Automatic Object Extraction from Airborne Laser Scanning Point Clouds for Digital Base Map Production

Widyaningrum, E.

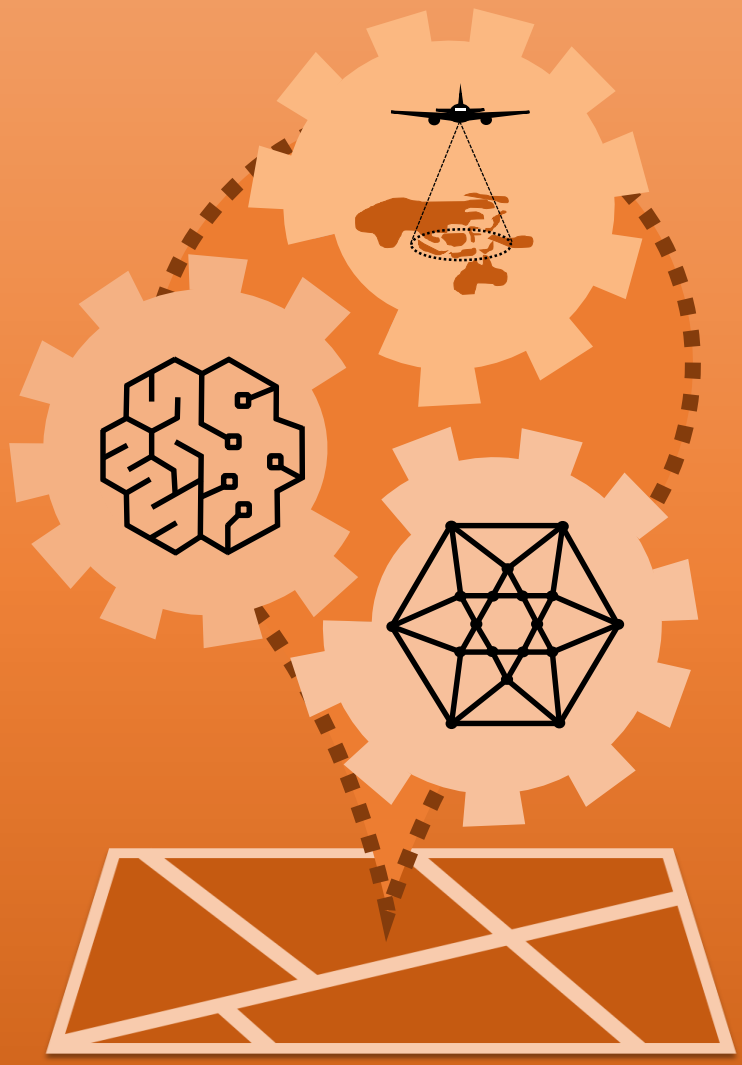
DOI
10.4233/uuid:8900fac8-a76c-482a-b280-e1758783b5b3

Publication date
2021

Document Version
Final published version

Citation (APA)
Widyaningrum, E. (2021). *Automatic Object Extraction from Airborne Laser Scanning Point Clouds for Digital Base Map Production*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:8900fac8-a76c-482a-b280-e1758783b5b3

Automatic Object Extraction from Airborne Laser Scanning Point Clouds for Digital Base Map Production



Elyta Widyaningrum

Automatic Object Extraction from Airborne Laser Scanning Point Clouds for Digital Base Map Production

Elyta Widyaningrum

Automatic Object Extraction from Airborne Laser Scanning Point Clouds for Digital Base Map Production

Elyta Widyaningrum

Automatic Object Extraction from Airborne Laser Scanning Point Clouds for Digital Base Map Production

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus prof.dr.ir. T.H.J.J. van der Hagen
chair of the Board for Doctorates,
to be defended publicly on Wednesday 10 March 2021 at 10:00 o'clock

by

Elyta WIDYANINGRUM

Master of Science in Land Management and Land Tenure
Technical University of Munich, Germany

born in Yogyakarta, Indonesia

This dissertation has been approved by the promotors:
Prof.dr.ir. R.F. Hanssen
Dr. R.C. Lindenbergh

Composition of the doctoral committee:

Rector Magnificus	Chairman
Prof.dr.ir. R.F. Hanssen	Delft University of Technology, promotor
Dr. R.C. Lindenbergh	Delft University of Technology, promotor
Independent members:	
Prof.dr.ir. P.J.M. van Oosterom	Delft University of Technology
Dr.ir. B.G.H. Gorte	University of New South Wales, Australia
Dr. H. Ledoux	Delft University of Technology
Prof.Dr.-Ing. M. Gerke	Technische Universität Braunschweig, Germany
Dr.ir. A.K. Mulyana	Indonesian Geospatial Information Agency (BIG)
Prof.dr.ir. H.W.J. Russchenberg	Delft University of Technology, reserve member

Keywords: airborne laser scanning point clouds, aerial images, base map, object extraction, building outline, road centerline, deep learning.

ISBN: 978-94-6366-382-3

Printed by Ridderprint BV
An electronic version of this dissertation is available at http://repository.tudelft.nl.

To my family

Acknowledgements

Thanks to Allah SWT, the God almighty, for giving me a blissful life and wonderful opportunities I have so far. I experienced new things and colorful moments, especially during my PhD. My research on automatic point clouds processing is first initiated when I was witnessing the bottleneck of the Indonesian large scale base map production. Acknowledging that data processing took longer time than its acquisition gave me an insight that there should be a way to accelerate it.

I am deeply grateful to my promotor and daily supervisor Roderik Lindenbergh for his invaluable assistance, critical feedbacks, and thorough reading of all my papers. I have learn a lot from Roderik and I am immensely thankful for his support, trust, and freedom to conduct this research. I would like to express my sincere gratitude to Ramon Hanssen, my promotor, for insightful comments and suggestions for this manuscript. Additionally, I would like to thank to my former supervisor, Ben Gorte, for providing me with technical support during one year supervision and during my research exchange in University of New South Wales Australia which was influential in shaping my research methods. Extended gratitude for GRS secretariat teams – Debbie, Lidwien, and Suzanne for always helping me with various non-technical and important administrative matters during my study.

I would like to thank my friends and colleagues of optical laser scanning group – Kaixuan Zhou, Aadrian van Natijne, Mieke Kuschnerus, Linh Truong Hong, Yufu Zhang for valuable discussions and cherished time spent together in the department. My appreciation also goes out to my BIG friends for their support all during my studies. I would like to extend my gratitude to Indonesian Endownment Fund and Research (LPDP) for the scholarship that allowed me to conduct this PhD research.

I would also like to express my deepest appreciation to my doctoral defense committee: Peter van Oosterom, Markus Gerke, Hugo Ledoux, Ben Gorte, Ade Komara Mulyana, and Herman Russchenberg, for their thorough reading of my thesis work and their valuable feedbacks.

Finally, I would like to express my gratitude to my Dad for his endless love, support, and sincere prayer. Many thanks for my sister, Anita, Adinda, and my brother, Aditya, for the wonderful time together. Last but not least, special thanks to my little family, Agung Indrajit and Felix Aria Indrajit, for their tremendous love, understanding, and supporting me to finish my thesis. You both make my life complete.

Summary

A base map provides essential geospatial information for applications such as urban planning, intelligent transportation systems, and disaster management. Buildings and roads are the main ingredients of a base map and are represented by polygons. Unfortunately, manually delineating their boundaries from remote sensing data is time consuming and labour intensive.

Airborne laser scanning (ALS) point clouds provide dense and accurate 3D positional information. Automatic extraction of buildings and roads from 3D point clouds is challenging because of their irregular shapes, occlusions in the data, and irregularity of ALS point clouds.

This study focuses on two particular objectives: (i) accurate *classification* of a large volume of ALS 3D point clouds; and (ii) smooth and accurate building and road *outline extraction*. To achieve the classification objective, we perform point-wise deep learning to classify an ALS point cloud of a complex urban scene in Surabaya, Indonesia. The point cloud is colored by airborne orthophotos. Training data is obtained from an existing 2D topographic base map by a semi-automatic method proposed in this research. A dynamic-graph convolutional neural network is used to classify the point cloud into four classes: bare land, trees, buildings, and roads. We investigate effective input feature combinations for outdoor point cloud classification. A highly acceptable classification result of 91.8% overall accuracy is achieved when using the full combination of RGB color and LiDAR features.

To address the objective of outline extraction, we propose building and road outline extraction methods that run directly on ALS point cloud data. For accurate and smooth *building* outline extraction, we propose two different methods. First, we develop the ordered Hough transform (OHT), which is an extension of the traditional Hough transform, by explicitly incorporating the sequence of points to form the outline. Second, we propose a new method based on Medial Axis Transform (MAT) skeletons which takes advantage of the skeleton points to detect building corners. The OHT method is resistant to noise but it requires prior knowledge on a building's main directions. On the contrary, the MAT-based method does not require such orientation initialization but is more sensitive to noise on building edges. We compare the results of our building outline extraction methods to an existing RANSAC-based method, in terms of geometric accuracy, completeness of building corners, and computation time, and demonstrate that the MAT-based approach has the highest geometric accuracy, results in more complete building corners, and is slightly faster than other methods.

For *road* network extraction, we develop a method based on skeletonization, which results in complete and continuous road centerlines and boundaries. In our study area, several roads are disrupted and disconnected due to trees. We design a tree-constrained approach to fill road gaps and integrate road width estimated from a medial axis

algorithm. Comparison to reference data shows that the proposed method is able to extract almost all existing roads in the study area, and even detects roads that were not present in the reference due to human errors.

We conclude that our object extraction methods enable a complete automatic procedure, extracting more accurate building and road outlines from ALS point cloud data. This contributes to a higher automation readiness level for a faster and cheaper base map production.

Samenvatting

Een basiskaart biedt essentiële geospatiale informatie aan voor applicaties zoals stadsplanning, intelligente transportsystemen, en rampbeheer. Gebouwen en wegen zijn de hoofdbestanddelen van een basiskaart en worden door veelhoeken voorgesteld. Spijtig genoeg is het manueel afbakenen van hun grenzen op basis van remote sensing gegevens tijds- en arbeidsintensief.
Puntenwolken verkregen via laserscannen vanuit de lucht (ALS) leveren dichte en nauwkeurige 3D-positie-informatie. Automatische extractie van gebouwen en wegen op basis van puntenwolken is uitdagend omwille van hun onregelmatige vormen, occlusies in de gegevens, en onregelmatigheden van de ALS-puntenwolken.
Deze studie richt zich op twee bepaalde doelstellingen: (i) nauwkeurige classificatie van grote aantallen ALS-3D-puntenwolken; en (2) vloeiende en nauwkeurige contourafbakening van gebouwen en wegen. Om de classificatiedoelstelling te verwezenlijken voeren we puntsgewijze deep learning uit om een ALS-puntenwolk van een complexe stadsscene in Surabaya, Indonesië, te classificeren. De puntenwolk is met behulp van orthofoto's vanuit de lucht ingekleurd. Trainingsgegevens worden verkregen van een bestaande 2D topografische basiskaart door middel van een halfautomatische methode die in dit onderzoek voorgesteld wordt. Een dynamisch-grafisch convolutioneel neuraal netwerk wordt gebruikt om de puntenwolk in vier klassen te classificeren: onbebouwde grond, bomen, gebouwen, en wegen. We onderzoeken combinaties van effectieve invoerfeatures voor de classificatie van buitenpuntenwolken. Een zeer aanvaardbaar classificatieresultaat van 91.8% aan algemene nauwkeurigheid werd bereikt bij het gebruik van de volledige combinatie van RGB-kleuren en LiDAR features.

Om de doelstelling van contourafbakening te adresseren, gebruiken we voor gebouwen en wegen afbakeningsmethodes die rechtstreeks op gegevens van ALS-puntenwolken worden uitgevoerd. Voor nauwkeurige en vloeiende contourafbakening van gebouwen stellen we twee verschillende methodes voor. Ten eerste, we ontwikkelen de geordende Hough transformatie (OHT), wat een uitbreiding is van de traditionele Hough transformatie, door de sequentie van punten die de contour vormen, expliciet te gebruiken. Ten tweede, we stellen een nieuwe methode voor die gebaseerd is op Mediale Astransformatie (MAT) skeletten en die gebruik maakt van de skeletpunten om hoeken van gebouwen te detecteren. De OHT-methode is bestand tegen ruis maar vereist voorkennis over de belangrijkste richtingen van een gebouw. Integendeel, de MAT-gebaseerde methode vereist geen dergelijke initialisatie van de oriëntatie, maar is gevoeliger voor ruis op de randen van gebouwen. We vergelijken de resultaten van onze methodes voor contourafbakening voor gebouwen met een bestaande methode die op RANSAC is gebaseerd, op basis van geometrische nauwkeurigheid, volledigheid van de hoeken van gebouwen, en rekentijd. We tonen dat de op MAT gebaseerde aanpak de

hoogste geometrische nauwkeurigheid heeft, wat resulteert in meer complete hoeken van gebouwen, en tevens licht sneller is dan andere methodes.

Voor de afbakening van wegennetwerken ontwikkelen we een methode die gebaseerd is op skeletisering, wat resulteert in volledige en ononderbroken middellijnen en randen van wegen. In ons studiegebied zijn verschillende wegen verstoord en onderbroken door bomen. We ontwikkelen een boombeperkte aanpak om hiaten in wegen op te vullen en wegbreedtes, die op basis van een mediaal as-algoritme geschat worden, te integreren. Vergelijking met referentiegegevens toont dat de voorgestelde methode in staat is om bijna alle bestaande wegen in het studiegebied te extraheren, en zelfs om wegen te detecteren die niet aanwezig waren in de referentiegegevens omwille van menselijke fouten.

We concluderen dat onze extractiemethodes voor objecten een compleet automatische procedure toelaten om meer accurate gebouw- en wegcontouren op basis van gegevens van ALS puntenwolken te extraheren. Dit draagt bij aan een hoger niveau van automatiseringsgereedheid, voor een snellere en goedkopere productie van basiskaarten.

Table of Contents

List of Acronyms

2D	Two Dimensional
2.5D	Two and a half Dimension
3D	Three Dimensional
AHN	Actual Height Data of the Netherlands
AI	Artificial Intelligence
ALS	Airborne Laser Scanning
CAD	Computer Aided Design
CNN	Convolutional Neural Network
CRF	Conditional Random Field
DBSCAN	Density Based Clustering Spatial Clustering of Applications with Noise
DGCNN	Dynamic Graph Convolutional Neural Network
DL	Deep Learning
DSM	Digital Surface Model
DTM	Digital Terrain Model
FL	Focal Loss
FN	False Negative
FP	False Positive
GIS	Geographic Information System
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPU	Graphics Processing Unit
HA	Hough Accumulator
HPC	High Performance Computing
ICA	International Cartographic Association
INS	Inertial Navigation System
IRN	Intensity, Return number, Number of returns
KDE	Kernel Density Estimator
LiDAR	Light Detection and Ranging
MA	Medial Axis
MAT	Medial Axis Transform
ML	Machine Learning
MLP	Multi Layer Perceptron
MRF	Markov Random Field
MVCNN	Multi View Convolutional Neural Network
NDVI	Normalized Difference Vegetation Index
NMA	National Mapping Agency
OA	Overall Accuracy
OHT	Ordered Hough Transform

RANSAC	Random Sampling Consensus
RF	Random Forest
RGB	Red, Green, Blue
RGI	Red, Green, Intensity
RGBIR$_n$N	Red, Green, Blue, Intensity, Return number, Number of returns
RMSE	Root Mean Square Error
SCE	Softmax Cross Entropy
SRGI	Indonesian Geospatial Reference System
SPG	Super Point Graph
SOR	Statistical Outlier Removal
SVM	Support Vector Machine
TP	True Positive
UTM	Universal Transverse Mercator
WGS	World Global System

1

Introduction

1.1 Motivation

Extracting outlines of topographic objects, like streets or houses, for mapping is a labor-intensive and time-consuming task. This research proposes methodology to extract GIS map elements (buildings and roads) automatically from airborne laser scanning (ALS) point cloud data enriched by aerial orthoimage color information.

1.2 Background

The use of geospatial information is increasing rapidly. There is growing recognition in both governments and the private sector that geospatial data is a vital component of effective decision making (UN-GGIM, 2015). In this digital era, geospatial technologies are revolutionizing the economy (World Bank, 2019). Geospatial information becomes a part of daily life and used to guide the mobilization of people and goods, to perform spatial modeling of an area, and to provide analysis of investment and business.

Geospatial information is the spatial aspect describing the location and position of an object or event that is beneath, on or above the earth surface, which is expressed in a particular coordinate system (Stock and Guesgen, 2016). A more complex definition could include different location-related datasets combined into layers that show information such as land use and population density. At its simplest, geospatial information is defined as the basic information found on a base (topographic) map. Once geospatial information is created, it can be used many times to support a multiplicity of applications (UN, 2015). The production of a map involves a long and complex procedure. Considering the importance of base maps in providing geospatial information for public needs, there is a need for more automated approaches to ensure the community get the right data at the right time.

This research focuses on the automation of the process of acquiring topographical object outlines, buildings and roads, for base mapping from ALS point cloud data. In the following, the digital map production workflow is introduced. Next the role of ALS point clouds in digital base mapping is explained, and general overview on map automation using artificial intelligence are discussed.

1.2.1 Digital base mapping

Maps are abstract models of reality which are designed to facilitate the extraction of all sorts of metrical and topological properties. They convey information on location, direction, distance, height, magnitude, connectivity and spatial association (Visvalingam, 1990). Digital mapping refers to the extraction and representation of spatial objects and their relationships in a complete, explicit and coherent but not redundant form on or by a computer. A digital map is a compact, structured, and elegant representation of geospatial data and their attributes on a computer-based system (Visvalingam, 1989).

A digital map is generally represented by two data types: raster and vector. Raster data stores information in the form of a matrix of grid cells or pixels. Each cell or pixel is represented by the cell address of its corner or center location. Each cell also has discrete attribute value assigned to it. Aerial and satellite images are examples of raster data, which are stored into different file formats such as: .geotiff, .img, .bil, etc. Vector data represents information into three basic types: points, lines, and polygons that delineate object outlines. A pair of points forms a line segment, and an ordered set of connected line segments forms a line. A set of connected lines where the start and end point have the same coordinates forms a polygon. Points composing a line or a polygon have coordinates and can be assigned with additional attribute information. Several data file types for vector format are: .shp, .dwg, .las, etc. Characteristics of raster and vector data for representing geographic features are illustrated in Figure 1.1.

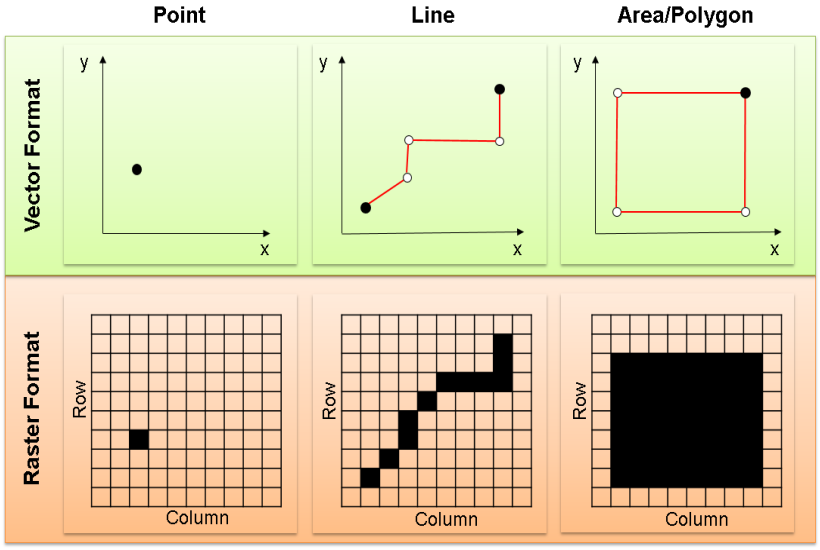


Figure 1.1 Digital representation of geographic features (point, line, and area) in vector (green) and raster format (orange). In vector format, each geographic feature is defined spatially by Easting (x) and Northing (y) coordinates. In raster format, data is represented as a grid structure where each grid cell or pixel is referred to by its particular column and row index.

Considering the current base map necessity, many people, organizations and institutions prefer to produce maps in vector format above raster format. Compared to the raster format, the vector format has some advantages which are described as follows:

- **crisp** – vector data looks crisp at any scale with discernable boundaries, from the highest to the lowest zoom level;
- **attribute** – vector data can be conveniently enriched by numerous attributes in text format as well as integer, and floating numbers;
- **size, speed and resources** – vector data is usually less than 50% of the size of raster data which requires less storage, less time to transmit, and less resources for processing.
- **data sharing and integration** – vector data is easier to be shared and integrated with other spatial and non-spatial data, for example associating population density into an administrative boundary layer.
- **style** – vector data allows various cartographic styles to be applied according to the user needs.

According to their purpose, different digital map provides different geospatial information. A (topographic) base map has general purpose to represent the visible features of landscapes such as buildings, relief, water bodies, and roads and serve as reference (in terms of geometry and positional) for thematic maps. Such reference map is regarded as scientific documents that should fulfill high standards on positional accuracy (Visvalingam, 1990). Base maps are produced at different scale, level of detail, and quality (American Society of Photogrammetry, 1980). For urban areas, topographic objects need to be mapped at a scale of 1:10,000 or larger. As many countries suggest topographic base maps as reference for thematic maps, cartographic enhancement and the geometric accuracy are important issues when topographic base maps are created (Hoehle, 2017).

Using Indonesia as research area, an illustration of a complete base map production workflow using traditional methods is presented in Figure 1.2. Traditional methods employ humans to interpret and delineate map objects. The map production workflow consists of five main steps: object delineation, DTM (Digital Terrain Model) generation, topology validation, toponym surveying and field checking, and database synchronization and cartography.

Based on the Regulation of Geospatial Information no.11/2018 on the Technical Analysis of the Implementation of Geospatial Information that is used to estimate project time length and cost in Indonesia, it is known that object delineation takes most time and costs, about 40% of the total allocated time and budget. Thus, automating this step is believed to strongly accelerate the map production workflow.

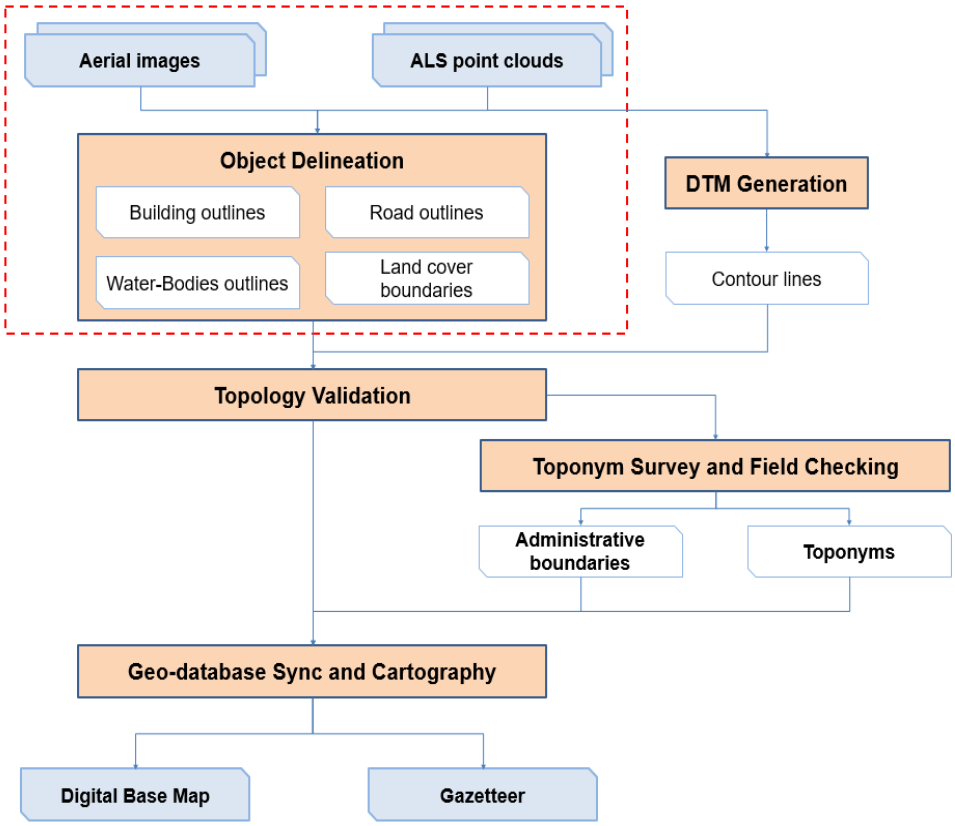


Figure 1.2 A digital base map production using airborne images and ALS point clouds workflow using traditional methods, consists of 5 main tasks: object delineation, DTM generation, topology validation, toponym (geographical names) and field checking survey, and geodatabase synchronization and Cartography. The most time consuming and expensive task, object delineation, is inside the red dotted box. This particular workflow is from Indonesia.

1.2.2 ALS point clouds as input for mapping

Airborne Laser Scanning (ALS) is recognized as one of the major data acquisition techniques in the community of photogrammetry, remote sensing, and computer vision. An ALS system, is able to acquire accurate and dense 3D LiDAR point clouds representing the topographic surface. Another main strength of ALS systems lies in the fact that the LiDAR signal is able to penetrate small gaps in vegetation and other semi-transparent objects above the surface. 3D point clouds have several characteristics, which make data processing very challenging: they are unordered, unstructured, and irregular. ALS point clouds are often considered to have 2.5D geometric structure (Su et al., 2016) and in general contain no RGB color information. Here, 2.5D point cloud

refers to the projection of the points along a specific direction (in this case the Z-direction). For example a building is seen only from the top although few points of building facades are also presented. Figure 1.3 visualizes different geospatial data over the same area. This example includes ALS point cloud data, aerial orthoimages, and the corresponding 1:1000 base map.

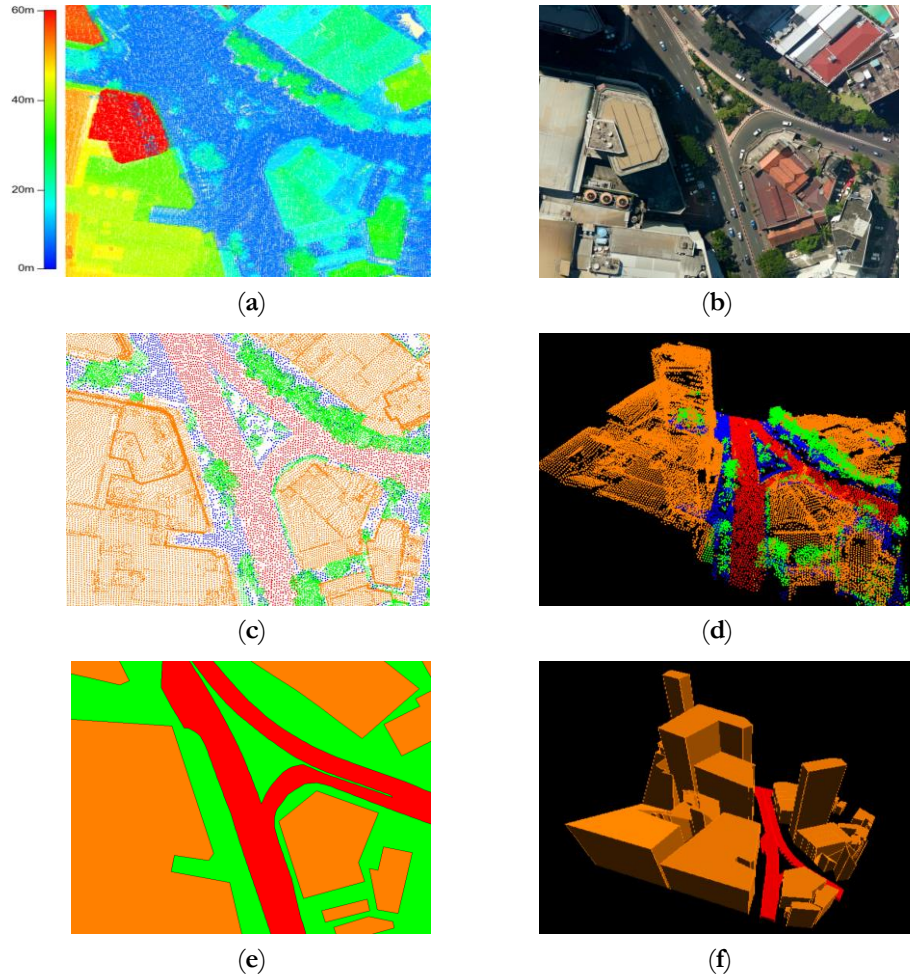


Figure 1.3 Visualization of different geospatial data over the same urban area in Surabaya, Indonesia. (**a**) colored ALS point cloud by height; (**b**) aerial ground orthoimage shows shadowed area caused by high buildings; (**c**) 3D visualization of labeled ALS point cloud colored by object class; (**d**) ALS point cloud colored by object class (blue: ground, green: trees, orange: buildings, red: roads); (**e**) the top view of 1:1000 base map (green: ground and vegetated areas, orange: buildings, red: roads); (**f**) 3D visualization of building and road of 1:1000 base map.

In comparison to aerial images, ALS point clouds may have limitations in representing sharp edges. This is because, in general, the laser does not always exactly hit an object along its edges. But, ALS point clouds are considered superior to aerial images for building extraction, since ALS is shadow independent and is free from relief distortion. Furthermore, LiDAR data provides a unique data source and has advantages over satellite and aerial images in capturing urban features with accurate height information, especially for building extraction (Yu et al., 2010).

As stated by Petrie and Toth (2009), the appearance of laser scanning system have been affecting geospatial data acquisition since laser scanning has an outstanding ability to represent and portray topography, building structures, and foliage with precise, fine, three-dimensional details. Compared to traditional 2D images, 3D LiDAR point clouds have the capability of providing significantly richer geometrical information cues for analyzing objects and environments (Han et al., 2020) e.g. position, size, shape and object orientation (Vosselman et al., 2005; Lin et al., 2020). However, LiDAR point clouds have limited reflectance properties regarding the object surface compared to aerial images. Regarding the object boundaries, LiDAR point clouds are usually not as good as aerial images due to their intrinsic characteristics (irregular and unstructured).

Given the advantages and disadvantages, many studies suggested to combine both data to improve the degree of automation and the robustness of object extraction (Schenk and Csatho, 2002). For the time being, the question of how to optimally use ALS data together with aerial imagery has still not been fully solved (Jarzabek-Rychard and Maas, 2017).

1.2.3 Map automation and artificial intelligence

Automation is a process, operation, or a system by a machine for repetitive tasks to limit human intervention. Map production is the process to extract and compile geographical data on a map. A workflow is a process or procedure in which certain tasks are completed. Therefore, an automatic map production workflow can be defined as a machine-driven process that results in the completion of tasks related to the extraction, compilation or construction of a map product.

Technology to automatize mapping is necessary to provide the community with up-to-dated and reliable geospatial information. For more than 50 years, automation of map production has been recognized to increase processing speed and improve map quality. Tobler (1959) stated that most basic tasks in cartography can be automated and that the volume of maps produced in a given time will be increased while costs are reduced. Robinson et al. (1995) stated the importance to compile a map in a way that is as easy as possible. This means that to perform an automated workflow, it is necessary for map-makers to sort out certain things associated to inputs and outputs, such as: the data sources (including their spatial and temporal resolution, geometric accuracy, cost, permissions, etc.), the map specification (including map projection, scale, area, map

element, mapping method, etc.), and the final map representation (including data output format, printed map, and web map).

In recent years, automation and artificial intelligence are used interchangeably in reducing human resources for effective and efficient data processing tasks. Artificial intelligent is a science that mimics human intelligence and behavior to solve problems and finish certain tasks. Based on this definition, automation can be used with or without artificial intelligence. Automation is often applied for fix procedures programmed by humans or simply just by replacing workers without the need to learn or solve problems.

Automatic object extraction for mapping consists of two primary tasks: object classification and outline extraction. Machine learning has the capability to parse and learn from data, and make the best possible decision based on what it has learned. This makes machine learning powerful for remote sensing classification. A special type of machine learning, deep learning, has received increasing interest in the last years. This is due to its capability to design features from data and make intelligent decisions on its own. Deep learning has proven to be faster and more accurate in classification of complex and huge data than machine learning (Najafabadi et al., 2015). We define the relation between automation and artificial intelligence, including machine and deep learning, as presented in Figure 1.4.

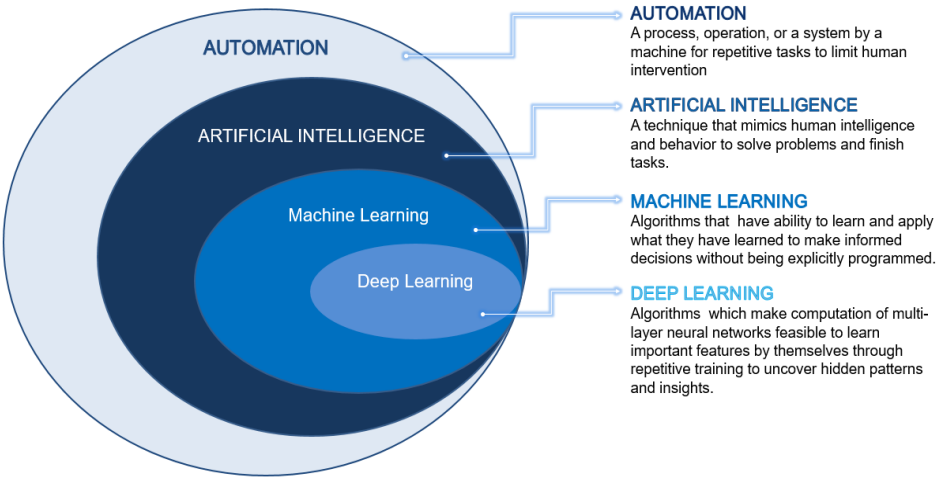


Figure 1.4 Relation between automation, Artificial Intelligence (AI), Machine Learning (ML), and Deep Learning (DL). Automation can be applied using AI. ML is a subset of AI that enables machines to automatically learn and use experience to improve performance. DL is a subset of ML where multiple layers of a neural network are stacked together to create a huge network to map input into output. Illustration is adapted from Goodfellow et al (2016).

The demand for digital base maps, especially for buildings and roads, for various applications is increasing. As a consequence, authoritative organizations like National Mapping Agencies (NMA), face challenges to produce the digital base map in a short period of time with limitations on financial and human resources (Eidenbenz et al., 2000). To cope with higher product demands, increased productivity and lower costs, automation tools in the production should be employed. As laser scanning point cloud data provides very reliable and detailed surface descriptions, it is obvious that automated point cloud processing for map production may lead to significant benefits.

1.3 Problem formulation

The main problem with current traditional map production, with its manual detection and delineation of objects in remote sensing data, is that it is labour intensive, and costly (Idris et al., 2012). Moreover, traditional map production may lead to inconsistencies, subjectivity and blunders. However, automatizing map production in a time- and cost-efficient way while maintaining map quality has its own problems, in particularly for urban areas. These problems are caused by the high diversity and complexity of real-world scenes (Poullis, 2013): the same object may have different shape, size, color, and texture, depending on acquisition time, sensor, and acquisition geometry (Awrangjeb et al., 2014). As an example, building roofs may have different shape, color, and size, and their appearance is further affected by small details (trees, antennas, terrasse chairs, etc.). The fact that the same objects may have different object characteristics makes automation even more problematic.

The use of 3D point clouds for automatic map object extraction remains problematic due to the absence of structure, order and semantics as well as the inherent irregularity, incompleteness, and ambiguity of point clouds. These problems explain why 3D point clouds are not very well described and difficult candidates for procedural and algorithmic programming (Kanevski et al., 2009). We conclude that due to the diversity of object appearances and the complexity of object structures, the problem of the automation of point cloud object extraction is still an active field of research.

Given the issues, in this following, we focus on discussing the particular problems with the automatic map element extraction from ALS point clouds for two objectives: (i) accurate classification of a large volume of ALS 3D point cloud; (ii) smooth and accurate buildings and road outline extraction.

1.3.1 Classification and segmentation

Scene classification is a key prerequisite for automatic object extraction. This means that the quality of object extraction is largely determined by the classification accuracy. Classification can be defined as the process of categorizing data (pixels, voxels, or points) into multiple homogenous groups, where data of the same group will have

similar properties. In this research, we define a point cloud classification as a process to assigning each point of a point cloud with a specific class or semantic label, also known as semantic segmentation or point labeling.

Classification or semantic segmentation of point clouds is considered non-trivial work in complex urban scenes (Bláha et al., 2016). Although a large number of remote sensing classification techniques have been studied for more than half a century, it is still difficult to determine a unique optimal classification method due to object variations, object complexities, and occlusions (Weng, 2012; Hu et al., 2014). Various landscapes and different data resolutions (spatial and temporal) may require different classification techniques and settings. For point clouds, classification is even more challenging due to their high redundancy, uneven sampling density, and lack of explicit structure (Nguyen and Le, 2013; Kang and Yang, 2018). The two most trivial parts of classification are discriminative features and proper classifiers (Zhu et al., 2017).

Nowadays, deep learning is the fastest growing technique in pattern recognition, computer vision, and data analysis (Zhu et al., 2017). As indicated by its name, deep learning is able to learn and extract high dimensional features from training data by itself. The deep learning capability to extract high-level features, complex abstraction, and data representations from large volumes of data makes it attractive for remote sensing data processing, particularly for detection, classification, and semantic segmentation tasks. Deep learning, which is mostly applied to structured grids (images), has been implemented with more and less success for geospatial 3D point clouds. Using deep learning to directly process 3D point clouds is challenging due to several factors e.g. high dimensionality, sparsity, and the unstructuredness of point cloud data (Guo et al., 2020). Earlier approaches overcame this challenge by transforming the point cloud into a structured grid (image or voxel) which lead to the increase of computational costs or loss of depth information (Bello et al., 2020).

PointNet (Qi et al., 2017) pioneered 3D point-wise deep learning methods that directly process point clouds for object detection and classification task. However, there are problems remaining when directly from ALS point clouds using deep learning to classify urban objects need to be solved. First, deep learning can take a large number of good training samples to extract the high-level features and learn hierarchical representation of data with large variety and veracity (Zhang et al., 2018). Nevertheless, the determination of an optimal number and quality of training samples to obtain acceptable classification accuracy is still unknown. The quality of training samples is closely related to correct point labeling, presence of noise, and sufficient object type's representation. Second, deep learning involves many parameters and settings, which are not intuitive to be linked to the real world. Up to now, many point-wise deep learning architectures work well for 3D indoor point clouds. The implementation of indoor point-wise deep learning for 2.5D ALS point cloud, needs to be studied further as it requires at least additional parameter tuning. Third, the irrelevant input feature might

cost a great deal of resources during the training process of neural networks. Good feature selection results can improve learning accuracy, simplify learning results, and reduce learning time for deep learning methods (Cai et al., 2018). Combination of airborne point clouds and images is expected to increase point-wise deep learning classification. ALS point clouds have the advantage at having accurate 3D coordinates and several additional off-the-shelf features such as intensity, return number, and number of returns. On the other hand, aerial images offer spectral color information that may provide more distinctive features but could also add more noise.

1.3.2 Building and road extraction

Map elements (building polygons and road centerlines) are essential for a wide range of applications such as urban planning, disaster response, intelligent transportation system, etc. On the map, buildings are represented by their outlines. Currently, delineating building outlines and road centerlines requires extensive manual work in map production. It is time consuming and expensive since it requires labor to draw building outlines or road centerlines one by one, especially in urban scenes. This is one of the reasons that makes base map production or updating difficult to be completed in a specific time.

One promising solution for enhancing base map production is by automatizing the process of object outline extraction or vectorization. Automatic processing aims at the reduction of processing time, costs, and human errors. Many studies have been investigating automated procedures for object outline extraction in the last two decades. Yet, we consider it an open problem due to its complexity and large variation in building and road structures.

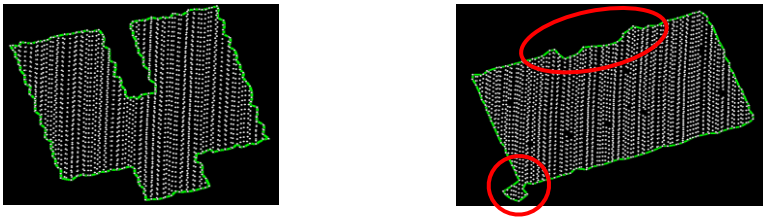
The building roof outlines extracted in this research are designed to meet specifications and assumptions as specified below:

- a building is represented by the size and shape of a 2D representation of its roof. That is, the size and shape of the buildings are similar to the roof as seen from the top. Buildings with overhanging roofs will have similar sizes and shapes on the map as its roof;
- small details or interior parts inside the building roof will not be considered. For example, in extracting outlines, a building with a gable roof will be treated as a flat roof;
- the expected result should meet the requirements of at least 1:5.000 map scale. Any building of an area of 2.5 m x 2.5 m or larger should be present in the map;
- the positional accuracy of building roof outlines should be similar to or better than 1.5 meter.

The road centerlines and outlines use assumptions and must meet specifications as specified below:

- the road is represented by the size and shape of a 2D representation of the road including the road centerline and outlines;
- the road should be topologically correct and should not have gaps due to occlusions (trees and cars);
- the expected result should meet the requirements of 1:5.000 map scale. A road having a width more than or equal to 2.5 m should be represented by a polygon and centerline. A road having width less than 2.5 m is represented by its centerline only;
- the positional accuracy of a road centerline should be at least 1.5 meter.

Buildings outlines are obtained by connecting all edge points. Edge points are provides by the concave hull (Moreira and Santos, 2007) or the alpha-shape (Edelsbrunner, Kirkpatrick and Seidel, 1983), but their output is usually jaggy or wavy as shown in Figure 1.5.a. When using these well-known bounding hull algorithms, flaws on the object edge (inside the red ellipses in Figure 1.5.b) may still remain as they do not consider the object characteristics (Guercke and Sester, 2011). Existing work on line regularization that process directly on a point cloud to obtain smooth and complete outlines have certain limitations (e.g. only consider two building primary directions, limit the building direction at certain angle differences).



(**a**) (**b**)

Figure 1.5 Noisy and jaggy building outlines due to data imperfections. (a) for a building without tree disturbance, initial building outlines are typically jaggy and wavy; (b) flaws (inside red circles) exist on the building edge due to trees close to the building.

Although a number of advancements in road extraction from ALS point clouds have been made, there are still two notable unsolved problems: how to obtain accurate road point from a point cloud and how to avoid the influence of attached areas (e.g., parking lots and bare grounds) on road extraction (Hui et al., 2016). Similar to the building case, extracted road centerlines are often jaggy or wavy because of noisy road edges. In addition, an extracted road network may also suffer from occlusions by trees and cars, which can disturb the network topology. For example, a tree with dense and wide

canopy on the side of the road hampers road visibility which in the end causes inaccurate centerlines, road gaps, or network discontinuity. To conclude, an automatic process to extract building and road outlines for a base map using ALS point cloud data involves as sequential steps: classification and segmentation, edge point detection, and line regularization. Each step may have error which can be propagated and will accumulate in the last step. Given these facts, the following the research questions are formulated.

1.4 Research questions

Based on the problems discussed in the previous section, the main objective of this thesis reads:

to develop automatic methods to extract topographic object outlines for digital base mapping using airborne laser scanning point cloud data supported by aerial images.

We aim to minimize human efforts in producing object outlines required for digital base maps. Object outlines in this research are the polygons representing shapes of topographic object, in this case buildings and road networks. To extract such outlines, high resolution and accurate data is required. Airborne laser scanning (ALS) provides dense and accurate 3D points representing the topographic surface which is essential for object extraction. Aerial images provide color information that can be used in addition to increase the quality of object outline results.

Given the above main objective, the main research questions and their sub-questions are defined as follows:

1. **How to accurately classify huge point cloud data into several classes in a way feasible for routine map production using deep learning?**
   One of the core problems in classification and segmentation is how to select and use effective features to obtain classification efficiency and result accuracy. Several machine learning methods to classify point clouds have been introduced. Different machine learning methods have different advantages and limitations. One of this research main concerns is related to classification of large volumes of ALS point cloud data to obtain an acceptable accuracy result. Therefore, it is essential to choose a classification method properly fit to the data characteristics and objectives. Deep learning gained much popularity due to its supremacy in terms of accuracy and handling huge amount of data. Given the advantages of deep learning, we attend the following sub-questions related to point cloud classification method:
   1.1. How to effectively utilize additional features from aerial images to increase the accuracy of ALS point cloud classification?

1.2. How to provide good and cheap training samples for ALS point cloud classification?

2. How to accurately extract complete and smooth road networks from given segmented road points?

Similar to buildings, the accuracy of road network outlines and centerline extraction depends on the road points classification and segmentation quality. The difficulties of road extraction from ALS point clouds lie in the fact that existing road detection methods mainly used cues based on color, monochromatic intensity, and texture. Different areas may have different road color and patterns. Distinctive features are sometimes hard to find in some study areas. Variations in the road neighborhood are sometimes insufficient to separate road from non-road objects. For example, a front yard of a house with color similar to a road sometimes can be detected as road. Moreover, road network extraction involves network structure (topology) which add complexity to the task. In this aspect, the following two sub-questions are derived:

2.1 How to obtain a complete and accurate road centerlines from given segmented road points, where these road points may be affected by gaps and noise?

2.2 How to obtain outlines representing the actual road borders?

3. How to accurately extract straight and smooth building outlines from given segmented building points?

The accuracy of building outlines depends on the quality of point cloud classification and segmentation. However, the high complexity of real world scenes and huge data volumes may lead to imperfect classification and segmentation results. ALS point cloud irregularity with its limited color information (only intensity), makes classification and segmentation even more challenging. In many cases, buildings and trees are confused in classification and segmentation results. Regardless the color, points on buildings and trees sometimes share similarity in pattern or characteristics that may lead to confusion. Points on building roofs are often classified as tree points or vice versa, as buildings are often adjacent to trees. Such conditions may results in over- and under building segmentation, which causes false building outline extraction results. Various classification methods using different approaches have been introduced, but due to LiDAR point cloud characteristics, flaws in classification and segmentation results may still exist. Thus, the following sub-questions will be addressed:

3.1 How to mitigate the effect with of noise and flaws on building edges?

3.2 How to accurately acquire complex building outlines of arbitrary shape?

1.5 Scope and limitations

In this research, the proposed methods are tested on ALS point clouds in combination with RGB color from aerial orthoimages (also known as orthophotos). This research uses datasets acquired from airborne optical systems and assumes that the input data sources are correct, well confirmed, and ready for further processing. Therefore, we will not discuss processing stage such as point cloud or image acquisition, geo-referencing, and co-registration (strip or bundle adjustment). To provide automatic map object extraction, we consider only the extraction methods of the two most expensive map objects i.e. buildings and road networks. This research is expected to generate map elements that at least meet the 1:5.000 map scale specifications (one meter positional accuracy). The map have at least 2D vector format specified as GIS vector data (e.g shapefile) using the same coordinate reference system as the original input data. The expected 2D vector map is supposed to be 3D-ready data, which means that it allows association or attributing object height (Z-drapping) for further step. Thus, assigning Z-value (object height) to obtain 3D vector data from our 2D vector results is not considered in this research.

1.6 Outline and research methodology

This chapter briefly describes the motivation, background, problems, research objective as well as scope and limitations. Figure 1.6 gives an overview of the methodology applied in this research. The main research objective is formulated and sub-divided into three questions. The three main chapters presenting new methodology (Chapter 3 to 5) address the three research questions as described in Section 1.4 and correspond to scientific papers that have been previously published either in a scientific journal or in conference proceeding. All the papers were subject to a full paper peer-reviewing process.

Chapter 2 provides an overview of automatic digital base map production and challenges on using ALS point clouds. A general overview on work on automatic object outline extraction from ALS point cloud data, including classification and segmentation, and line extraction, is presented. This chapter is intended to provide sufficient foundation to build upon in subsequent chapters.

Chapter 3 examines different feature combinations and loss functions for the classification of colored ALS point clouds using a point-wise deep learning approach. The point cloud is colored by ground orthophotos, which were acquired at the same time from the same platform as the ALS point cloud. We try different input features combination to investigate the importance of different off-the-shelf features from ALS point clouds and aerial images. Different loss functions are also applied to minimize the effect of class imbalance as exists in our study area caused by a combination of dense buildings separated only by small roads. This chapter also provides an integrated procedure for road vectorization starting from a large size airborne point cloud

sampling in an urban scene. Parts of this chapter have been published in the proceedings of the 40th Asian Conference on Remote Sensing, Daejeon, South Korea (Widyaningrum and Lindenbergh, 2019) and the Remote Sensing journal (Widyaningrum et al., 2021).

Chapter 4 introduces building outline extraction from ALS point clouds using a method called Ordered-list Hough Transform (OHT). Our literature review shows that methods on building outline extraction use regularization to obtain smooth and accurate building outlines. Nevertheless, the currently acquired quality (in terms of geometric accuracy, straightness, and completeness) of the extracted building outline results need to be improved, especially for complex buildings. Hough transform, which is initially invented to identify complex lines in images, was applied to perform outline detection from point clouds data although its performance on detecting different sizes and orientations of buildings automatically remains a problem. We design the criteria that meet the challenge and has successfully demonstrated this on three different study areas of different characteristics. ALS point clouds of each study area are classified and segmented by different segmentation methods to test and evaluate the stability of the proposed method. Chapter 4 was published in the Remote Sensing journal, see Widyaningrum et al. (2019).

Chapter 5 presents our work on building outline extraction from ALS point cloud using a so-called Medial Axis Transform (MAT) approach. Considering that existing methods on building outline extraction mainly use specific rules to determine the building orientations which then lead to limitations to detect outlines for buildings of arbitrary orientation, we suggest a primitive-free approach based on skeletonization to extract accurate and straight outlines of buildings of arbitrary orientation. A shrinking circle method is applied to building edge points for obtaining medial skeleton points. Based on the segmented medial points and medial descriptors, building corner points can be estimated. A comparison to outcomes of existing building outline extraction methods of the same study area is presented. Chapter 5 was published in the Pattern Recognition journal, see Widyaningrum et al., (2020).

Chapter 6 gives conclusions and recommendations for future work.

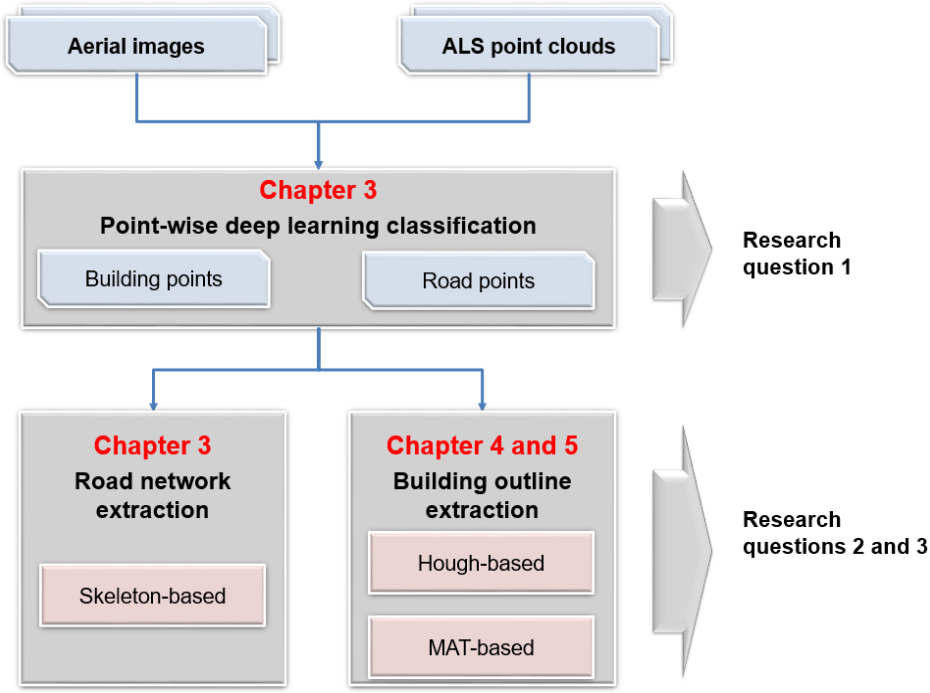


Figure 1.6 Overview of the methodology developed in this research.

2

Automatic Building and Road Extraction

In this chapter, a general overview on the use of ALS point clouds for automatic building and road extraction is given. Building and road extraction delineate their physical boundaries in planimetric which may implicitly include the classification and segmentation problem. Base map specifications and ALS point cloud characteristics are reviewed in Section 2.1. As prerequisite for extracting object outlines, ALS point cloud classification and segmentation are required, which is discussed in Section 2.2. Section 2.3 describes line extraction methods and line regularization methods, which are particularly useful for building outline and road network extraction. Section 2.4 provides a short summary of this chapter.

2.1 ALS point cloud for mapping

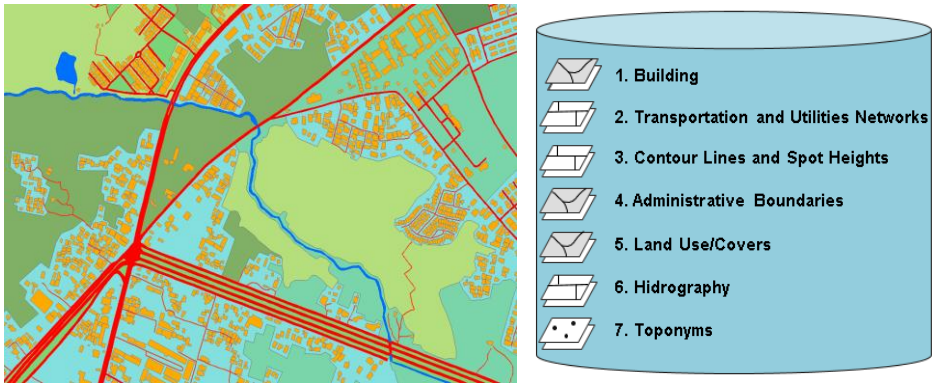
The importance of automatic point cloud processing is increasing with the interest of using laser scanning for various applications, in particular for mapping. Despite the advantages in providing extensive and accurate 3D data within reasonable time, methods to automatically process point clouds are still open for further development. Considering their characteristics, ALS point clouds have both advantages and disadvantages with respect to automatic object extraction (Schenk and Csatho, 2002). Digital base map requirements and point cloud characteristics are presented in Section 2.1.1 and Section 2.1.2, respectively.

2.1.1 Digital base map

The definition of base map varies and evolves over time. According to the American Society of Photogrammetry (1980), a base map is the graphic representation of the earth surface at a specified scale of selected fundamental map information and is used as a framework upon which additional data of a specialized nature may be compiled. The International Cartographic Association/ICA (1996) defines a map as a symbolized image of geographical reality, representing selected features of characteristics resulting

from the creative effort of its author's execution of choice and is designed for use when spatial relationships are of primary relevance. The last two decades, digital mapping has been introduced. A base map is defined as a layer that provides essential information on common land features upon which mapping applications may be performed and from which more specialized data may be derived (Decker, 2000). In the current definition, a base map provides not only geospatial information but should also include spatially associated attributes and should be easy to share and integrate. Figure 1.1 shows a digital representation of a base map consisting of several layers: buildings (orange), transportation networks (red), water bodies (blue), etc.

There are two major phases in creating a map: data acquisition and map production. At the data acquisition phase, data collection activities are performed to obtain the main input data for map production such as aerial images, point clouds, satellite images, etc. At the next phase, map production, acquired data is processed into a map. The map production phase involves several data processing activities such as data alignment/registration, classification, delineation/vectorization, editing, cartography and database processing.



(**a**) (**b**)

Figure 2.1 Digital representation of the Indonesian base map. (**a**) 1:10.000 digital base map of Bandung city, Indonesia, in vector format; (**b**) a digital base map is managed as geospatial information database containing several main layers.

To create a good map, certain criteria and specifications need to be fulfilled. As an example, for Indonesia, the required object completeness is 85% for buildings while 90% accuracy for road and water bodies (Regulation of Head of Geospatial Information Agency, 2014) is required. The geometric accuracy is set at 30% of the map scale. For example, to make a 1:5.000 map, the required geometric accuracy is 30% x 5000 = 1.5 meter. This means, any object on the map is allowed to deviate from its real location by 1.5 meter maximum. Further details on the 1:5000 base map specifications for Indonesia are presented in Table 2.1.

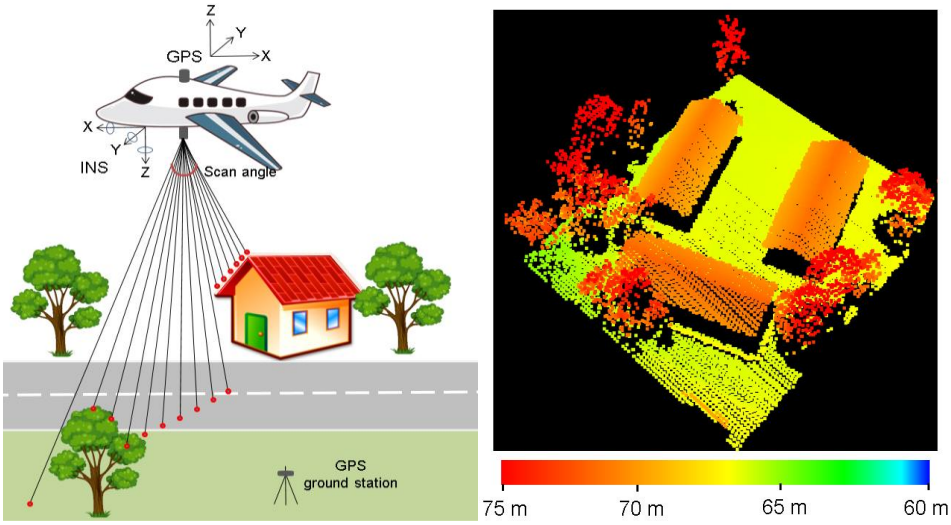
Table 2.1 Specifications of the Indonesian 1:5.000 base map.

Criteria	Buildings	Roads
Coordinate and Reference System	UTM - WGS84	UTM - WGS84
Elevation Reference (Geoid)	Indonesian SRGI (Indonesian Geospatial Reference System)	Indonesian SRGI (Indonesian Geospatial Reference System)
Geometric accuracy	1.5 meter	1.5 meter
Minimum objects area to be delineated as a polygon	Building size ≥ 2.5 x 2.5 meters	Road width ≥ 2.5 meters
Maximum object width to be delineated as a line	n. a	Yes, if road width < 2.5 meters
Sharing/merging boundary	Yes, if the distance between buildings ≤ 1 meter	No
Completeness	85%	90%

2.1.2 ALS point cloud characteristics

For more than two decades, Airborne Laser Scanning (ALS) has been used for fast collection of data over large areas in a timely manner (Kabolizade et al., 2010). An ALS system is a sensor platform, which uses laser-based measurements of the distance between the aircraft carrying the platform and the ground (Vosselman and Maas, 2010; Höfle and Rutzinger, 2011). A typical ALS system mounted on an aircraft contains several instruments: (i) the laser scanner to emit the pulses to the target on the ground and receive the backscattered pulse; (ii) an Inertial Navigation System (INS) to record the aircraft orientation; (iii) a high precision Global Positioning System (GPS) to record the position of the aircraft; and (iv) a computer interface to control, record, and manage communication among devices and data storage.

Figure 2.2 illustrates the ALS point cloud acquisition technique over the earth surface. The laser pulses reflect off objects on the surface including buildings, trees, and road. The laser instrument records the travel time required for each pulse to hit the object and travel back to the instrument. Using the location and orientation of the laser scanner (from the GPS and INS), the scan angle and the range distance to the object is used to compute the 3D (X, Y, Z) coordinates of each pulse return. This produces a collection of return locations, which is known as a point cloud. A set of 3D points acquired by ALS equipment is called as a point cloud. Along with the 3D position, additional information (intensity, return number, number of returns, full pulse recording) can be collected and added to each point of the resultant dataset (Tomljenovic, 2016).



(**a**) (**b**)

Figure 2.2 ALS point cloud acquisition. (**a**) Principle of ALS point cloud acquisition. The distance to the target on the ground is precisely determined by the travel time of the emitted pulse to the surface and back to the sensor device. GPS and INS are used to track the location and orientation of the aircrafts. Most pulses travel at an angle, while some pulses are directly nadir. Smaller scan angles increase the point density; (**b**) 3D visualization of ALS point cloud colored by elevation.

3D point clouds can represent almost any type of physical object, landscape, or geographic region at all scales with any precision (Richter, 2018). Based on the projection direction, ALS point clouds are considered as 2.5D data as it only represent an object from one direction, which is from the top direction. ALS point clouds have several benefits such as the ability to penetrate dense vegetation, no effect of relief displacement, lighting conditions insensitivity, and multiple returns information. Moreover, ALS data allow for a highly automated processing workflow (Jarzabek-Rychard and Maas, 2017).

The use of ALS point clouds for topographic mapping is still not efficient due to bottlenecks that hinder the automatic processing of point cloud data due to the their core internal characteristics of being irregular, unstructured, and unordered (see Figure 2.3). A point cloud is **irregular** with regard to its density. Points in a point cloud are not evenly sampled across different parts of an object/scene. A point cloud is **unstructured** because it is not arranged on a regular grid. Each point is scanned independently and its distance to neighboring points is not always fixed. In contrast, pixels in images represent a 2D grid, and spacing between two adjacent pixels is always fixed. A point cloud is **unordered** because its points are stored as a somehow unordered list in a file. The order in which the points are stored does not change the scene represented, which makes it invariant to permutations.

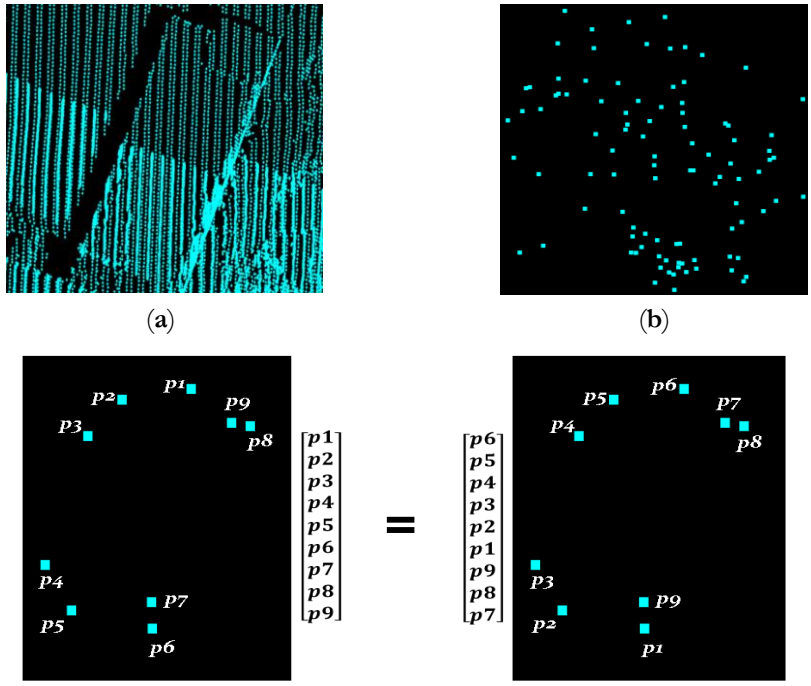


Figure 2.3 Point cloud internal characteristics. (**a**) irregularity means that points are sometimes not evenly sampled or have different point density (the upper part is sparser than the lower part); (**b**) unstructured means that each point is independent and the distance to the neighboring points is not fixed; (**c**) unordered means that points are stored as an unordered list that lacks topology and connectivity.

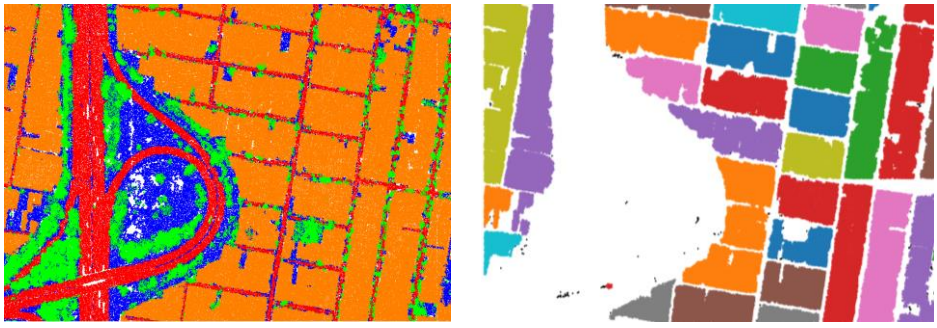
Döllner (2020) added other point cloud characteristics, including:

- **discrete representation** – discrete samples of shapes without restriction regarding topology or geometry;
- **incompleteness** – due to the discrete sampling, representations are incomplete by nature;
- **ambiguity** – the semantics (e.g. surface type, object type) of a single point generally cannot be determined without considering its neighborhood;
- **per-point attributes** – each point can be attributed by additional per-point data such as color or surface normal;
- **massiveness** – 3D point clouds may consists of millions or billions of points.

2.2 Point cloud classification and segmentation

In general, automatic obtaining object outlines can be achieved through a particular sequence of processing stages, detection and extraction. Object detection is essentially classification and segmentation of the input data which result in detected objects with coarse boundaries. Object extraction is then carried out to delineate the shape of detected object by vectors (Rottensteiner and Clode, 2009).

Different definitions exist for data classification, segmentation, and clustering since it is applied in diverse studies and applications such as computer science, medical engineering, archeology, biology, image and signal processing, and remote sensing. To avoid confusion, in this research, we refer to point cloud classification or semantic segmentation as the process to label each point with a class label so that all points in the dataset are categorized into the assigned classes. While segmentation is defined as a process to group nearby points having similar geometric characteristics into a segment. We use segmentation to partition a classification result into several groups according to their 2D or 3D position. In our context, classification results in points labelled as building, while segmentation partitions classified building points into individual buildings or several building blocks. Figure 2.4 illustrates the difference between classification and segmentation.



(**a**) (**b**)

Figure 2.4 Point cloud classification and segmentation results on a small subset of Surabaya city, Indonesia. (**a**) Classification results in points labelled according to their classes (blue represents bare land, green represents trees, orange represents buildings, and red represents road); (**b**) segmentation results in different building blocks, all consisting of points having the class labels buildings (different colors indicate different building blocks).

Similar to 2D image classification, 3D point cloud data is also greatly benefitting from current rapid development of machine learning. Existing classification techniques running directly on 3D point clouds are categorized into unsupervised and supervised methods, which will be discusses in paragraph 2.2.1 and 2.2.2, respectively. A literature review on the current hot classification approach, deep learning, is presented in Section 2.2.3.

2.2.1 Unsupervised approach

An unsupervised approach assigns unspecified classes to points based on a predefined parameterization without a prior information about class labels of the given dataset. Many segmentation methods for extracting topographic objects from point clouds have been developed in the last decades. In machine learning, a feature is defined as the individual property or characteristic of an object or phenomenon being observed (Bishop, 2006). The feature space is the set of all possible feature vectors, where a feature vector contains the feature values of one data point. For example, a color image, where color is parameterized by its RGB values, has a 3D RGB feature space.

A higher level of information of a LiDAR point cloud can be obtained by aggregating homogenous segments that are mainly determined by geometric constraints within neighborhoods (Sithole, 2005). The idea of this approach is to cluster similar points, merge them, and then separate them from other clusters according to certain parameters (such as planarity, roughness, shape, spectral values, etc.) to identify candidate objects for further processing. Several studies in literature categorized unsupervised methods into four categories: region-based, edge-based, graph-based and cluster-based techniques. Each method has different advantages and limitations which will be discussed in the following.

Region-based

Region-based segmentation mainly uses the assumption that neighboring points within one region have similar properties (Vosselman, 2013). If neighboring points are similar, they belong to one cluster/segment. These algorithms identify patches and seeds and then connect or group the patches based on adjacency features (such as normal, curvature, slope, etc). Nardinocci et al (2003) applied region-growing using height differences to generate planar segments. The geometry and topology of regions is expressed using graph theory, where segment nodes carry information on segment size. Rules are applied to extract segments representing ground, vegetation, buildings, and other objects. Rabbani et al. (2006) proposed segmentation of point clouds using a smoothness constraint that worked well for highly accurate point clouds. Unfortunately, this method may become problematic for less accurate point clouds. Shen et al. (2012) proposed a segment-based classification method by implementing surface growing to segment LiDAR points into different clusters and considering multiple echoes to distinguish ground from non-ground measurements. At last, rule-based classification is performed on the segmented point clouds. Vosselman (2013) developed a segmentation strategy by combining different approaches to segment large point clouds. Even though these techniques often succeed to satisfy their purposes, they have limitations to determine region borders accurately (Nguyen and Le, 2013) and are not well suitable for segmenting parts of a point cloud that cannot be described by surfaces (Vosselman et al., 2017).

Edge-based

Edge based techniques segment point clouds by detecting planimetric edges forming boundaries of regions, and then group the points inside the closed boundaries. This technique often applies rasterization of a LiDAR point cloud (Bhanu et al., 1985; Jiang et al., 1996). The performance of this type of methods depends on the quality of the edge detector. Although edge-based methods allow fast segmentation, they may have accuracy problems and often detect disconnected edges that make it difficult for a filling procedure to identify closed segments. This is due to situations that commonly occur in point cloud data: high sensitivity to noise and uneven point density (Nguyen and Lee, 2013; Castillo et al., 2012). Moreover, these techniques have limitations to work on point clouds directly.

Model-based

Model based techniques aim to fit geometric primitive shapes such as planes, cylinders, cubes, cones or spheres by using a mathematical model to segment points. Points conform to the mathematical representation of the primitive shape are labelled as one segment. Two most prominent algorithms for model fitting are RANSAC (Fischler and Bolles; 1981) and the Hough transform (Duda and Hart, 1972). Both algorithms have been widely implemented for 3D point cloud segmentation. Rabbani and van den Heuvel (2005) presented an efficient Hough based algorithm for automatic detection of cylinders in point clouds. Several algorithms used RANSAC for plane segmentation of building roofs (Chen et al., 2013), building facades (Bauer et al., 2005), and indoor scenes (Li et al., 2011). Oehler et al., (2011) used both Hough transform and RANSAC to perform coarse to fine plane segmentation from indoor and outdoor 3D point clouds.

Tarsha-Kurdi et al. (2007) compared RANSAC and 3D Hough transform to detect planar roofs from airborne LiDAR point cloud data. They conclude that RANSAC is more efficient both w.r.t sensitivity to point cloud characteristics and processing time. Model-based algorithms use mathematical principles that are fast and robust to outliers. Their main limitation is inaccuracy when dealing with complex shapes and different point cloud sources (Nguyen and Lee, 2013; Poux et al., 2016).

Cluster-based

This technique groups data into clusters based on available or computed attributes. Afterwards, the points in each cluster are labelled by segment Id. This segmentation approach offers flexibility in accommodating spatial relations and attributes to incorporate different cues into the segmentation process. However, the limitation of these approaches is they are highly dependent on the quality of derived attributes. The attributes of point cloud data should be computed precisely to produce the best separation among different classes. K-means, mean shift, and fuzzy clustering are

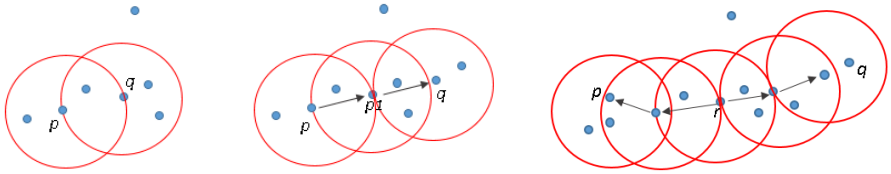
among well-known clustering-based algorithms. In machine learning, this group of segmentation methods is called spatial clustering (Ilanthiraiyan and Krishnan, 2014).

Filin and Pfeiffer (2006) introduced an ALS point cloud segmentation method based on cluster analysis in feature space. They introduced the slope-adaptive neighbourhood, which uses distance as a key feature. Biosca and Lerma (2008) presented an unsupervised robust clustering approach based on fuzzy methods to extract homogeneous segments from an unorganized point cloud. Sampath and Shan (2009) proposed a framework for the segmentation and reconstruction of buildings from ALS point clouds. The surface normals of all planar points were clustered by extended fuzzy k-means clustering. The problem related to the number of clusters to extract was solved by using a potential-based approach which considers both geometry and topology for determining cluster similarity. The method only focuses on tackling planar roofs of buildings.

Compared to other non-parametric (partitioning or hierarchical) segmentation methods, some studies indicate that density-based algorithms are best suited for extracting features from point clouds data (Ghosh and Lohani, 2013; Tran et al., 2013). One well-known density based method, DBSCAN (Density-based Spatial Clustering of Applications with Noise) proposed by Ester et al. (1996), has the ability to find clusters of arbitrary shape as well as to detect outliers of large spatial data without pre-knowledge on the number of clusters. This technique employs two major parameter inputs: radius distance (eps) and a given minimum number of points ($minPts$).

DBSCAN uses the following definitions and conditions (see Figure 2.5):

- **Definition 1:** Neighborhood $N_{eps}(p) = \{q \in D | dist_{(p,q)} \leq eps\}$.
  Point p is a core point if at least has $minPts$ points within distance eps (eps is the maximum radius from p) of it (including p). Those neighborhood points are said to be directly density-reachable from p.
- **Definition 2:** directly density-reachable if: $p \in N_{eps}(q)$ and $|N_{eps}| \geq minPts$. Point q is directly density-reachable from p if point q is within eps distance from p and p must be a core point.
- **Definition 3:** density-reachable.
  Point q is density-reachable from p if there is a chain of points $p_1, \ldots, p_n$ where $p_1 = p$ and $p_n = q$, such that each p_{i+1} is directly reachable from p_i (all the points on the chain must be core points, except q).
- **Definition 4:** density-connected.
  Point q is density-connected to point p if there is a point r where both p and q points are density-reachable from point r.
- **Definition 5:** outliers.
  Point not reachable from any other point is noise or outlier.



(**a**) (**b**) (**c**)

Figure 2.5 Definitions and conditions of DBSCAN. Given a radius distance of point q from point p, $eps = 1$ meter and minimum number of points, $minPts = 4$. (**a**) Directly density-reachable; (**b**) Density-reachable; (**c**) Density-connected.

To find a cluster, DBSCAN starts with an arbitrary seed point p and retrieves all neighbouring points (density-reachable) from p that are located within a given radius distance (eps) and contains a given minimum number of points ($minPts$). Outliers are defined once $minPts$ cannot be achieved within the given eps. The cluster will grow as long as nearby points within the eps distance from seed p fulfill the $minPts$ threshold. In case $minPts$ within distance eps is not fulfilled, a point or group of points is considered as outliers. During the cluster growing, outliers may change into a member of the cluster once they are in the eps distance from the respective seed. To grow the next cluster, the next seed is chosen that does not belong to any cluster. The clustering stops once all points are assigned. In this research, we implement DBSCAN clustering from 3D point data.

Unsupervised classification can perform well even when no prior knowledge is available and is less labor intensive, as labeling the training samples is not required. However, these methods have main drawbacks: they neglect the possible correlation between different features and rely on specific mathematical principles that may not be universally valid for different dataset.

2.2.2 Supervised approach

Supervised approaches require labelled samples to learn the classification boundaries automatically from training data (Bishop, 2006). Supervised methods offer more flexibility than unsupervised classifiers that may have difficulty to tune different set of discriminant rules and thresholds to classify the points of different cases. Supervised classification has been successfully applied, using methods such as decision trees, Random Forest, SVM (Support Vector Machine), Bayesian networks, etc. However, such traditional machine learning approaches have one major limitation that is the requirement for designing and extracting explicit discriminative features, which may not always be successful for complex scenes. All of these supervised algorithms rely on discrimination functions in using representative attributes acquired from training data. Weinmann et al. (2015) describes a general procedure of point cloud classification consisting of four main stages: neighborhood selection, feature extraction, feature selection, and classification (Figure 2.6).

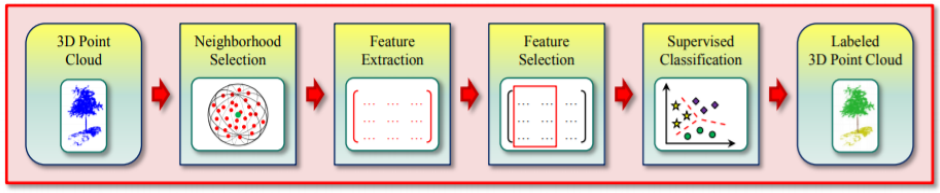


Figure 2.6 General procedure of 3D point cloud classification consisting of four stages: neighborhood selection, feature extraction, feature selection, and supervised classification. Image source: Weinmann, 2015.

SVM is based on statistical learning theory and has the aim of determining the location of decision boundaries that produce the optimal separation of classes in feature space (Cortes and Vapnik, 1995). Breiman (2001) proposes Random Forest (RF) based on the idea to combine many weak classifiers to obtain a strong classifier. Random Forest consists of several decision trees. Each tree in the Random Forest is created by drawing a subset of training data through the so-called bagging method. To train the trees, the bagging procedure will randomly select e.g. two thirds of the samples from the training data. The remaining samples are used in an internal cross validation for estimating the random forest performance. Several machine learning classifiers use statistical contextual models to reduce noise e.g. Conditional Random Fields (CRF) and Markov Random Fields (MRF). CRF takes into account the topological or geometric relationships between different objects that exist in e.g. data representing an urban environments and was successfully applied to obtain three main land cover classes: vegetation, building and ground (Niemeyer et al., 2012).

Studies on point cloud classification using supervised approaches mainly use hand-crafted geometric features as information cue (Weinmann et al., 2015; Landrieu et al., 2017). Features used for classification need to be defined and selected in advance, which can be difficult as this requires domain expertise.

2.2.3 Deep learning

Deep learning has been recognized as an advanced technology for big data analysis with a large number of successful cases in image processing, speech recognition, object detection, etc (Zhang, et al., 2018; Zhou et al., 2019). Its ability to provide a reliable way to solve difficult task and computationally expensive problems makes deep learning popular. For scene detection and classification, handcrafted feature design and selection, which is a difficult task in regular machine learning, are no longer required when using deep learning (Xie et al., 2020).

Deep learning is a subset of machine learning in artificial intelligence that uses multiple artificial neural networks or hidden layers to progressively learn high-level abstractions from training data using hierarchical architectures (Guo et al., 2016). One of the most well-known deep learning architectures for object extraction and

classification are Convolution Neural Networks (CNN) which are mostly applied for image analysis (Figure 2.7). One component equals at least one layer in a CNN. Three basic components of CNN are summarized as follow:

- **Convolutional layer** learns and detects important features of the input (such as edges, lines, color gradients in the image). It has a set of filters whose parameters including weights need to be trained and adjusted during the training. A typical convolution uses 3x3 or 5x5 kernel size or receptive field.
- **Pooling layer** to down-sample the information of the outputs from convolutional layer to make faster computation and more robust to variations in the position of features in the input. This is usually incorporated between two successive convolutional layers. By stacking a convolutional and a pooling layer, higher level of feature representations are gradually extracted. Max pooling, which keeps the maximum value of a matrix window, is a typical pooling layer.
- **Fully connected layer** represents the feature vector for the input and makes prediction. This layer takes all neurons in the previous layer and connect them to every single neuron of the current layer to generate global semantic information. Fully connected layer is usually located in the few last layers of CNN.

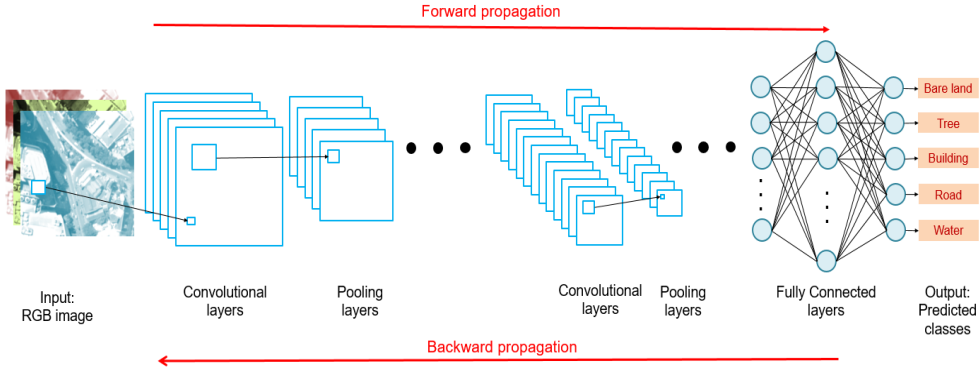


Figure 2.7 General CNN architecture consisting of 3 main components: convolutional layers, pooling layers, and fully connected layers. Any layer residing between input and output is known as hidden layer. The process generating the output (forward propagation) flows in one direction from the input to the output layer sequentially. To minimize the error and optimize the network's parameters, the network computes loss function derivatives at the output layer and propagates them back towards the input layer (back propagation).

In general, a training process in deep learning consists of six main steps:

1. **Initialization** to assign initial weights applied to all neurons.
2. **Forward propagation** is when the network assigns an input and passes through the hidden layers to produce an output, information flows forward through the network.

3. **Loss function** to represent how far the network's prediction is from the true labels by calculating the differences between prediction and the true labels, given the current weights.
4. **Back propagation** to iteratively minimize the error (differences between prediction and training label) by gradient descent and updating the weights.
5. **Weight update** to adjust the weight value according to the back propagation results.
6. **Iteration until convergence** is required as the weights may be adjusted by small values at a time. Therefore, several iterations are required by the network to learn.

As point clouds are unstructured and unordered, it is almost impossible to use standard convolutional operators like in CNNs. Regular convolutional neural networks use convolution operation which are performed on data that is ordered, regular, and on a structured grid (e.g. images). Directly convolving kernels against features associated with the points will result in neglection of shape information and variance to point ordering (Zhang et al., 2020). Early approaches overcome these challenges by converting the point cloud into a structured grid format (Bello et al., 2020). Therefore, we categorized the deep learning works focusing on point cloud classification into three approaches: multiview-based, voxel-based, and point-based. Both multiview-based (MVCNN, SnapNet, and SnapNet-R) and voxel-based (VoxNet, PointGrid, SEGCloud) use an indirect approach that requires conversion of the point cloud into a regular structure. Point-based approaches work directly on point clouds (PointNet, PointNet++, SPG, etc.).

PointNet (Qi et al., 2017) is a pioneering deep learning architecture that works directly on unstructured 3D point clouds. PointNet not only accelerated the speed of computation but also improved the performance of semantic segmentation (Zhang et al., 2019). To classify a point cloud, PointNet uses a network composed of a succession of fully-connected layers, instead of convolution operators. To use the network, each input point $\boldsymbol{P_i} \in \mathbb{R}^D$ $(i = 1, \dots, N)$ is represented by its D-dimensional feature vector consisting of 3D coordinates (x, y, z) with additional features such as RGB color, normals, etc.

As shown in Figure 2.8, PointNet processes unorganized point cloud data in three stages. First, each unordered point is fed into a multi-layer perceptron (MLP) and mapped onto a C-dimension feature vector to extract independent features. Second, independent features are processed in a max pooling layer to aggregate N point feature vectors invariant to permutations. Third, the C-dimension global features are mapped into an F-dimension output vector using another MLP.

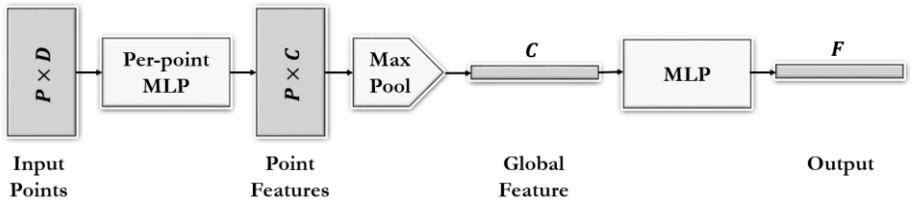


Figure 2.8 Basic processing stages and components of PointNet. The network takes directly N points as its input. Each D-dimensional of input point is mapped into a C-dimension feature vector through MLP. Per-point (local) features are aggregated into a global feature vector by max pooling layers. The global feature is then mapped onto F-dimensional output vector consisting of classification scores.

Crucial problems PointNet tries to solve are:

- Ordering problem due to lack of structure in point clouds. PointNet uses a symmetric function to map input points to higher-dimensional space. A symmetric approximate function takes n vectors as input and outputs a new vector that is invariant to the input order e.g. $sum(a,b) = sum(b,a)$ or $average(a,b) = average(b,a)$ or $max(a,b) = max(b,a)$
- Transformation invariance due to possible rotation and translation. PointNet uses a spatial transformation network (T-nets) to estimate an affine transformation matrix to spatially align input point clouds and features which makes the network invariant to geometric transformation.

Although PointNet has a beneficial effect on point cloud classification and segmentation, by design, the network does not consider the local neighborhood structure and spatial correlation between points (Qi et al., 2017). Therefore, Qi et al. (2017) proposed PointNet++ to take into account the distance between points by using a hierarchical neural network. However, this model still not directly determines geometric features between points. A Dynamic Graph CNN (DGCNN) is then proposed by Wang et al. (2018) to overcome this problem. The DGCNN architecture incorporates a graph-based CNN approach to capture the local geometry of points by an edge convolution operation on a k-nearest neighbor graph which is iteratively updated. In addition, SuperPoint Graph (SPG) as proposed by Landrieu and Simonovsky (2018) offers a rich representation of contextual relationships between object parts rather than points. The model adaptively partitions the point cloud into geometrically homogenous simple shapes to build a Superpoint Graph which is then fed into a graph neural network for producing semantic labels. However, the method does not explicitly model the local spatial relationships among points, thus acquiring less shape awareness (Liu et al., 2019). PointCNN (Li et al., 2018) uses a so-called *X*-conv operator to weigh and permute input points and features before they are processed by a typical convolution on the transformed features. Another deep learning architecture that processes points directly is KPConv (Kernel Point Convolution),

introduced by Thomas et al. (2019). Instead of using a multi-layer perceptron, KPConv applies 3D convolution kernels to define the radius neighborhoods as input and processes them with weights spatially located by a small set of kernel points.

Deep learning architectures that directly process the point cloud are considered as the most effective and efficient as they do not need projection on multiple 2D images or 3D volumes (Ge et al., 2018). Current work on deep learning indicate the need of further improvement and investigation (Szegedy et al., 2016; Gu et al., 2018; Zhang et al., 2019). For example, the amount of sufficient good quality training samples, computational efficiency, training procedure acceleration. Furthermore, one major barrier for applying deep learning is that it requires skill and experience to select suitable hyperparameters such as the number of layers, kernel size, learning rate, etc.

2.3 Line extraction

Efforts to automate the extraction of object outlines from remote sensing data form a major research direction in the photogrammetric and computer vision communities (Agouris et al., 2004). Automatic object extraction defined in this research aims at obtaining topographic objects and their geometric reconstruction by vectors that can be used in GIS or Computer-Aided Design (CAD) systems.

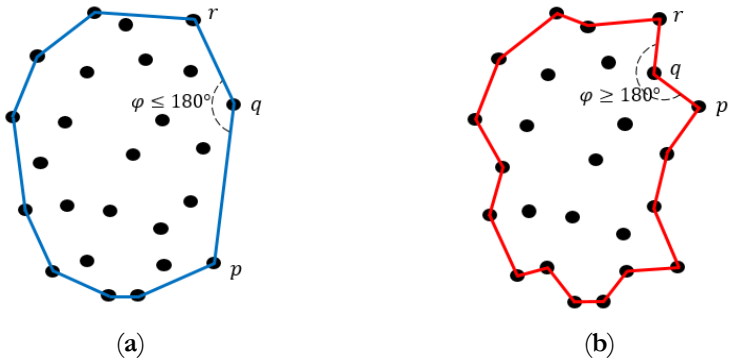
Several methods use an indirect approach to obtain outlines of topographic objects by deriving first a Digital Surface Model (DSM) from an ALS point cloud (Weidner and Foerstner, 1995; Brenner, 2000; Hollaus et al., 2010). Some other methods use point clouds directly to obtain an outline by detecting primitive shape representations such as lines and planes. Using ALS point clouds to extract outlines of man-made urban objects may suffer from point spacing irregularity effects that creates noisy boundaries. Thus, to obtain straight and smooth object boundaries, outline extraction has at least two main tasks: edge detection and line regularization. In this research, edge detection is defined as the process of selecting edge points of an object forming an outline. Line regularization is the process to smooth jaggy outlines caused by unstructuredness of edge points and additional data problems. In the following, we further discuss relevant methodologies for edge detection and line regularization.

2.3.1 Edge-aware shape analysis

Consider a set of points in 3D or 2D Euclidean space representing a building. Finding the most outer points can be categorized as edge point detection. Thus, a building outline is formed by connecting all line segments, in which each line segment connects two consecutive edge points. The problem to find simple and planar polygons outlining a finite point set in 3D or 2D Euclidean space arises in many practical applications. As an example, the task of creating a 3D building model comprises the detection of their outlines or footprints (Vosselman, 1999). Outline detection is a representation of a shape which is a classical problem in computational geometry.

Convex and concave hulls

The convex and concave hull are both representations of the area occupied by a set of finite points. In geometry, the convex hull of a set of point in a plane is the smallest polygon enclosing the points so that there are no concavities or inward corners in the polygon boundaries. The concave hull is loosely defined as a polygon tightly enclosing all points while allowing also inward corners, see also Figure 2.9.



(**a**) (**b**)

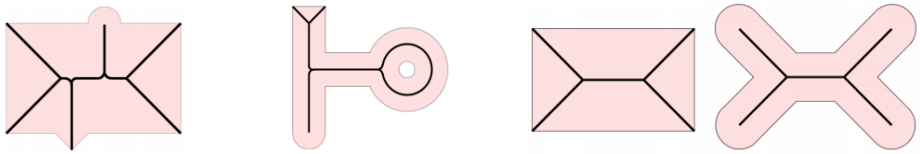
Figure 2.9. Convex and concave hulls outlining the shape of a set of points. (**a**) a convex hull always has interior angle φ between two neighboring edges equal to or less than 180 degrees; (**b**) a concave hull in general occupies smaller area than the convex hull as interior angles φ between two consecutive edges $\overline{pq}$ and $\overline{qr}$, are allowed to be greater than 180 degrees.

Representing the shape of finite point sets by simple polygons becomes a challenge if the resulting outlines needs to be non-convex and straight with few edges and angles (Pohl and Feldmann, 2016). Non-convex shapes are those shapes in which at least two sides are pushed inwards (Figure 2.9.b). Alpha-shape (Edelsbrunner, 1983) is introduced as a solution to the problem, in which a predefined parameter alpha is used as a size-criterion to determine the level of detail and capture the intuitive notion of the object shape. To address the same problem, Galton and Duckhum (2006) suggested nine evaluation criteria for concave hull algorithms, which was followed by several works such as *k*-nn concave hull (Moreira and Santos, 2007), *X*-shape (Duckham et al., 2008), alpha-concave hull (Asaeedi et al., 2017), etc. Having the same goal, many studies consider alpha-shape as a concave hull approach. Several methods identified edge points and apply boundary line detection to obtain straight and smooth line using either concave hull or alpha-shape (Dorninger and Pfeifer, 2008; Lach and Kerekes, 2008; Wei, 2008; Albers et al., 2016).

Skeletonization

Another shape analysis approach, skeletonization (Blum, 1967) derives a compact and thin line segment representation of an area or surface. The resulting skeleton, also known as medial axis, has the ability to maintain the geometrical and topological

properties of a shape such as its connectivity, length, direction, and width which makes it to some extent possible to recover the original shape (Figure 2.10). The main drawback of skeletonization is the sensitivity to the presence of noise or small undulations on the object edge as are common for point clouds. Skeletonization methods available in literature can be categorized into four main methods: geometry-based (Voronoi, Delaunay triangulation, shrinking ball), distance function-based (Euclidean distance map); morphological thinning-based (straight skeleton, fast parallel thinning), and obtained by a general field function (linear transform, Newtonian model).



(**a**) (**b**) (**c**)

Figure 2.10 Skeleton characteristics. (**a**) skeleton (black line) is sensitive to noise or small changes on the object edge; (**b**) skeleton representing topological structure of the original object shape; (**c**) skeletons of similar shape and size can be extracted from different object shapes. Different objects may result in exactly the same skeleton.

Works on topographical object extraction like outline and centerline extraction can greatly benefit from skeletonization characteristics. However, object extraction can normally not be solved satisfactorily by applying a basic skeletonization method alone (Haunert and Sester, 2008). This is because different methods have different advantages and limitations. For example, a fast parallel thinning algorithm can reconstruct the original object shape but it requires a binary image as an input (see Figure 2.11) and it is difficult to estimate the distance from a skeleton (centerline) to its corresponding object edges. On the other hand, a shrinking ball algorithm can directly process points, but more steps are required to deliver a linear skeleton as the algorithm results in skeletal points instead of a line. Furthermore, skeletonization usually results in jaggy or wavy skeleton branches. Thus, it is likely that modification or additional processing is required to obtain smooth and straight object outlines and centerlines.

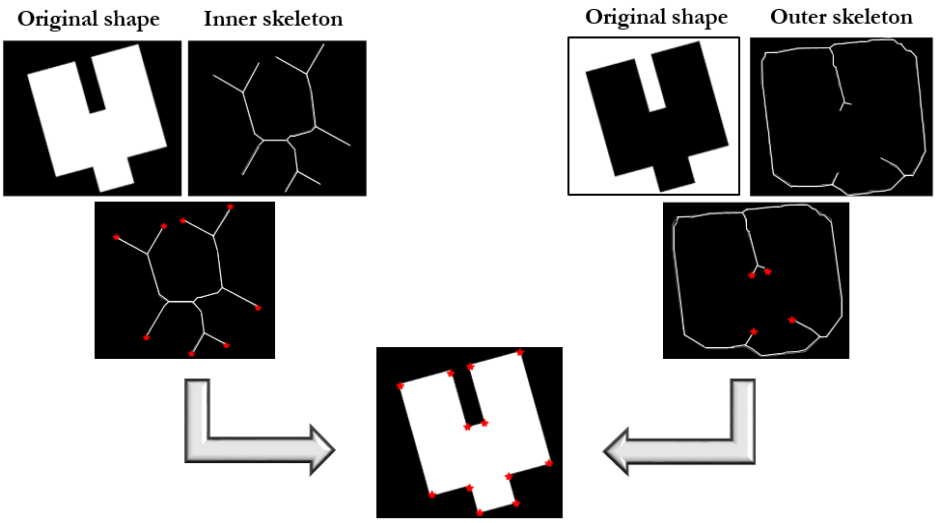


Figure 2.11 Reconstruction of original shape from the skeleton using a fast parallel thinning algorithm. The original shape can be reconstructed by estimating the position of all end points of the skeleton. A binary image is required as input for this algorithm.

2.3.2 Line regularization

Two well-known algorithms are used as basis for line fitting: RANSAC (Fischler and Bolles, 1981) and Hough transform (Hough, 1959; Duda and Hart, 1971; Ballard, 1981). The RANdom SAmple Consensus (RANSAC) algorithm is a simple but powerful algorithm which has ability to fit a mathematical primitive to a set of data affected by noise. RANSAC generates candidate lines based on a minimum number of two observations (data points) required to estimate the model parameters. The basic steps of the RANSAC algorithm are summarized in algorithm 1 as follows:

Algorithm 1. RANSAC

1. Randomly select the minimum number of points required to determine the model parameters;
2. Estimate the parameters of the model;
3. Determine the number of points from the dataset which fit within a predefined tolerance ε. The fitting points are the inliers;
4. If the fraction of the number of inliers over the total number of points; exceeds a predefined threshold τ, re-estimate the model parameter using all inliers and terminate;
5. Otherwise, repeat steps 1 through 4 (maximum of N iterations)

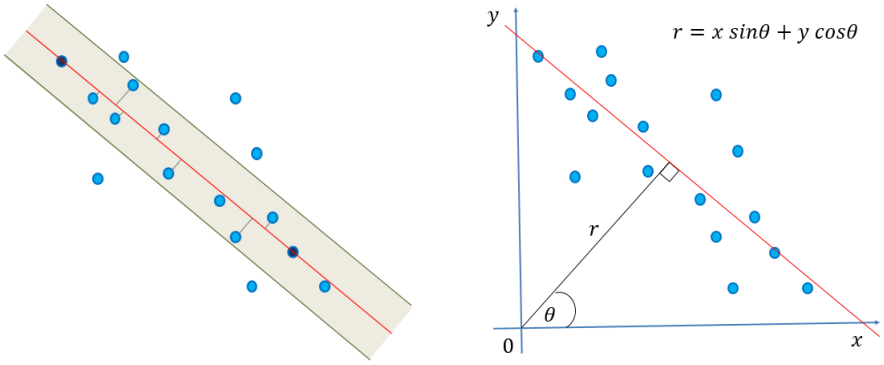
The key idea of Hough transform is to find a line model based on a voting scheme in parametric domain. A line in object space (x, y) is converted into parametric space,

e.g. angle and distance from the origin (r, θ), or slope and intercept (m, b). The number of points from all possible parametric combinations are stored in an accumulator matrix. Based on the local maxima shown in the accumulator matrix, best fitting lines can be detected. Algorithm 2 describes the general step of Hough transform.

Algorithm 2. Hough transform

1. Parameterize each point from object space coordinates (x, y) into parametric coordinates (r, θ);
2. Store the number of points voting for the same parametric coordinates into a binned accumulator matrix;
3. Find the parametric coordinates of all local maxima
4. Back-transform parametric local maxima to the real object space coordinates

RANSAC and Hough transform are recognized as powerful tools to detect straight line which gives good results even in the presence of noise (Jacobs et al., 2013). In extracting straight and smooth lines, both algorithms can be used to regularize a noisy boundary or fuzzy line. Several improved algorithms for shape recognition use either RANSAC or Hough transform as basis. The Hough transform is more accurate and stable while RanSAC is more computational effective (Jacobs et al., 2013; Kröger et al., 2016). For detecting planar roof of ALS point cloud, Tasha-Kurdi et al., (2007) compares 3-dimensional RANSAC and Hough algorithms. It turns out that RANSAC provides more efficient and better results than Hough transform. Figure 2.12 illustrates line fitting methods using RANSAC and Hough transform.



(**a**) (**b**)

Figure 2.12 Line fitting methods. (**a**) RanSAC fits a line to noisy points based on the distance of each point to a line model obtained from two points (black points). Any point located within a certain distance from a line is categorized as inlier. The best line is the one with most inliers; (**b**) Hough transform performs line parameterization by converting the line parameter from object space (x, y) into parametric space (r, θ). The best line obtaines most votes in parameter space.

Other approaches of line smoothing or regularization that have also been implemented in GIS (Geographic Information System) operations include Douglas-Peucker (1973) and Visvalingam-Whyatt (1993) algorithms. To smooth a jaggy line, Douglas-Peucker chooses the points to be kept while Visvalingam-Whyatt algorithm removes points between two end-points iteratively. However, regularizing outlines using these two line smoothing algorithms (e.g. Douglas-Peucker) is not always effective, especially not for buildings, due to unintentional removal of certain corners (Pohl and Feldman, 2016). Moreover, both algorithms have limitation considering the presence of noise that is preserved in the results.

2.4 Summary

In this chapter, we reviewed state of the art of methods related to object outline extraction from ALS point cloud. We presented a general information on point cloud processing and methodological framework associating several required methods for conducting automatic map production from ALS point cloud which include at least two big tasks: classification and line extraction. Although several approaches have been widely used, some potential improvements are required to increase the performance and results quality. Considering the point cloud characteristics, extracting outlines of topographic objects directly from a point cloud is a challenging task but this point-wise approach also provides richer features than images that are helpful for automation. The following chapters discuss extraction methods of building and road outlines from ALS point cloud data proposed by this research.

3

Automatic vectorization of urban map objects from a colored ALS point clouds using DGCNN deep learning and skeletonization

Urban map objects (buildings and roads) are essential input for urban planning, smart city concepts and traffic management. However, extracting and updating such vector-based map manually is time consuming and labor intensive. This paper presents a methodology to extract urban map objects automatically from an Airborne point cloud combined with color information from an airborne orthophoto. First, we classify points into urban land cover classes using DGCNN (Dynamic Graph Convolutional Neural Network). Second, medial axis transformation derived skeletons are used to get initial building outlines from the building points, while thinning process is implemented to obtain road centerlines. Third, building outlines and road centerlines are refined: from the classified building points, building blocks outlines are extracted by a 2D shrinking circle method. Next, line simplification and road gap filling is conducted. Final road outlines are obtained based on the actual road width estimated by a medial axis approach. The proposed method is tested on the metropolitan area of Surabaya, Indonesia. The point-wise classification achieves a highly acceptable result with 91.8% overall accuracy when using the full combination of spectral color and LiDAR features. Based on the evaluation results, our method resulting in road polygons with 80.6% recall and 72.6% precision rate.

This chapter is organized as follows: Section 3.1 gives background on the study. Section 3.2 describes related work on classification of ALS point clouds and map vectorization. Section 3.3 provides a description of the study area and specifies the data used in the research. The methodological framework, consisting of classification and vectorization of urban objects, is presented in Section 3.4. Section 3.5 presents and discusses results. Finally, conclusions and recommendations are given in Section 3.6.

3.1 Introduction

Maps provide the integrative platform for all digital data that has a location dimension (UNGGIM, 2018). Decision-making, including urban planning, intelligent transportation systems, and disaster management depends on the availability of maps. Modern digital map production should be able to provide quality maps efficiently (faster and cheaper). Urban maps are crucial in situational decision making where building and roads are regarded as primary map ingredients (Nyerges and Jankowski, 2009; Indrajit et al. 2019). To efficiently obtain and maintain up-to-date maps, automatic extraction of man-made structures such as buildings and roads became an important research topic for geoinformatics (Hinz et al., 2001; Kabolizade et al., 2010). Automatic extraction remains challenging for several reasons: first, buildings and roads have different characteristics, in terms of shape, size, and color. Second, input data used for extraction is affected by noise and occlusions caused by objects (trees, cars, towers, etc.) in the scene.

Airborne Laser Scanning (ALS) point clouds and aerial photos are the main very high resolution and accurate input data available to map cities. Both data have different characteristics and complementary advantages. Aerial photos are rich with spectral information, giving a representation similar to what humans see in the real world, while an airborne LiDAR point cloud provides an accurate 3-dimensional representation of urban objects (Shan and Toth, 2009). In case of edge detection, LiDAR point clouds may not always represent object boundaries well. On the other hand, aerial photo pixels clearly represent continuous and firm object boundaries. Combination of both data is expected to increase the degree of automatic processing as well as the quality of the result (Rottensteiner and Clode, 2009).

In the last few years, there is significant development in research on deep learning to solve traditional machine learning problems (Griffiths and Boehm, 2019). Traditional machine learning use features defined by humans which may not be sufficient for certain complex cases. Due to its ability to process large datasets and to learn representations from complex data acquired in real environments (Carrio et al. 2017), deep learning is a promising tool to improve the performance and quality of automatic scene classification. Further work is required, however, to obtain urban map objects that are represented by their boundaries.

Skeletonization provides an effective and compact representation of objects by reducing its dimensionality to a skeleton or centerline or medial axis while preserving the objects topology and geometric properties (Saha et al. 2016). Skeletonization also allows to completely recover the shape of the original object. Considering the benefits of skeletonization, their application for urban map objects boundary extraction is promising. Thus, we designed a pipeline to obtain accurate urban object boundaries automatically from a colored airborne point cloud using a novel skeletonization approach.

The key contributions of this study are as follows:

(1) To investigate an effective feature combination for the classification of huge outdoor ALS point clouds colored by aerial photos for complex urban landscapes having class imbalance due to the presence of many buildings.

(2) To design and implement an integrated skeleton-based approach to vectorize a smooth road centerlines and boundaries from the classified road points.

(3) To provide an effective approach to extract complete urban map objects including building and trees automatically from the classified points.

3.2 Related work

Previous studies on map elements extraction network extraction use various input data and methods. In general, urban map element extraction contains two major tasks: ALS point cloud classification and vectorization (outline extraction). Point cloud classification defined in this study refers to a task assigning predefined class or semantic label (e.g. ground, building, or tree) to each individual 3D point of a given point cloud, which is also known as semantic segmentation or class labeling. Vectorization is defined as extracting the boundary shape of an object in vector format (polygon, line, or point).

3.2.1 Classification of ALS point clouds

Classification of urban remote sensing data remains challenging as it usually involves large datasets while urban scenes are notoriously complex. Furthermore, point cloud data have particular characteristics that make classification even more challenging: they are unordered and unstructured, often with large variations in point density and occlusions (Balado et al. 2019).

Nowadays, there is interest in using deep learning approaches to solve classification and segmentation tasks (Ghosh et al., 2019). Deep learning for 3D point cloud data has been developed. Some methods apply dimensionality reduction by converting 3D data into multi-view images (MVCNN, SnapNet, etc.), other methods organize point clouds into voxels (SegCloud, OctNet, etc.), or directly use 3D points as input (PointNet, PointNet++, etc.). Inspired by PointNet (Qi et al., 2017), several point-wise deep learning methods classify 3D point cloud data using a network composed of a succession of fully-connected layers. However, PointNet limitations on capturing the spatial correlation between points, triggered several alternative point-wise deep learning network architectures such as SuperPoint Graph (Landrieu and Simonovsky, 2018), PointCNN (Li et al., 2018), and DGCNN (Wang et al., 2018).

Deep learning has been used and adopted in various applications where the global interpretability still lacks a well-established definition in literature (Ish-Horowicz et al., 2019). A central question regarding the interpretability: how can humans understand the reasoning behind the model predictions? A common interpretation approach is to

identify the importance of each input feature to optimize the prediction results (Singla et al., 2019).

In the context of neural networks, a model may have difficulties in learning meaningful features (Horwath et al., 2020). Most experiments on point-wise deep learning use benchmark indoor point clouds (e.g. Stanford S3DIS dataset) with input features consisting of 3D coordinates (x, y, z), color information or RGB (Red, Green, Blue), and normalized coordinates (n_x, n_y, n_z). Implementations for airborne point clouds with different input features are available in the literature. Soilan et al. (2019) implemented multi-classification (ground, vegetation, building) using PointNet applied on an ALS point cloud. They replaced RGB features as used in the original PointNet publication by LiDAR-derived features: Intensity, return number, and height of the point with respect to the lowest point in a 3x3m neighborhood. Even though the classification accuracy achieved 87.77%, there is high confusion between vegetation and buildings. Wicaksono et al. (2019) used a DGCNN to classify an ALS point cloud into building and non-building classes by two different feature combinations: with and without color features. Based on their results, they stated that color features do not improve the classification and suggested further research to address the incorporation of color information. In contrast, using a so-called sparse manifold CNN, Schmohl and Soergel (2019) obtained a 0.8% higher overall accuracy when using additional color information on their test set segmentation. Xiu et al. (2019) classify ALS point cloud data concatenated with color (RGB) features from an orthophoto using a PointNet architecture. By applying RGB features, overall accuracy increased by 2%, from 86% to 88%. Additionally, Poliyapram et al. (2019) propose end-to-end Point-wise LiDAR and a so-called Image Multimodal Fusion Network (PMNet) for classification of an ALS point cloud of Osaka city in combination with aerial image RGB features. Their results show that the combination of Intensity and RGB features could improve overall accuracy from 65% to 79% while the performance in identifying buildings improved by 4%.

Despite the result inconsistencies of previous work using RGB features in ALS point cloud classification, to the best our knowledge, impact on the classification of using RGB image features in high-rise building areas where relief displacement has not yet been analysed. We conclude that the beneficial effect of using RGB features in ALS point cloud classification is unclear and indecisive. A possible explanation for the inconsistency of the results are problems in the fusion of the ALS point cloud and the color information.

Effective classification with imbalanced data is still an important research area for real-world applications (Johnson and Khoshgoftaar, 2019). Most of the imbalanced class distribution relates to loss in performance (Hensman and Masko, 2015). Lin et al. (2018) introduce a focal loss function to address class imbalance in object detection in a case of extreme imbalance between foreground and background pixels.

3.2.2 Vectorization

3.2.2.1 Road vectorization

The complex appearance and topology of roads explains why road extraction remains a difficult task in remotely sensed data (Bendouda and Berrached, 2018). Clode et al. (2007) propose road centerline extraction by convolving a binary road image with a complex kernel, called Phase Coded Disk, which uses phase to encode the angle of a line. An initial road boundary is derived via supervoxels and graph cuts (Zai et al. 2017). Then, CNN-based boundary completion is used to complete road boundaries affected by gaps. To obtain a smoother and more accurate boundary, road centerlines extracted from GNSS trajectory points and satellite images are used as completion guidance for a generative adversarial network (GAN) model. Still, the method has a large error distance (the average distance between added points and their nearest corresponding point on the ground truth boundary) on curvy roads.

Boyko and Funkhouser (2011) partitioned road points into several patches using a road map and then fitted a ribbon-snake (width extended active contour) to model road boundaries. In an area where several snakes meet or touch, the method results in jagged and disrupted road areas due to the absence of inter-snake interaction. Moreover, the method does not explicitly deal with road intersections and requires parameter tuning in terms of the energy field, which limits the automation. Widyaningrum and Lindenbergh (2019) used parallel thinning skeletonization of road binary images to extract the road centerline and then applied a connectivity-based approach to extract the network topology and regularize the skeleton to obtain smooth lines. However, the road polygon (road edges) are often not accurate as the method uses the same width for all roads in the study area and not applicable for curvy roads.

Xu and Poullis (2019) implemented a post-processing refinement step which converts road pixels classified by the SegNeXt architecture into final road network vectors using iterative Hough transform. A set of study areas (Tokyo, Chicago, Boston, and Amsterdam) were used to determine how post-processing affects the accuracy and completeness of the extracted road network. Their results produce on average comparable results and in some cases better than the DeepRoadMapper (Mattyus et al., 2017) and RoadTracer (Bastani er al., 2018) method.

3.2.2.2 Building vectorization

Building outline extraction methods can be classified into two groups: indirect and direct approaches. Direct method directly work on 3D point clouds, while indirect approaches require prior conversion from point cloud into other data representation (e.g. height model). A direct method implemented by Sampath and Shan (2007) used a modified convex hull combined with hierarchical least-squares regularization to obtain building outlines. Lach and Kerekes (2008) perform boundary extraction from point cloud data using a 2D alpha-shape and consecutive regularization. Albers et al. (2018)

used a Hough transform to regularize building boundary points selected by a 2D alpha-shape. The intermediate result got repositioned based on energy minimization. Huang and Sester (2011) reconstruct building footprints by first segmenting the 3D point cloud with a 3D Hough transform and then use roof heights and ridge information as additional parameters for the statistical reconstruction of a building footprint using a method called Reversible Jump Markov Chain Monte Carlo (RJMCMC). The proposed method only consider rectangular shapes.

Using indirect method, Zhao et al. (2016) used boundary tracing of building regions detected by connected component analysis. First a point cloud is converted into a Digital Surface Model (DSM), then trees are removed by NDVI filtering, next regularization is applied based on principal directions. Jarzabek-Rychard (2012) report on straight-line extraction using Random Sample Consensus (RANSAC) in a height image derived from a LiDAR point cloud. Regularization is then performed by merging close parallel line segments and adjusting angles according to a mean direction estimated from the longest line segments.

Based on the existing work related to our study, it is concluded that:

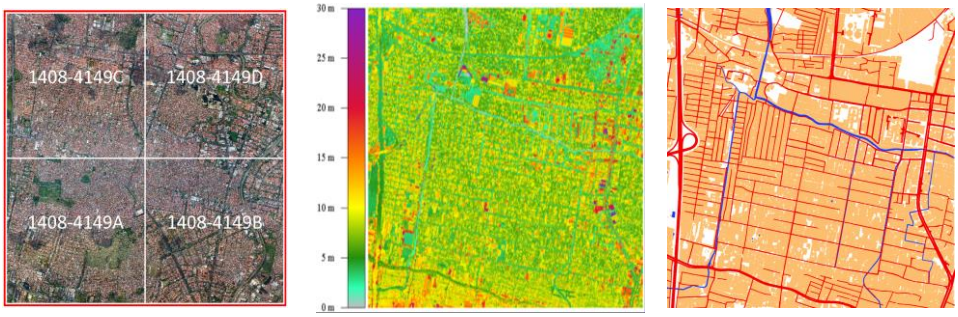
1. Finding optimal input feature combination for airborne point cloud classification using a deep learning approach incorporating RGB color image remains an open issue due to inconsistencies between different research results.
2. As class imbalance is naturally inherent in many remote sensing classification problems, providing a sufficient amount of good quality training samples without overfitting the data is still an important research topic.
3. There is still an open discussion to improve the quality of road network extraction results, in particularly when dealing with road intersections, gaps, and curved roads.

3.3 Study area and data specification

The study area is located in the second-largest Indonesian metropolitan area, Surabaya city in West Java Province. The city is characterized by dense settlement areas with various types of well-connected roads. Surabaya city is home to numerous high-rise buildings and skyscrapers. Although many parks exist, vegetation in Surabaya city is dominated by trees. The study area covers 21.5 km², and coincides with four Indonesian 1:5.000 base map sheets: 1408-4149A, 1408-4149B, 1408-4149C, and 1408-4149D as shown in Figure 3.1.a.

The ALS point cloud is captured by an Optech Orion H300 instrument and has an average density of about 30 points/m². The aerial photos captured at the same time by a tandem camera have a spatial resolution of 8 cm with less than 15 cm positional accuracy. The ALS point cloud used for our research consists of 354.197.545 points. The ALS point cloud comes divided into two classes: ground and non-ground points.

The dataset is projected in the UTM49 South coordinate system using the WGS84 geoid. Both LiDAR point cloud and aerial photos are acquired from the same platform at the same time in 2016. The reference data used to label the points and to evaluate the final results is an Indonesian 1:1.000 base map from 2017. The base map is acquired by manual 3D delineation from the same aerial photos.



(a) (b) (c)

Figure 3.1. Input data and coverage of Surabaya city, Indonesia. (a) Orthophoto of the study area covering the four indicated 1:5000 map sheets; (b) ALS point cloud covering the 1408-4149C map sheet colored by elevation; (c) The 1:1.000 base map of the 1408-4149C map sheet containing buildings (orange), roads (red), and water-bodies (blue).

3.4 Methodology

This study takes a point cloud colored by an orthophoto as an input to estimate automatically 2D urban map objects. These consist of building blocks and road networks in vector format (polygon or polyline). The object boundaries represent the footprints of buildings and road outlines that are orthogonally projected on a horizontal plane. In other words, the output of this study are smooth and accurate boundaries of building blocks and road networks to be presented on a large scale 2D map.

Our methodological workflow consists of two main tasks: classification and vectorization (Figure 3.2). The point-wise classification task is performed first to automatically label the points into several object classes. This includes training set preparation, training by DGCNN, and evaluation of the classification results. Vectorization is then performed to extract road centerlines and building block outlines to be presented on a map. In the following, proposed methods are described focusing on Steps 2 and 3: classification and vectorization.

3.4.1 Point-wise deep learning classification

The goal of classification is to obtain class labels for each point of a point cloud. We aim for four point cloud classes: bare land, trees, buildings, and roads. Due to the limited number of LiDAR points covering water in the study area, a water class is not included. Vegetation other than trees is not required for our map.

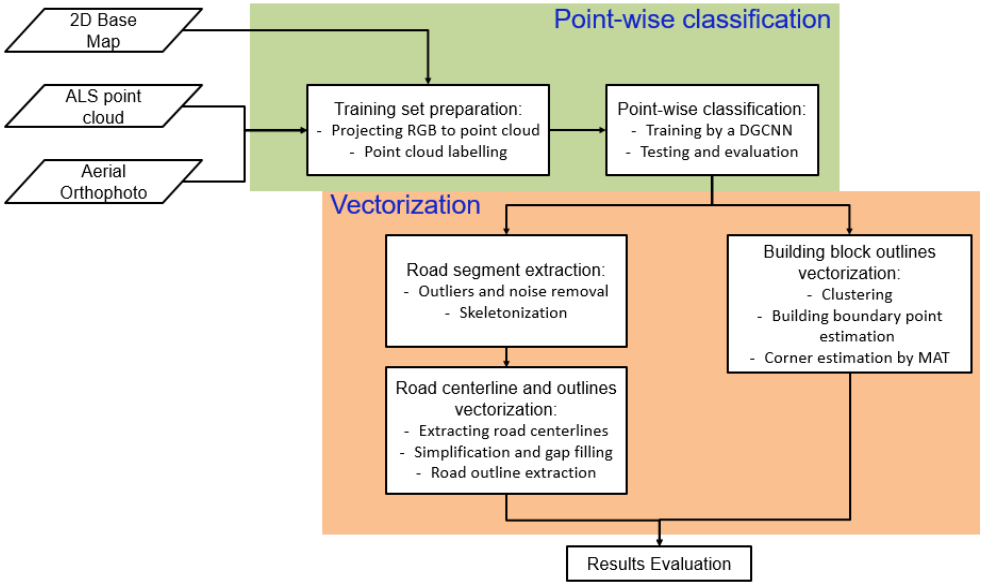


Figure 3.2. The methodological workflow to extract urban map objects (buildings and roads) automatically from ALS point cloud and super-imposed orthophoto.

This study uses a Dynamic Graph CNN (DGCNN) architecture proposed by Wang et al. (2018) for classification. DGCNN is a pointwise neural network architecture that combines PointNet and a graph CNN approach. The network architecture uses a spatial transformation module and estimates global information, like PointNet. The Graph CNN approach captures local geometric information while ensuring permutation invariance. It extracts edge features through the relationship between a central point and neighbouring points by constructing a nearest-neighbor graph that is dynamically updated.

3.4.1.1 Training set preparation

As our method requires spectral information, the first step is to project RGB (Red Green Blue) color information from an orthophoto onto the ALS point cloud data by nearest neighbor. Next, the point cloud data is downsampled to 1 meter 3D spacing for efficiency and to facilitate the capturing of global information. For the classification task, the 1408-4149C map sheet is used as test data and the remaining map sheets (1408-4149A, 1408-4149B, and 1408-4149D) are used as training samples (see Figure 3.1.a).

To label the points, 2D building and road polygons from the 1:1000 base map of Surabaya city were used. Although the point cloud data used in this study is already classified into ground and non-ground points, two challenges needed to be solved when

using a 2D base map to label the point cloud. First, the corresponding base map does not provide information on trees. Second, as we use 2D base map polygons to label the points, the labeled building and road points may include many mislabeled points in case trees exist above buildings or roads. Therefore, we perform hierarchical filtering to label tree and bare land points based on surface roughness. The method is also intended to improve the quality of the training samples by removing likely mislabeled points on buildings and road points.

The labeling criteria are as follows:

- From the non-ground points, points are labeled as building using 2D building polygons of the base map. Using the same method, ground points are labeled as road. Remaining points are labeled as bare land.
- From the points labeled as building or road, any point that has surface roughness above a threshold is re-labeled as tree. The surface roughness is estimated for each point based as the distance to the best fitting plane estimated using all neighboring points inside an area of 2m × 2m. Given the resulting roughness values for both trees and building points in selected test areas, and given the fact that tree canopies in the study area have a minimum diameter of about 3m, the roughness threshold is set empirically to 0.5m.
- A Statistical Outlier Removal (SOR) algorithm is performed to remove remaining outliers. We set the threshold for the average distance ($\bar{d}$) = 30 and multiplier of standard deviation = 2. This means, the algorithm calculates the average distance of 30 k-neighboring points and then removes any point having distance more than $\bar{d} + 2 * standard\ deviation$.
- As final step, training samples data are converted to the hdf5 (.h5) format by splitting each part of the area into blocks of size 30m × 30m with a stride of 15m. These thresholds are set empirically for our data and study area to ensure local geometry is captured efficiently by the network.

3.4.1.2 Point cloud classification by a DGCNN

DGCNN is chosen as the network to perform point cloud classification. DGCNN architecture (see Figure 3.3) incorporates a so-called EdgeConv module to capture local geometric features from points, which is missing in previous point-wise deep learning architectures (Xie et al. 2019). EdgeConv constructs a local graph between a point and its k-nearest neighbor points and applies convolution-line operations on the graph edges. DGCNN uses PointNet (Qi et al. 2017) as basic architecture but combines it with graph CNNs. Instead of using fixed graphs, like other graph CNN methods, EdgeConv updates its neighborhood graphs dynamically for each layer of the network, thereby effectively increasing the spatial coverage of the neighborhoods, as the convolution step between layers down-samples the point cloud.

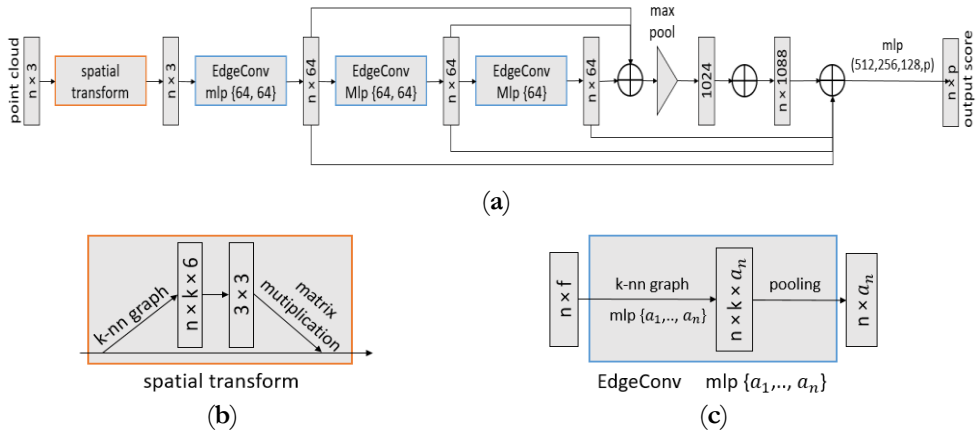


Figure 3.3 The DGCNN components for semantic segmentation architecture (a) The network uses spatial transformation followed by three sequential EdgeConv layers and three fully connected layers. A max pooling operation is performed as a symmetric edge function to solve for the point clouds ordering problem, i.e. it makes the model permutation invariant while capturing global features. The fully connected layers will produce class prediction scores for each point; (b) A spatial transformation module is used to learn the rotation matrix of the points and increase spatial invariance of the input point clouds; (c) EdgeConv which acts as multi-layer perceptron (MLP), is applied to learn local geometric features for each point. Image source: Wang et al. (2018).

Each EdgeConv block applies an asymmetric edge function $h\Theta(x_i, x_j) = h\Theta(x_i, x_j - x_i)$ across all layers to combine both the global shape structure (by capturing the coordinates of the patch center x_i) and the local neighborhood information (by capturing $(x_j - x_i)$) as shown in Figure 3.4. Similar to PointNet and PointNet++, the aggregation operation to downsample the input representation in DGCNN is max pooling.

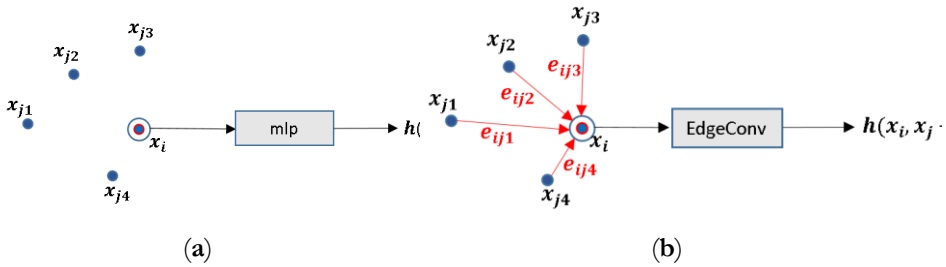


Figure 3.4 Basic differences between PointNet and DGCNN. (a) The PointNet output of the feature extraction $\boldsymbol{h(x_i)}$, is only related to the point itself, (b) EdgeConv of the DGCNN elaborates the local geometry relationship $\boldsymbol{h(x_i, x_j - x_i)}$ of a point to its neighbourhood. Here, a k-nn graph is constructed with $k = 4$.

Suppose a point cloud is given with n points, each consisting of F features, denoted by $X = \{x_1, \dots, x_n\} \subseteq R^F$. In our case, feature dimensionality of $F = 9$, as each point consists of 3D coordinates and 6 other features (e.g. RGB color, normalized spatial location, Intensity, etc.). Recall that a graph consists of a set of objects, V (nodes) and their relations, E (edges). For each point, x_i , DGCNN constructs a labeled k-nearest neighbor star graph whose vertices are x_i and the nearest neighbors, while its edges (i, j_1),…,(i, j_k) connect x_i to the points $x_{j1}, \dots, x_{jk}$ closest to x_i. The k-nearest neighbours of a point dynamically changes from layer to layer of the network and are computed sequentially. The k-nearest neighbours of a point dynamically changes from layer to layer of the network and are computed sequentially. The edge function is defined as $e_{ij} = h\Theta(x_i, x_j)$, where $h\Theta: R^F \times R^F \rightarrow R^{F\prime}$ is a non-linear function that contain learnable parameters Θ, and $\Theta = (\theta_i, \dots, \theta_k)$ is the weights of the filter to be optimized in each edge convolutional layer.

3.4.1.3 The choice of feature combinations and loss functions

Similar to PointNet and DGCNN, each point in the point cloud input is attributed with a 9 dimensional feature vector consisting of three spatial coordinates $(\mathrm{x}, \mathrm{y}, \mathrm{z})$ and six additional features. Candidate additional features are: spectral color information (Red, Green, Blue), normalized 3D coordinates (n_x, n_y, n_z), and off-the-shelf LiDAR features (Intensity, Return number, and Number of returns). Normalized 3D coordinates (n_x, n_y, n_z) are used as additional features by PointNet, PointNet++, DGCNN, and other networks to boost the translational invariance of the algorithm (Qi et al. 2018). Normalized 3D coordinates, in which the point cloud original coordinates are transformed to a common space coordinate (ranging from 0 to 1) by dividing $\mathrm{x}, \mathrm{y}, \mathrm{z}$ values by theirs maximum value of each tile, are expected to give global information. In the indoor case, normalized point coordinates provide a strong indication on the type of object (e.g. floor always has Z value close to 0, walls have X or Y values at 0 or 1, etc). This study tries to investigate the effectiveness of such normalized coordinates in the outdoor scenarios of orthogonal airborne point clouds. To evaluate the contribution of each feature, we compare four different sets of feature vectors as presented in Table 3.1.

Table 3.1 Different feature combinations

Feature Set	Set Name	Features
Set 1	RGB	$\mathrm{x}, \mathrm{y}, \mathrm{z}, \mathrm{R}, \mathrm{G}, \mathrm{B}, n_x, n_y, n_z$
Set 2	IRnN	$\mathrm{x}, \mathrm{y}, \mathrm{z}, \mathrm{I}, \mathrm{Rn}, \mathrm{N}, n_x, n_y, n_z$
Set 3	RGI	$\mathrm{x}, \mathrm{y}, \mathrm{z}, \mathrm{R}, \mathrm{G}, \mathrm{I}, n_x, n_y, n_z$
Set 4	RGBIRnN	$\mathrm{x}, \mathrm{y}, \mathrm{z}, \mathrm{R}, \mathrm{G}, \mathrm{B}, \mathrm{I}, \mathrm{Rn}, \mathrm{N}$

Description of different feature Sets presented in Table 3.1 is explain as the following.

- Feature Set 1 uses the default feature combination, as widely used by indoor point cloud benchmarks (e.g. S3DIS dataset). It consists of 3D coordinates, RGB color, and normalized coordinates.
- Feature Set 2 replaces RGB color by LiDAR features IRnN (Intensity, Return number, and Number of returns) to evaluate the importance of LiDAR features.
- Feature Set 3 combines two color channels (Red and Green) with LiDAR intensity to investigate the importance of spectral features.
- Feature Set 4 combines full RGB color features and LiDAR IRnN features and excludes normalized coordinates to evaluate the importance of global geometry.

During training, 4096 points are uniformly sampled from each training block of size 30m × 30m, to form data batches with a consistent number of points, while all points are used during testing. We used 9 features for training, therefore, the size of the data fed into the network is 4096×9. We use $k = 20$ nearest neighbors for each point to construct the k-nearest neighbor graph. For all experiments, the final model is obtained after running 51 epochs, optimized by an Adam optimizer with an initial learning rate of 0.001, a momentum of 0.9, and a mini batch size of 16. The 3D point cloud semantic segmentation using DGCNN is performed on the High Performance Computing (HPC) environment of Delft University of Technology, consisting of 26 computing nodes. For training, two Tesla P100-16GB GPUs were used.

Deep neural network learns to map a set of input to a set of output based on the training data. At each training step, the network compares the model predictions to actual labels to determine and increase the model performance. Typically, gradient descent is used as optimization algorithm to minimize the error and update the current model parameters (weights and biases). To calculate the model error, a loss function is used.

A class imbalance exists in our study area because it is heavily dominated by buildings (59%) and trees (19%). Therefore, we investigate two different loss functions that are incorporated within the DGCNN architecture: Softmax cross entropy loss and Focal loss (Lin et al., 2017).

(1) Softmax Cross Entropy (SCE) loss. This is a combination of a Softmax activation function and cross-entropy loss. Softmax is frequently appended to the last fully connected layer of a classification network. Softmax converts logits, the raw scores output by the last layer of the neural network, into probabilities in the range 0 to 1. The function converts the logits into probabilities by taking the exponents of the given input value and the sum of exponentials of all values in the input. The ratio between the exponential input value and the sum of exponential values is the output of softmax. Cross entropy describes the loss between two probability distributions. It measures the similarity of the predictions to the actual labels of the training samples.

Consider a training dataset $D = \{(x_i, y_i) | i \in \{1,2,\dots,M\}\}$ with x_i input data within a batch of size M, and y_i is the i-th label target (one-hot vector) of C classes. $f(x_i)$ denotes the dimensional feature vector before the last fully connected layer of C classes. W_j and b_j, $j \in \{1,2,\dots,C\}$ represent the trainable weights and biases of the j-th class in Softmax regression, respectively. Then the SCE loss is written as follows:

$$\mathcal{L}_{SCE} = -\sum_{i=1}^{M} \log\left(\frac{exp\left(W_{y_i}^T f(x_i) + b_{y_i}\right)}{\sum_{j-1}^{C} exp(W_j^T f(x_i) + b_j)}\right) \quad (3.1)$$

(2) Focal loss is introduced to address accuracy issues due to class imbalance for one-stage object detection. Focal loss is a cross-entropy loss that weighs the contribution of each sample to the loss based on the classification error. The idea is that, if a sample is already classified correctly by the network, its contribution to the loss decreases. Lin et al (2017) claim that this strategy solves the problem of class imbalance by making the loss implicitly focus on problematic classes. Moreover, the algorithm weights the contribution of each class to the loss in a more explicit way using Sigmoid activation. The focal loss function for multi-classification is defined as:

$$\mathcal{L}_{FL} = -\sum_{i=1}^{C} (y_i \log(p_i)\,(1-p_i)^{\gamma}\alpha_i + (1-y_i)\log((1-p_i)p_i^{\gamma}(1-\alpha_i)) \quad (3.2)$$

where C denotes the number of classes; y_i equals 1 if the ground-truth belongs to the i-th class and 0 otherwise; p_i is the predicted probability for the i-th class; $\gamma \in \{0, +\infty\}$ is a focusing parameter; $\alpha \in \{0,1\}$ is a weighting parameter for the i-th class. The loss is similar to categorical cross entropy, and they would be equivalent if $\gamma = 0$ and $\alpha_i = 1$.

3.4.1.4 Classification evaluation

To evaluate the prediction accuracy of different feature combinations and loss functions, the training set results are evaluated. This study uses several performance metrics as follows:

- Overall accuracy, indicating the percentage of correctly classified points of all classes from the total number of reference points. This metric shows general performance of the model, thus, may provide limited information in case of class imbalance.
- Confusion matrix, a summary table reporting the number of true positives, true negatives, false negatives and false positives of each class. The matrix provides information on the prediction metrics per-class and the types of errors made by the classification model.
- Precision, Recall, and F1-score. Precision and Recall are metrics commonly used for evaluating classification performance in Information Technology and related to the False and True Positive Rates (Raschka, 2014; Tharwat, 2020). The recall (also known as completeness) refers to the percentage of the total points

correctly predicted by the model, while the precision (also known as correctness) refers to the percentage of the results that are relevant. The F1-score is a weighted average of precision and recall to measure model accuracy.

3.4.2 Road network vectorization

In this research, road network vectorization aims at obtaining the centerline and boundary of roads, starting from airborne 3D points classified as road. The centerline of the road is represented by a polyline while the road surface boundary is represented by a polygon. Following the methodology in Figure 3.5, road vectorization is divided into two major tasks: skeletonization and road completion. Skeletonization extracts the road centerline while completion smooths and completes both road centerline and boundary.

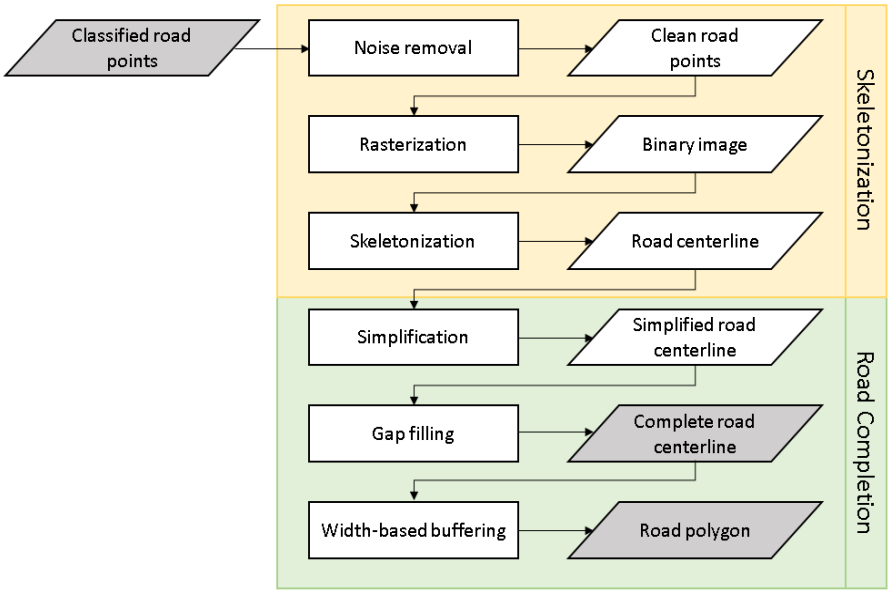


Figure 3.5 Pipeline of road network vectorization consists of two main tasks: skeletonization (inside the yellow box) and road completion (inside the green box). The method requires points classified as road as input data.

3.4.2.1 Road skeletonization

The first step of the proposed road network vectorization removes noise that remained in the classified road points. This study considers isolated points or groups of less than 10 points as noise or outliers. We apply DBSCAN clustering (Ester et al., 1996) to remove noise as this algorithm has the capability to handle arbitrary shapes including roads.

The second step, 2D rasterization, converts a set of noise-free road points to obtain a gridded image representing road and non-road pixels. A Kernel Density Estimator or KDE (Parzen, 1962) is used to estimate the density of points per-unit-area by a kernel function. Each grid of the output raster has a value determined by the density of neighboring points within a given radius. The KDE function is selected to exclude the area outside the road and to avoid gaps on the road caused by occlusions such as cars or trees. A weighted-distance Gaussian kernel of radius 2.5m is applied to obtain a density image of cell size 0.5m.

Third, skeletonization is performed to estimate the road centerline. Fast parallel thinning introduced by Zhang and Suen (1984) is applied to estimate the 2D centerline of a road. The basic principle of the algorithm is to remove all pixels, layer by layer, starting from the boundary of the shape until only those pixels belonging to the skeleton remain. The algorithm requires a binary input image and preserves skeleton connectivity by using sub-iteration for each boundary removal iteration. The binary image consists of road (value 1) and non-road pixels (value 0). The parallel thinning algorithm only operates on pixels of value 1, representing the road.

As illustrated in Figure 3.6.a, the black pixels assigned with number 1 represent the original road pixels. The algorithm processes only points P1 fully surrounded by black pixel neighbors in a 3×3 window as in Figure 3.6.b. That is, black pixels at the border and corners of the image are excluded. Next, thinning proceeds by removing non-skeleton pixels (change value of black pixel from 1 to 0) simultaneously until the skeleton pixels (white pixels with 1) remain. The skeleton result of the parallel thinning algorithm is presented in Figure 3.6.c.

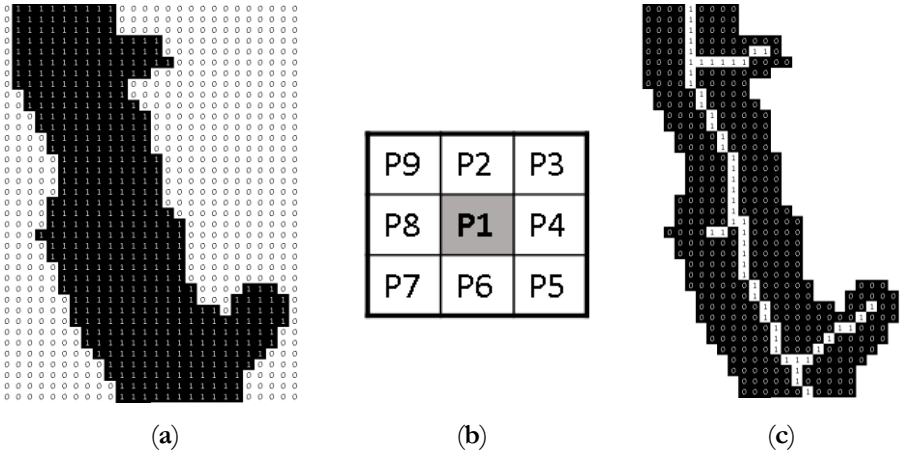


Figure 3.6 Road skeletonization obtained by parallel thinning of binary image. (**a**) Input binary image (black: road pixel, white: non-road pixel); (**b**) neighbors of $P1$ in 3×3 window; (**c**) Skeleton result (black: road pixel; white: skeleton pixel).

The first sub-iteration checks any pixel P1 to be removed or set to 0 if it satisfies all conditions as follows:

C.1 Pixel P1 is black and has eight neighbors;

C.2 $2 \leq S(P1) \leq 6$ with $S(P1)$ is the number of black (non-zero) neighboring pixels of P1;

C.3 The number of transition from white to black (0 → 1) of the neighbors ordered clockwise, equals 1;

C.4 At least one of (P2, P4, P6) is white ($P2 \times P4 \times P6 = 0$);

C.5 At least one of (P4, P6, P8) is white ($P4 \times P6 \times P8 = 0$).

The second sub-iteration re-examines each pixel and set its value to 0 whenever it satisfies both C.1 to C.3 and C.4' and C.5'.

C.4' At least one of ($P2, P4, P8$) is white ($P2 \times P4 \times P8 = 0$);

C.5' At least one of $(P2, P6, P8)$ is white $(P2 \times P6 \times P8 = 0)$.

3.4.2.2 Road centerline simplification

Skeletonization results in jaggy centerlines, therefore line simplification is applied. Visvalingam-Whyatt (1993) simplification is used. As illustrated in Figure 3.7, the algorithm iteratively removes vertices of least perceptible change based on triangles formed by three consecutive vertices. For any smallest triangle having an area smaller than a given area threshold, the middle vertex is removed. After each vertex removal, triangles are recomputed and the process is repeated. We set the area threshold to 0.3 meter to simplify centerlines, which is sufficient to preserve the shape of curved roads as exist in our study area.

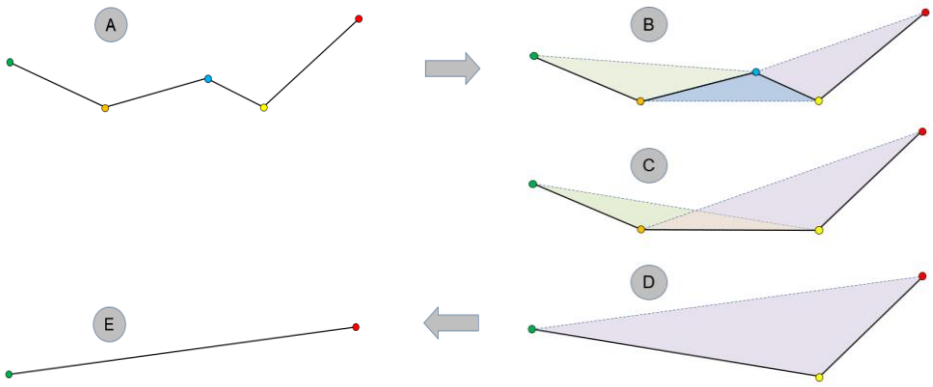


Figure 3.7 Road centerline smoothing by a Visvalingam-Whyatt approach. (**a**) the centerline input consists of line segments (black) and its points; (**b**) triangles of 3 consecutive points are formed; the blue triangle has the smallest area; (**c**) remaining line segments and points after deletion of the blue point, the green triangle is the smallest; (**d**) remaining line segments and points after deletion of the orange point; (**e**) remaining end points resulting in a smoothed centerline.

3.4.2.3 Road completion

We propose a tree-constrained extended dangle completion approach to fill in road gaps due to dense trees. A road gap is defined as an area that belongs to the road but due to some reasons is not detected, resulting in a disconnected road network. In this step, classified tree points are required to indicate which gaps on the road should be completed.

Polygons representing tree coverage are extracted from points classified as tree using an alpha-shape (Edelsbrunner et al., 1983) based boundary. The proposed algorithm finds dangle points by searching for end-points of road lines. A radial buffer polygon, also called dangle space, is created for each dangle point. Based on the average of corresponding road width, the determined buffer radius to estimate the dangle space is 4 meter. Next, the intersection between dangle space and tree polygon is examined. A road gap is detected if two different dangle spaces intersect one tree polygon (Figure 3.8.a). From this pair of dangle polygons, a line connecting the corresponding dangle points is created. The line is used to fill in the road gap. The newly created line is then added to the existing road centreline (Figure 3.8.b).

(**a**) (**b**)

Figure 3.8 Road centerline completion by a tree-constrained extended dangle method. (**a**) A missing centerline segment is identified as non-empty intersection between tree polygon (green) and the dangle polygon (white circle) of a dangle point (purple); (**b**) road gap filling adds new road segments (red lines inside dotted white circles) forming a complete road centerline.

Width-based buffering is applied to obtain road polygons. The road width is estimated by a Euclidean distance transform of the skeleton to the boundary. The distance transform finds the minimum distance between any point in the image to the closest zero pixel of the binary road image. The Euclidean distance is estimated by the following equation:

$$d_i = \sqrt{\sum_i^n \left(p_i - {b_i}^2\right)} \tag{3.3}$$

p_i= input points of the skeleton

b_i= the background point (value=0) where the smallest distance occurs to input point p_i.
n= number of input points

Using the same binary image as for skeletonization, we estimate the road width by a distance transform. Such method results in another type of skeleton that encodes the maximum distance between any skeleton pixel and its corresponding edge pixels. The estimated width is assigned to the smoothed centerline by a spatial join. In this case, the smoothed centerline segment records the average of all the distance values that is intersect to any distance skeleton pixel. The final width used for the next step is the average of these widths. The smoothed centerline segment is buffered according to the estimated width.

3.4.2.4 Road evaluation

For evaluating the geometric accuracy of resulting road vectors, this study implements a set of area-based quality metrics, which are calculated based on polygon. To estimate the True Positives (TP), False Positives (FP), and False Negatives (FN), the evaluation method requires polygons created by buffering the road centerline results and the centerline references (see Figure 3.9).

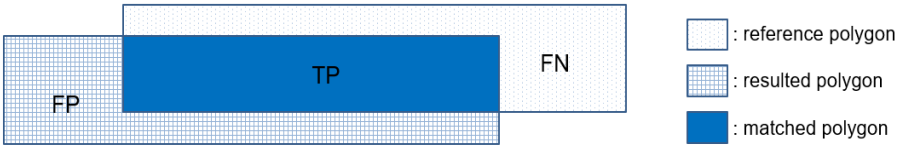


Figure 3.9 Polygon matching principle to estimate the total matched area between road result and reference polygons. Matched area is considered as TP (blue), area only found in the result is considered as FP (gridded area), otherwise FN (white).

In addition, the extraction rate is also computed by comparing the length of road segments in the result and in the reference within 1.5 meter buffer. Road quality metrics (Completeness and Correctness) are then computed using Equation 3.4 to 3.5 below.

$$Completeness = \frac{TP}{TP+FN} \times 100\% \quad (3.4)$$

$$Correctness = \frac{TP}{TP+FP} \times 100\% \quad (3.5)$$

Completeness describes how well reference data is explained by the extracted road and is determined by the ratio of area/length of the matched reference to the total area/length of the reference data. Correctness represents the percentage of correctly extracted road data and is determined by the ratio of the area/length of the matched extraction to the total area/length of matched extraction (Rottensteiner and Clode, 2009).

3.4.3 Building vectorization

Our study area is located in a metropolitan city. Buildings in such area, which share building walls, form a building block. The building blocks are basic information essential for practical urban planning and design (Spiller and Agudelo, 2011) and urban spatial analyses (Salomons and Point, 2012). To extract building polygons, we use our previous work presented in Chapter 5 (Widyaningrum et al., 2020) that extended a 2D shrinking circle algorithm (Ma et al., 2012) to extract skeletal building points. The shrinking circle method works directly on points so rasterization is not necessary. The shrinking circle method estimates the skeletal points of a shape by fitting maximally inscribed circles to the given boundary points, based on nearest neighbors and normal vectors. The skeletal points are the center points of all maximally inscribed circles. The workflow of building block outline extraction is illustrated in Figure 3.10.

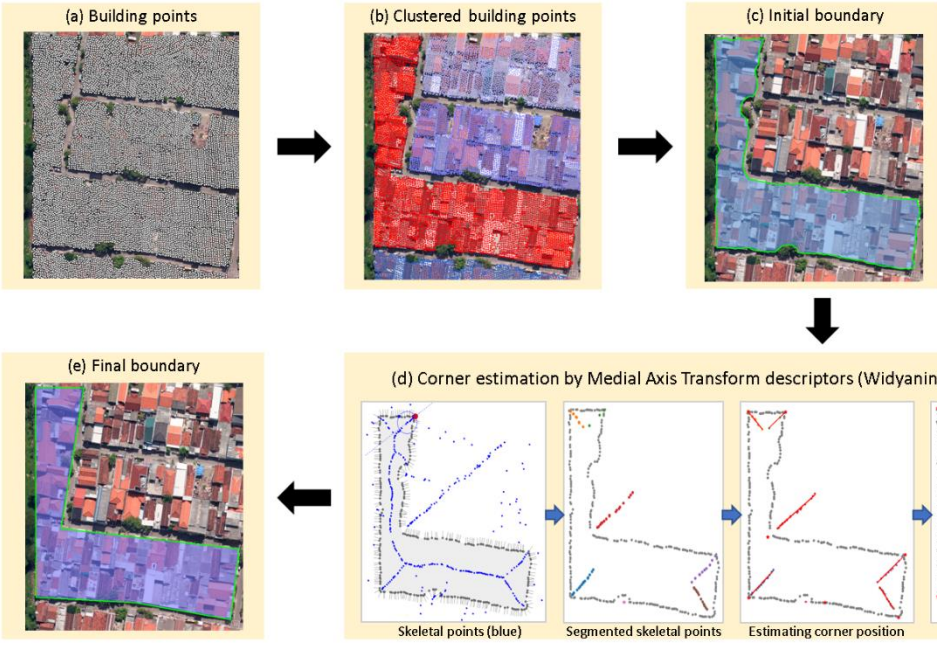


Figure 3.10 Building outline extraction using medial axis descriptors. (**a**) building points classified by a DGCNN; (**b**) clustered building point, different color different cluster; (**c**) initial building block boundary (green line) is estimated by alpha-shape algorithm; (**d**) corner estimation by a MAT descriptors; (**e**) final smooth building block outline (green).

On the classified building points, DBSCAN clustering algorithm is performed to divide building points into several building clusters. For each building cluster, boundary points are selected by an alpha-shape algorithm. On the resulting boundary points, a so-called shrinking circle method is applied to generate the medial building block points.

Here, medial refers to a specific skeleton definition. The medial axis descriptors (circle radius, angles and normals) of all skeletal points are extracted and used for the next process, the corner-based segmentation. The real corner position is estimated by fitting a line to each skeletal segment followed by extrapolating the zero-radius location on that fitted line. Final smooth and accurate outlines are obtained by connecting the corner points based on theirs indices.

3.5 Results and discussion

This section, at first, provides quantitative and qualitative results on point cloud classification by DGCNN with different feature combinations. Second, the evaluation of the vectorization results are presented. Third, supplementary discussion related to the use of two different loss functions and the use of RGB features from orthophotos in case of high-rise buildings is presented.

3.5.1 Classification results

To investigate the best feature combination to classify ALS point cloud colored by an orthophoto using a deep learning approach, three different metrics (completeness/recall, correctness/precision, and F1-score) along with Overall Accuracy (OA) are used. For ALS point cloud classification, four different feature combinations and two loss functions are compared. The total number of samples used for training is 30.929.919 points, dominated by building points (59%). Trees, bare land and road classes are sampled by 21%, 13% and 7%, at the points respectively.

Table 3.2 shows the classification results of all predefined feature combinations and loss functions used in this study. Based on the evaluation results, feature Set 4 achieves the highest overall accuracy (91.8%) and F1-score for all classes. In general, the use of normalized coordinate features (n_x, n_y, n_z) in combination with other features is not as effective as the combination of spectral color with LiDAR features. The use of full RGB color and off-the-shelf LiDAR features significantly improves the F1-score of trees class by at least 7% and buildings by 5.7%.

Table 3.2 Point cloud classification results of different feature combinations.

Feature Set	Feature vector	OA (%)	F1-score (%)			
			Bare land	Trees	Buildings	Roads
Set 1	$x, y, z, R, G, B, n_x, n_y, n_z$	83.9	83.0	80.3	87.3	75.1
Set 2	$x, y, z, I, Rn, N, n_x, n_y, n_z$	85.7	84.2	81.6	89.1	79.0
Set 3	$x, y, z, R, G, I, n_x, n_y, n_z$	83.9	83.5	79.9	87.4	74.9
Set 4	$\mathbf{x, y, z, R, G, B, I, Rn, N}$	91.8	87.7	88.6	94.8	84.1

Table 3.3 Per-class classification metrics of different feature combinations.

Feature Set	OA (%)	Bare land (%)		Trees (%)		Buildings (%)		Roads (%)	
		Prec.	Recall	Prec.	Recall	Prec.	Recall	Prec.	Recall
Set 1	83.9	76.2	**91.2**	81.6	79.0	87.6	87.0	83.9	68.0
Set 2	85.7	77.5	92.1	88.4	75.7	86.9	91.4	84.2	74.5
Set 3	83.0	76.3	92.1	82.3	77.7	86.9	87.8	86.1	66.2
Set 4	**91.8**	**85.2**	90.4	**93.3**	**84.3**	**93.4**	**96.2**	**86.9**	**81.6**

Based on the class quality metrics presented in Table 3.3, the potential of different feature combinations to predict different land cover classes in our test area is discussed below:

- Bare land class

The lowest recall rate is achieved by feature Set 4 RGBIRnN. This indicates that normalized 3D coordinates are useful to detect bare land correctly. The highest recall rates are achieved by feature Set 4. This is because feature Set 4 produces much less False Positives (FP), which makes the precision rate higher. Combination of LiDAR intensity and normalized coordinates (Set 2 and 3) effectively maintains a high number of points correctly classified as bare land, indicated by high recall (92.1%). On average, the road class has the lowest recall rates while the bare land class always has the lowest precision. This indicates that there is high confusion between bare land and road classes, which we assume mainly happens due to the presence of open areas having similar height and color such as parking areas, front yards, and backyards.

- Tree class

Feature Set 4 obtains the highest recall and precisions rate with a score of 84.3% and 93.3%, respectively. The use of both RGB color and LiDAR information in feature Set 4 significantly increases the tree detection by almost 11% compared to the other feature Sets. In general, the main source of error are trees misclassified as buildings, which particularly occurs for trees adjacent to buildings. Our results also show that there are more trees misclassified as building than buildings detected as tree which results in recall rates that are always lower than precision.

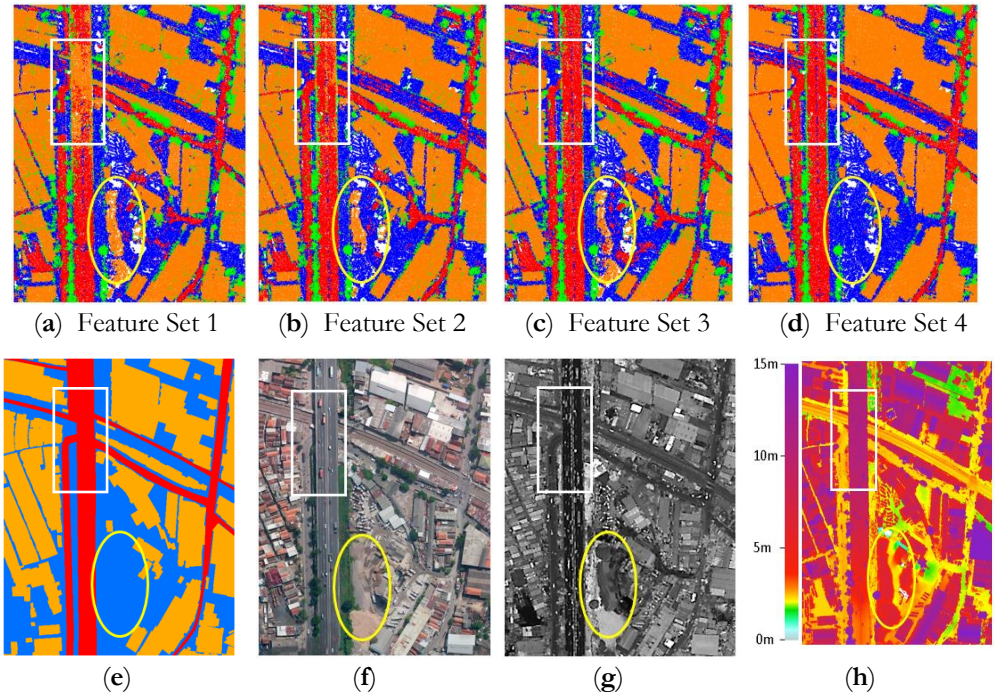
- Buildings

The recall and precision rate of building detection did remarkably improve when using feature Set 4. Likely, the decreasing number of confusions between buildings and trees induces higher building classification accuracy. The biggest error sources for building classification are small details on roofs and building façades classified as tree.

- Roads

Although the road detection accuracy is not as good as other classes, the highest recall and precision rate is achieved by feature Set 4 with 81.6% recall and 86.9% precision. Using RGB and intensity (Set 4) as input features significantly improves the recall rate of roads by reducing the number of road points detected as bare land.

Figure 3.11 visualizes the classification results of different feature combinations over a subset of our test area in comparison to the following data sources: base map, orthophoto, LiDAR intensity, and Digital Surface model (DSM).



(**a**) Feature Set 1 (**b**) Feature Set 2 (**c**) Feature Set 3 (**d**) Feature Set 4

(**e**) (**f**) (**g**) (**h**)

Figure 3.11 Samples of different feature Set results. (**a**) to (**d**): classification results of four feature combinations in comparison to (**e**) base map, (**f**) aerial orthophoto, (**g**) LiDAR intensity, and (**h**) Digital Surface Model (DSM). In (**a**) to (**e**) blue color represents bare land, green represents trees, orange represents buildings, and red represents roads, respectively.

The white rectangle highlights an area where most classification results fail to detect a highway and an adjacent road of different height. Feature Set 1 results in misclassification of some points on the overpass highways as buildings and the adjacent road below the highway as bare land points (see white rectangle in Figure 3.11.a). Because roads, buildings, and bare land have similar geometric characteristics (e.g. planarity), using LiDAR intensity (Figure 3.11.g) in addition is beneficial to increase the roas classification accuracy.

A sand pile exists in the study area due to construction work at the time of data acquisition (indicated by yellow ellipses in Figure 3.11). Only feature Set 4 correctly classifies the sand pile points as bare land while the other feature Sets falsely classify the sand pile as building. This suggests that using complementary airborne LiDAR and spectral orthophoto features is increasing detection accuracy.

3.5.2 Vectorization results

3.5.2.1 Road evaluation

For road centerline and outline vectorization, road points produced by feature Set 4 are used as this feature combination returns the best quality result. Using the method described in Section 3.4.2, a road centerline is extracted from the input road points by skeletonization. Visvalingam-Whyatt simplification method regularizes the jaggy centerline as resulting from the skeletonization step. Skeleton-based road network vectorization consists of road skeletonization and road completion (Figures 3.12).

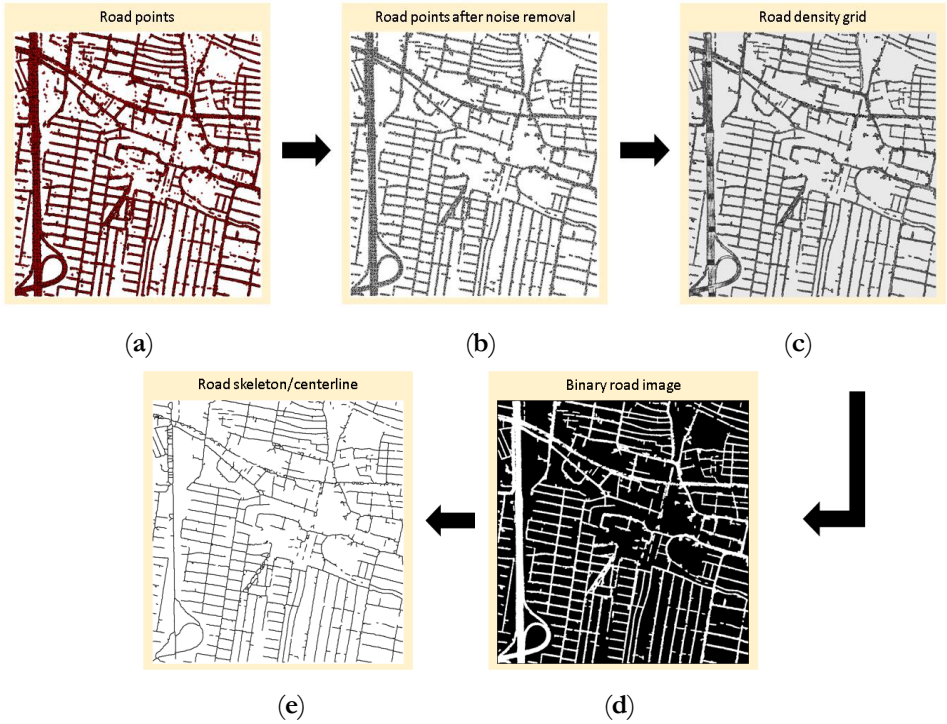


Figure 3.12 Road skeletonization workflow. (**a**) road points obtained by DGCNN classification; (**b**) Noise-free road points after DBSCAN clustering; (**c**) rasterization of the road points using a kernel density estimator; (**d**) Binary road network image; (**e**) Continuous road centerline obtained by a parallel thinning skeletonization.

Figures 3.13.a and 3.13.b show the road centerline result and reference, respectively. The method successfully extracts almost all existing roads in the area while maintaining network topology. Narrow roads in dense settlement areas are also extracted by our method. Road polygons, which incorporate estimated road width, are presented in Figure 3.13.c. Applying area-based evaluation metrics described in section 3.4.2, shows that our method is able to deliver road polygon result with 80.6% completeness and 72.6% correctness.

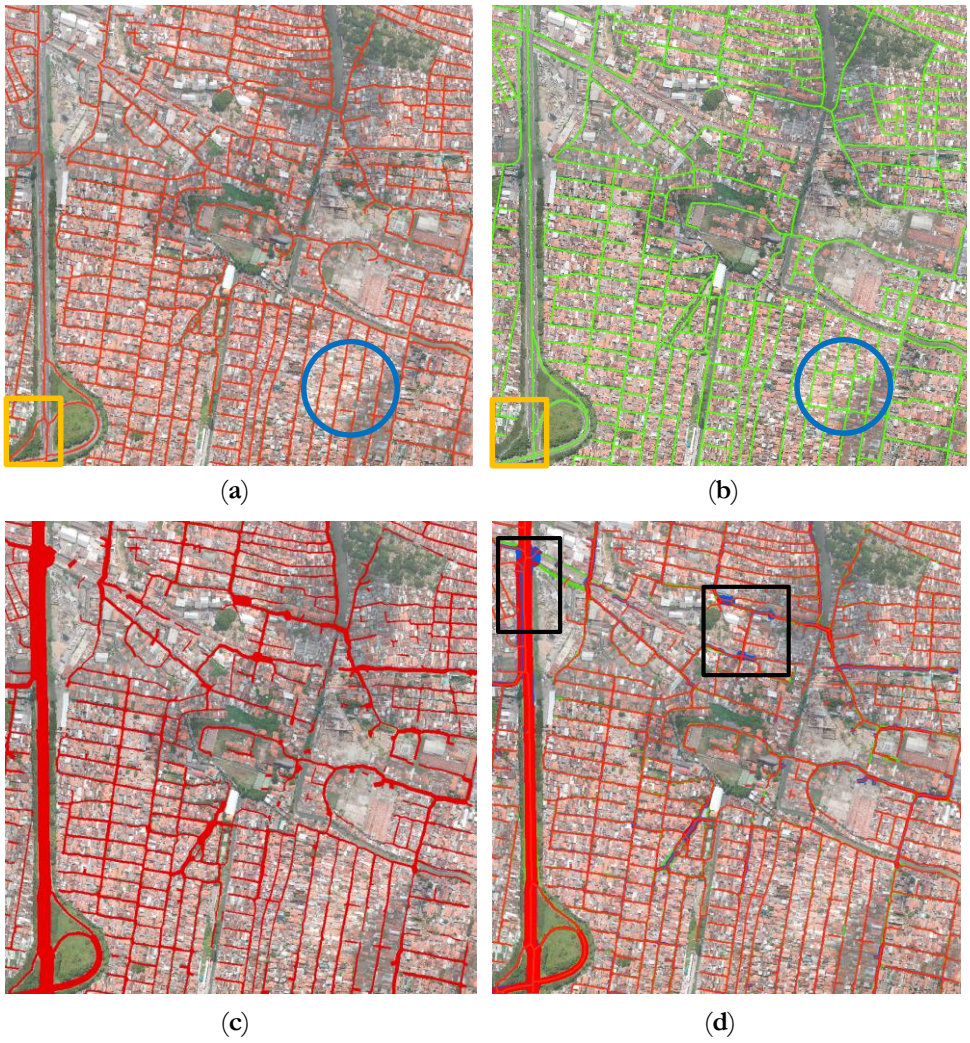


Figure 3.13 Road centerline extraction results overlaid on orthophoto. (**a**) road centerlines from the proposed method (red); (**b**) reference centerline (green); (**c**) road polygons from the proposed method (red); (**d**) road network evaluation results: red indicates matched areas (true positive), green indicates undetected road areas (false negative), while blue indicates roads only detected by our method (false positive).

We also evaluate the quality of road centerline results. Using 1.5 meter positional accuracy, the road centerline result is correct if it lies within 1.5 meter buffer from the reference centerline. Based on this evaluation, the method results in a completeness and correctness of 79.3% and 80.2%, respectively. Our method can detect several roads which are not presented in the reference centerlines (see comparison of two orange rectangles in Figure 3.13.a and 3.13.b) which possibly happens due to human error. Several road segments not presented in our result are usually small road paths which

sometimes covered by building roofs (see blue circles in Figure 3.13.a and 3.13.b).

As shown in Figure 3.13.d, most false positive areas (blue) are found at complex road configurations like a highway adjacent to a road of different height or a road adjacent to a paved park (indicated by the black rectangles). In these two cases, our method detects two adjacent roads instead of one. Parking areas are often classified as road which also contributes to the false positive rate. False negatives are mostly correspond to several road segments not detected by our method. There are also cases where the resulting road polygons have smaller width than the reference, especially at the highway borders, which increases the false negative rate.

3.5.2.2 Building evaluation

Building outline vectorization is performed using the feature Set 4 classification result. Given the segmented building points, MAT-based approach successfully extract the corner of buildings and building blocks accurately. Figure 3.14 shows an area where building points are clustered into several blocks. In addition, the comparison between building block outlines and reference building outlines from the 1:1.000 base map are shown in Figure 3.14.c and 3.14.d, respectively.

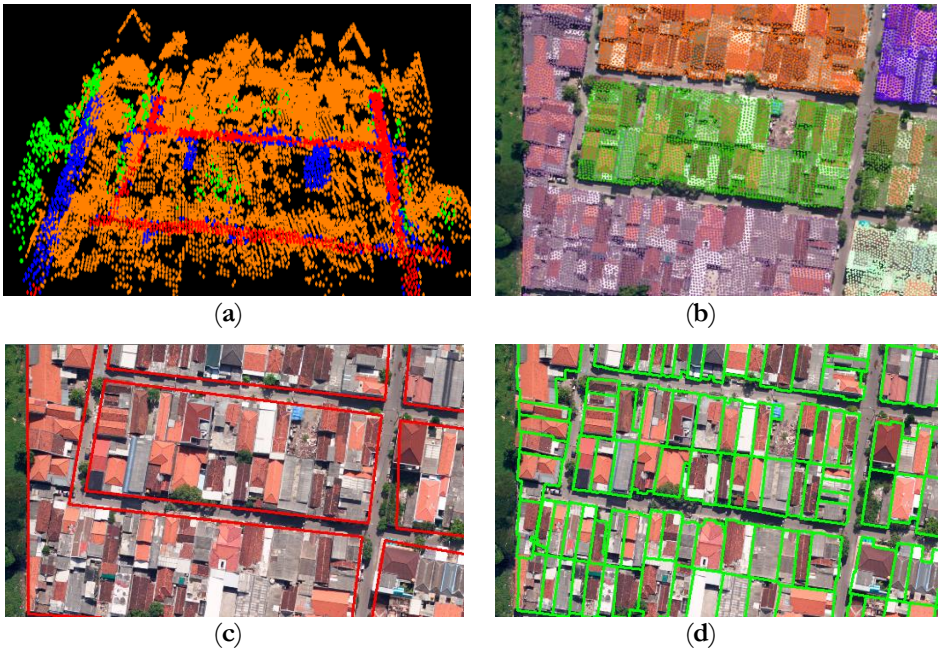


Figure 3.14 Building points of dense settlement forming a block. (**a**) 3D visualization of point classified as building shows the building roof irregularity of the area; (**b**) clustering the building points results in block of buildings rather than individual buildings; (**c**) the building block outlines result (red) extracted by a MAT method; (**d**) individual building outlines reference (green) shows human subjectivity in interpreting the building boundaries.

In some parts of our study area, the building points are grouped into several building blocks instead of individual buildings as buildings are closely located together. The lack of gaps between buildings, variation in roof shapes and color makes it difficult to separate building points in individual buildings. Sparser point cloud density caused by down-sampling during the classification makes it even more challenging.

3.5.3 Supplementary discussion

In this discussion we discuss two effects in the DGCNN classification: (i) the influence of the loss function used and (ii) the influence of so-called relief displacement, which causes some mismatch between ALS point clouds and RGB imagery.

3.5.3.1 Loss function

Even though class imbalance exists in our study area, the overall accuracy is not necessarily increased by applying a Focal Loss (FL) function as may be expected. Table 3.4 shows the results on feature Set 4 when using two different loss functions: SCE and FL (α = 0.2, γ=2).

Table 3.4 Point cloud classification results comparing two different loss functions, SoftMax Cross Entropy (SCE) and Focal Loss (FL) on feature Set 4. Results are quantified in terms of Overall Accuracy (OA) and F1-score.

Loss function	Feature vector	OA (%)	F1-score (%)			
			Bare land	Trees	Buildings	Roads
SCE	x, y, z, R, G, B, I, Rn, N	91.8	87.7	88.6	94.8	84.1
FL	x, y, z, R, G, B, I, Rn, N	88.1	81.8	85.3	92.7	68.6

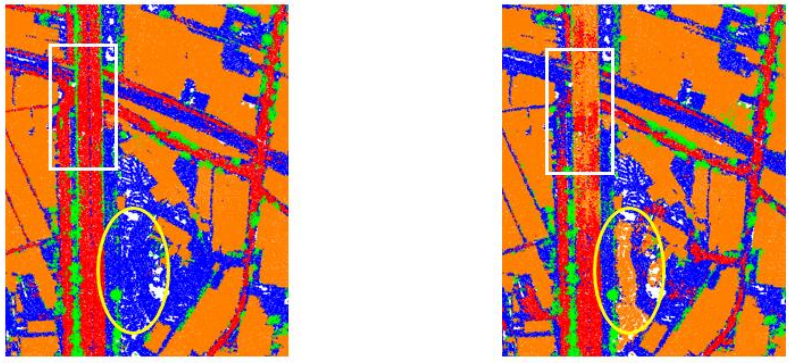
The overall accuracy (OA) of the results of feature Set 4 decreases by 3.7% if using FL. However, the F-1 score for the bare land and roads classes drops by ~6% and ~15%, respectively, if using FL. Our explanation for this is that the loss function focuses on decreasing the loss of the classes that produces large amount of misclassified points, in this case buildings and trees, thereby somehow neglecting bare land and notably roads.

Based on the confusion matrix presented in Table 3.5, Feature Set 4 in combination with FL has the highest precision rate (86.7%) for trees, but the number of correctly detected tree points is lower than for the other feature Sets. This is because FL focuses on increasing the detection rate by evaluating the errors of the dominant class so that the number of misclassified tree points is decreasing. For building classification, the highest recall (93.7%) is achieved when using FL with only 0.3% recall difference SCE. For road classification, the use of FL doubled the number of false negatives compared to SCE.

Table 3.5 Confusion matrix matrix for results obtained by applying DGCNN on Feature Set 4 when using Focal Loss. The matrix contains numbers of points.

Feature set 4 (RGBIRnN)		Reference				Precision
		Bare land	Trees	Buildings	Roads	
Prediction	bare land	340,132	770	33,529	78,364	75.1%
	trees	304	553,175	105,094	42	84.0%
	building	18,099	83,704	1,552,315	3,367	93.7%
	road	20,557	97	763	112,952	84.1%
Recall		89.7%	86.7%	91.8%	58.0%	**88.1%**

The white rectangles and yellow ellipses in Figure 3.15 indicate (a) an area where some parts of the highway are misclassified as building and, (b) a sand pile is misclassified as building when using Focal Loss (FL). For our purposes, the SCE loss function performs better then FL in the DGCNN architecture.



(**a**) (**b**)

Figure 3.15 Comparison of two classification results obtained using two different loss functions. (**a**) classification results using SCE loss function results in more complete roads; (**b**) classification results using FL. using FL returns incomplete roads (white rectangle) and falsely classifies a sand pile as building (yellow ellipse).

3.5.3.2 Relief Displacement

One drawback of using aerial photos is the positional shift of high elevated objects (e.g. high rise buildings). This effect is called relief displacement and is caused by variation in camera angle. Displacement errors increase with the height of the object and the distance to the acquisition location. Objects suffering from relief displacement in photos usually have bigger size in the photo than in reality, as some parts of vertical walls are exposed and buildings appear to lean to one specific direction. In aerial photo classification, relief displacement is considered as one of the main sources of mislabeling (Chen et al., 2018). Figure 3.16 shows a relief displacement error in an orthophoto of a leaning building that blocks a lower building and nearby trees.

The use of ALS point clouds is a way to detect objects blocked by high-rise buildings that even a human operator cannot identify in an orthophoto. For example, part of a building in Figure 3.16.c (highlighted by a white ellipse) is automatically and correctly detected by our method (yellow outline) but is missing in the building reference (pink outline). This means that even though we use ground orthophotos to color the point clouds, which, as a consequence, results in wrongly coloured points in case of relief displacement, the network we employ still classifies the points correctly. It is likely that during training, DGCNN is able to learn and give smaller weight or big penalties to the color features in case relief displacement exist, thereby favoring the geometric point cloud information.

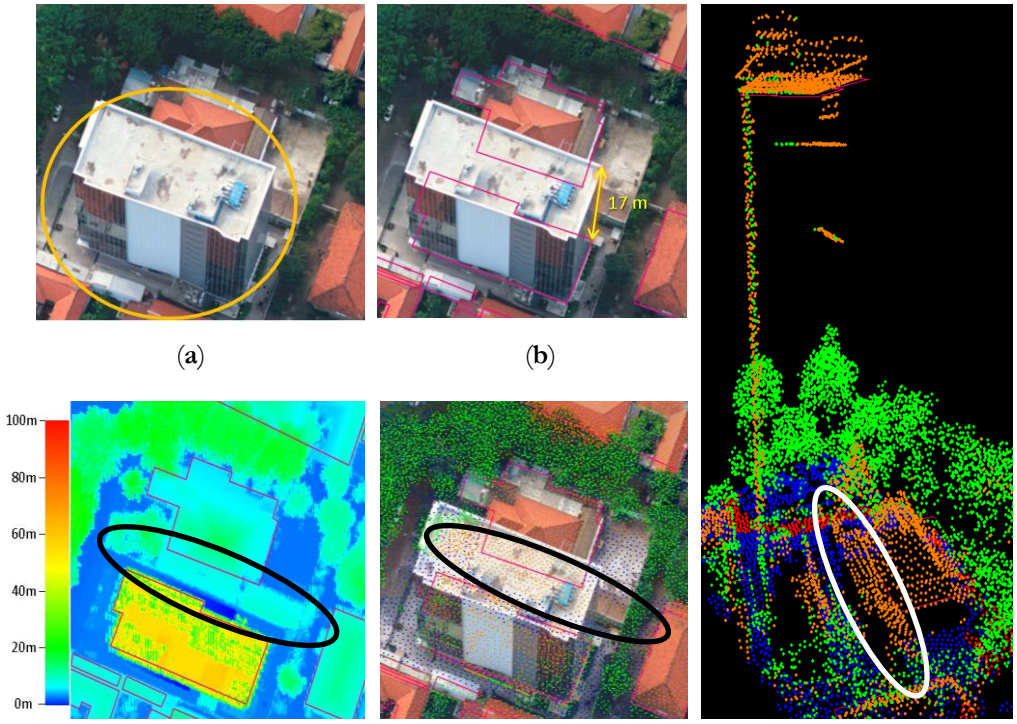


Figure 3.16 Relief displacement makes a high rise building block an adjacent lower building and trees. (**a**) The leaning of a building (inside orange circle) on an orthophoto indicates relief displacement; (**b**) the building in the orthophoto has an offset of up to 17 meter from the reference polygon (pink outlines); (**c**) the LiDAR DSM indicates a part of building that does not exist in the base map/reference (inside the black ellipse); (**d**) DGCNN can detect building points (orange) correctly including the missing building part (inside the black ellipse); (**e**) 3D visualization of the classified building points including the missing building part (inside the white ellipse).

3.6 Conclusion and future works

In this study, a complete methodology to automatically extracting the urban map objects (roads and buildings) using effective point-wise deep learning and skeletonization of a colored ALS point cloud. The proposed method shows promising results for extracting urban map objects automatically from the combination of real world ALS point clouds and orthophotos. We labeled the training samples used for classification using the best available public vector data from a 1:1000 base map. Experiments were conducted to test various feature combinations and two loss functions to classify outdoor point cloud data using the recent DGCNN architecture. Based on the classification results, the combination of full RGB image features and airborne LiDAR features outperforms other feature sets and significantly increases the classification accuracy. The Softmax Cross Entropy loss function performs better than Focal Loss, although the latter loss function was included in the testing because of some class imbalance in our input data.

The output of the classification step is used as input for road and building outline vectorization. Consider the road networks characteristics, skeletonization facilitates efficient extraction of continuous centerlines and network topology. Our road extraction method successfully delivers 89% completeness of existing roads. We found that the use of kernel density is partly useful to remove the small gaps on the road due to cars. We demonstrate the novel of road completion task based on tree-constrained approach to fill the road gaps. Still there is space for some improvement, especially considering the positional accuracy of road intersections that is affected by some skeletonization characteristics. Due to the orthogonality of the ALS point cloud, our method does not perform well on multi-level roads of different altitude.

Further research should include multi-scale processing to enable individual building and tree crown extraction. The development of an optimal input feature and block size selection procedures are also worth to be studied further. Such procedures should largely replace the current empirical for increasing deep learning classification accuracy.Our research indicates that 3D deep learning matured so much that it is now actually able to extract geometric information as required for digital maps at near-operational quality, but in a much shorter time than traditional workflows. The applicability of our method to data representing other cities and countries as well as possible extensions to rural environments is an interesting direction for future research.

4

Automatic building outline extraction from ALS point clouds by ordered points aided Hough transform

Many urban applications require building polygons as input. However, manual extraction from point cloud data is time- and labor-intensive. Hough transform is a well-known procedure to extract line features. Unfortunately, current Hough-based approaches lack flexibility to effectively extract outlines from arbitrary buildings. We found that available point order information is actually never used. Using ordered building edge points allows us to present a novel ordered points–aided Hough Transform (OHT) for extracting high quality building outlines from an airborne LiDAR point cloud. First, a Hough accumulator matrix is constructed based on a voting scheme in parametric line space (θ, r). The variance of angles in each column is used to determine dominant building directions. We propose a hierarchical filtering and clustering approach to obtain accurate line based on detected hotspots and ordered points. An Ordered Point List matrix consisting of ordered building edge points enables the detection of line segments of arbitrary direction, resulting in high-quality building roof polygons. We tested our method on three different datasets of different characteristics: one new dataset in Makassar, Indonesia, and two benchmark datasets in Vaihingen, Germany. To the best of our knowledge, our algorithm is the first Hough method that is highly adaptable since it works for buildings with edges of different lengths and arbitrary relative orientations. The results prove that our method delivers high completeness (between 90.1% and 96.4%) and correctness percentages (all over 96%). The positional accuracy of the building corners is between 0.2–0.57 m RMSE. The quality rate (89.6%) for the Vaihingen-B benchmark outperforms all existing state of the art methods. Other solutions for the challenging Vaihingen-A dataset are not yet available, while we achieve a quality score of 93.2%. Results with arbitrary directions are demonstrated on the complex buildings around the EYE museum in Amsterdam.

This chapter is organized as follows: Section 4.1 provides the background of the research. Section 4.2 describes related work on 2D building outline extraction. The methodological framework is presented in Section 4.3 while Section 4.4 describes

different test sets and data preprocessing steps used in this research. Section 4.5 presents sensitivity analyses and experiments, followed by Section 4.6 that presents and discusses results. Finally, conclusions and recommendations are given in Section 4.7.

4.1 Introduction

The detection of straight and accurate building outlines is essential for urban mapping applications like 3D city modeling, disaster management, cadaster, and taxation. To accommodate the high demand of various applications, accurate building outline extraction requires an automated procedure. Rottensteiner and Briese (2002) stated that in the building reconstruction task, building boundary determination is a crucial but difficult step. In recent years, automatic approaches for detecting building roof outlines are still intensively studied. In urban remote sensing, automatic building line detection has a low success rate due to scene complexity, incomplete cue extraction, and sensor dependencies (Sohn and Dowman, 2007).

LiDAR point clouds have become one of the most commonly used input data for large-scale mapping. For efficiency purposes, the need to optimize LiDAR data usage has increased rapidly. As LiDAR is able to provide accurate three-dimensional (x, y, z) point clouds free from relief displacement, the use of LiDAR data to extract building polygons automatically has become a key target for researchers and practitioners within the geospatial industry. However, extracting building boundaries from point clouds is still a challenging task because LiDAR points do not always exactly hit the edge of a building. As a result, LiDAR point clouds feature jagged edges instead of straight and continuous lines. In addition, different kind of building roof configurations (size, shape, color, etc.) and the surrounding context increase the difficulties to design an automated method. A robust approach is required to adapt to different kinds of buildings and overcome the influence of noise. Efforts on building outline extraction were also conducted on the combination of LiDAR point clouds and aerial images to use each of their advantages. Unfortunately, to fuse different input data is not easy as building representations may suffer from relief displacement or building distortion in image scenes (Widyaningrum and Gorte, 2017). In many cases, the geometric position of images and LiDAR point clouds hardly match.

Machine learning approaches, such as Support Vector Machine (SVM), Random Forest (RF), deep learning, etc., undeniable provided a breakthrough in the field of point cloud processing. Machine learning has been widely used to improve object extraction (e.g., building, road, trees, etc.), classification, and segmentation. Many geo-applications (base map production, cadaster, road inventory, etc.) require fine object boundaries to generate Geographic Information System (GIS) vector data as a final product. However, machine learning methods use neighborhood information to obtain learned or handcrafted features. Notably at borders between segments, such as at building outlines, such features are fuzzy. As a consequence, extracting sharp edges is difficult

for machine learning methods and results typically do not meet map production requirements. Therefore, post processing is necessary. Currently, a limited number of image-based building delineation tools exist, including BREC (Gamba et al., 2009), as well as point cloud-based commercial software such as TerraScan and ENVI. Nevertheless, the quality (geometric accuracy, straightness, and completeness) of the extracted building outline results need to be improved, especially for complex buildings (Golubiewski et al., 2019; Susetyo et al., 2018). This research aims to provide an alternative solution to extract accurate and straight building outlines from point clouds automatically.

The problem of line detection method is one of establishing meaningful groups of edge points that lies along straight lines. Hough transformation is a well-known model-based approach that uses length-angle or slope-intercept parameters to detect lines (Princen et al., 1992). Hough transform was introduced by Paul Hough in 1962 to detect curves in images and was applied to the field of computer vision by Duda and Hart (1972), who encouraged the use of the length from origin R and orientation angle θ for line parameterization. It was designed to solve a number of computer vision problems. Vosikis and Jansa (2008) stated that Hough transformation is a powerful tool for automated building extraction and creation of digital city models, but also that the degree of automation is still highly correlated to the quality of the input data. Another challenging problem of the use of the Hough transform is the limited accuracy of the object extraction, which is sensitive to the resolution of the accumulator space and to the noise in the data (Herout and Jansa, 2008; Lee and Kweon, 1997). Performance on detecting different sizes and orientations of buildings automatically also remains a problem.

This research proposes a new method to extract accurate building outlines from ALS (Airborne Laser Scanner) point clouds automatically using an extension of Hough transform that exploits lists of ordered points to define line segments and corners. We provide the following three significant contributions to overcome common issues when dealing with the use of Hough transformation for line extraction:

1. Detection of arbitrary directions. Regularization should not hamper the extraction of consecutive roof edges that are not perpendicular or roof edges with an orientation not matching the overall orientation of a building;
2. Extraction of different interrupted segments of different lengths belonging to the same line. Instead of a line, collinear line segments should be distinguishable for preserving the original building geometry;
3. Robustness to noise, flaws, and irregularity. The shape and size of a building should be preserved in case jaggy points or due to objects exist (e.g., trees) adjacent to the building causing flaws in the building segmentation result.

4.2 Related work

Several methods to extract building outlines from point clouds have been proposed in the past. This literature review first discusses building extraction methods using various remote sensing techniques, followed by the use of LiDAR data and regularization for building extraction. Second, the use of Hough transform for line extraction is discussed.

Recent studies on edge detection using deep learning techniques focus more on natural images (Bertasius et al., 2015; Shen et al., 2015; Liu et al., 2017; Yang et al., 2016; Wang et al., 2019, Kelm et al., 2019) and remote sensing images (Lu et al., 2018). Resulting edges are often thick and noisy and require post processing before thinned and sharp boundaries are obtained (Deng et al., 2018). In 2018, Microsoft conducts building footprints extraction for the US and Canada areas from satellite images by first classify building pixels using a deep learning toolkit (called CNTK), followed by a polygonization step that converts building pixel into polygon. The polygonization is conducted by imposing a priori building properties that are manually defined and automatically tuned. Some of these a priori properties are:

- Building edge should have some minimal some minimal length, both relative and absolute, e.g., 3 m;
- Consecutive edge angles are likely to be 90 degrees;
- Consecutive angles cannot be very sharp, i.e., smaller than some auto-tuned threshold, e.g., 30 degrees;
- Building angles likely have few dominant angles, meaning that all building edges are forming an angle of (dominant angle $\pm$ nπ/2).

Yu et al. (2018) claim to present the first edge-aware deep learning network for 3D reconstruction from point cloud data, namely EC-Net. Edge-aware means here that the network learns the geometry of edges from training data, and during testing, it identifies edge points and generates more points along edges (and over the surface). This method has limitations in cases of large holes and otherwise incomplete data. Sharp edges around tiny structures that are severely under-sampled may not be extracted because the training patches become too large for tiny structures.

Many studies combine different type of remote sensing to acquire accurate building outlines. Sohn and Downman (2007) proposed a method for building footprint extraction from a combination of IKONOS and LiDAR data. They apply a model-driven approach on a LiDAR point cloud and a data-driven approach on satellite images. Li et al. (2013) present an automatic boundary extraction method by combining LiDAR and aerial images to handle various building shapes. Their method consists of three main steps. First, roof patch points are detected from filtering, building detection, and removal of wall points. Second, initial edges are obtained using a Canny detector constrained by buffer areas of edges extracted in the first step. In the final step, roof patches and initial edges are fused using mathematical morphology to form complete

building boundaries. It is stated that the boundary result contains redundancies, which need further simplification. The low point density causes a high number of false negatives and false positives.

Zhao et al. (2016) propose building footprint extraction and regularization using connected components from airborne LiDAR data and aerial images. Building candidates are separated from a LiDAR Digital Surface Model (DSM) using connected operators and trees are removed using NDVI values derived from the image. Building boundary lines are traced by a sleeve line simplification algorithm and are regularized based on the principal building direction. This research identifies different sources of errors like the regularization process, DSM interpolation, and vegetation points near the buildings. Awrangjeb (2016) determined building outlines from point cloud data by boundary tracing and regularization to preserve high detail boundaries and return high pixel-based completeness. Errors occur due to failures to estimate a dominant direction. The method is extended by Akbulut et al. (2018) to smooth jagged building boundaries generated by an active contour algorithm. LiDAR point clouds and aerial images were combined to improve the segmentation quality of the active contour method. Siddiqui et al. (2016) performed a gradient-based approach to extract building outlines from both LiDAR and photogrammetric images. Gradient information obtained from LiDAR height and local color matching is used to separate trees from buildings. Prominent building orientations are regularized based on the assumption that building edges are mainly oriented at 0° (parallel), 90° (perpendicular), 45° (diagonally), 22.5°, or 11.25° to each other. The proposed method is able to deliver consistent results. However, their method is designed to extract buildings with flat and sloped roofs. Xie et al. (2018) presented a method for hierarchical regularization of building boundaries in noisy ALS and photogrammetric point clouds consisting of two stages. First, boundary points are shifted along their refined normal vector and divided into piecewise smooth line segments. In the second stage, parallel and vertical relationships between line segments are discovered to further regularize edges. 2D building footprints extraction was tested on two ISPRS Toronto benchmark datasets and obtained 0.77m and 0.68m RMSE.

Several studies focus on the utilization of LiDAR data as a single input for building extraction and apply regularization to improve the result (Zhao et al., 2016; Siddiqui et al., 2016; Awrangjeb et al., 2014; Dorninger and Pfeifer, 2008; Gilani et al., 2016; Huang et al., 2018; Sampath and Shan, 2007). Regularization is applied to enforce rectangularity and orthogonality of human-made objects. Lach and Kerekes (2008) report on boundary extraction from LiDAR point cloud using 2D α-shapes and apply consecutive regularization. Line simplification based on a sleeve-fitting approach is applied once the edge points are extracted. Then, regularization is used to force boundary line segments to be either parallel or perpendicular to dominant building orientations. In this research, a quantitative analysis and geometric accuracy of the result is not given. Dorninger and Pfeifer (2008) use mean shift segmentation to detect a building and use 2D α-shape generalization to extract initial roof outlines from an airborne LiDAR point cloud.

Based on the angular direction of subsequent line segments and connected linear components of the α-shape, regularization is then applied to enforce orthogonality and parallelism of linear components. Hence, the adjusted building edges are either parallel or orthogonal, and the method is not applicable for a building that has more than two edge directions.

Sampath and Shan (2007) modified a convex hull algorithm to trace building boundaries from raw point cloud data and determine dominant directions. Then, regularization is applied using hierarchical least-squares to extract building outlines such that the slopes of line segments are either equal or perpendicular. However, the regularization quality was found to be dependent on the point density of the LiDAR data and only considers two dominant directions. Gilani et al. (2016) propose building detection and regularization using multisource data which are ALS point cloud data, orthoimages, and Digital Terrain Models (DTM). Candidate buildings are identified using connected component analysis from a building mask generated from ALS data. Building outlines are then detected by hierarchical clustering and filtering. Building footprints are generated using image lines and extracted building boundaries. Regularization begins with the selection of the longest line and it next adjusts nearby lines. The regularized building outlines may deviate from the correct building orientations since the result depends on the selection of longest lines.

A comprehensive review on the use of Hough transforms in image processing and computer vision is presented in Illingworth and Kittler (1988). Mukhopadhyay and Chaudhuri (2015) present a comprehensive and up-to-date survey of Hough transform on various issues of Hough transforms, which even after 51 years of discovery is a lively topic of research and applications. Herout et al. (2013) specifically review the use of Hough transforms for line detection. Morgan and Habib (2002) used Hough transforms from a TIN model of LiDAR point clouds to determine building boundaries. Triangles incident to the building edges (internal breaklines) that connect buildings and ground points are selected and used to obtain triangle centers. These points are fed into a Hough transform to detect lines. Because of limited point density and smaller numbers of extracted triangles on some short building boundaries, the Hough transform detects less building lines than it should. Guercke and Sester (2011) used Hough transforms to simplify and straighten the shape of building footprints extracted from LiDAR data. A jagged building outline is divided into small line segments and is then transformed into Hough space. Line hypotheses are determined based on the dominant direction detected as peaks in Hough space. These hypothesis lines are then refined by least squares to form a closed polygon. The method has problems on buildings with multi-parallel short building edges (stair-like shape) due to peak detection failures.

Iterative Hough transform is proposed (Dalitz et al., 2017) to detect building edges from 3D point clouds. Each line is optimized with an orthogonal least square fit. After a line is found, points belonging to this line are removed and the Hough transform procedure is repeated until no points are left or a sufficient number of lines is found. However, the proposed method has drawbacks such as memory insufficiency and overflow in the accumulator matrix because many points belong to a specific line. Another drawback is a premature stop of the iteration process due to many identical points with the same coordinates, which then yield no line direction due to zero covariance matrix. The Hough Transform suffers from several well-known problems including spurious peaks and quantization effects. Miller (2016) extracted building edges from a DSM generated from a LiDAR point cloud. Each edge is converted to lines using a Hough transform to get the building footprint. Inaccuracy of the extracted building footprint of a large dormer and tall extruding roof structure are mainly caused by rotation and bigger pixel size (down sampling) that ultimately reduced the performance of their approach.

Oesau (2015) proposes a multiline extraction method for shape detection of mobile laser scanner (MLS) point cloud data. 2D line segments are extracted through a Hough Accumulator that combines both a Hough transform and global maxima in a discrete parameter space. However, over-simplification is introduced by the coarse resolution of the Hough Accumulator. Albers et al. (2016) use Hough transform to extract building line segments from airborne LiDAR point clouds. The building edge points are selected by a 2D α-shape algorithm and then repositioned based on energy minimization using three terms: distance estimated line to input points, angle between consecutive lines, and line segment length. The proposed method allows presenting more than one building orientation but is reported that to work for consecutive segments with 45° and 90° angle difference (angle $\in \{45°, 90°, 135°, 180°\}$). Hohle (2017) generates straight building polygons from aerial images using Hough transform. It follows an orthogonality and parallelism scheme by assuming that consecutive building edges are orthogonal.

In summary, obtaining accurate building boundaries are still an open problem. Prior Hough transform works mentioned above have certain limitation either to determine arbitrary building orientation or in accurate peak detection. Thus, the absence of building detection of arbitrary directions from point cloud motivates us to develop an automatic approach to extract building outlines accurately from a given point cloud.

4.3 Methodology

The goal of this research is to obtain the 2D outline or bounding polygon of a building automatically. We propose a hierarchical approach to select accurate lines by generating a point accumulator matrix from ordered building edge points.

The expected result from this research is a set of 2D building polygons in vector format that meet the map following specifications:

1. A building is defined based on a nadir representation of its roof;
2. The building outline consists of straight-lines edges that form a closed polygon;
3. The extracted 2D building outlines shall at least meet the criteria for Indonesian 1:5,000 map scale specifications regarding the positional accuracy and level of detail (SNI 8205, 2015). That is the expected building outline has a positional accuracy of at least 1 m. The minimum building size that must be extracted is equal to 2.5 × 2.5 meter.

We propose a general framework for obtaining building outlines and demonstrate its ability by applying it to different test sets. The general framework of this research consists of five major steps: pre-processing, edge point selection, building line segment detection using Ordered point-aided Hough Transform (OHT), line segment intersection, and 2D closed building polygon extraction from ordered building corners. The whole framework is shown in Figure 4.1.

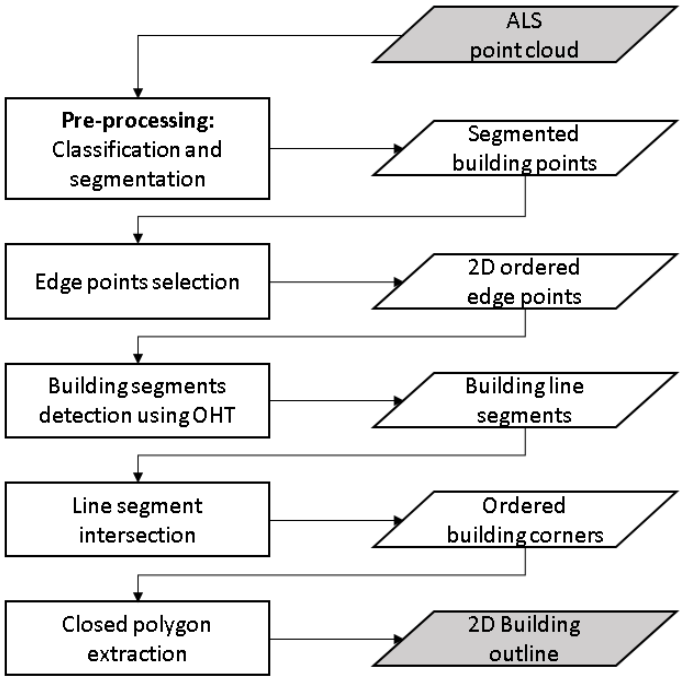


Figure 4.1 The general procedure for extracting high quality straight building outlines.

An overview of the proposed building outline extraction is illustrated in Figure 4.2.

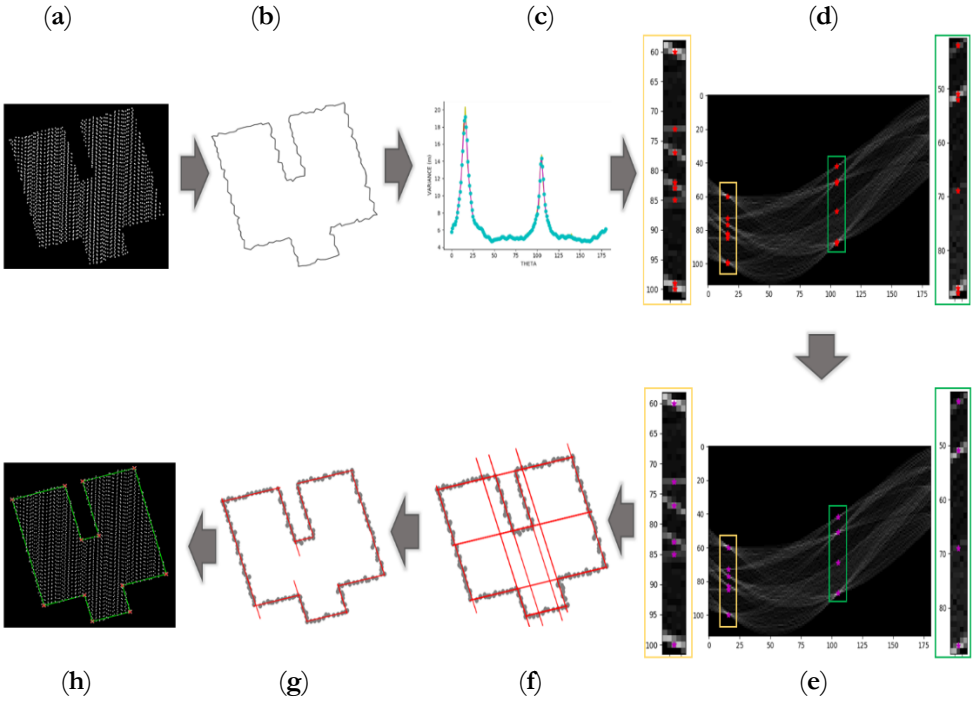


Figure 4.2 The proposed Ordered point aided Hough Transform (OHT) workflow for building outline extraction. (**a**) Building points; (**b**) concave hull of a building roof; (**c**) detection of dominant directions using local maxima; (**d**) detection of initial hotspots along dominant directions yields14 initial hotspots; (**e**) reduction to 10 filtered hotpots; (**f**) 10 lines corresponding to filtered hotspots; (**g**) Point accumulator analysis yields 12 segments; (**h**) segment intersection identifies 12 corners.

The novelty of our method lies in the use of ordered points, which to the best our knowledge has never been used to detect building lines of different length in Hough space. The capability of the proposed method to detect arbitrary building orientations provides another advantage over existing methods. The proposed OHT (approach consists of the following steps:

1. Extract ordered 2D edge points from a given building segment by applying K-NN concave hull;
2. Parameterize all possible lines through the 2D edge points, and store the distances to the origin r of these lines in a matrix R. A Hough accumulator matrix HA counts accumulated points of the same orientation angle θ and distance r;
3. Detect dominant building directions;
4. Identify candidate cells representing prominent lines along dominant directions;

5. Create an Ordered Point List matrix OPL to store lists of ordered points. OPL is then used for detecting and filtering line segments, generating building corners, and forming a closed polygon.

4.3.1 Classification and segmentation

Our method requires ordered building edge points. Only the 2D coordinates of the edge points will be kept to extract building outlines. For this task, we apply the concave hull K-Nearest Neighbor algorithm (Moreira and Santos, 2007) that uses the value of k as the only parameter to control the smoothness of the result. In the beginning, this algorithm finds its first vertex (point A) based on the lowest Y value. Then it will search k-nearest points (for k = 3: point B, C, D), as candidate for the next vertex. Point C will be assigned as the next vertex if it has the largest angle of right-hand turn measured from the horizontal line through point A. In the next step, the k nearest points of point C are queried, and the selected vertex is appointed once it has the largest angle of right-hand turn from line A–C. The process is repeated until the first vertex is selected once again as candidate.

Higher k will lead to smoother polygons. In this research, the k values vary from 3 to 11 depending on the point density. A building with irregular point intervals may require higher value of k to derive suitable edges.

4.3.2 Hough accumulator matrix

The key idea of the Hough transform is to select straight-line candidates based on a voting scheme in a parameter space. To parameterize a line, we use two parameters: distance to the origin, r, and orientation angle, θ. The mapping relations of a point in object space (x, y) and (θ, r) parameter space is specified in Equation 4.1.

$$r = x\cos\theta + y\sin\theta \tag{4.1}$$

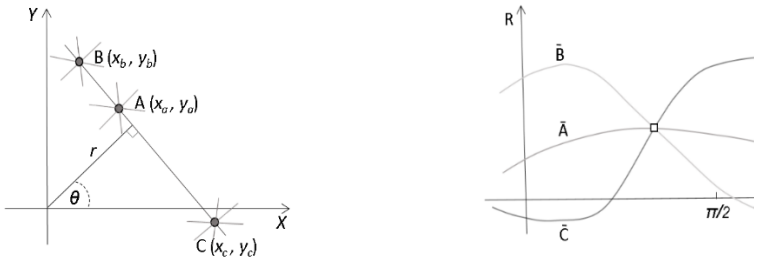
The chosen polar (θ, r) parameters are advantageous over the slope-intercept (m, c) parameterization since the (m, c) may have a singularity when the slope of the line is infinite. The number of rows and columns of the matrix is adjusted according to the bin sizes of the two parameters. Each cell contains a number of lines having θ and r values.

As illustrated in Figure 4.3.a, one line defined by a pair (θ, r) may contain different building edge points, here A, B, and C. The fan of all lines passing through one point (Figure 4.3.a) corresponds to a sinusoidal curve in Hough space (Figure 4.3.b), while each point in Hough space corresponds to one straight line in object space.

For a given set of n ordered LiDAR points $P_i = (x_i, y_i)$, for $i = 1,2, \dots, n$ forming a building roof boundary, a matrix R containing line distance r parameter values is created as follows:

1. Fix origin *O* at $(min\ x_i, min\ y_i)$ for (x_i, y_i), $i = 1,2, \dots, n$ and initiate a matrix R with dimension $(n \times 181)$;
2. For point P_i and for each $\theta\epsilon\{1°, 2°, \dots, 180°\}$, determine $r_i(\theta)$ using Equation (3.1);
3. Store r_i in matrix R at position (i, θ).

Matrix R stores r-values for lines of different angles (0° to 180°) through each point P_i. Next, it will be considered if points share lines. Lines common to many points are likely to define a building edge. To identify common lines, i.e., lines defined by different points but sharing the same (θ, r) values, the Hough Accumulator (HA) is created next. HA requires binning of r and θ to represent the location of its cells. Each cell in HA stores the number of points with matching θ and r. Different cells in HA represent different combinations of θ and r values. The higher the number in a HA cell, the more likely the cell produces a correct line for an edge.



(**a**) (**b**)

Figure 4.3 Hough transform using (θ, r) parameters for detecting lines. (**a**) In object space, a line represented by an angle-distance (θ, r) passing through three points *A*, *B*, and *C*; (**b**) In Hough space, this line appears as the point of intersection of the curves $\bar{A}$, $\bar{B}$, and $\bar{C}$.

HA is a 2D array of size (number of $bin_r \times 180$). The trade-off between the number of bins of the matrix and the number of available observations is crucial. Too many bins may lead to a sparse representation of the density that will decrease the ability to detect prominent lines. On the other hand, too few bins will reduce the resolution and accuracy of the building line results. We recommended that the bin width of r, (bin_r), is set according to the average point interval.

4.3.3 Detection of arbitrary building directions

Building shapes and other man-made objects are often characterized by certain geometrical regularities (Hohle, 2017), mostly appearing as perpendicularity or parallelism. However, in a reality, not every building is constructed using such geometrical regularities. Therefore, determining possibly arbitrary building direction is an important strategy for the building extraction process.

One limitation of Hough Transform is that when the number of lines increases, the correlated error around the peak in the parameter space could cause ambiguities for line (Leavers, 1992). To limit the search space for selecting line candidates, the proposed algorithm uses local maxima detection instead of global maxima. Local maxima are detected by identifying peaks in the graph of the variance of θ along the columns of the matrix HA. The variance is defined as the average of the squared deviations from the mean number of lines along the θ column of matrix HA. Finding local maxima means to detect accumulator cells that have higher vote than their neighborhood (peak). For detecting peaks, we first apply a Savitzky Golay filter (1964) to denoise the data. The basic idea of Savitzky Golay filtering is to replace each point by the corresponding value of a least squares fit of a low order polynomial fitted to points in a window centered at that point.

In the smoothed variance data, peaks are detected if they meet two criteria: normalized threshold (*amp*) and minimum distance (*mindist*) between each detected peak. The normalized threshold will select peaks with higher amplitude than the threshold. Figure 4.4 illustrates the one-dimensional peak detection from a smoothed variance input (magenta graph) to define the direction. In most cases, the difference between two dominant building directions is close to 90°.

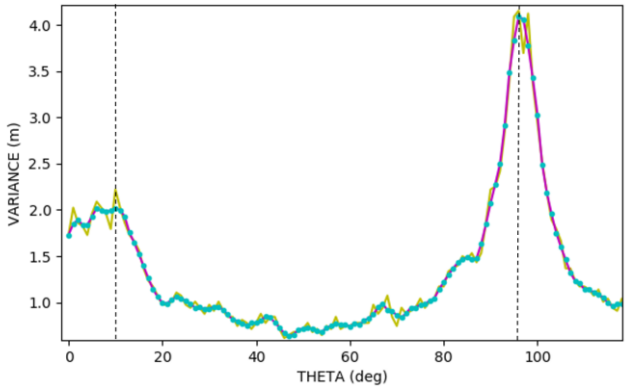


Figure 4.4 Peak detection in smoothed data. The yellow graph represents the original variance and the magenta line represents smoothed data obtained after Savitzky Golay filtering. Vertical dashed-lines indicate detected dominant directions.

4.3.4 Hotspot selection

After dominant directions are detected, the algorithm will next search along the corresponding columns for cells in matrix HA that have at least *minL* edge points. These cells are then preserved as initial hotspots. An initial hotspot is a candidate cell to represent a building line. The minimum edge length (*minL*) parameter is set based on the required building length, ℓ, as $minL = \ell/d$, where d denotes the point interval. If, for example, the point interval is 0.5 m and the minimum length of the building edge to be extracted ℓ is 2.5 m, the required threshold $minL = 5$.

However, in HA, different column adjacent hotspots (one cell difference) may represent lines belonging to the same edge. This happens because some noise from the same edge points result in slightly different (θ, r) combinations. Hotspot filtering is applied by searching for any adjacent hotspots along a dominant direction. Only the hotspot that has maximal $HA_{(i,j)}$value is kept. Figure 4.2.d and Figure 4.2.e, respectively, present the result of initial hotspot detection and hotspot filtering.

4.3.5 Ordered point list

One of our main contributions is to extract building outlines using a so-called Ordered Point List (OPL) matrix. This matrix is generated based on the classic Hough accumulator. OPL has the same dimension as HA and the same (θ, r) parameters. The difference is that HA stores just the number of accumulated points voting for a line while OPL stores the actual ordered lists of points voting for a line. This means that each cell in $OPL_{(\theta,r)}$ contains an ordered list of points P_i that are on the parametric line $r = x\cos(\theta) + y\sin(\theta)$

To obtain more accurate and complete building edges, in OPL, the contents of each filtered hotspot is merged with its adjacent cells of the same column ($\Delta bin_r = 0$ and $\Delta bin_r = 1$). Matrix OPL_m contains the merged point members. The difference cell value between HA, OPL, HA_m, and OPL_m of specific (θ, r) is llustrated in Figure 4.5. Points accumulated in HA (Figure 4.5.a) are specified in OPL (Figure 4.5.c). The red cell marks one of the hotspots. Point members of hotspot cells of merged OPL_m (a red cell in Figure 4.5.d) are adapted by include neighboring cells. As an example, $OPL_{m(105,87)}$ has 11 additional points that complement the existing ordered point members of $OPL_{(105,87)}$.

4.3.6 Segment detection and filtering

Our algorithm yields arbitrary main directions that are used to select prominent lines. For some buildings with a complex shape, a false main direction may get detected because the corresponding θ has the highest vote in HA. Therefore, we apply hierarchical filtering to eliminate false lines resulting from a wrong main direction.

Using filtered hotspots and the merged Point Accumulator matrix OPL_m as main input, the algorithm measures the distance of each point belonging to a filtered hotspot, to the hotspot line parameterized by the pair (θ, r). Then, it counts the number of hotspot points that have a distance more than bin_r. A hotspot will be removed from the list if one of two following conditions holds:

- It has at least 3 points having a distance more than bin_r;
- It has a mean distance *d* bigger than half bin_rvalue.

This mean distance threshold is set based on an empirical observation for detecting false lines and distinguishing those from correct but noisy lines.

A line resulting from the traditional Hough transform cannot distinguish different segments belonging to the same line. In this case, two different building edge segments that share the same (θ, r) will not be detected. The proposed edge extraction algorithm requires segments, instead of lines, for producing a closed polygon. In this research, a segment is defined as a part of a line that is bounded by two building corner points. Note that a building line may contain more than one segment.

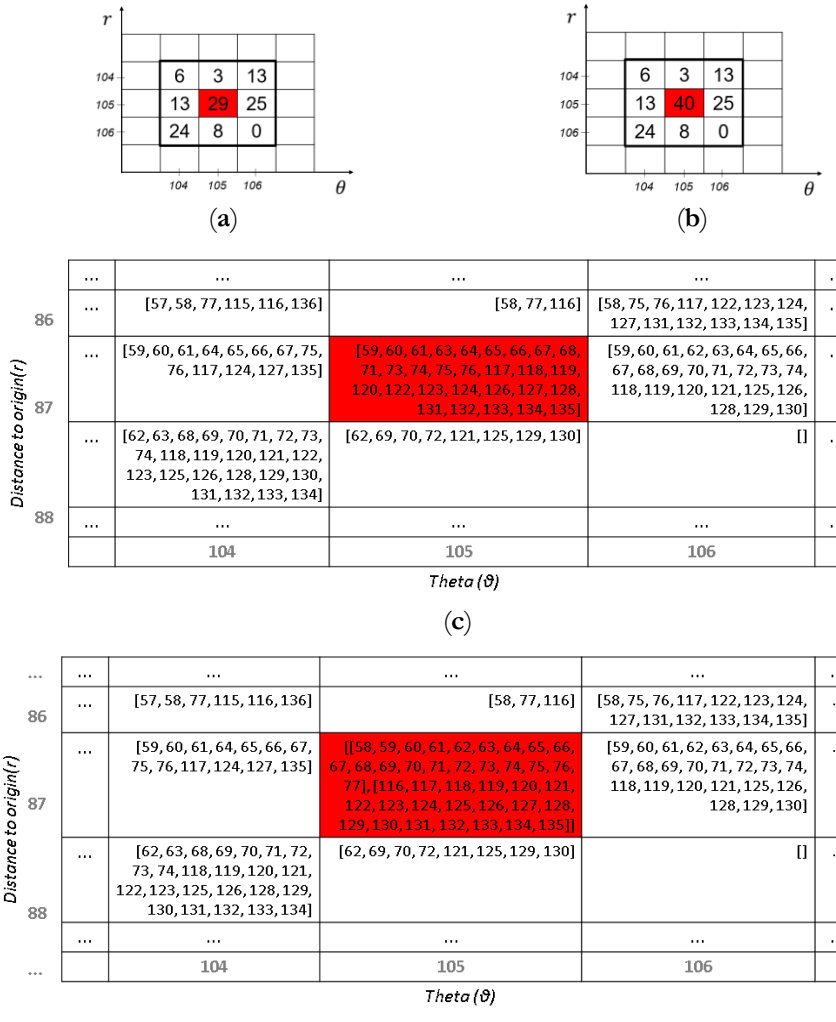
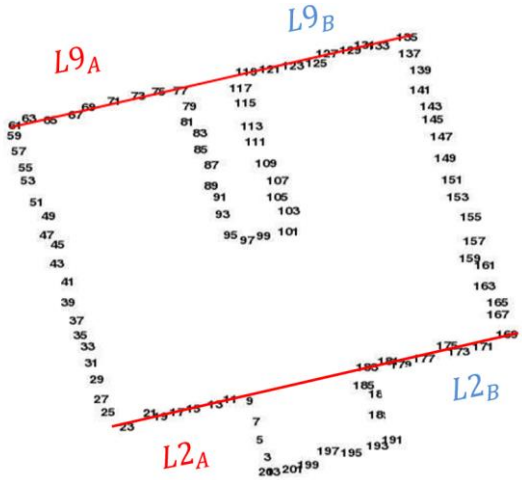


Figure 4.5 Cells in HA, OPL, HA_m, and OPL_m for r= 86–88 and θ= 104°–106°. Red cells correspond to hostspots. (**a**) $HA_{(\theta,r)}$cell containing accumulated numbers of points voting for a line; (**b**) $HA_{m(\theta,r)}$ cell containing merged accumulated number of points from its adjacent cell; (**c**) $OPL_{(\theta,r)}$ cell containing lists of points voting for the same line; (**d**) $OPL_{m(\theta,r)}$cell containing lists of merged points from its adjacent cell.

Segment detection exploits ordered edge points stored in OPL_m. Each segment has a θ, an r, and a number of ordered points. Multiple segments on a line are identified by a gap or point jump between ordered points in the list. We set the gap threshold to 2 (two points jump). In this case, a gap is detected if there are at least two consecutive points missing from the list of ordered points.



(**a**)

$L1$	[0, 1, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202]
$L2$	[[8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23], [168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183]]
$L3$	[22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61]
$L4$	[40, 41, 94, 95, 96, 97, 98, 99, 100, 101, 150, 151, 152]
$L5$	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 72, 73, 201, 202]
$L6$	[76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 197, 198, 199]
$L7$	[100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 192, 193]
$L8$	[118, 119, 120, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192]
$L9$	[[58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77], [116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135]]
$L10$	[134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169]

(**b**)

Figure 4.6 Representation of ordered edge point distribution of a building. Lines are separated in segments by gap identification. (**a**) Line $L9$ (red) consists of two segments: $L9_A$ and $L9_B$; (**b**) List of points supporting the same line. The line points in $L2$ and $L9$ are both divided over two segments, indicated by red and blue numbers.

An example of segment detection is illustrated in Figure 4.6. The red line in Figure 4.6.a consists of two segments as a gap exists in the list of ordered points ($L9$) as presented in Figure 4.6.b. Line L9 has two different segments. Point members of segment $L9_A$ are marked in red (from 58 to 77), while the point members of segment $L9_B$are marked in blue (from 116 to 135). The two segments are separated by a large point interval. A segment is then identified and selected as a list containing a minimum number of points *min*L. The minimum segment length is adjusted according to the output requirements. As an example, $L1$ in Figure 4.6.b consists of two lists of ordered points separated by a big gap. Nevertheless, it will return one segment only because the first list of ordered points only has two members, points 0 and 1.

After all segments are identified, matrix OPL_m. contains different segments. These segments are sorted based on the first element of the list of points. Finally, segment filtering is performed. This last filtering step is needed to remove false segments that may remain in the result. A shorter segment having all points the same as a longer segment will be removed if one of two following conditions holds:

- The first point is not assigned as the last point of a longer segment;
- The last point is not assigned as first or last point of a longer segment.

Lists of points are input for corner extraction. First, all building segments are sorted based on their lowest participating point label. Then, the algorithm extracts intersections from consecutive segments. A closed building polygon is formed by connecting all consecutive segment intersections.
From two given parametric lines,

$$r_1 = x\,cos\theta_1 + y\,sin\theta_1 \tag{4. 2}$$

$$r_2 = x\,cos\theta_2 + y\,sin\theta_2 \tag{4. 3}$$

The intersection point $c(x_c, y_c)$ is computed as

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} \begin{bmatrix} cos\theta_2 & sin\theta_2 \\ cos\theta_1 & sin\theta_1 \end{bmatrix} \tag{4. 4}$$

Table 4.1 summarizes different kind of point and line representations used in our proposed method. A point in object space is translated into a row in R, as a curve in HA, and as part of a list in OPL. Lines and segments in object space are represented by a subset of a column in R and as a cell in HA. A cell in HA shows the number of points belonging to the same line, while a cell in OPL shows the list of points of a line. Hence, in OPL, lines and segments are represented by list of several points.

Table 4.1 Summary for object representation of different matrix.

Object Space	Matrix R	Matrix HA	Matrix OPL
Point	Row	Curve	Part of a list
Line	Subset of column	Cell / Point	Long list
Segment	Sub of subset	Cell / Point	Long list

4.3.7 Validation

Two different quantitative analyses are performed to evaluate the result of the building outline extraction: performance metrics and positional accuracy. The ground truth used as reference to assess our results is described in Section 4.4. We used three performance metrics (Rutzinger et al., 2009) to evaluate the building polygon results, completeness (*Cp*), correctness (*Cr*), and quality (*Q*). The performance metrics are calculated based on an area comparison of buildings in the reference data and in the result in the unit m².

The positional accuracy is a geometric validation that evaluates if the quality of the extracted building polygons meets the geometric accuracy criteria. The positional accuracy is determined using a Root Mean Square Error (RMSE) value. The squared root of the average of the squared differences between corner positions (X and Y coordinate) in the reference and in the result is calculated to estimate the RMSE.

$$RMSEx = \frac{\sqrt{\sum(X_{res} - X_{ref})^2}}{n} \tag{4.5}$$

$$RMSEy = \frac{\sqrt{\sum(Y_{res} - Y_{ref})^2}}{n} \tag{4.6}$$

$$RMSEr = \sqrt{RMSEx^2 + RMSEy^2} \tag{4.7}$$

Where

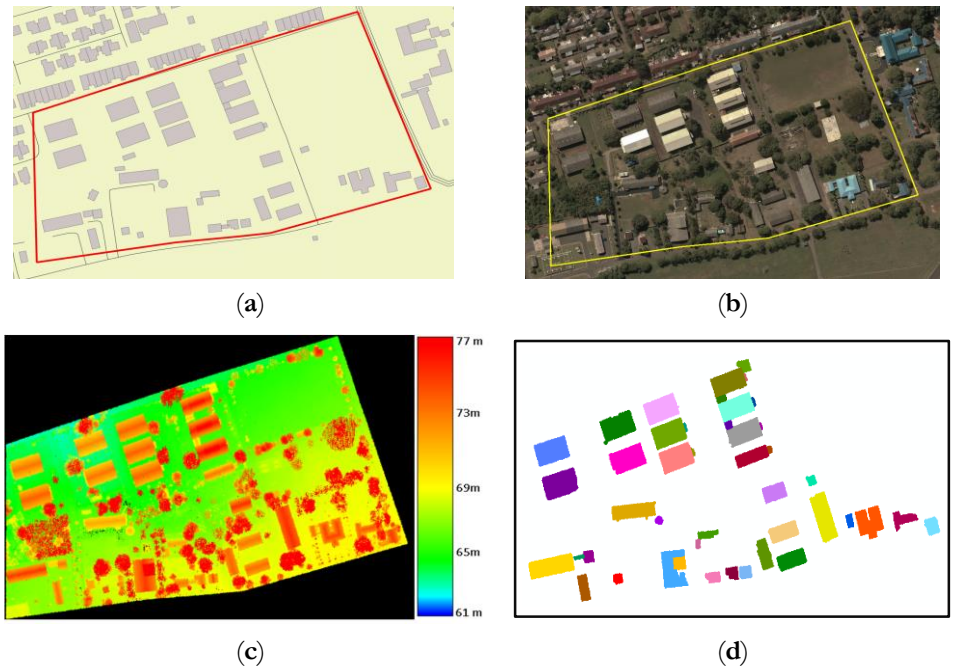
X_{res}, Y_{res} = Coordinates of resulting corner points
X_{ref}, Y_{ref} = Coordinates of corner points in the ground truth
n = Total number of corner points

4.4 Test area and preprocessing

4.4.1 Test set Makassar

The test set is located in a newly built sub-urban area of Makassar city (Figure 4.7.b), Sulawesi island of Indonesia. The LiDAR point density is between 8–10 points/m² (ppm) and was acquired in 2012 using a Leica ALS70. The total study area is 1.2 km² (Figure 4.7.c). A topographic base map in vector format is used as ground truth (Figure 4.7.a).

The LiDAR data we use has already been filtered into ground and non-ground points using TerraScan software. The software implements a Progressive TIN Densification, originating by Axelsson (2000), to filter non-ground points. From the non-ground points, the building roof points are separated from the tree points using two thresholds: point distance to planar surface (0.2 m) and minimum segment size (30 m^2).



(**a**) (**b**)

(**c**) (**d**)

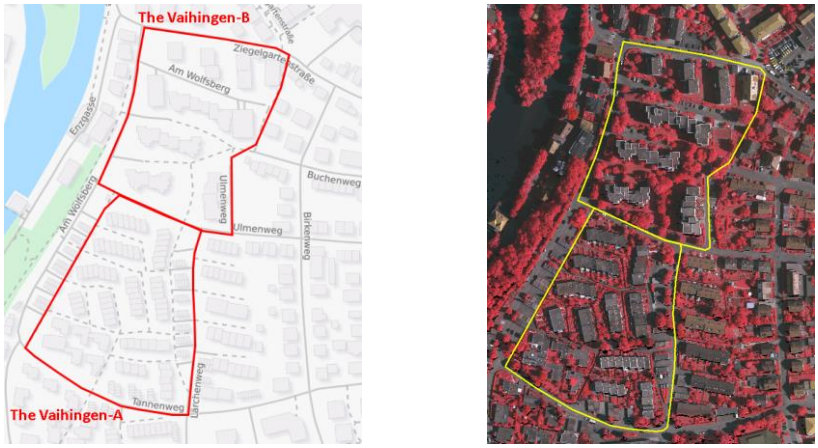
Figure 4.7 Test set of Makassar. (**a**) Test set area (inside red outline) overlaid to Indonesian 1:10.000 base map; (**b**) Test set areas (inside yellow outline) overlaid to aerial orthoimage. (**c**) ALS point cloud of the Makassar test set colored by height; (**d**) Clustered building points. Different color indicates different segment. ©BIG.

The 3D building points as output by TerraScan are then segmented into different clusters using the DBSCAN algorithm. DBSCAN segmentation (Ester et al., 1996) requires two parameters: radius distance (*eps*) and minimum number of points (*minPts*). To find a segment, DBSCAN starts with an arbitrary seed point *p* and then retrieves all neighboring points (density-reachable) from p that are located within a given *eps* and contains a given *minPts*. Outliers are defined once *minPts* cannot be achieved within the given *eps*. The cluster will grow as long as nearby points within the *eps* distance from seed p fulfill the *minPts* threshold. In case *minPts* within distance *eps* is not fulfilled, a point or group of points is considered as outlying. During the cluster growing, outliers may change into a member of one of the clusters once they are within the *eps* distance from the active seed point. To grow the next cluster, a next seed that does not belong

to any cluster is chosen. The clustering stops once all points are assigned. The parameter thresholds used for the Makassar dataset are *eps* = 1.2m and *minPts* = 3. This means that the required minimum number of points assigned as a cluster within 1.2 m from the seed points is three points. The segmented 3D building points resulting from DBSCAN after size filtering are shown in Figure 4.7.d.

4.4.2 Test set Vaihingen

The second area of study belongs to the Vaihingen test set provided by the ISPRS (International Society for Photogrammetry and Remote Sensing). The LiDAR point density varies between 4–7 points/m² (ppm). This data was acquired in August 2008 by a Leica ALS50 airborne LiDAR system. There are two sub areas, Vaihingen-A and Vaihingen-B, as presented in Figure 4.8. The Vaihingen-A dataset consists of residential buildings of complex shape surrounded by trees. Ground truth for Vaihingen-A comprises a set of building references from OpenStreetMap (OSM), confirmed by true orthophotos provided by the ISPRS. The Vaihingen-B test set, which is basically the same dataset as Vaihingen Area 2 as described on the ISPRS webpage, is characterized by complex high-rise buildings that have several roof layers at different height. This benchmark dataset is chosen and used by several similar studies (Zhao et al., 2016; Siddiqui et al., 2016; Gilani et al., 2016; Huang et al., 2018) to test and compare their algorithms. For Vaihingen-B, we use 2D building outlines in vector format as provided by ISPRS as ground truth.



(**a**) (**b**)

Figure 4.8 The test sets of Vaihingen. (**a**) Test set areas (inside red outlines) overlaid to OSM map; (**b**) Test set areas (inside yellow outlines) overlaid to ISPRS orthoimage. ©OSM and ISPRS.

For the Vaihingen-A test set, we use the surface growing function of Point Cloud Mapper (PCM) developed by Vosselman et al. (2004) to perform point cloud segmentation. Surface growing thresholds implemented in this study are as follows:

- Seed surface selection. At least 10 points out of 20 nearest points within a 1-m radius are used to check for neighborhood planarity and apply a 3D Hough transform. Seeds are extended to other points located within a 1-m radius with a height difference of less than 20 cm to the fitted plane. The bin sizes of the distances and angles of the 3D Hough transform are set to 20 cm and 3 degrees;
- Growing expansion. Once a seed segment is found, region growing looks for adjacent points belonging to the same plane. Points are assigned to a plane if the distance to the corresponding plane is 50 cm maximum.

A different segmentation method is applied for Vaihingen-B. First, the point cloud is classified using the LAStools software developed by Rapidlasso. We then preserved only the planar points using plane detection to remove remaining tree points. The planar points are then segmented using DBSCAN (*eps* = 1.2m and *minPts* = 3). The test set of the Vaihingen-A and the Vaihingen-B, as well as the segmentation and classification results (Figure 4.9).

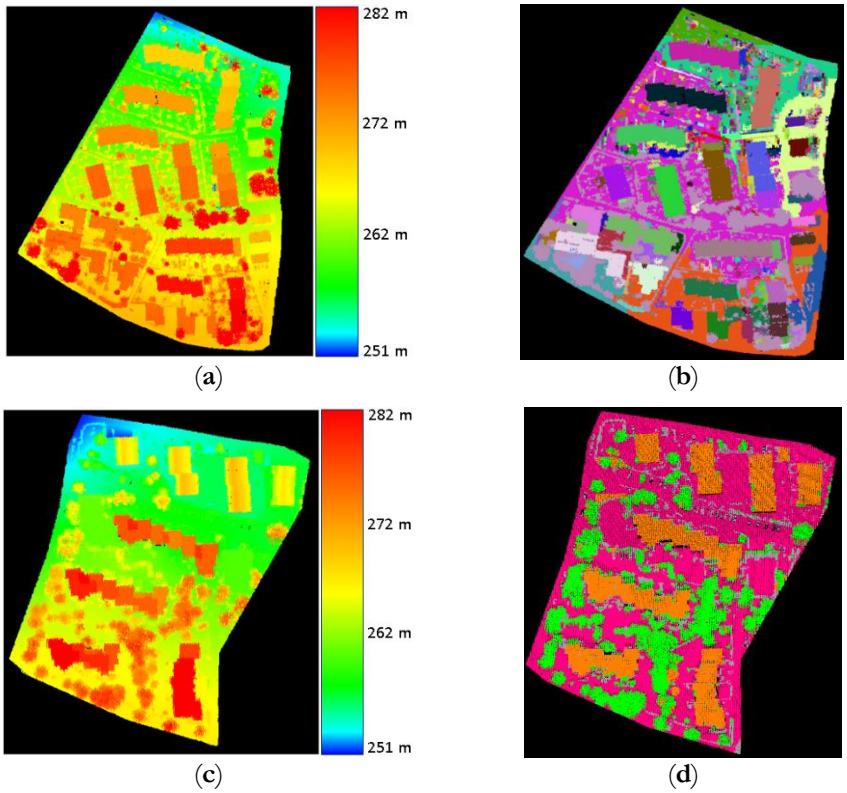
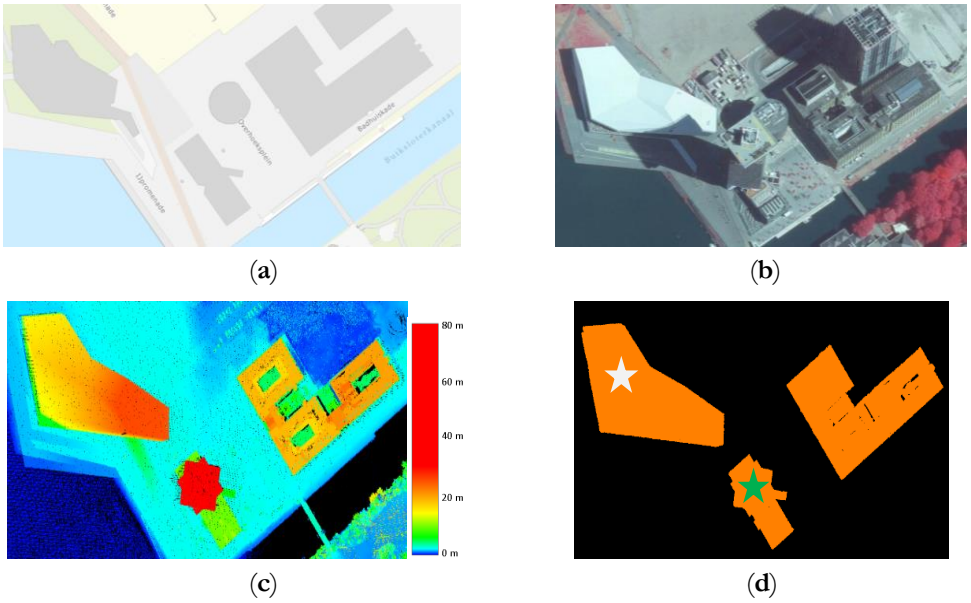


Figure 4.9 Two test sets of Vaihingen benchmark dataset. (**a**) ALS points of Vaihingen-A; (**b**) segmented points of Vaihingen-A; (**c**) ALS points of Vaihingen-B; (**d**) classified points of Vaihingen-B. ©ISPRS.

4.4.3 Test set Amsterdam

An additional point cloud dataset sampling the EYE-Amsterdam neighborhood (Figure 4.10.a) demonstrates the ability of our algorithm to extract complex buildings of multiple arbitrary directions. We use an open source AHN3 point cloud dataset downloaded from PDOK, the Netherlands. Point clouds of AHN3 acquired in 2014 are already classified into several classes (ground, building, water, etc.). The AHN3 classified building points of our EYE-Amsterdam study area is shown in Figure 4.10.d.



(**a**) (**b**)

(**c**) (**d**)

Figure 4.10 EYE-Amsterdam test set. (**a**) Map of the EYE-Amsterdam; (**b**) 2017 Aerial image of the EYE-Amsterdam; (**c**) 2014 AHN3 point cloud; (**d**) AHN3 classified building points (orange). ©PDOK of the Netherlands and ESRI-NL.

4.5 Sensitivity analysis and experiments

This section describes common issues that highlight specific features of our algorithm to answer the research objectives. The last paragraph presents a sensitivity analysis of the parameters used.

4.5.1 Detecting multiple arbitrary direction

The EYE building (as indicated by a white star in Figure 4.10.d) has a unique shape, which makes it impossible to apply perpendicularity rules. The A'DAM Lookout building (as indicated by a green star in Figure 4.10.d) is another complex building in this area with five building directions of different edge length. The ability of our proposed algorithm for extracting outlines of such complex building shapes is illustrated

in Figure 4.11. Based on the detection of multiple arbitrary building directions (Figure 4.11.a), the building segments are identified (Figure 4.11.b). As the final output, based on corners of consecutive building segments, a closed building polygon is formed (Figure 4.11.c).

Herout (2013) stated that the accuracy of the Hough Transform is strongly dependent on the detection of maxima in parameter space. As local maxima for detecting arbitrary building directions depend on the number of votes, it is possible that edge points of a complex building dominantly vote for an incorrect building direction. This situation may cause the algorithm to detect false building lines that are supported by many edge points but are not part of any building segment.

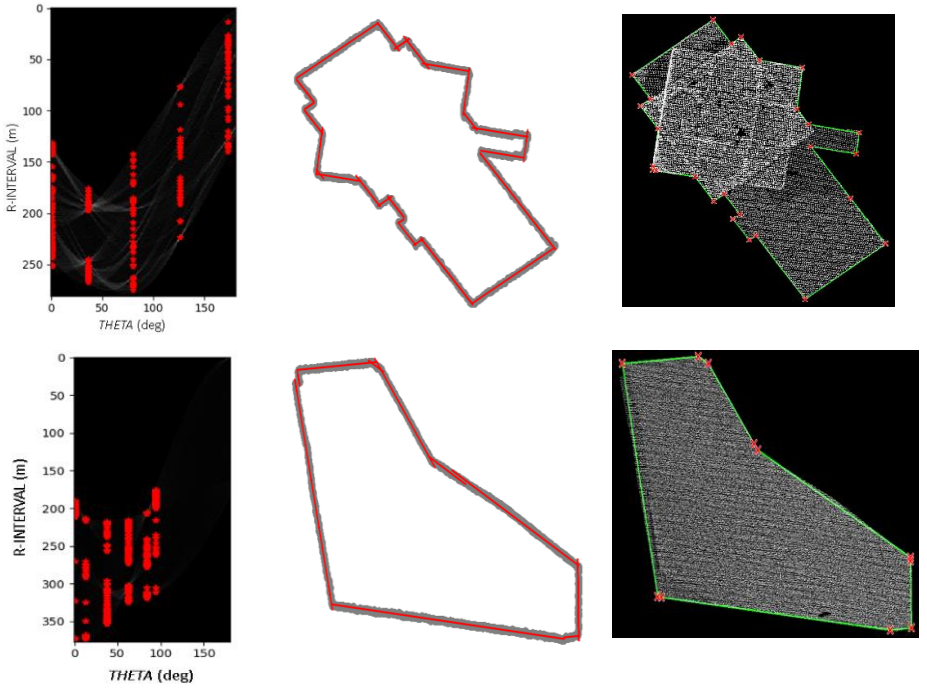


Figure 4.11 Extraction of buildings with multiple arbitrary directions. (**a**) Hotspots (red points) of multiple arbitrary directions; (**b**) Line detection; (**c**) Building outlines results.

Figure 4.12 illustrates the detection of a false dominant direction resulting in a false building line. The false main direction is likely caused by a special building geometry shape that allows most of the edge points from two different directions to vote for the same direction. In this case, parallel short building edges result in false lines. However, by adding a filtering step to our workflow, the algorithm is able to eliminate and remove these false lines. In Figure 4.12.a, initial hotspots are detected based on three dominant

directions at $\theta = 7°$, $\theta = 78°$, and $\theta = 96°$. The yellow line, at $\theta = 78°$, marks the falsely detected dominant direction corresponding to the false green lines in Figure 4.12.b. All lines corresponding to initial hotspots are shown in Figure 4.12.b, where false lines are shown in green and correct lines in red. Figure 4.12.c shows the results of our proposed segment filtering procedure.

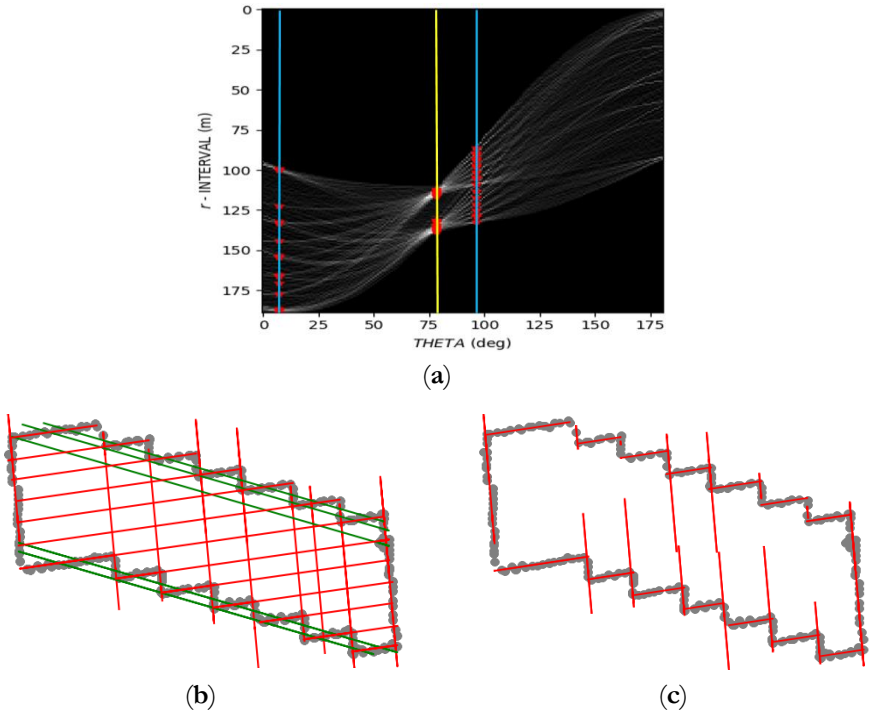
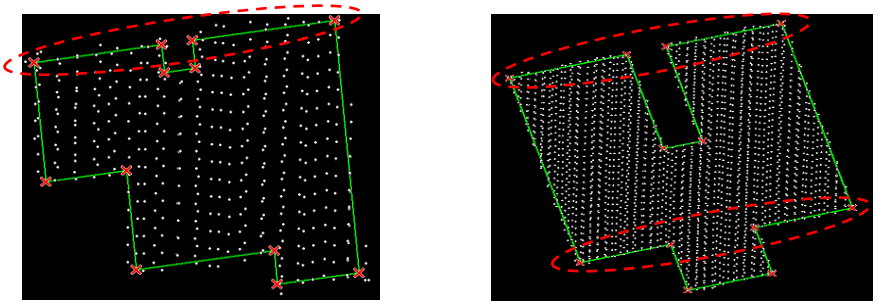


Figure 4.12 Outline extraction of a building with one false dominant direction. (a) Initial hotspots (red) in three dominant directions are detected where the highest peak (yellow line) is a false dominant direction; (b) Representation of all initial lines including five false lines (green); (c) Line segments after filtering.

4.5.2 Extraction of different interrupted segments of different length

One drawback of traditional Hough Transform is its difficulty to distinguish different segments that are collinear as any segment with the same (θ, r) parameters will be considered as the same line (1996). Moreover, in the application of traditional Hough transform, short segments only result in low peaks, which makes them difficult to identify (Lee and Kweon; 1997; Guerreiro and Aguiar, 2012). Because of this limitation, it is a problem to identify short building edges especially if long edges exist of different direction.

Figure 4.13.a and Figure 4.13.b, respectively, show that building outlines of different length and building outlines of collinear line segments (as indicated by red ellipses) are correctly extracted using the proposed method. By exploiting gaps in the ordered edge points of each line, segments that are collinear can be detected and separated.



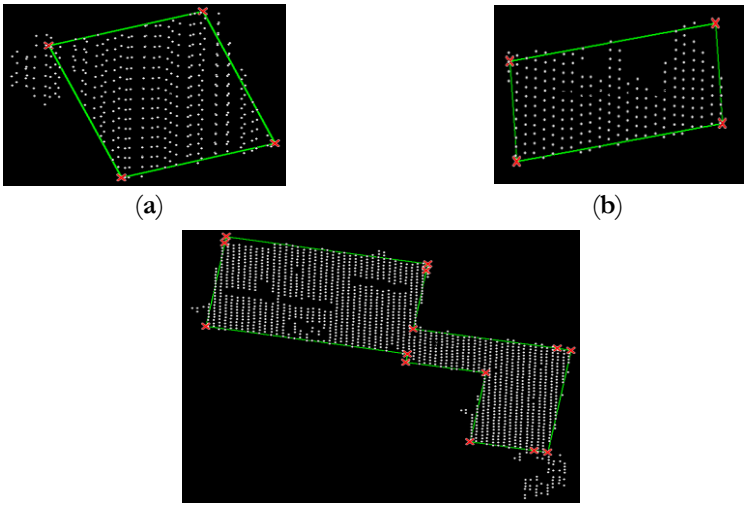
(**a**) (**b**)

Figure 4.13 Building outline result of different segment lengths with two collinear segments. (**a**) Outline extraction of a building that has two collinear line segments (inside the red ellipse); (**b**) Outline extraction of a building that has four collinear line segments (inside the red ellipse).

4.5.3 Robustness to noise and irregularity

In general, our method has the ability to create correct outlines of noisy edge points and straighten up such imperfect building edges. Adjacent trees may cause flaws or irregularities in building segmentation results. This situation, later on, may induce an incorrect building outline. Our proposed method eliminates and removes the influence of such building irregularity. Figure 4.14.a and 4.14.c, respectively, show the outline results in case of over-segmentation for a rectangular and complex building shape by using the proposed method. Both buildings have flaws due to connected trees. In such cases, the algorithm has two possibilities to produce a correct building outline. First, due to sparse and irregular distribution of tree points, the number of edge points of the tree is less than the minimum length edge ($minL_e$) threshold. Second, in case the number of edge tree points is more than $minL_e$, the extracted line segments of the tree edges do not form a fully closed polygon. As the proposed algorithm uses consecutive lines based on ordered points to extract the corresponding corners, extraction of false corners is avoided.

The capability of our algorithm to deal with irregularities may extend to an incomplete building roof case (under-segmentation). Figure 4.14.b shows the outline of a building roof that is partially covered by an adjacent tree. Correct lines are still obtained although there is a gap consisting of missing edge points in the middle building edge part.



(**a**) (**b**)

(**c**)

Figure 4.14 Building outline results in case flaws exist in the segmentation results. (**a**) Outline of a building connected to a tree; (**b**) Outline of a building roof partially covered by a tree; (**c**) Outline of a complex building shape connected to a tree.

4.5.4 Sensitivity analysis

We have implemented a data-driven method that may require parameter tuning for areas of different characteristics. Parameters need to be set for using the proposed procedure are discussed as follows:

- **Bin size:** to determine cell size and distribute the edge points in a specific row (bin_r) and column (bin_θ). We fix bin_θ at (1 degree), while the size of the bin_r is variable, depending on the point density. For the Makassar and the Vaihingen datasets, bin_r varies between 0.3 to 0.6 m.
- **Local maxima:** to detect dominant building directions. Two parameters to determine dominant directions are the amplitude (*amp*) and the minimum distance (*mindist*) between each detected local maximum. The recommended threshold for *amp* is 0.5 m and *mindist* is 30°. In case of complex building shapes and noisy data, an *amp* threshold between 0.15–0.2 m and a *mindist* value between 15°–30° is recommended.
- **Minimum number of points:** to determine initial hotpots. The *minL* value is determined based on the length of the building edge that is required to be extracted. We applied a smaller edge threshold for the Vaihingen test set as this dataset has complex buildings with many short edges (± 1.5 m). The edge threshold for Vaihingen is set to 3 points. For AOI-1 Makassar, *minL* is set to five points. This threshold is set as the minimum length of a building segment length that needs to be extracted is 2.5 m.

Recap of parameter setting used in this research is presented in Table 4.2.

Table 4.2 Parameter settings for different test sets.

Test set	bin_r	amp	*mindist*	*minL*
Makassar	0.3–0.6 m	0.5 m	30°	2.5 m
Vaihingen-A	0.3–0.6 m	0.15–0.2 m	15–30°	1.5 m
Vaihingen-B	0.3–0.6 m	0.15–0.2 m	15–30°	1.5 m

Each Hough Accumulator matrix HA cell contains the number of edge points that vote for the same (θ, r) value. Bin size determination is a crucial step that influences the point distribution of edge points into each cell in HA. If the bin size is set too big, it may result in the coinciding of different edges or less accurate line results. On the other hand, if we set the bin size too small, some particular building edges may not be identified. The bin size for θ (bin_θ) is set to 1°. Experiments show that the determined bin_θ is sufficient to find line candidate. For point clouds with a point interval between 0.3 and 0.7 m, we set the bin_r size to 0.5 m.

We conducted an additional experiment to identify a sufficient bin_r value as well as to evaluate the sensitivity of the bin_r parameter. Two different building shapes (simple and complex) with a point interval from 0.4 to 0.7 m are used for this experiment. As simple building case, a building with a rectangular roof with 4 lines/corners is used. As complex case, we select a building with 24 lines/corners. The results of the experiment are presented in Table 4.3.

Table 4.3 Evaluation of different bin size.

Bin size (cm)	Mean	Std dev	No. of Main Directions	No. of Resulting Segment	No. of Remaining Points	RMSE (m)
Simple Building (four lines/corners)						
20	0.071	0.043	3	6	5	0.224
40	0.126	0.105	2	4	0	0.206
50	**0.121**	**0.101**	**2**	**4**	**0**	**0.198**
60	0.167	0.161	2	4	0	0.223
80	0.223	0.195	2	4	0	0.248
Complex Building (24 lines/corners)						
20	0.086	0.059	3	11	62	N/A
30	0.140	0.109	4	17	19	N/A
40	**0.153**	**0.133**	**3**	**24**	**0**	**0.203**
50	0.192	0.168	3	28	0	0.346
60	0.259	0.206	3	24	0	0.362
70	0.284	0.248	3	33	0	0.473
80	0.363	0.288	3	36	0	0.927
90	0.390	0.356	3	26	6	N/A

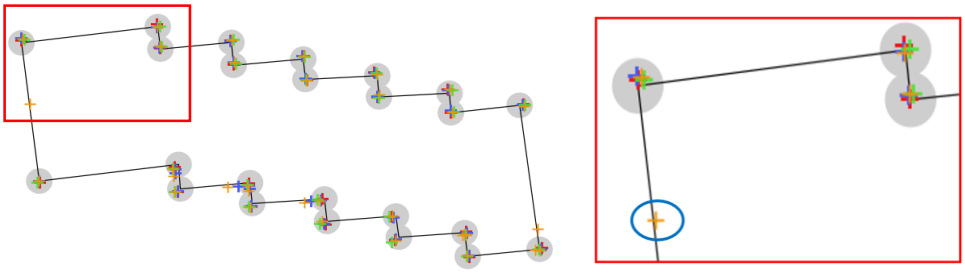
For the experiment, all required parameters have the same value except bin_r. The results indicate that the smaller the bin size, the smaller the average of standard deviation and the RMSE. However, at the same time, the possibility to produce additional segments or corners that are not always correct increases. We also present the total number of edge points that is not used by any line to notify any incomplete or suspicious line result. In Table 4.3, a simple (square shape) building with bin_r = 30 cm and bin_r = 20 cm yields three dominant directions that could result in false lines. However, for bin_r = 30 cm, our method is able to eliminate these false lines by obtaining four building line segments with acceptable RMSE. Meanwhile, for bin_r = 20 cm, our method fails to deliver a complete building outline, as five points remain unused.

The sensitivity of bin_r is increasing for a complex building. As this building has many line segments with significant length difference between the short and long segments (short edges have 3-4 points, while longer edges have 12–20 points), the determination of bin_r affects the line segments results (number of extracted segments and RMSE). Moreover, the small distance between several parallel building edges yields false dominant directions. In Table 4.3 a bin_r size that is smaller or bigger than the point interval (which is between 40 and 70 cm) results in an incomplete building outline, which is indicated by the presence of remaining points or no RMSE result (N/A).

The distribution of resulted segment intersections for different bin sizes (from bin_r = 40 cm to bin_r = 80 cm) is illustrated in Figure 4.15.a. Different intersections of different bin sizes are indicated by different colors. Grey circles indicate a 1-m buffer of reference building corner. Some intersections located in the middle of building edge (as indicated by the blue circle in Figure 4.15.b) are obtained using bin_r = 70 cm and bin_r = 80 cm. However, these corners do not affect the shape of the polygon, as they are collinearly located between two other corners.

The proposed building outline extraction works for single buildings. It requires a pre-processing step to select and acquire the outer building edge points. In certain cases, the proposed algorithm may fail to extract building outlines correctly due to heavy noise and errors in the segmentation. In this case, parameter tuning (bin_r and/or local maxima) may solve the problem, as the binning process of such irregular edge points may result in bad point division, and a voting scheme that makes the algorithm fail to detect correct local maxima.

Considering the aforementioned problems, a scheme for quality checking of the output is necessary to increase the applicability of our method for map production. We define a strategy to assist the quality checker in case of an incorrect or doubtful result. A missing line or a shifted line that causes incomplete or incorrect building polygons can be detected from the number of edge points that is not used by any extracted line. Based on this, the human operator may decide to perform a manual check and confirm the result. This procedure is expected to accelerate quality control and minimize manual editing.



(**a**) (**b**)

Figure 4.15 Experiment to evaluate the influence of the bin size parameter in case of extracting a complex building outline within a 1-m buffer of the reference corners (grey circle). (**a**) Scatter plot of building corners for different bin sizes. Different plus (+) color indicate corners of different *bin*$_n$; (**b**) Zoomed in a part of building corners.

4.6 Results and discussion

In the following, we will discuss the overall results of the proposed method. A more detailed discussion of the experiments for the Makassar test set and the Vaihingen test sets respectively, are given next. Finally, a comparison to previous results on the benchmark test set (Vaihingen-B) is presented.

4.6.1 General evaluation

Goal of this research is to provide a robust procedure for automatic building outline extraction from airborne LiDAR point clouds. Using the ground truth described in Section 4.4, we evaluate our method. Our method is able to achieve high completeness and correctness for both study areas as shown in Table 4.4. For Makassar, building polygons results achieve 91.8% completeness, 99.2% correctness and 91.1% quality. The quality metrics of the Vaihingen-A test set are also high, 96.4%, 96.5%, and 93.2%, respectively. Vaihingen-B achieves slightly less good quality metrics of 90.1% completeness, 99.4% correctness and 89.6% quality.

Table 4.4 Quality metrics of the building outline results and the concave hull results (Cp*: completeness,* Cr*: correctness,* Q*: quality)*

Test Set	Building Polygon	*Cp* (%)	*Cr* (%)	*Q* (%)
Makassar	OHT results	91.8	99.2	91.1
	Concave hull	92.1	98.3	90.7
Vaihingen-A	OHT results	96.4	96.5	93.2
	Concave hull	95.3	95.2	90.9
Vaihingen-B	OHT results	90.1	99.4	89.6
	Concave hull	90.9	99.6	90.6

We use different methods of point cloud filtering, classification, and segmentation to assure that our building extraction method is able to adapt to and is not limited to a specific processing workflow. The proposed workflow for extracting building outlines requires segmentation as a pre-processing step. The quality of the segmentation result influences the extracted building outlines. Irregular concave hull outlines as shown in Figure 4.2.b, are not suitable as input for map products. Table 4.4 shows a comparison between an object-based evaluation of our results and the concave hull of the segmented building points. Our results have better quality and correctness, with an increase of between 1% and 3%, except for Vaihingen-B. This proves that our algorithm is able to increase the quality of building outlines resulting from the concave hull and improve the segmentation result quality.

It should also be noted that the computational performance of the proposed method is not an issue. The average computation time of the proposed OHT method for delineating 2D building outlines on an Intel Core 2Duo CPU with a 2.4 GHz processor is about 0.579s per-building.

The completeness of our building outlines results is lower than the correctness. This means that, on average, building polygons are smaller than the ground truth data. As LiDAR points rarely sample a building edge, only points located within the building roof are used, which causes a shrinking of the polygons. We estimated the percentage of shrinking area using all buildings of simple (square) size that are well-segmented in our test sets. The average of the building shrinking is consistent at 4.24% for the Makassar test set and 4.36% for the Vaihingen test sets.

4.6.2 Results for Makassar

Indonesian base map specifications require that each building edge of at least 2.5 m should be presented on the map. Accordingly, the minimum length *minL* of a segment for the Makassar test set is five points (2.5m/0.5m). The bin_r size is set to 0.5 m. The ground truth data used as reference for Makassar is the Indonesian base map at a scale of 1:10.000. The base map is obtained by manual 3D-compilation of stereo-photos acquired at the same time and from the same platform as the LiDAR data we use.

The comparison between the extracted building polygons of Makassar and the ground truth is presented in Figure 4.16.b. From 42 buildings present on the base map, 37 buildings are extracted. The five missing buildings are indicated by blue stars. In addition, our method is able to extract one building that is not present in the ground truth (as indicated by a black arrow in Figure 4.16.b). This building has a size of 15.5 m by 43 m and should have been present in the base map.

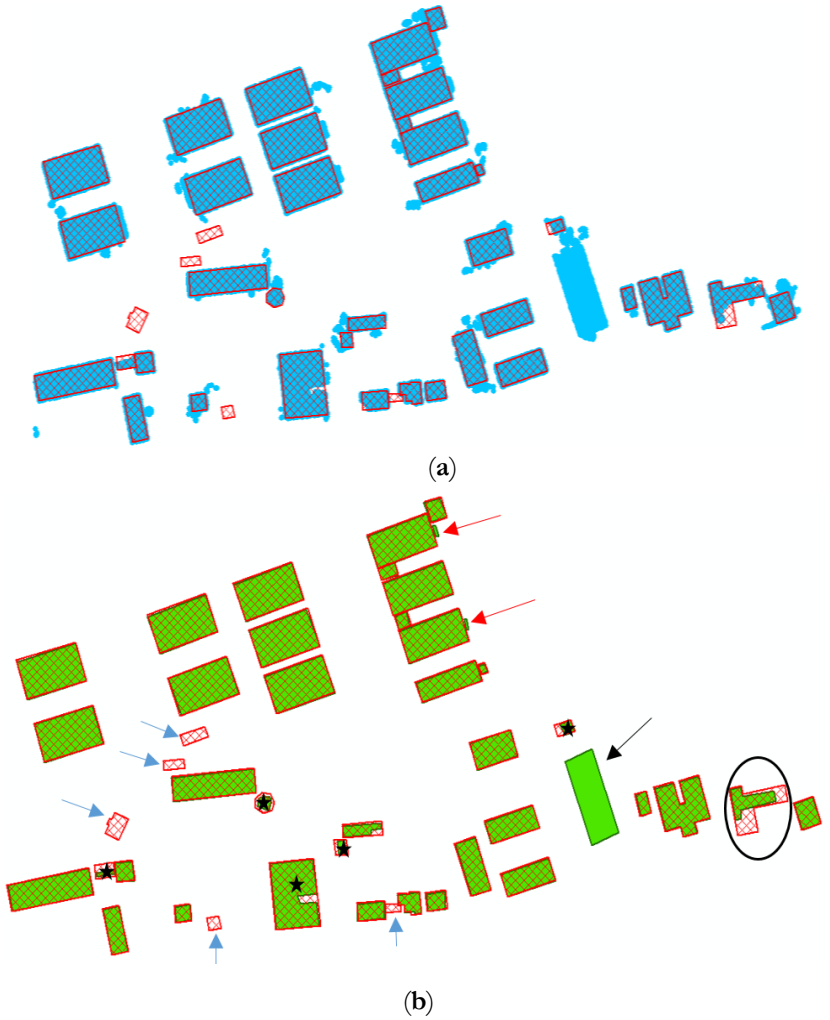


Figure 4.16 Illustration on how the proposed method regularizes building edges especially in case noise exist in the building segmentation input for the Makassar test set. (**a**) Building comparison between filtered building points (blue) and base map (red); (**b**) Building comparison between outlines generated by the proposed method (green) and base map (red).

Even though some noise exists in the segmented building points, the proposed method is still able to extract accurate building polygons. As highlighted in Figure 4.16.a, most filtered building segments (showed in blue) are noisy. High vegetation points connecting to the buildings cause most of the noise. Six buildings have a size or shape different from the base map (indicated by black stars) due to imperfect filtering and segmentation of building points. Low elevation of building roof points is assumed as the main cause why five buildings (indicated by blue arrows) are completely missing.

Positional accuracy is measured for buildings that exist in our result and the base map. RMSE is measured based on the coordinate differences between building corners from our result and reference data. The RMSE for complete building polygons is between 0.38–0.57 m. For a building where some parts of its roof are not completely detected (as indicated by the black circle in Figure 4.16.b), the building corners shift reaches 10.84 m.

During the experiment, the proposed method can actually extract building edges of a length less than the required 2.5 m from the Makassar data. The two buildings, as indicated by red arrows in Figure 4.16.b, are annexes of a size of 1.3 m by 6 m. These building annexes are likely not included in the map, as their size does not meet the 1:10,000 base map specification. Our method is able to extract such small building parts by applying the minimum edge length paramater $minL$ to $minL = 3$.

4.6.3 Results for Vaihingen

The chosen test sets of Vaihingen consist of high-residential buildings that have complex shape and are surrounded by trees. Several buildings have multiple roof layers of different heights and various length of edges. We set 1.5 m as the minimum length of a building line segment corresponding to $minL = 3$ for both of the Vaihingen test sets. The bin_r size setting is between 0.3 and 0.6 m.

We succeeded to extract all buildings in the Vaihingen-A and the Vaihingen-B test set. For the Vaihingen-A test set, the reference data is acquired by Open Street Map (OSM) validated by aerial orthophotos provided by ISPRS. For the Vaihingen-B test set, we use a set of building outlines provided by ISPRS as the ground truth. The RMSE result of buildings in the Vaihingen-A data that are completely segmented is between 0.2–0.37 m. As shown in Figure 4.17.b, five out of 27 buildings have different shape due to over-segmentation (three red stars) or under-segmentation (three black stars) due to dense trees above the building roof.

The RMSE of extracted building corners is between 0.19–0.96 m. According to the ground truth, the Vaihingen-B results have the lowest quality metrics among our three test sets. The main cause for a lower quality metric is misdetection of a vegetated building roof part of significant size (8.5 m by 9.5 m), marked by the black circle in Figure 4.17.c, which is likely a low roof covering an underground basement. The filtering process failed to detect this roof part as a building as the height difference between this basement roof and the ground is less than 1 m. The Digital Surface Model (DSM) of this subset area, as presented in Figure 4.18.b, shows that height information may not help to detect this kind of building (inside the white circle). In addition, some vegetated building roofs are not completely detected (indicated by brown circles in Figure 4.18.a and Figure 4.18.b). This happens because the trees and their surroundings disturb the expected planarity.

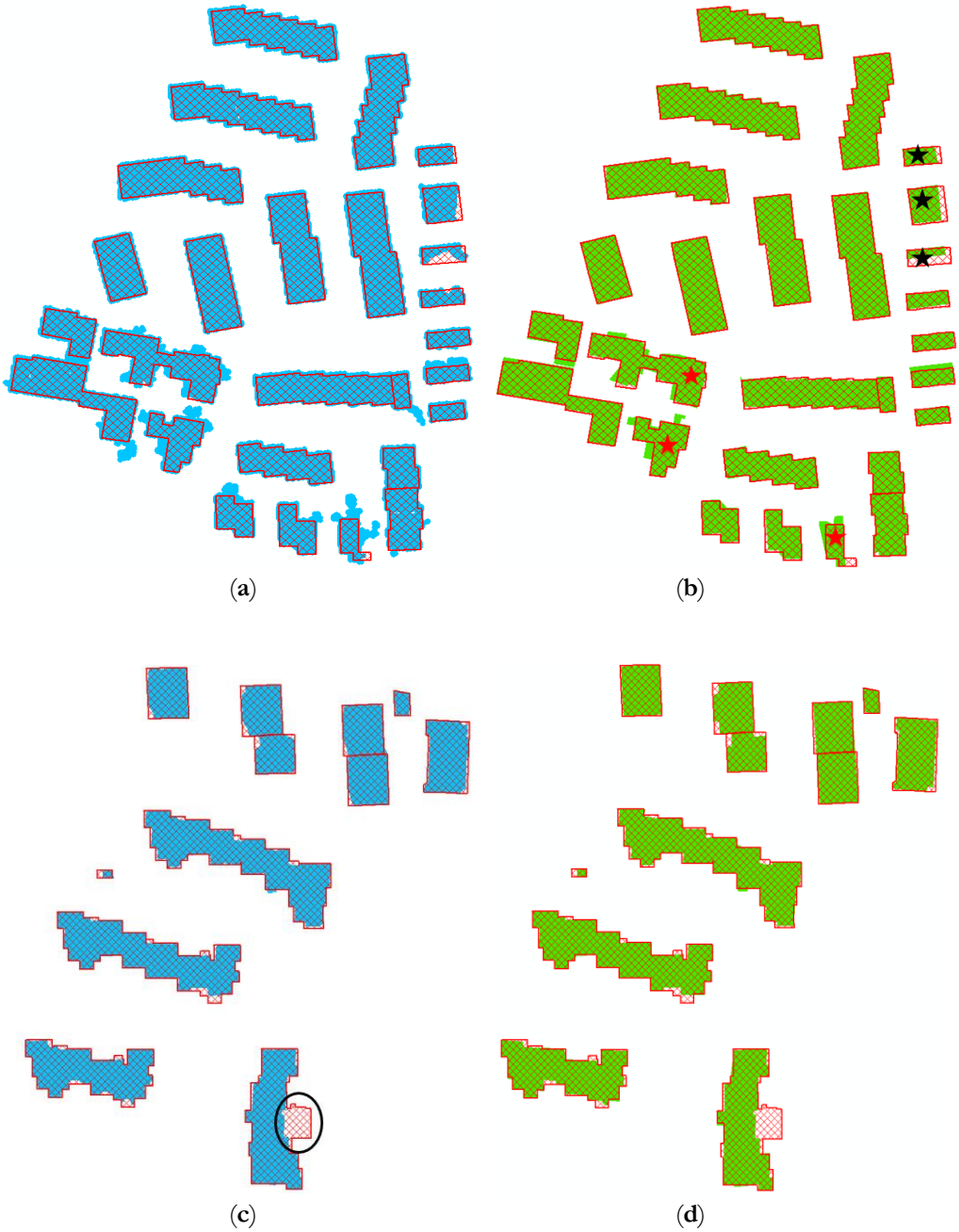
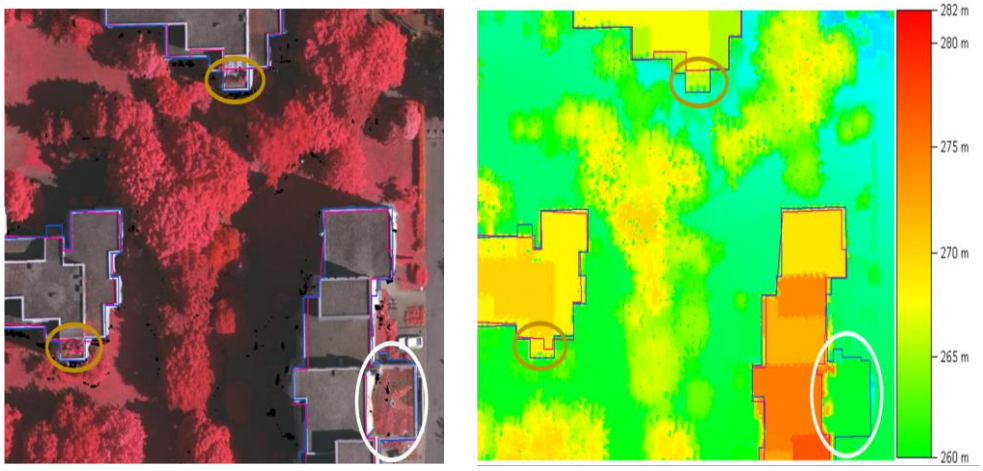


Figure 4.17 Comparison of building outline results with ground truth. (**a**–**b**) Vaihingen-A test area; (**c**–**d**) Vaihingen-B test area. Left: comparison of filtered building points (blue) and base map (red polygons); Right: outlines generated by the proposed method (green) and base map (red polygons).



(**a**) (**b**)

Figure 4.18 Vaihingen-B misses some part of building roofs due to vegetation (inside the brown circle) and low roof elevation (inside the white circle). (**a**) Overlay of building outline results (magenta) and reference (blue) with orthophoto; (**b**) Overlay of building outline results (magenta) and reference (blue) with Digital Surface Model (DSM).

In the results, several building polygons have a shape and size different from the ground truth because of tree points allocated to the segmented building points. High vegetation adjacent to buildings is the main cause for the high false positive rate of the building polygon results. Moreover, neighboring trees that cover some parts of the roof induce some false negatives. For example, in Figure 4.19.a, the building segment contains parts of four adjacent trees. These tree points have similar height as the building roof, which range from 275.5 to 276.5 m. The height difference to the mean is about 40 cm as shown in Figure 4.19.c. However, our line extraction algorithm is able to ignore 1 out of 4 trees as indicated by the yellow circle in Figure 4.19.a. An extra feature, such as intensity as shown in Figure 4.19.d may not work for obtaining a correct outline for this building case. A trade off in using the intensity value to remove trees will reduce the building completeness as there is a roof part covered by a big tree. An additional clustering step (using e.g., DBSCAN) may work to remove trees. Nevertheless, when a building roof is covered by dense trees (e.g., as marked by the purple circle in Figure 4.19.a), orthogonal input data (such as ALS point cloud data or airborne images) may not work to detect the building boundaries accurately.

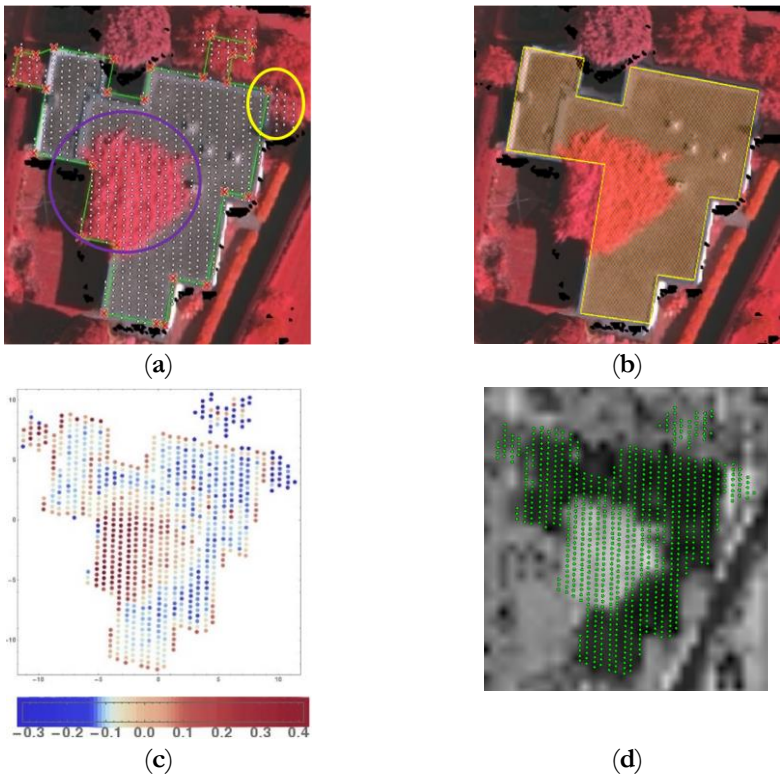


Figure 4.19 Trees connected to building lead to incorrect outline result. (**a**) Building outline result (green); (**b**) Reference building polygon; (**c**) Deviations from the mean roof height; (**d**) Intensity image of the neighboring building area.

4.6.4 Comparison to previous building outline works

An analysis is performed to compare the performance of the proposed method to previous methods. Several previous results are selected from the Vaihingen-B evaluation, which is available on the ISPRS web page (http://www2.isprs.org/commissions/comm3/wg4/results/a2_detect.html) and confirmed by the corresponding articles. We only consider methods aiming at obtaining straight building outlines that implement a data-driven approach and apply a regularization approach similar to our research objective. The selection is also limited to works that use ALS point clouds or a combination of ALS point clouds and aerial photos as input. We also include two results that are not presented on the ISPRS web page in our comparison.

As shown in Table 4.5, the MON3 represents Siddiqui et al. (2016) method has highest completeness but this method uses additional color information from aerial photos to get straight lines. Building boundaries in the Vaihingen-B test set are clearly recognizable as bold white pixels in the aerial photos. Hence, the low building roof (as

marked by the white circle in Figure 4.18. a) as well as vegetated roofs (as marked by the brown circle) can still be detected. However, the lower correctness metric of the MON3 method indicates that some buildings may be over-segmented. On the contrary, our method has best correctness and quality metric for this test set. The higher quality metric indicates that our method delivers complete and accurate building polygon results.

Table 4.5 Comparison to previous studies
(*Cp:* completeness, *Cr*: correctness, *Q*: quality)

Test set	Input Data	*Cp* (%)	*Cr* (%)	*Q* (%)
Awrangjeb, 2014 (MON2)	ALS	87.1	94.0	82.6
Siddiqui et al., 2016 (MON3)	ALS + Photos	**97.2**	84.3	82.3
Gilani et al., 2016 (FED_2)	ALS + Photos	88.8	84.5	76.4
Zhao et al., 2016	ALS + Photos	91.0	95.0	86.8
Huang et al., 2018	ALS	87.3	99.0	86.5
Proposed method	ALS	90.1	**99.4**	**89.6**

Our proposed method requires building roof points as input. Then, the 2D edge points are transformed to Hough space to obtain prominent building outlines. Based on this scheme, as long as building points are given, our method should work for arbitrary point cloud data including point clouds generated from images by for example dense image matching process.

4.7 Conclusion and future work

We have presented a framework for delineating 2D building outlines automatically from segmented ALS point clouds. An adaptive approach to obtain boundary lines of different building shapes and sizes is developed based on an extension of Hough transformation exploiting the order of points forming a building outline. Point votes for all possible lines passing through given edge points are stored in a Hough accumulator matrix. Based on the accumulator matrix, the algorithm uses local maxima for detecting dominant building orientations. The prominent lines are selected based on detected dominant directions. A hierarchical filtering approach by empowering the Hough transform with ordered edge points is introduced to select correct building segments and derive accurate building corners. Many problems that occur with the original Hough Transform are avoided.

Our enhanced Hough transformation method, which exploits the ordered points and regularity, gives a substantial improvement in the quality of building outline extraction as concluded from a comparison to existing benchmark results. Based on our extensive evaluation, the proposed procedure is able to deliver high completeness and correctness as well as high positional accuracy. Even though the Hough transformation

involves many matrices, aside from the pre-processing step, the processing speed to process the building outline is not an issue as the proposed algorithm works per single building. Another advantage of the proposed method is that it directly uses the point cloud. No data conversion or additional data is required. The proposed procedure is tested on different areas to verify the robustness of our method to the variation of different data specifications, and urban landscape characteristics. In case noise and small flaws exist in the data, the voting scheme of the Hough transform makes our method feasible for preserving the actual building shape and size. Implementation of the proposed outline extraction on different building segmentation attests that the use of the algorithm is not limited to a particular segmentation method.

We implemented a data-driven method that involves directed regularization that works effectively to detect multiple building orientations and derive accurate straight outlines. Our algorithm, accordingly, has limitation to detect curved outlines. As the algorithm requires edge points for delineating outlines, it is sensitive to the pre-processing steps, which are segmentation and concave hull. It may require parameter tuning for different dataset and different output requirements to achieve satisfactory results. Therefore, to guarantee an optimal result, understanding the input data and determining output criteria is necessary.

Extension of the present work should consider the implementation and performance evaluation of the proposed method for massive map production. Applying a robust classification and segmentation method for a larger area may become one of the challenges in future, as it may influence the outline extraction result. Application of Hough transform for curved buildings should also be elaborated.

5

Building outline extraction from ALS point clouds using Medial Axis Transform Descriptors

Automatic building extraction and delineation from airborne LiDAR point cloud data of urban environments is still a challenging task due to the variety and complexity at which buildings appear. The Medial Axis Transform (MAT) is able to describe the geometric shape and topology of an object, but has never been applied for building roof outline extraction. It represents the shape of an object by its centerline, or skeleton structure instead of its boundary. Notably, end points of the MAT in principle coincide with corner points of building outlines. However, the MAT is sensitive to small boundary irregularities, which makes shape detection in airborne point clouds challenging. We propose a robust MAT-based method for detecting building corner points, which are then connected to form a building boundary polygon. First, we approximate the 2D MAT of a set of building edge points acquired by the alpha-shape algorithm to derive a so-called building roof skeleton. We then propose a hierarchical corner-aware segmentation to cluster skeleton points based on their properties which are the so-called separation angle, radius of the maximally inscribe circle, and defining edge point indices. From each segment, a corner point is then estimated by extrapolating the position of the zero radius inscribed circle based on the skeleton point positions within the segment. Our experiment uses point cloud datasets of Makassar, Indonesia and EYE-Amsterdam, The Netherlands. The average positional accuracy of the building outline results for Makassar and EYE-Amsterdam is 65 cm and 70 cm, respectively, which meet one-meter base map accuracy criteria. The results imply that skeletonization is a promising tool to extract relevant geometric information on e.g. building outlines even from far from perfect geographical point cloud data.

In this chapter, some research background and an overview of related work is given in Section 5.1 and Section 5.2, respectively. Section 5.3 describes the methodological workflow using the MAT-based method to delineate building outlines. This is followed by a discussion of the results and conclusions as presented in Section 5.4 and Section 5.5, respectively.

5.1 Introduction

Mapping building roof outlines, also called building footprints, is essential for digital base map cartography, planning, surveillance, infrastructure management and sustainable city design. Several urban-related applications such as cadaster maintenance and building taxation require building outlines at a routine basis. Extracting building boundary lines manually is expensive and time consuming, especially in urban scenes. Research on extracting building outlines automatically from high-resolution data remains challenging due to the complexity of roof structures and variations in the design of our urban environment. Up to now, building outlines are typically digitized from multiple aerial images. The use of aerial images is preferred above other data sources as human operators can easily detect buildings on such images. For automation purposes, image disadvantages such as shadows, trees covering building roofs and color variations may increase the extraction error. Moreover, relief displacement may cause problems when using an orthoimage to obtain building boundaries in case of unfavourable image acquisition angles (Devi, 2014). Airborne Laser Scanner (ALS) point cloud data is an alternative data source. ALS point clouds have been used as a major data source for mapping applications for a few decades (Wang, et al., 2018). The ability of ALS point clouds to provide many accurate and undistorted 3D points makes it suitable as data source for object extraction. Man-made urban objects (buildings, roads, canals) typically have symmetric shape with straight lines and sharp corners. Such characteristics enable automatic boundary outline extraction from an ALS point cloud. Thus, the use of ALS point clouds for rooftop mapping in combination with an efficient algorithm is expected to provide better building outlines.

Medial axis transform (MAT), is a powerful shape extraction technique that provides a compact geometrical representation while preserving topological properties of the input shape (Tsogkas and Dickinson, 2017; Tagliasachi, 2016). The MAT was introduced by Blum (1967) to describe biological shapes. Since then, it has been used for applications in image processing and computer vision. However, MAT has a fundamental drawback, which is its instability to small perturbations of the input shape, which then may disturb the topology of the MAT branches (Peters, 2018; Bai et al., 2007). Moreover, wider deployment of MAT to extract shapes analysis from surveying quality data with its associated problems, is still challenging (Taglisachi, 2016).

In principle, the MAT can be implemented for urban mapping purposes, particularly to extract building shapes by detecting its corners. As illustrated in Figure 4.1, corner points (red) are detected when the tip of a skeleton branch (blue line) touches the building boundaries (black). However, generating a MAT skeleton from point clouds is a challenging problem as such data contain fuzzy borders, in particular, when data is missing (Huang et al., 2013). This makes the application of MAT for airborne point clouds difficult.

Corners are important local features and knowledge on their location can minimize further data processing without losing specific features of the original object shape (Chen et al., 2009; Ghosh,2015). Given an airborne point cloud of an urban area, we propose a method for extracting building outlines automatically by detecting accurate roof corner points based on MAT descriptors.

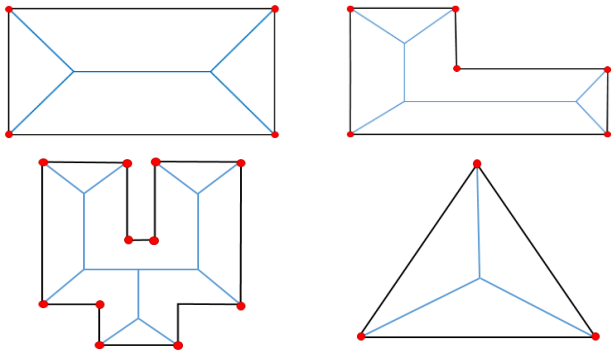


Figure 4.1 The MAT skeleton (blue lines) intuitively detects corners (red points) located at the intersection of the skeleton and the object boundary (black lines).

5.2 Related Work

Work on the development of building outline extraction from various remote sensing data has been intensified in parallel with the increased interest in GIS (Geographic Information System) digital map products (Taylor and Lovell,2012). Combining different data sources to extract building outlines is believed to increase the detection rate and accuracy compared to using a single data source as more features can be used. Nevertheless, fusion of data of different sensor types is not a trivial task as fusion is hampered by dissimilar resolution, alignment issues or mismatches in feature information caused by sensor characteristics or differences in viewpoint during acquisition (Furkuo and King, 2004). Manno-Kovacs and Sziranyi (2015) proposed an Orientation Selective Building Detection method to detect buildings from a combination of aerial and high-resolution satellite images. They apply active contouring for obtaining smooth and accurate building outlines. However, inhomogeneous buildings are sometimes only partially detected and any object connected to the building (e.g. trees) can result in false positive detections. Zhao et al. (2016), Awrangjeb (2016), and Li et al. (2013) combined LiDAR point clouds and aerial images to detect buildings and obtain smooth building outlines by regularization and mathematical morphology. Building outline errors occured due to failures to determine the building principal directions during regularization (Zhao et al., 2016; Awrangjeb, 2016) or line redundancies after simplification (Li et al., 2013) especially, in case of low point density. Our previous study (Widyaningrum et al., 2019) proposed an extended Hough

transform method using ordered lists of points to detect building boundary segments from airborne point cloud data. Hierarchical filtering is applied to remove spurious lines. That method outperformed existing state of the art methods in terms of correctness and quality metrics on benchmark dataset. However, the method is likely to introduce false corners, especially for buildings of complex shape, as spurious lines may still exist in the final step.

Various definitions of MAT or skeleton found in literature correspond to different methods for computing the MAT leading to different results with different properties. In general, MAT algorithms typically focus on deriving the geometric location of the centerline or medial axis of a surface, so-called skeletonization (Leymarie and Kimia, 2007). Up to now, numerous skeletonization methods and their application for 2D and 3D object description are available in literature. The existing skeletonization methods are often categorized into four main approaches:

- Morphological thinning-based methods that was first applied for discrete binary images (Pavlidis, 1978), and improved by Huang et al. (1987) and other related notions on the development of 3D thinning algorithm (Couprie and Bertrand, 2016; She et al., 2009).
- Geometry-based methods using medial axis transformation for planar shape (Lee, 1982) including Voronoi diagrams (Culver et al., 2004) and Delaunay triangulation (Reddy and Turkiyyah, 1995).
- Distance-based functions such as skeleton generation and centerline by the distance transform (Niblack et al., 1992), skeletonization by discrete Euclidean distance maps (Ge and Fitzpatrick, 1996), and Euclidean skeleton based on connectivity criterion (Choi et al., 2003).
- General-field functions which are generated by functions rather than use distance function, for example by replacing the nonlinear distance with a linear transform (Ahuja and Chuang, 1980), using Newtonian potential model to replace the distance function (Chuang et al., 2000), and using Electrostatic Field Theory (EFT) function to generate potential distribution inside the object (Grigorishin and Yang, 1998).

Reviews on skeletonization methods and its applications have been discussed by Saha et al. (2016; 2017), Pavlidis (1980), and Amenta et al. (2001). As many skeletonization algorithms were designed more for image analysis, we limit the scope of our study on skeletonization for point cloud data.

Several works on MAT using geographical data for various purposes have been conducted. Haunert and Sester (2008) applied straight skeleton extraction to derive linear representations of polygons and road centerlines from a cadastral dataset. Yirci et al. (2013) extracted detailed pedestrian networks by generating a centerline using two skeleton operators (straight skeleton by parallel thinning and medial axis by Voronoi

diagram). Methods for river centerline extraction based on Delaunay triangulations remain challenging for certain complex situations e.g. a scenario with a skeleton branching in different directions (McAllister and Snoeyink, 2000; Regnauld and Mackaness, 2006). Broersen et al. (2017) used the 2D skeleton of a Voronoi diagram and the 3D skeleton of a shrinking ball for identifying watercourses and deriving its centerlines from classified aerial point clouds. Widyaningrum and Lindenbergh (2019) extract the road network of an urban area from a colored point cloud using parallel thinning skeletonization (Zhang and Suen, 1984). However, a further generalization step maintaining the road topological order is required for smoothing jaggy skeletal lines yielded by the parallel thinning algorithm.

Ma et al. (2012) estimate a 3D medial axis point using a shrinking ball approach based on nearest neighbors and normals from a given set of points. Their work is claimed as a faster and simple method that approximates the 3D MAT based on maximally inscribed balls tangent to two surface points whose center is positioned on a normal line. The ball centers are the medial axis points. The shrinking ball algorithm is not only accurate and computationally efficient but is also considered as the most simple and fast existing surface-skeletonization method (Tagliasacchi et al., 2016). This method is considered as suitable for the geographical case as it is point based, simple, fast, and scalable (2018). However, to use this method, fine sampling is required to directly obtain a high quality skeleton. The fact that the shrinking ball approach can only result in unstructured skeleton points while disregarding the topology of the skeleton branches makes this algorithm not directly applicable for applications involving surveying data. For use in practical applications, the MAT consisting of medial points and corresponding maximally inscribed circles (in 2D) or balls (in 3D) needs further processing (Au et al., 2008; Chaussard et al., 2009; Kustra et al., 2015).

5.3 Methodology

Our research focuses on the adaptation of MAT for extracting building outlines from noisy point cloud data required for mapping and spatial modeling purposes. We extend the work on the iterative shrinking ball algorithm and develop a strategy to exploit skeleton features to accomplish the goal of accurate building outline extraction. A new approach for skeletal point segmentation is also proposed in this research. The proposed method achieves state-of-the-art in handling noisy surface boundaries and reconstructing the building outlines. It requires minimal human interaction by optimizing the use of skeleton-based features. Specifically, the contributions of our work are as follows:

- We integrate skeleton-derived features and global features to perform robust skeletal points (MAT) segmentation handling varying point density and noise level.

- We combine ordered surface point indices and skeletal-derived features to detect corner points.
- We introduce the use of skeletal-derived features to estimate building corner positions accurately.

Overall, our method overcomes some traditional pitfalls of using MAT techniques in case of noisy input.

As this research requires using the MAT in 2D space, we adapt the 2D shrinking circle algorithm by Ma et al. (2012). The general workflow of our proposed method for automatically extracting building roof outlines consists of four main steps (see Figure 5.2). First, building boundary points are extracted by an alpha-shape algorithm (Edelsbrunner, 1983). Next, the boundary points are transformed into its 2D MAT or skeleton points using the 2D shrinking circle algorithm. Third, we then apply our MAT segmentation to segment the MAT points by exploiting their geometric attributes. The segments are then used to detect corner points. Fourth, polygonization is carried out to form a 2D closed polyline based on the detected corner points.

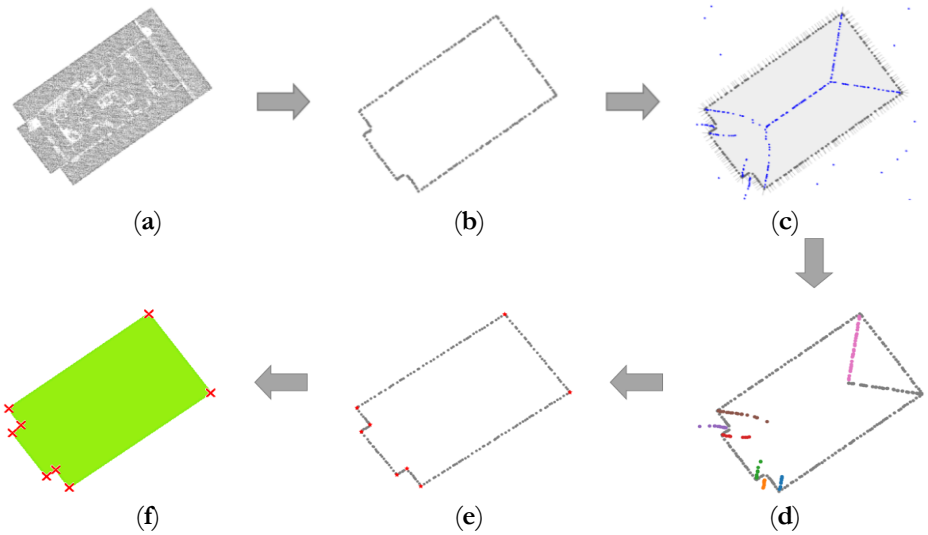


Figure 5.2 Proposed methodological workflow for extracting the building outlines in clock-wise order. (**a**) ALS building roof points as input; (**b**) 2D building boundary points; (**c**) 2D MAT extraction (blue points); (**d**) MAT points segmentation and outlier removal, different color for different segment; (**e**) corner points (red) estimation; (**f**) polygonization based on the estimated corners (red crosses).

This research uses the extended shrinking circle approach that implements the denoising heuristic as proposed by Peters (2018). We define the skeleton of an object surface $\boldsymbol{S}$ as a set of center points c of maximally inscribed circles $\boldsymbol{B}(c, \rho)$ in $\boldsymbol{S}$ (see Figure 5.3) where ρ denotes the radius of such circle.

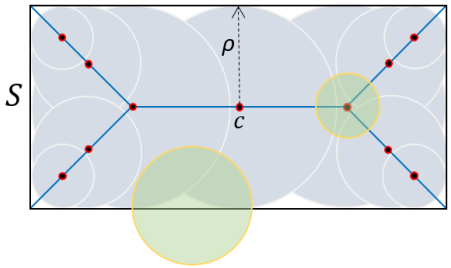


Figure 5.3 The skeleton (blue line) of a rectangular shape with its corresponding inscribed circle (grey) and medial axis point c (red point).

The 2D skeleton points are also called medial axis points. By associating the circle radius ρ function to the set of medial axis points, we obtain the so-called Medial Axis Transform (MAT). As shown in Figure 5.3, the medial axis points (red points) form the MAT skeleton (blue lines) of a rectangular object $\boldsymbol{S}$. Each maximally inscribed circle (in grey) touches at least two points of the boundary of $\boldsymbol{S}$ (black outline). Center points of any circle that is not maximal or not inscribed in $\boldsymbol{S}$ (green circles) are dismissed and not considered as medial axis points.

To provide a clear and coherent narrative, further details on each methodological step are provided in the following: Section 5.3.1 describes the alpha-shape algorithm. Section 5.3.2 and 5.3.3 provide necessary details on the shrinking circle method and skeletal points extraction, respectively. Skeletal point segmentation is described in Section 5.3.4. The last step, corner point estimation and building outline generation, is explained in Section 5.3.5. Evaluation methods for building outline extraction used in this research are discussed in Section 5.3.6.

5.3.1 Alpha-shape

Given the segmented building points, creating building outlines starts with boundary point selection by the alpha-shape algorithm as introduced by Edelsbrunner (1983). An alpha-shape is well known for its capability to preserve small shape details of a finite point set at a required level of detail. The 2D alpha-shape is constructed based on the 2D Delaunay triangulation of the input points. The method identifies boundary points that are defined in terms of a parameter $\alpha \geq 0$, which controls the level of detail of the boundary shape. Given a set S of points on a plane and a value of α, the algorithm works as follows:

1. Compute the Delaunay triangulation DT(S) of S. All edges in DT(S) are candidate for the alpha-shape S_α.
2. For all edges e of DT(S) with end points p and q, say:
    a. Find two circles B_{pq1} and B_{pq2} of radius α with center $c_{pq}(1)$ and $c_{pq}(2)$ containing end points p and point q of the same edge e. The circles are defined in terms of the below circle centers:

$$c_{pq}(1,2) = \left(\frac{x_p + x_q}{2} \pm \sqrt{\alpha^2 - \left(\frac{\|e\|}{2}\right)^2} \left(\frac{y_p - y_q}{2}\right), \frac{y_p + y_q}{2} \pm \sqrt{\alpha^2 - \left(\frac{\|e\|}{2}\right)^2} \left(\frac{x_p - x_q}{2}\right) \right) \quad (5.1)$$

Where $\|e\|$ is the length of the edge between end points p and q.

b. If at least one of the circles contains no points from S in its interior, e is a valid boundary edge (Figure 5.4.a), otherwise the edge is removed (Figure 5.4.b).

3. The union of all valid boundary edges forms the alpha-shape S_α (Figure 5.4.c)

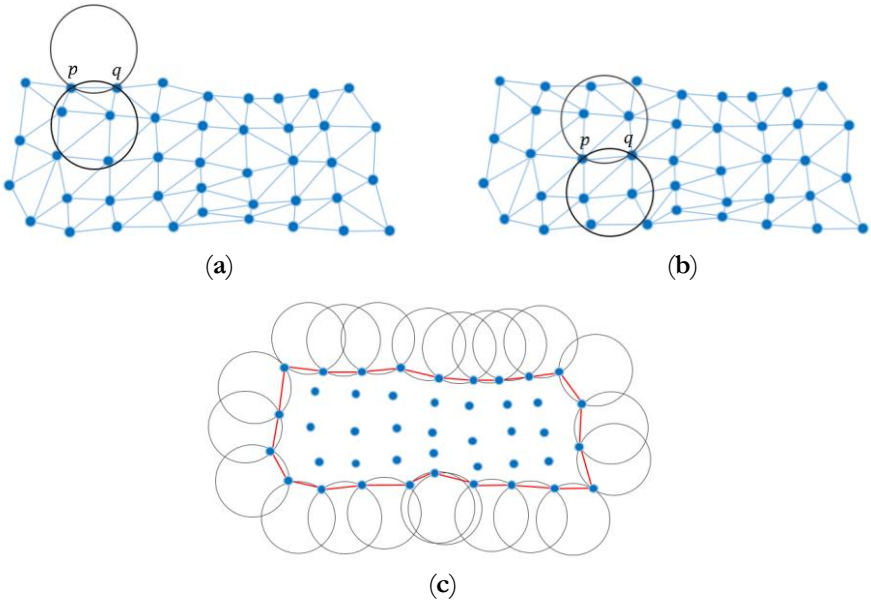


Figure 5.4 The alpha-shape criteria. (**a**) Delaunay edge connecting points p and q is a boundary edge as one of the circles, here the top circle, is empty; (**b**) Delaunay edge connecting points p and q is not a boundary edge as both circles are not empty; (**c**) complete boundary polygon (red line) of the point set as indicated by empty circles of radius α (black circles).

The value of α is a real number with $0 \leq \alpha \leq \infty$. As α approaches 0, the shape may shrink, develop holes and may become disconnected. In the extreme case, the value of $\alpha = 0$ results in the data points itself. When α increases towards infinity, the alpha shape approaches the convex hull of the set S of points. In case of geospatial point clouds, the point density is often varying, and is depending on sensor characteristics and measurement geometry and an appropriate value of α should be chosen accordingly. To identify boundary points of unordered building roof points in our study areas, we decided empirically for an α-value between 0.3 and 0.5.

5.3.2 The shrinking circle principles

Given are a set of noisy edge points V on a surface $\boldsymbol{S}$ with corresponding normal vectors N. The MAT points are defined as the set of centers c and corresponding radius ρ of maximally inscribed circles $\boldsymbol{B}(c, \rho)$ in $\boldsymbol{S}$ that are bi-tangent to the boundary $\boldsymbol{S}$. The circle $\boldsymbol{B}$ and corresponding circle center c are denoted as medial circle and medial axis point, respectively.

The basic principles of the shrinking circle method (see Figure 5.5) are as follows:

1. A medial circle touches the surface in at least two points (p, q) where $p, q \in \boldsymbol{S}$.
2. Following the line defined by normal vector N_p of edge point p, the radius ρ of a circle $\boldsymbol{B}_p$ decreases iteratively until $\boldsymbol{B}_p$ touches $\boldsymbol{S}$ at q, where $q \neq p$ and the circle center c is on the line through N_p. Iteration stops if the maximal $\boldsymbol{B}_p$ circle is found.
3. A medial circle is a maximal empty circle, which means it contains no surface points.

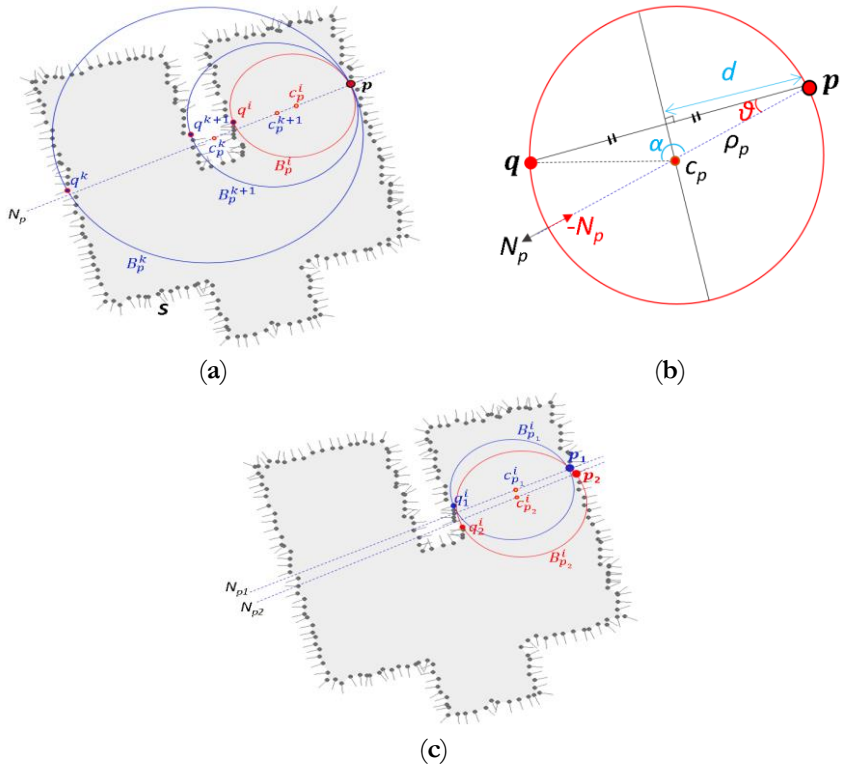


Figure 5.5 Basic principles of the shrinking circle algorithm applied on noisy building edges. (**a**) Shrinking circle iteration of point p; (**b**) Circle center c_p is the intersection of N_p and perpendicular line of bisector $\overline{pq}$; (**c**) Medial circles of two consecutive points p_1 and p_2 results in two medial axis points c_{p1}^i and c_{p2}^i.

5.3.3 Skeletal points extraction

To obtain the MAT of surface $\boldsymbol{S}$, medial axis points $c(p)$ are computed. Hence, the maximal inscribed medial circle $\boldsymbol{B}$ for all sample points p in $\boldsymbol{S}$ is computed by the following steps:

1. An initial circle $\boldsymbol{B}_{init}$ of p is defined based on an initial radius ρ_{init}. The ρ_{init} value is set sufficiently large e.g. equal to the largest distance between two input points.
2. Given ρ_p^k, where $k = \{1, 2, .., i\}$ denotes the k-th iteration step, the circle center c_p^k is given by:
$$c_p^k = p - N_p \rho_p^k \tag{5.2}$$
3. Find the surface point $q_p^k \in \boldsymbol{S}$ closest to c_p^k such that $q_p^k \neq p$.
4. Test for circle maximality for the circle defined by q_p^k and p:
   a. If the distance from c_p^k to q_p^k equals the radius of the circle ρ_p^k, the circle B_p^k is maximal and c_p^k is a medial axis point.
   b. Otherwise, compute the radius of the next shrinked circle ρ_p^{k+1} using the following equations:
$$\rho_p^{k+1} = \frac{d(p,\ q^{k+1})}{2cos\theta_p^{k+1}} \tag{5.3}$$
Where:
$$cos\theta_p^{k+1} = \frac{N(p - q^{k+1})}{d(p,q^{k+1})} \tag{5.4}$$
$$d(p, q^{k+1}) = \frac{|p - q^{k+1}|}{2} \tag{5.5}$$

The iteration will stop when the medial axis point as described in step 4.a. is found. Figure 5.5.a. shows consecutive shrinking of a circle touching $\boldsymbol{S}$ at point p, which results in a medial circle $\boldsymbol{B_p^i}$ and a medial axis point c_p^i in the last iteration.

Given a defined inside and outside of surface $\boldsymbol{S}$, the MAT consists of two components: one part inside surface $\boldsymbol{S}(N_p)$, consisting of the so-called inner medial axis points, and another one outside surface $\boldsymbol{S}(-N_p)$, the outer medial axis points.

For each $p \in \boldsymbol{S}$, the corresponding inner and outer MAT points are computed by iterating steps 2 to 4. The inward normal N_p is used for the inner MAT calculation, while the outward normal $-N_p$ is used for the outer MAT calculation. Figure 5.5.b. shows the geometry for calculating the medial axis point c_p and the direction of the normal vector N_p for the inner circle (black arrow) and outer circle (red arrow).

Noise handling is an essential step to overcome the sensitivity of MAT to noisy boundaries. In case a small bump or noise exist on the input surface, a circle may get shrunk too much which then likely results in undesirable medial axis points. Such overly shrinked circle, typically has a small separation angle α. The separation angle α (see

Figure 5.5.b) is the angle between line $p - c_p$ (the line connecting point p and medial axis point c_p) and line $q - c_p$ (the line connecting point q and medial axis point c_p).

$$cos\alpha = \frac{\overrightarrow{c_p p} \cdot \overrightarrow{c_p q}}{|\overrightarrow{c_p p}| \cdot |\overrightarrow{c_p q}|} \tag{5.6}$$

The denoising heuristics technique presented by Peters (2018) is used to select the so-called good circle that is often computed during the above described shrinking approach. The good circle is defined as the last circle in the shrinking approach with a separation angle α_k bigger than the separation angle threshold α_{min}.

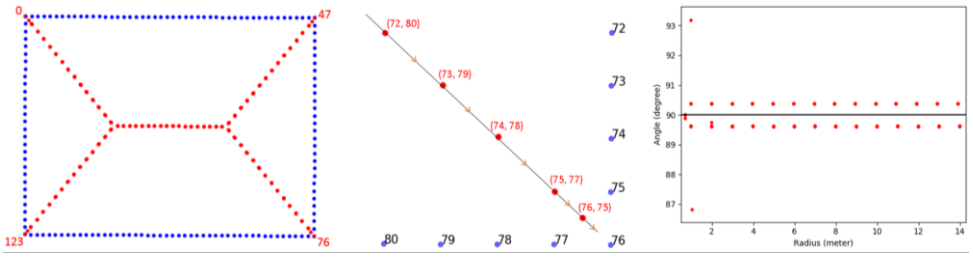
After this step, every medial axis point $\boldsymbol{c_p}$comes with a number of attributes. Attributes of each MAT point $\boldsymbol{m_p}$ are medial axis point $c_p$coordinate, radius ρ, separation angle α, indices of surface point p and q, and normal vector N_p or $-N_p$. Theoretically, using these MAT attributes, the geometry of $\boldsymbol{S}$ can be reconstructed completely.

5.3.4 MAT point segmentation

MAT attributes provide rich information that can be used to group MAT points into different medial segments or branches. In our case, segmenting the points into different branches is used for detecting the corner points. One issue when obtaining a skeleton from a point cloud with the shrinking circle algorithm is that the point cloud provides an unstructured point sampling of $\boldsymbol{S}$. Also for the MAT points resulting from steps 1 to 4 in Section 5.3.3, adjacency relations are initially not known. For further application of the MAT, two useful observations to identify MAT point connectivity are as follows:

- MAT points heading towards the same turning point or corner are considered as one segment. Fine sampled surface points of a square shape $\boldsymbol{S}$ in Figure 5.6.a result in fine MAT points in which some of the MAT points gradually approach a specific turning point. In this sense, each MAT point created from a maximally inscribed circle, touching at surface point p and q appoint to a turning point that is equally located between surface point p and q. As illustrated in Figure 5.6.b, the median value of two surface points p and q (in red text) is similar to the corner's index (76).
- MAT points are expected to have a separation angle α close to 90°. As shown in Figure 5.6.c, MAT points of a rectangular shape have separation angles that are distributed around 90°.

In practice, surface points are not perfectly distributed and noise-free and are not as regular as shown in Figure 5.6.a. Small perturbations on the surface boundary create so-called skeletal noise (Reniers et al., 2008; Giesen et al., 2009). When detecting shape corners, skeletal noise may induce false segments, which then results in false corners.



(**a**) (**b**) (**c**)

Figure 5.6 Characteristics of skeletal points used for segmentation in case of uniform edge point spacing. (**a**) MAT points (red) heading to the same turning point of the edge are considered as one segment. Turning points of this shape are: 0, 47, 76, and 123; (**b**) Median of the corresponding (p, q) of MAT point (red) is the same as the turning point index (76); (**c**) Distribution of MAT points (red) angle that are close to 90°.

Our segmentation criterion relies on the proximity of the turning point location of the boundary point. Intuitively, MAT points that lie close to each other and have similar features values are grouped together. Moreover, we expect that the radius ρ will gradually change along a segment branch.

Based on aforementioned observations, we use three global thresholds and four MAT-derived features for segmenting the MAT points. The global thresholds are not related to MAT and are defined to increase the segmentation accuracy. The global thresholds are:

G.1. A buffer distance *bf* from the object surface $\boldsymbol{S}$. Only a MAT point located within the specified buffer will be considered for segmentation. This threshold is used to exclude unusable outer MAT points resulting from the maximal circle of two edge points with outward normal. These MAT points are typical noise located far from the surface points.

G.2. Minimum number of points for each segment *minPts*. Any segment having less points than the given *minPts* is considered as segment noise.

G.3. Point index interval Δpt sets the minimum distance between two candidate corner points, as expressed below:

$$\Delta pt \geq \frac{l}{r} - 2 \tag{5.7}$$

In Equation 5.7, l is the minimum required edge length and r is the point cloud interval. The point index interval Δpt criterion is designed to avoid having false or extra corners at a certain minimum determined edge length in case of short and noisy boundaries. For example, given a set of points with 0.5 m point cloud interval r, we require to extract building edges of minimum length l = 2.5 m, thus, Δpt is set to 3. Imagine that point 13 in Figure 5.7 has the same medial properties as point 11, 16, and 20. Point 13 will not be considered as a corner point as it has less than 3 point difference to point 11.

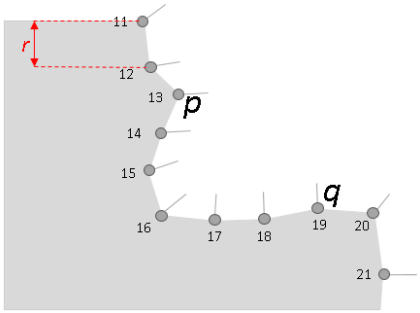


Figure 5.7 Noisy point edge may indicate a false corner like point 13. By applying point index interval $\Delta pt = 3$, point 13 will not be detected as a corner.

The customized features derived from the MAT attributes, or MAT-derived features are described as follows:

F.1. A MAT point $\boldsymbol{m_p}$ having a separation angle α close to 90° is considered for segmentation. Here 'close' is specified by the separation angle difference threshold, $\partial\alpha$. MAT points outside the given $\partial\alpha$ threshold are considered as skeletal noise. This means, a MAT point $\boldsymbol{m_p}$ will be considered for segmentation if it has a separation angle α_p between $90° + \partial\alpha$ and $90° - \partial\alpha$.

$$90° - \partial\alpha \leq \alpha_p \leq 90° + \partial\alpha \tag{5.8}$$

F.2. Each edge point $p \in \boldsymbol{S}$ is assigned a unique index. For the corner-aware segmentation, MAT points $\boldsymbol{m_p}$ of similar characteristics are expected to belong to the same cluster. This is assessed by considering the point indices of the surface points of $\boldsymbol{S}$. Assume $p, q \in \boldsymbol{S}$. Let p_{idx} and q_{idx} denote the point indices of p and q, respectively. The median value med_{pq} is obtained by Equation 5.9.

$$med_{pq} = p_{idx} + \frac{q_{idx} - p_{idx}}{2} \tag{5.9}$$

For example, p_{idx}=13 and q_{idx}=19 in Figure 5.7 results in med_{pq}= 16. Different MAT points with similar med_{pq} value likely belong to the same MAT segment.

F.3. The normal angle differences δN between the normal of point p_i and the normal of the previous point p_{i-1} and the normal of the next point p_{i+1} defined by $(|Np - Np_{i-1}|)$ and $(|Np - Np_{i+1}|)$, respectively.
The normal angle differences δN between 2 consecutive edge points is used to determine candidate corners K_p for $p \in \boldsymbol{S}$. Detection of candidate corners K_p is used to obtain first estimate of the number of building corners which is later used to remove false segments in case of noisy edges. At noisy edges, segments with small median difference may be formed which later may result in false corners. Consider two adjacent normal vectors N_{p_i} and $N_{p_{i-1}}$ of point p and p_{i-1} respectively, compare Figure 5.8. The angle δN between the two normal vectors N_{p_i} and $N_{p_{i-1}}$ is obtained via Equation 5.10.

$$\cos(\delta N_{p_i p_{i-1}}) = \frac{N_{p_i} \cdot N_{p_{i-1}}}{|N_{p_i}| \cdot |N_{p_{i-1}}|} \tag{5.10}$$

A surface point p initiates a candidate corner point K_p if the angular differences δN to its two adjacent points (p_{i-1} and p_{i+1}) are above the given angle threshold δN. That is expressed in Equation 5.11.

$$(\delta N_{p_i p_{i-1}}, \delta N_{p_i p_{i+1}}) \geq \delta N. \tag{5.11}$$

For example: for $\delta N = 20°$, if $\delta N_{p_i p_{i-1}} \geq 20°$, and $\delta N_{p_i p_{i+1}} \geq 20°$, then the surface point p is a candidate corner.

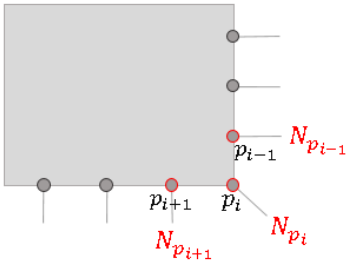


Figure 5.8 Normal vectors (grey lines) of edge points (black points), where p_i is the turning point and N_{p_i} is the corresponding normal vector.

F.4. The maximum median index difference threshold ∂K_p. This feature is used to avoid false corners and additional segments caused by perturbation or noise on the surface particularly near to the corners.

Given a set of MAT points $M = \{(c_p, \rho, \alpha, p, q, N)\}$ with 6 features ρ, α, p, q, N per medial axis point c_p, the MAT point segmentation works as follows:

1. Select MAT points $\boldsymbol{m_p}$ located within a certain buffer distance of *bf* from the object surface $\boldsymbol{S}$ using global threshold G.1. For most of our buildings, a buffer distance *bf* = 35 meter is sufficient.
2. Select only MAT points $\boldsymbol{m_p}$ having an acceptable separation angle α_p as specified by the MAT-derived feature F.1. This step eliminates skeletal noise that typically has a separation angle value away from $90°$. In our case, a separation angle difference $\partial\alpha = 20°$ is sufficient for the medial segmentation.
3. Compute the median value med_{pq} of the filtered MAT points from step 2 using the MAT-derived feature F.2.
4. Identify all possible candidate corners K_r and put them in a list $K_1, K_2 \ldots, K_i$. Candidate corner K_r is added to the list if it satisfies all the MAT-derived criteria specified in F.3, F.4, and global feature G.3. We use threshold values $\partial K_r = 3$, $\delta N = 15°$ and $\Delta pt = 2$, respectively.
5. Given a candidate corner K_r from step 3, the algorithm searches for MAT points with median value med_{pq} similar to r_{idx}.

6. If $\left|med_{pq} - r_{idx}\right| \leq \Delta pt$, then MAT point $\boldsymbol{m_p}$ is assigned to medial segment $\boldsymbol{Mseg}(r)$.
7. Any medial segment $\boldsymbol{Mseg}$ having less member points than *minPts,* as defined in G.2, will be removed. This step will eliminate false segments that may be formed in case of flaws on the edges.

Medial segments are used next to estimate real corners where one medial segment corresponds to one corner.

5.3.5 Corner point estimation

Instead of appointing edge points as corners, we rather estimate the position of corners based on the medial axis point positions and their corresponding radius. The radius ρ of the maximally inscribed circles of MAT points will gradually decrease towards corners. Each medial segment ideally contains a set of MAT points with gradually decreasing radii. The location where the radius will become zero $(c_{\rho=0})$, typically identifies the location of the corner point corresponding to a medial segment is estimated as follows. Figure 5.9 shows how the radius ρ of the MAT point depends linearly on the x-coordinate. Therefore, a line is fitted by PCA (Principle Component Analysis) through the (x, ρ) points of the segment at hand. The x- coordinate corresponding to zero radius $(\rho = 0)$ of the fitted line L_r (as indicated by the blue line in Figure 5.9) is reported as the x coordinate of the corner point. The y-coordinate of the corner point is obtained in a similar way.

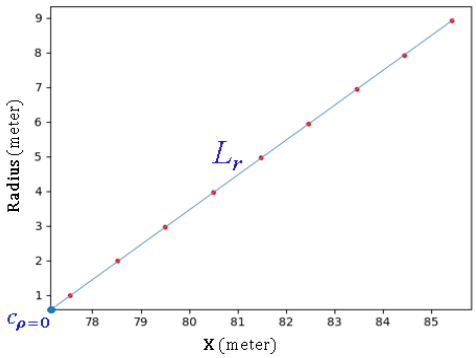


Figure 5.9 Estimating the x-coordinate of the corner (blue point) by predicting where radius $\rho_p^k = \mathbf{0}$ on L_r (represented by a blue line).

In case of an edge with heavy noise, false segment may remain. A spatial filtering step is necessary as final filter to remove any spurious estimated corner c. This spatial filtering step preserves any corner point that is within a specified radius from the surface points $p \in \boldsymbol{S}$. In our case, we use 1 meter as we require building outline result to have positional accuracy at least 1 meter. As final step, a closed building polygon is obtained by connecting all corner points consecutively referring to the point indices.

5.3.6 Building outline evaluation metrics

Two different evaluation metrics are applied for evaluating the performance of the proposed workflow in fulfilling the required building outline specification: corner geometric accuracy and corner detection accuracy. To evaluate the geometric accuracy, we compared the position of corners coordinates of the building outline results to the reference (Zheng et al., 2013). Positional accuracy, also known as geometric position accuracy or location accuracy, is used as main indicator to measure how well the building polygons are positioned with respect to its true position within an absolute georeferenced system. We use the RMSE (Root Mean Square Error) to measure the average of the squared differences between building corner positions (X and Y coordinate) in the reference and in the result. The RMSE of a complete building is calculated for all detected building corners with respect to the position of corresponding reference corners.

$$RMSEx = \frac{\sqrt{\sum(X_{res} - X_{ref})^2}}{n} \tag{5.12}$$

$$RMSEy = \frac{\sqrt{\sum(Y_{res} - Y_{ref})^2}}{n} \tag{5.13}$$

$$RMSEr = \sqrt{RMSEx^2 + RMSEy^2} \tag{5.14}$$

Where:

X_{res}, Y_{res} = Coordinates of resulting corner points

X_{ref}, Y_{ref} = Coordinates of corner points in the ground truth

n = total number of corner points

Due to the complexity of some building, not all corners may be detected completely. Therefore, we evaluate the corner detection accuracy by means of three retrieval measurements: recall, precision and F1-score (Makhoul et al., 1999). Precision is used to measure the exactness or fidelity, whereas recall is used to measure the completeness. The F1-score is the weighted mean of precision and recall.

$$Precision = \frac{TP}{TP + FP} \tag{5.15}$$

$$Recall = \frac{TP}{TP + FN} \tag{5.16}$$

$$F1\ score = \frac{2\ .Precision\ .Recall}{Precision + Recall} \tag{5.17}$$

For this purpose, a corner point is considered a True Positive (TP) if it is located within 1-meter radius from the corresponding corner reference, while any undetected corner including corners with an offset of more than 1 meter from the corresponding

reference corner is considered a False Negative (FN). Any corner in the result that does not exist in the reference is considered as False Positive (FP).

In the building polygon in Figure 5.10, the number of correct corners (TP) is 4, while the number of false corners FP (inside the green ellipse) is 1, and the number FN of undetected corners or corners with an offset of more than one meter (as indicated by the blue circles) is 2. This configuration gives Precision = 0.8, Recall = 0.67, and its F1-score is 0.73.

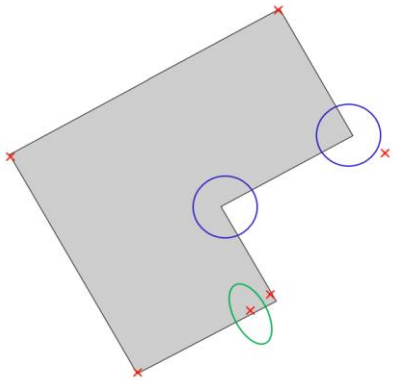


Figure 5.10 Illustration of building corner detection accuracy based on the number of correctly detected corners within 1-meter radius from the corner reference. Red crosses indicate building corner results. In blue, two 1-meter circles around an undetected reference corner are indicated. The green ellipse indicates a corner that does not exist in the reference.

Figure 5.11 summarizes the proposed method to detect building corners from a given cluster of building points (Figure 5.11.d). The parameter thresholds, as discussed above, are set empirically depending on the point density and the required specification.

Additional pre-processing is necessary in case clustered building points are not available. In this case, an initial classification and/or semantic segmentation processing step is required. We use a classified ALS point cloud whose points are labelled according to their object class (building, ground, and unclassified). To group points belonging to one building, points are clustered by applying the DBSCAN algorithm (Ester et al., 1996). As a result, different buildings will have different cluster number and theirs points are labelled according to the corresponding cluster number. Once the clustered building points (as presented in Figure 5.11.d) are available, boundary points need to be extracted (Figure 5.11.e), for which we use the alpha-shape algorithm. The resulting building boundary points are then used as input for the MAT shrinking circle algorithm (blue points Figure 5.11.f). MAT points filtering and segmentation is applied based on their separation angle and median index value (Figure 5.11.g). Each medial segment generates a different corner candidate. Positions of the corners are extrapolated linearly using PCA (Figure 5.11.h).

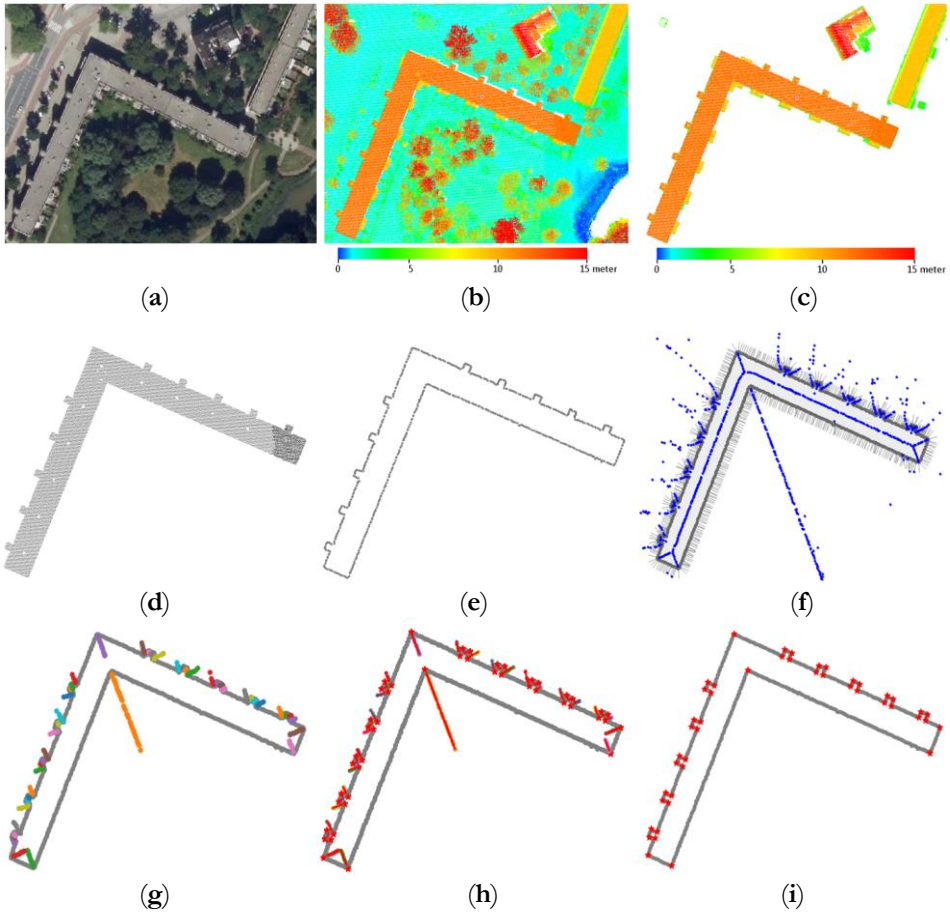


Figure 5.11 Overview of the proposed corner detection method using MAT descriptors. (**a**) Aerial photos of the building; (**b**) ALS point cloud of the building; (**c**) Classified building points; (**d**) Clustered building points; (**e**) Building boundary points; (**f**) Skeleton (MAT points) of building edge points (blue); (**g**) Segmented MAT points. Different colors mark different segments; (**h**) Linear extrapolation to estimate corner point positions; (**i**) Detected corner points.

5.4 Results and discussion

5.4.1 Experiments of the study areas

For the experiment, we use three study areas with different landscape characteristics and airborne LiDAR point cloud specifications. The first dataset represents a sub-urban area of the city of Makassar, Indonesia. The point cloud data was captured in 2012 by a Leica ALS70 instrument and has 7-11 ppm (point per meter) point density.

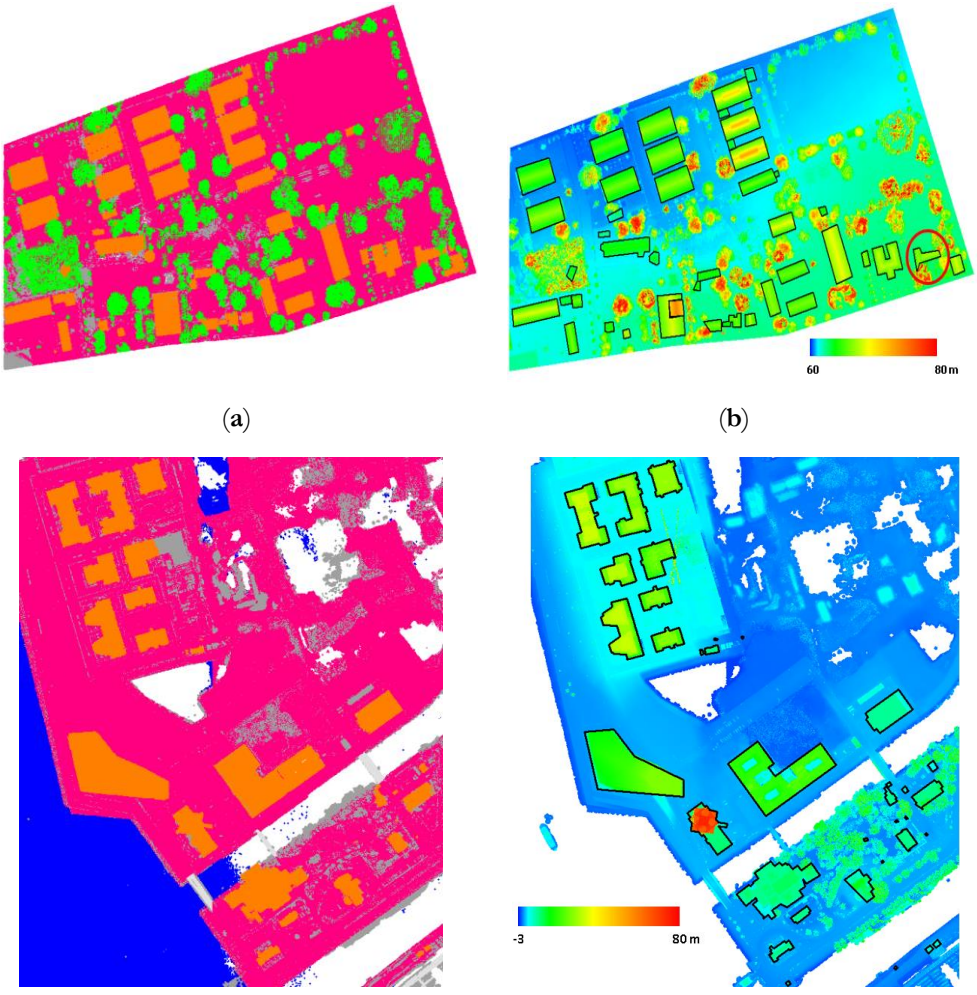
The second dataset is a Dutch national AHN3 point cloud sampling the area of EYE-Amsterdam, the Netherlands. The AHN3 data has a point density of at least 10 ppm and was acquired in 2014. Most buildings in this area have a public or business function. This test set is selected as many of the buildings in this area are considered to have a high complexity in terms of shape and size.

Building points of the Makassar dataset, as presented in Figure 5.12.a. in orange, were classified using LAStools. For EYE-Amsterdam, we used the provided building classification of the AHN3 dataset (shown as orange points in Figure 5.12). The alpha-shape algorithm is then applied to derive the outline of each building. We do not discuss the details of the pre-processing steps further as it combines well-known methods in the field of GIS and remote sensing.

For the Makassar area, the topographic base map scale 1:10.000 is used as ground truth data. The topographic base map is generated from manual 3D delineation from stereo-images with the same acquisition time as the Makassar airborne point cloud data. For validating the Eye-Amsterdam results, the Dutch building registration dataset BAG (*Basisregistratie Adressen en Gebouwen*) of the 2019 building dataset is used. However, we noticed that several BAG buildings have different shape and size compared to the AHN3 buildings due to different data acquisition time. Thus, the RMSE of the corners is calculated for unchanged building.

In this research, the required positional accuracy for the outline result is at least 1-meter. For the Makassar dataset, the average RMSE for 36 buildings in the study area is ~ 65 cm, which meets our requirements. There is an exception for one incomplete building (indicated by a red circle in Figure 5.11.b), that has a RMSE of more than one meter. In this case, the building roof is partially covered by dense trees resulting in poor building segmentation. Based on the number of corners from the reference data, the precision, recall and F1-score of the Makassar test set are 0.99, 0.95, and 0.97 respectively.

Building outline results for the EYE-Amsterdam dataset, as shown in Figure 5.12.d, are also good at 0.7 meter RMSE on average. For this dataset, there are no classification issues as we use available building points provided by the AHN3. In this case, one of the biggest factors that influences the accuracy of the result is the definition of building roof, in case an overhanging roof exist. This means that the overhanging roof is likely included in a building in AHN-3 data but not in the BAG data, which results in discrepancies. Figure 5.13.b shows overhanging roofs on one building in the EYE-Amsterdam area (marked as **A** in Figure 5.13.a). Another issue is found on a building with a curved building outline. The algorithm could not detect points on a curved line, as the normal angle differences δN between edge points is small (less than five degrees). Decreasing the normal angle differences δN threshold would not always mitigate this problem, as it will increase the number of false corners due to noise of the edge points.



(**a**) (**b**)

(**c**) (**d**)

Figure 5.12 Study areas and building outline results. (**a**) Classified point cloud of the Makassar dataset consists of buildings (orange), ground (pink), and vegetation (green); (**b**) Comparison of the Makassar outline result (black line) to the Digital Surface Model (DSM) image; (**c**) Classified AHN3 point cloud of the EYE – Amsterdam dataset consists of buildings (orange), ground (pink), water (blue), and unclassified points (gray); (**d**) Comparison of the EYE-Amsterdam outline result (black line) to the Digital Surface Model (DSM) image.

(**a**) (**b**)

Figure 5.13 Comparison of Bag building polygon (red) and the AHN3 building points (grey) in case of overhanging roof. (**a**) Top view building definition of BAG polygon (red) and the AHN3 building points (grey); (**b**) Building "**A**" has overhanging roof (red circle) ©GoogleStreetView.

5.4.2 General overview

Compared to our previous study on building outline extraction using ordered points aided Hough transform (Widyaningrum et al., 2018), the MAT approach has higher sensitivity to noisy edge points and variations in point density. Small bumps on the edges affect the normal direction of the corresponding point, which later affects the corner detection result. However, the MAT approach has a similar accuracy as our previous work at 0.7 meter RMSE for the MAT-based method and 0.57 for the OHT method. Our proposed method is able to handle one of the shrinking ball algorithm limitations that requires the surface normal for each sample point (Ma et al., 2012).

The use of the alpha-shape algorithm makes it possible to orient the normals of each edge point automatically when performing inner and outer MAT computation. Inner and outer MAT points together effectively detect all building corners.

Figure 5.14 shows the ability of our method to detect corners of various building shapes with different point density. Figure 5.14 row 1 demonstrates how our algorithm works on sparse building edge points that later result in false segments. The red circle in Figure 5.14.b row 1 shows three medial segments around a corner. Two of them are considered as false segments. However, the algorithm delivers the correct corner points only (inside the red circle in Figure 5.14.c row 1) even though false segments exist. The method also works in case small perturbations exist on the boundary e.g. due to trees (see orange circles in Figure 5.14.a row 3). There is one MAT point (the red point inside the yellow ellipse in Figure 5.14.a row 3) resulting from a noisy edge part, that is correctly ignored by the medial segmentation procedure. On the other hand, the algorithm may fail to extract non-linear shapes like rounded edges (see blue circles in Figure 5.14. row 2) as the algorithm cannot detect significant normal change for boundary points on the rounded edge. In this case, the algorithm can only detect two candidate corners from two medial segments, positioned at both ends of the rounded edge.

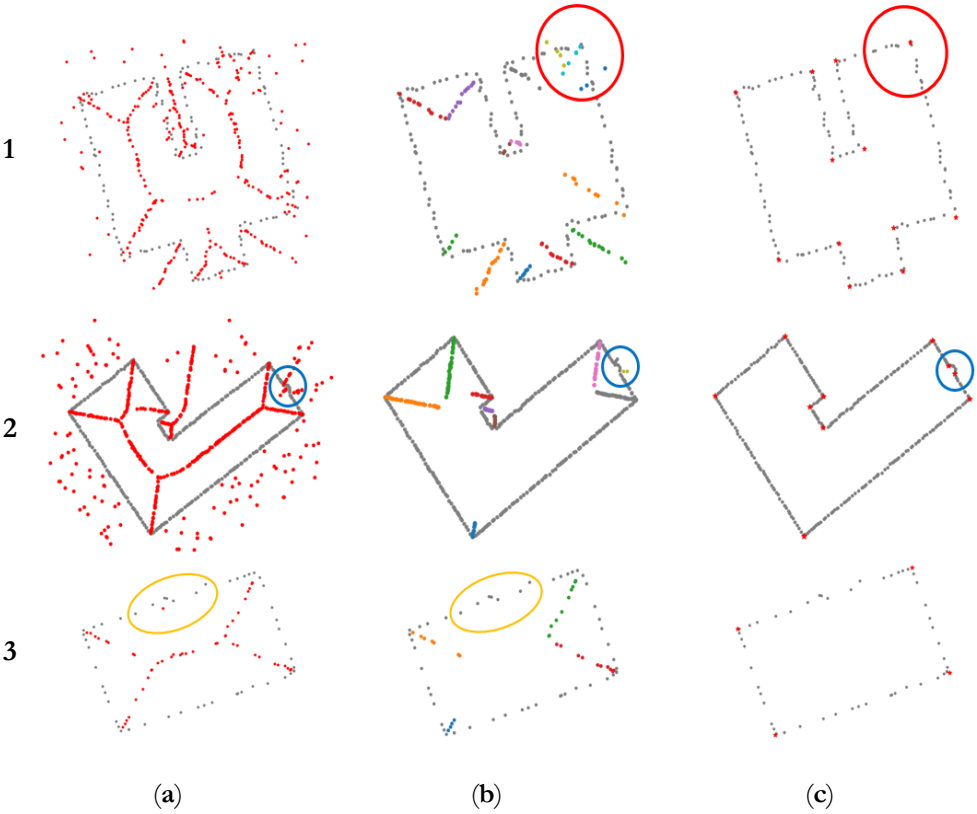
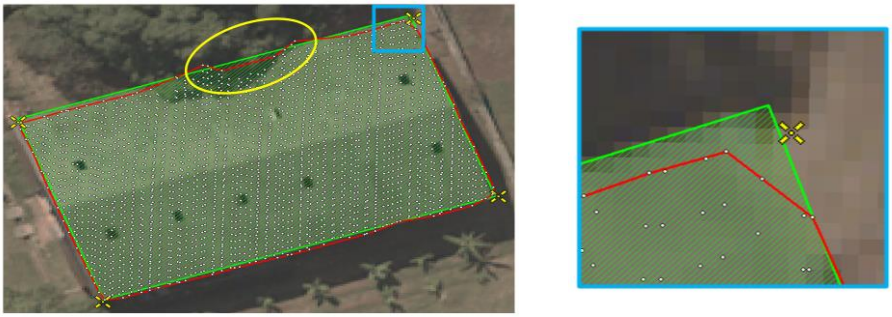


Figure 5.14 Outline results for different building shapes. (**a)** The MAT points (red) of the edge points (gray); (**b)** Segmented MAT point; (**c**) Resulting corner points (red).

As shown in figure 5.15.a, our method is able to obtain building corners (yellow points) that are close to the reference building polygon (green) even when there is a perturbation in the boundary (inside the yellow ellipse). The proposed method, particularly, improves the alpha-shape outline (red). Figure 5.15.b compares between corners from our method, the alpha-shape outline result, and the building outline reference.

In addition, we found out that several MAT properties have not been investigated yet, particularly for the outline extraction, e.g. the curvature of consecutive MAT points indicating an asymmetric shape of two edges of different length, may be used for locating but also characterizing corners accurately.



(**a**) (**b**)

Figure 5.15 Corner identification comparison between our proposed method and the reference and the alpha-shape outline in Makassar study area. (**a**) Corners extracted from our method (yellow) compare to alpha-shape outline (red line) and the building map polygon (green polygon); (**b**) Zoomed-in of the blue rectangle in Figure 5.15.a.

5.4.3 Comparison analysis

In this section, we compare results of our proposed method to those of existing methods on building outline extraction. For this purpose, a small subset of the AHN3 airborne point cloud of Amsterdam is used. The two other building outline extraction methods applied to this test area are: the Ordered Hough Transform (OHT) method proposed by Widyaningrum et al. (2018) and a RanSAC-based segmentation and regularization method by Lucas and Van Tilburg (2019). All these methods use an alpha-shape algorithm to select boundary points, which later result in building corners. The RanSAC-based segmentation and regularization method requires primary building orientations as it regularizes all boundary lines with respect to these orientations and its perpendicular orientations. Another regularization approach, OHT, applies the so-called extended Hough transform on a list of ordered boundary points that enable to detect arbitrary building directions and extract different boundary segments.

The comparison metrics considered are the building corners geometric accuracy (RMSE), the computation time of building outline extraction after the boundary points are selected, and the corner detection accuracy in terms of recall, precision, and F1-score.

As shown in Table 5.1, our proposed method has the highest geometric accuracy as well as F-1 score. The RanSAC-based method has higher Recall value as our proposed method, but is likely to have more undetected corners. However, the proposed method has the lowest number of false positive corners. The average computation time for the three methods is considered comparable, although in fact the proposed method is the fastest. However, each method may have different strengths and weaknesses when it is applied on a complex building roof shape.

Table 5.1 Evaluation of different building outline extraction methods

Approach	RMSE (m)	Corners Detection Accuracy			Avg comp time (sec)
		Precision (%)	Recall (%)	F1-score (%)	
Hough-based (Widyaningrum et al., 2018)	0.434	89.48%	95.15%	91.91%	2.10
RanSAC-based (Lucas & Van Tilburg, 2019)	0.548	91.89%	96.14%	93.65%	1.90
MAT-based (the proposed method)	0.414	94.17%	94.55%	93.82%	1.71

Using the building shown in Figure 5.16 as one example, a reference corner inside the brown circle is detected by OHT, but not by the proposed method. The RanSAC-based method results in two corners with an offset close to 1-meter from the reference. The proposed method fails to detect one building corner (inside the brown circle) due to a wide angle between two consecutive building outline segments that is close to 180°. In this case, the shrinking circle failed to produce a separation angle α close to 90° needed for corner detection.

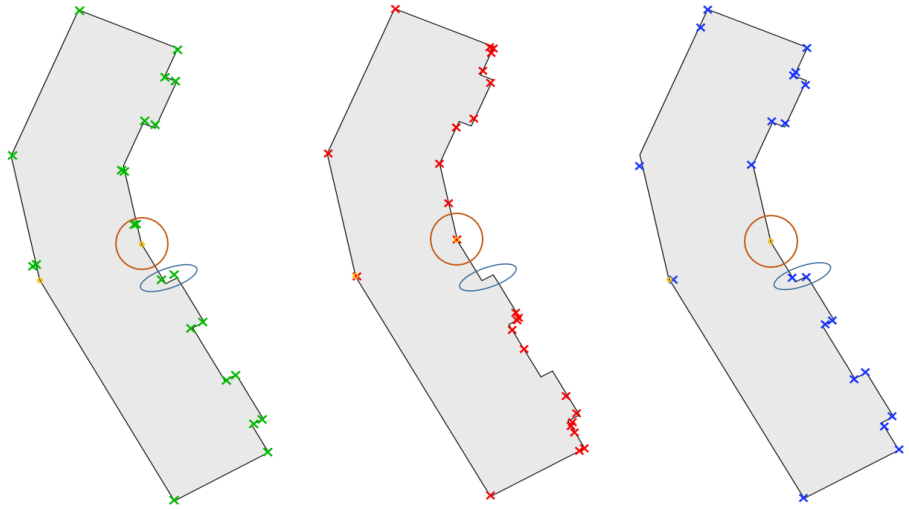


Figure 5.16 Building corners resulting from three different methods compared to building reference (gray polygon). Results notably vary within the brown circle and blue ellipse. (**a**) Corners extracted by the RanSAC-based method (green crosses); (**b**) Corners extracted by the OHT method (red crosses); (**c**) Corners extracted by the proposed method (blue crosses).

Another difference in the results in Figure 5.16 is indicated by the blue circles. The OHT method fails to detect two corners of a short building edge of one meter length, while the RanSAC-based method as well as our proposed method successfully detect both corners. All methods produce false corners but the proposed method has the smallest number of false corners.

5.4.4 Computational and complexity analysis

We implemented our method in Python2.7 on an Intel Core 2Duo CPU with 2.4 GHz processors. The computation time for our corner detector algorithm varies from 0.28 to 0.99 second per building, depending on the building size, shape and point density.

Recall that the proposed method has four main steps: boundary point selection by an alpha-shape algorithm, MAT extraction by a shrinking circle approach, MAT points segmentation, and estimation of corners by PCA. However, in case building points are not yet available, an additional classification and segmentation step is required. One of the most common method for 3D point cloud segmentation is seeded region growing. This type of algorithms selects a seed point and adds a point from the neighborhood if it meets a certain criterion. As reported by Shih and Cheng (2005), and Deschaud and Goulette (2010), the computational complexity of seeded-based region growing for partitioning a point cloud consisting of n 3D points into N segments is $O(n\ log\ n)$. Rabbani et al. (2006) improved plane detection using a smoothness constraint based on the normal angle difference between neighboring points and reported a time complexity of $O(n + N\ log\ n)$.

The proposed algorithm works on individual building segments. Suppose a segment contains u number of building points, the computational complexity of our four main steps are as follows:

1. The alpha-shape algorithm for selecting boundary points using Delaunay triangulation, as reported by (Varytimidis, 2016), has a time complexity of $O(u\ log\ u)$.
2. Given v boundary points extracted by, e.g. the alpha-shape algorithm, the shrinking circle algorithm used a KD-Tree to search its nearest point with time complexity of $O(v\ log\ v)$. The shrinking circle algorithm has a computational complexity of $O(v^2)$ because the algorithm may have to visit all medial axis points c_p for every boundary point (Ma et al., 2012).
3. Medial axis points segmentation has a time complexity of $O(v^2)$, but performance in practice is close to $O(v)$. Each MAT point m is checked for its median value based on its corresponding boundary points (p_{idx}, q_{idx}) which takes $O(v)$ time. The selection of candidate corner points based on the angular differences between two neighboring boundary points (p, q) takes $O(v)$ as well. The selection of MAT points m belonging to the same segment based on the similarity of median value of two boundary points (p, q) and corner point indices

also has a time complexity of $O(v^2)$, which leads to an overall time complexity of $O(v^2)$.

4. The overall time complexity of PCA is $O(d^3 + d^2 m)$ where m is the number of sample points and d is the number of features (Makhoul et al., 1999). Here, $d = 2$. For estimating the corner position corresponding to zero radius on a line fitted by the PCA, the time complexity is therefore linear in the number of sample points.

If all points of a building segment were boundary points, the computational complexity would be $O(v^2)$ time. This computational complexity for the total procedure of medial axis segmentation is not a problem because the algorithm works just on the boundary points of a single building. Point cloud segmentation is in practice the heaviest computation as it involves the complete 3D point cloud data. For example, in our Makassar test area, the building segmentation step considered all 464.191 points, while the medial axis segmentation considers 43 individual buildings. The number of points per-building in the Makassar test area varies from 96 to 3185 points, that later results in a number of building boundary points varying from 23 to 156 points. Note that the processing of individual buildings can also be parallelized easily.

5.5 Conclusion and future works

In this research, we have presented a procedure for automatically extracting building outlines from airborne point clouds based on the MAT descriptors generated by the 2D shrinking circle method. Our approach takes advantage of MAT invertibility with its medial axis and the corresponding radius function that allows reconstructing the exact object shape. Building classification is conducted first. A set of building boundary points is then extracted using an alpha-shape algorithm. After applying a shrinking circle algorithm to the input boundary points, MAT points are obtained. To identify corners, we introduce a corner-aware segmentation to group MAT points to their corresponding medial branch. The segmentation combines both global thresholds and several MAT-derived features. Next, the algorithm fits a line to all MAT points of a segment. Based on the corresponding radii of the MAT points, the corner point location is estimated by extrapolating the position where the radius is zero on the fitted line.

The positional accuracy results of the estimated corner points indicate that our method provides a completely new and promising tool for reconstructing the geometric shape of building roofs from scattered airborne point clouds by using the MAT approach. The proposed method performance is highlighted on a number of complex building shapes in and around the EYE building in Amsterdam, The Netherlands. The ability of the proposed method to obtain accurate corners and complete shapes indicates the robustness of our method to small perturbations on the building edge. In case of sparse point intervals, densification of edge points may help to increase the

accuracy of the MAT result. Meanwhile, our method has limitations to obtain outlines of an object with a curved or circular shape. In comparison to state-of-the-art methods on building roof outline extraction, our proposed method shows a promising result in acquiring accurate building corners geometrically. Compare to RanSAC and Hough transform based methods, our method is a primitives-free approach that does not require orientation initialization.

Though current skeletonization methods show a progressive development, deployment for wider applications is still challenging. Different applications may require different skeletonization methods and/or MAT descriptors. For future work, we will consider extending the MAT technique for reconstructing other digital map objects, such as road networks, ridges, or streamlines. The application of the proposed workflow for extracting curved lines and for a larger area is also interesting to be investigated further.

6

Conclusions and Recommendations

The main objective of this research is to develop automatic methods to extract object outlines for digital mapping using ALS (Airborne Laser Scanning) point cloud data in possible combination with aerial images. In this chapter, we present the key conclusions (Section 6.1), list the main contributions (Section 6.2), and recommendations for further research (Section 6.3).

6.1 Conclusions

This research presents methodology to automatically extract the outlines of two types of urban man-made objects, buildings and roads. To accelerate map production and minimize human assistance, automatic delineation of topographic objects is found to be the key challenge to overcome. Airborne laser scanning point clouds in combination with RGB color from aerial images are used to increase the degree of automation as well as to maintain the accuracy of the results.

This research concludes that building and road outlines can be automatically extracted directly from point cloud data by combining and developing state of the art techniques from machine learning, remote sensing, computer vision, and computational geometry. Our workflow consists of point cloud classification, segmentation, and outline extraction. For classification, we implement a point-wise deep learning technique and use additional RGB information to increase the accuracy of the results. Tests demonstrate that we are able to obtain high classification accuracy results in terms of precision and recall. Object extraction is then performed starting from the previously classified points to obtain smooth building and road outlines. In point clouds, object boundaries are unstructured and not as straight as seen in aerial images. To mitigate the effect of jaggy and imperfect building edges, we introduce two novel methods for line regularization, ordered Hough transform and medial axis transform-based method, both resulting in straight and smooth polygon outlines. We propose skeleton-based methodology for automatic road outline extraction completed by a novel gap filling method to provide a complete and continuous road network. This research is practical

in its contribution to automatic object delineation resulting in cheaper and faster map production.

To answer the main objective, we defined three research questions specified by sub-questions as listed in Chapter 1. Discussions answering the research questions are presented in the following.

Research question 1 – How to accurately classify huge point cloud data into several classes in a way feasible for routine map production using deep learning?

Classification partitions a point cloud into several predefined object classes. This step is a prerequisite for object outline extraction. Routine map production involves huge amount of input data and therefore requires effective scene classification. Current most quickly developing classification approach, deep learning, shows promising results in handling large amount of data. Given an ALS point cloud enriched by an aerial orthophoto dataset, the third research question is specified in the following two sub-questions.

Sub question 1.1 – How to effectively utilize additional features from aerial images to increase the accuracy of ALS point cloud classification?

ALS point clouds and aerial images are two complementary data sources for detailed and accurate mapping. Aerial images are rich with spectral features, while ALS point clouds directly provide 3D geometric features. Optimal feature combination for airborne point cloud classification using a deep learning approach incorporating RGB color information from remote sensing images has not yet been discussed and tested thoroughly, because deep learning parameters are not intuitive to implement for real world applications.

We investigate four different input combinations of input features from ALS point clouds and aerial orthophotos to evaluate each feature contribution to the point cloud classification. Input features used in this research include a combination of off-the-shelf LiDAR (3D coordinates, intensity value, return number and number of returns) and image features (RGB color) as well as tailored features (normalized 3D coordinates). The feature combinations considered are: (i) 3D coordinates, RGB, and normalized coordinates; (ii) 3D coordinates, LiDAR features, and normalized coordinates; (iii) 3D coordinates, two color channels (Red and Green) and LiDAR intensity; and (iv) 3D coordinates, RGB and LiDAR features, excluding normalized coordinates. Based on the evaluation metrics, we found that combination (iv), combining LiDAR and RGB color, achieves the highest overall accuracy (92%) as well as F1 score for all classes. Other feature sets obtain an overall accuracy within the range of 84% – 86%. The use of full RGB orthophoto colors in combination with the LiDAR point cloud significantly improves the classification results.

Sub-question 1.2 - How to provide good and cheap training samples for ALS point cloud classification?

Deep learning requires a large amount of high quality training samples. The generation of training samples for accurate classification results, especially for airborne point cloud data, is a non-trivial task. In Chapter 3, we provide a method for creating high-quality free training samples for 3D semantic segmentation of new airborne point cloud data using an existing 2D vector base map for four land cover classes: bare land, buildings, trees, and roads.

An existing base map is a useful source of information to label training data. However, there are several challenges to extract free 3D training samples using a 2D base map. First, the base map data used to label the point cloud may not fully provide all required object polygons, especially for trees. Second, as we use base map polygons to label points, the labeled building and road points may include many mislabeled points, as there are trees covering buildings or roads.

In this study, the labeling procedure is started with point cloud filtering to separate ground and non-ground points. From the non-ground points, building points are labelled using building polygons of the base map. Using the same method, road points are labelled from the ground points, and remaining points are labelled as bare land. Next, tree points are identified by surface roughness and height filtering. As several building polygons may contain tree points, we apply roughness filtering to eliminate tree points labelled as building.

The proposed point cloud labeling procedure delivers good classification results of 84% up to 94% overall accuracy depending on the feature combination. The detection rate for buildings has the highest F1-score (between 87% and 95%), while the detection rate for roads gives the lowest F1-score (between 75% and 84%). Roads have the smallest detection rate probably because roads cover less area compared to the other classes. The labeling procedure is considered cheap, and fast as it required less than eight hours to label all points in our study area of size ± 5 km^2.

Research question 2 – How to accurately extract complete and smooth road networks from given road points?

On maps, road networks are usually represented by centerlines or outlines or both. A road network is characterised by a connected line arrangement, enabling people to move from one location to another. Such connectivity is known as network topology. Therefore, it is important that a road extraction method preserves topology and the local connectivity of a road network. Another challenging task in road network extraction is to extract a network completely, which is difficult, as roads are often

occluded by cars or dense trees. Given the road extraction challenges, the second research question is addressed by the following two sub-questions.

Sub-question 2.1 - How to obtain a complete and accurate road network from given segmented road points, where these road points may be affected by gaps and noise?

Several existing road extraction methods are susceptible to introduce wrong road branches due to road gaps and noise. Therefore, in Chapter 3, we introduce a practical gap-free approach for extracting complete road networks from segmented road points using a skeletonization approach. Skeletonization provides an effective and compact representation of objects by reducing their dimensionality to a skeleton or centerline while preserving objects topology and geometric properties. On the classified road points, DBSCAN clustering is applied to remove noise and outliers. Next, we convert road points into a binary image using a kernel density estimator. We found that such a kernel density estimator is useful to partly fill small gaps on the road caused by cars. To extract road centerlines, an existing fast parallel thinning algorithm is used. Parallel thinning captures the centerline of a road while its so-called deletion rule accommodates topology preservation. Hence, the topological correctness of the centreline result is guaranteed.

As ALS point clouds provide bird's eye view of objects, road points can be occluded by dense trees or cars which cause unwanted gaps on the road. Such gaps may cause incomplete road extraction results that disrupt and disconnect the network. In our case, most disruptions are caused by dense trees. Therefore, we propose a novel road completion method based on a tree-constrained approach. To fill road gaps, tree polygons need to be extracted first. Dangle point identification is performed to detect road gaps. A dangle point is defined as the end-point of a line which is not connected to any other line. The algorithm adds lines to any road gap based on the intersection of the dangle points and tree polygons within a specified distance. Our road completion technique effectively fills in most road gaps in our study area. The results demonstrate that the procedure is applicable in obtaining complete coverage of the road network while maintaining its topology.

Sub-question 2.2 - How to obtain outlines representing the actual road borders?

Road outlines are obtained, based on the actual road width estimated by the medial axis approach sketched above. Width-based buffering is applied to obtain road polygons. The road width is estimated by an Euclidean distance transform of the skeleton to the boundary. The distance transform finds the minimum distance between any road pixel and the closest non-road pixel of the binary road image. Such method results in another type of skeleton that encodes the maximum distance between any skeleton pixel and its corresponding edge pixel. The estimated width is assigned to the smoothed centerline by a spatial join. Finally, the smoothed centerline segment is buffered according to the

estimated width. The results show that the method is efficient and easy to implement in providing road borders according to its actual width. Although road width deviations of up to two meter happen at a specific highway example, in general, the boundaries of our road polygons have less than one meter difference from the reference.

Research question 3 – How to accurately extract straight and smooth building outlines from given building points?

Buildings, in most cases, have planar surfaces bounded by smooth boundaries. The first research question is related to the methodology for extracting building outlines from unstructured point cloud data whereas specific issues are addressed by the following two sub-questions.

Sub-question 3.1 - How to mitigate the effect of noise and flaws on the building edges?

LiDAR points on building edges are in general irregular and noisy. Existing bounding hull methods have limitations to deal with noisy edges of buildings of complex shape. For example, the concave hull or alpha-shape results in jagged building outlines while the convex hull is not suitable to represent edge points of a building of concave shape. Trees near buildings sometimes heavily affect the detection of building roofs, leading to disrupted building edges and non-smooth outlines.

To overcome the effect of noise and flaws on building edges, in Chapter 4, we proposed an outline extraction method to obtain straight and smooth building outlines that preserve the original building shape and size. The ordered Hough transform (OHT) method is developed as an extension of the classical Hough transform by exploiting the order of points forming the building outline. From given building edge points obtained by a *k*-nn concave hull method, a Hough accumulator matrix is constructed to store the number of edge points voting for the same parametric model. Dominant building directions are estimated by the variance of angles. Hierarchical filtering and clustering is performed to detect parameters having most votes (hotspots). A matrix containing a list of ordered edge points enables the detection of line segments of arbitrary direction, resulting in straight and smooth building roof polygons.

In case noise and small flaws exist in the building outline data, the voting scheme of the Hough transform enables the preserving of the actual building shape and size. We applied the OHT method in three different urban areas of different characteristics: Makassar - Indonesia, Vaihingen - Germany, and Amsterdam - the Netherlands. We compared the proposed method to other building outline extraction methods, also using ALS point clouds as input, on the Vaihingen benchmark dataset. Comparison shows that our OHT method has best correctness (99.4%) and quality (89.6%) metrics on the given test set. The higher quality metric indicates that our method delivers complete and accurate building polygon results.

Sub-question 3.2 - How to accurately acquire complex building outlines of arbitrary shape?

Buildings are ubiquitous man-made objects which sometimes have arbitrary shape and size. Existing methods have limitations in extracting short edges or in correctly detecting multiple building orientations which lead to inaccurate outline results. Therefore, there is a need for a simple and primitive-free approach to extract building outlines. A reliable skeleton-based shape descriptor, the medial axis transform (MAT), has been used for various shape analysis tasks, such as the approximation, recognition and retrieval of shapes as well as topology representation and data reduction of complex geometric models.

In Chapter 5, we introduced a MAT-based building outline extraction method that works directly on point clouds. Medial axis-based shape descriptors have strong advantages for the efficient and invariant geometrical representation of shapes and perform well in object reconstruction because they preserve the complete boundary information of a given shape. To obtain MAT skeleton points, a shrinking circle algorithm is applied to the building boundary points. To identify corners, we propose a corner-aware skeleton segmentation method to group MAT points belonging to the same medial skeleton branch. Next, the algorithm fits a line to all MAT points of a skeleton segment. Based on the corresponding radii of the MAT points, the corner point location is estimated by extrapolating the position where the MAT radius is zero, which in theory coincides with the corner of a shape.

We evaluate our MAT-based method by comparing it to other building outline extraction methods, one RANSAC-based method and one Hough transform-based method (Chapter 4). For this purpose, classified AHN3 airborne point clouds of Amsterdam are used. Three comparison metrics considered are the building corners geometric accuracy (root mean square error), the computation time, and the corner detection accuracy in terms of recall, precision, and F1-score. The MAT-based approach has the highest geometric accuracy (RMSE = 0.4 m) and F-1 score (93.8%) which demonstrates the robustness of our MAT-based method in predicting the building corners. The average computation time for the three considered methods is comparable, although in fact the proposed method is the fastest. Compare to the RANSAC and the Hough-based methods, the MAT method is a primitive-free approach that does not require orientation initialization. Thus, the MAT-based method shows a promising result in acquiring building corners for complex roof structures of arbitrary shape.

6.2 Main contributions

The four main contributions of this research are summarized as follows.

1. We prove that the use of RGB color from aerial orthophotos can significantly increase the overall accuracy of airborne point cloud classification which at the same time shows the robustness of point-wise deep learning approaches to the effect of false RGB color due to relief displacement, see Section 3.5.1 and Section 3.5.3.
2. We introduce and validate an efficient procedure to provide good and cheap training samples for ALS point cloud classification using deep learning.
3. We propose the first skeleton-based road centerline extraction method which provides a complete procedure to fill road gaps and to extract road boundaries using corresponding estimated road width, see Section 3.4.2.
4. We propose new methods on 2D building outline extraction from ALS point clouds based on two different approaches, Hough-based and skeleton-based, as presented in Section 4.3 and Section 5.3, respectively. These methods are evaluated and were shown to outperform existing state-of-the-art building outline extraction methods in several aspects, in particularly, in terms of correctness and quality.

List of publications

Journal articles:

1. **Widyaningrum**, E., Bai, Q., Fajari, M. K., and Lindenbergh, R. C. (2021). Airborne laser scanning point cloud classification using the DGCNN deep learning method. *Remote Sensing*, 13(5), 859.
2. **Widyaningrum**, E., Peters, R. Y., and Lindenbergh, R. C. (2020). Building outline extraction from ALS point clouds using medial axis transform descriptors. *Pattern Recognition*, 107447.
3. **Widyaningrum**, E., Gorte, B., and Lindenbergh, R. C. (2019). Automatic building outline extraction from ALS point clouds by ordered points aided hough transform. *Remote Sensing*, *11*(14), 1727.

Conference proceedings:

1. **Widyaningrum**, E., Fajari, M. K., Lindenbergh, R. C., and Hahn, M. (2020). Tailored features for semantic segmentation with a DGCNN using free training samples of a colored airborne point cloud. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 43, 339-346.

2. **Widyaningrum**, E. and Lindenbergh, R.C. (2019). Skeleton-based automatic road networks extraction from a colored orthophoto point cloud. In: *Proceedings of* the 40th *Asian Conference on Remote Sensing.* (**The Shunji Murai – Best Paper Award**, ACRS 2019, Daejeon – South Korea).

3. *Chen, Y., Gao, W., **Widyaningrum**, E., Zheng, M., & Zhou, K. (2018). Building classification of VHR airborne stereo images using fully convolutional networks and free training samples. *The International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 42(4). (**Best Poster Award**, ISPRS TC-IV Symposium 2018, Delft – the Netherlands).

4. **Widyaningrum**, E., Lindenbergh, R. C., Gorte, B. G. H., and Zhou, K. (2018). Extraction of building roof edges from LiDAR data to optimize the digital surface model for true orthophoto generation. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *42*(2). (**Best Young Author Paper Award**, ISPRS TC-II Symposium 2018, Riva del Garda – Italy).

5. Zhou, K., Gorte, B., Lindenbergh, R., and **Widyaningrum**, E. (2018). 3D building change detection between current VHR images and past LiDAR data. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42(2).

6. **Widyaningrum**, E., and Gorte, B. G. H. (2017). Comprehensive comparison of two image-based point clouds from aerial photos with airborne LiDAR for large-scale mapping. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42, 557-565.

7. **Widyaningrum**, E., and Gorte, B. G. H. (2017). Challenges and opportunities: one stop processing of automatic large-Scale base map production using airborne LiDAR data within GIS environment case study: Makassar city, Indonesia. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42, 365.

8. Octariady, J., Hikmat, A., **Widyaningrum**, E., Mayasari, R., and Fajari, M. K. (2017). Vertical accuracy comparison of digital elevation model from LiDAR and multitemporal satellite imagery. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42, 419.

9. **Widyaningrum**, E., Fajari, M., and Octariady, J. (2016). Accuracy comparison of VHR systematic-ortho satellite imageries against VHR orthorectified imageries using GCP. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *41*.

* All authors contributed equally

6.3 Recommendations for future work

The work for automatic object outline extraction from airborne laser scanning point clouds data provides insights on new developments and research. Recommendations are presented as follows.

First, we recommend an extension of the present outline extraction work that should consider the implementation and performance evaluation of the proposed method for a larger area with multiple map scales (varioscale). The application of the proposed workflow for extracting curved or circular outlines should also be investigated further. Furthermore, automatic 3D models generation should also be considered as future development of this study.

For road extraction, to smooth all centerlines at once is still problematic as curvy roads require different thresholds than straight roads. A more efficient line smoothing method or procedure requires further study. Through current skeletonization methods show progressive development, deployment for wider applications is still challenging. Different applications may require different skeletonization methods. For future work, we propose to extend the MAT technique for extracting and reconstructing other topographical objects, such as tree crowns, mountain ridges, or river channels.

Our research indicates that 3D deep learning matured so much that it is now actually able to extract geometric information as required for digital maps at near-operational quality, but in a much shorter time than traditional workflows. To increase applicability, research on the usability of a model trained for a specific task as the starting point for a model for another task (transfer learning) needs to be investigated further. The same holds for the reuse of a trained model at another study area (domain shifting) of different characteristics.

We have demonstrated the use of point-wise semantic segmentation to classify ALS point clouds followed by a post-processing step to obtain object outlines. Specific methods for application on ALS point clouds which can directly detect, segment and delineate individual objects, also referred to as instance and panoptic segmentation are expected to be designed in coming years.

In chapter 3, our evaluation results confirm that the use of RGB information from co-temporal aerial orthophotos significantly improves ALS point cloud classification. However, it is still worth to study the combination of ALS point clouds with very high resolution satellite images as an alternative, in case airborne orthophotos are not available. In addition, we recommend further investigation on the combination of ALS point clouds with airborne or spaceborne images acquired at different times.

Finally, any automation process is unlikely to deliver 100% correctness and completeness. Therefore it is still required to perform data checking and editing. For routine production, it is worth the effort to provide methods or tools to facilitate and accelerate such editing task. In addition, it is necessary to study which completeness and correctness values are sufficient for automatic acceptance of processing results.

Bibliography

Agouris, P., Doucette, P., and Stefanidis, A. (2004). Automation and digital photogrammetric workstations. *Manual of photogrammetry*, 949-981.

AHN3 point clouds data, obtained from https://www.pdok.nl/nl/ahn3-downloads, accessed on October 2018.

Ahuja, N., and Chuang, J. H. (1997). Shape representation using a generalized potential field model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2), 169-176. DOI: 10.1109/34.574801.

Albers, B., Kada, M., and Weichmann, A. (2016). Automatic extraction and regularization of building outlines from airborne LiDAR point clouds. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *41*, DOI: 10.5194/isprsarchives-XLI-B3-555-201.

Akbulut, Z., Özdemir, S., Acar, H., and Karsli, F. (2018). Automatic Building Extraction from Image and LiDAR Data with Active Contour Segmentation. *Journal Indian Society of Remote Sensing*, *46*, 2057-2068.

Amenta, N., Choi, S., and Kolluri, R. K. (2001). The power crust. In: *Proceedings of the sixth ACM symposium on Solid modeling and applications*, 249-266. DOI: 10.1145/376957.376986

American Society of Photogrammetry. (1980). Manual of Photogrammetry, 4th edition, Falls Church, Va.

Asaeedi, S., Didehvar, F., and Mohades, A. (2017). α-Concave hull, a generalization of convex hull. *Theoretical Computer Science*, 702, 48-59. DOI: 10.1016/j.tcs.2017.08.014.

Au, O. K. C., Tai, C. L., Chu, H. K., Cohen-Or, D., and Lee, T. Y. (2008). Skeleton extraction by mesh contraction. *ACM transactions on graphics* (TOG), 27(3), 1-10. DOI: 10.1145/1360612.1360643.

Awrangjeb, M. (2016). Using point cloud data to identify, trace, and regularize the outlines of buildings. *International Journal Remote Sensing*, *37*, 551-579, DOI: 10.1080/01431161.2015.1131868.

Awrangjeb, M., Lu, G., and Fraser, C. (2014). Automatic building extraction from LiDAR data covering complex urban scenes. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40, 25. DOI: 10.5194/isprsarchives-XL-3-25-2014.

Axelsson, P. E. (2000). DEM generation from laser scanner data using adaptive TIN models. *The International Archives of Photogrammetry and Remote Sensing*, 32, 110–117.

BAG data - Gebouwen (INSPIRE geharmoniseerd), obtained from http://geodata.nationaalgeoregister.nl/bag/wfs?service=WFS&request=GetCapabilities, accessed on May 2019.

Bai, X., Latecki, L. J., and Liu, W. Y. (2007). Skeleton pruning by contour partitioning with discrete curve evolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3), 449-462. DOI: 10.1109/TPAMI.2007.59.

Balado, J., Martínez-Sánchez, J., Arias, P., and Novo, A. (2019). Road environment semantic segmentation with deep learning from MLS point cloud data. *Sensors*, *19*(16), 3466. DOI: 10.3390/s19163466.

Ballard, D. H. (1981). Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, *13*(2), 111-122. DOI: 10.1016/0031-3203(81)90009-1.

Bastani, F., He, S., Abbar, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Madden, S., and DeWitt, D. (2018). RoadTracer: Automatic extraction of road networks from aerial images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4720-4728.

Bauer, J., Karner, K., Schindler, K., Klaus, A., and Zach, C. (2005). Segmentation of building from dense 3D point-clouds. In: *Proceedings of the ISPRS. Workshop Laser scanning Enschede*, 12-14.

Bello, S. A., Yu, S., Wang, C., Adam, J. M., and Li, J. (2020). Deep learning on 3D point clouds. *Remote Sensing*, *12*(11), 1729.

Bendouda, M., and Berrached, N. (2018). Urban road network extraction from remote sensing images using an improved F* algorithm. *Journal of the Indian Society of Remote Sensing*, 46(7), 1053-1060. DOI: 10.1007/s12524-018-0773-3.

Bertasius, G., Shi, J., and Torresani, L. (2015). High-for-low and low-for-high: Efficient boundary detection from deep object features and its applications to high-level vision. In: *Proceedings of the IEEE International Conference on Computer Vision*, 504-512.

Bhanu, B., Lee, S., Ho, C. C. and Henderson, T., 1986. Range data processing: representation of surfaces by edges. In: *Proceedings of the Eighth International Conference on Pattern Recognition*, 236–238.

Biosca, J. M., and Lerma, J. L. (2008). Unsupervised robust planar segmentation of terrestrial laser scanner point clouds based on fuzzy clustering methods. *ISPRS Journal of Photogrammetry and Remote Sensing*, *63*(1), 84-98. DOI: 10.1016/j.isprsjprs.2007.07.010.

Bishop, Christopher (2006). *Pattern Recognition and machine learning*. Berlin: Springer. ISBN 0-387-31073-8.

Bláha, M., Vogel, C., Richard, A., Wegner, J.D., Pock, T., and Schindler, K. (2016). Large-scale semantic 3D reconstruction: An adaptive multi-resolution model for multi-class volumetric labeling. In: *Proceedings IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), Las Vegas, USA.

Blum, H. (1967). A transformation for extracting new descriptors of shape. Cambridge: MIT press.

Boyko, A., and Funkhouser, T. (2011). Extracting roads from dense point **clouds** in large scale urban environment. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(6), S2-S12. DOI: 10.1016/j.isprsjprs.2011.09.009.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5-32. DOI: 10.1023/A:1010933404324.

Brenner, C., 2000: Towards fully automatic generation of city models. *International Archives for Photogrammetry and Remote Sensing*, 33(B3), 85-92.

Broersen, T., Peters, R., and Ledoux, H. (2017). Automatic identification of watercourses in flat and engineered landscapes by computing the skeleton of a LiDAR point cloud. *Computers & Geosciences*, 106, 171-180. DOI: 10.1016/j.cageo.2017.06.003

Cai, J., Luo, J., Wang, S., and Yang, S. (2018). Feature selection in machine learning: A new perspective. *Neurocomputing*, 300, 70-79. DOI: 10.1016/j.neucom.2017.11.077.

Castillo, E., Liang, J., & Zhao, H. (2013). Point cloud segmentation and denoising via constrained nonlinear least squares normal estimates. In: *Innovations for shape analysis*, 283-299. Springer, Berlin, Heidelberg.

Carrio, A., Sampedro, C., Rodriguez-Ramos, A., and Campoy, P. (2017). A review of deep learning methods and applications for unmanned aerial vehicles. *Journal of Sensors*. DOI: 10.1155/2017/3296874. DOI: 10.1155/2017/3296874.

Chaussard, J., Couprie, M., and Talbot, H. (2009). A discrete λ-medial axis. In: *International Conference on Discrete Geometry for Computer Imagery*, 421-433. Springer. DOI: 10.1007/978-3-642-04397-0_36.

Chehata, N., Guo, L., and Mallet, C. (2009). Airborne LiDAR feature selection for urban classification using random forests. *The International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 38, Part 3/W8.

Chen, D., Zhang, L., Mathiopoulos, P. T., and Huang, X. (2014). A methodology for automated segmentation and reconstruction of urban 3-D buildings from ALS point clouds. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *7*(10), 4199-4217.

Chen, J., Zou, L. H., Zhang, J., and Dou, L. H. (2009). The Comparison and Application of Corner Detection Algorithms. *Journal of multimedia*, 4(6). DOI: 10.4304/jmm.4.6.435-441.

Chen, Y., Gao, W., Widyaningrum, E., Zheng, M., and Zhou, K. (2018). Building classification of VHR airborne stereo images using fully convolutional networks and free training samples. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42(4).

Choi, W. P., Lam, K. M., and Siu, W. C. (2003). Extraction of the Euclidean skeleton based on a connectivity criterion. *Pattern Recognition*, 36(3), 721-729. DOI: 10.1016/S0031-3203(02)00098-5.

Chuang, J. H., Tsai, C. H., and Ko, M. C. (2000). Skeletonisation of three-dimensional object using generalized potential field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11), 1241-1251. DOI: 10.1109/34.888709.

Cortes, C., and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273-297. DOI: 10.1007/BF00994018.

Couprie, M., and Bertrand, G. (2016). Asymmetric parallel 3D thinning scheme and algorithms based on isthmuses. *Pattern Recognition Letters*, 76, 22-31. DOI: 10.1016/j.patrec.2015.03.014.

Culver, T., Keyser, J., and Manocha, D. (2004). Exact computation of the medial axis of a polyhedron. *Computer Aided Geometric Design*, 21(1), 65-98. DOI: 10.1016/j.cagd.2003.07.008.

Dalitz, C., Schramke, T., and Jeltsch, M. (2017). Iterative Hough transform for line detection in 3D point clouds. *Image Process. Line*, *7*, 184-196, DOI: 10.5201/ipol.2017.208.

David Douglas and Thomas Peucker. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2), 112-122. doi:10.3138/FM57-6770-U75U-7727.

Decker, D. (2000). GIS data sources. Wiley.

Deng, R., Shen, C., Liu, S., Wang, H., and Liu, X. (2018). Learning to predict crisp boundaries. In: *Proceedings of the European Conference on Computer Vision* (ECCV), 562-578.

Devi, S. (2014). Veena. Measurement of Relief Displacement from Vertical Photograph. *The International Journal of Science, Engineering and Technology Research*, 3, 2800-2805.

Döllner, J. (2020). Geospatial Artificial Intelligence: Potentials of Machine Learning for 3D Point Clouds and Geospatial Digital Twins. *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 1-10.

Dorninger, P., and Pfeifer, N. (2008). A comprehensive automated 3D approach for building extraction, reconstruction, and regularization from airborne laser scanning point clouds. *Sensors*, 8, 7323–7343. DOI: 10.3390/s8117323.

Douglas, D. H., and Peucker, T. K. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: the international journal for geographic information and geovisualization*, 10(2), 112-122. DOI: 10.3138/FM57-6770-U75U-7727.

Duckham, M., Kulik, L., Worboys, M., and Galton, A. 2008. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recognition*, 41, 3224-3236. DOI: 10.1016/j.patcog.2008.03.023.

Duda, R.O., and Hart, P.E. (1972) Use of the Hough transform to detect lines and curves in pictures. *Communication ACM*, 15, 11-15. DOI: 10.1145/361237.361242.

Edelsbrunner, H., Kirkpatrick, D. G., and Seidel, R. (1983). On the shape of a set of points in the plane. *IEEE Transactions on Information Theory,* 29(4), 551-559. doi:10.1109/TIT.1983.1056714.

Eidenbenz, C., Käser, C., & Baltsavias, E. P. (2000). ATOMI-Automated reconstruction of Topographic Objects from aerial images using vectorized Map Information. In *XIX ISPRS Congress, Commission III*. Institute of Geodesy and Photogrammetry, ETH-Hönggerberg.

ENVI. Available online: https://www.harrisgeospatial.com/Software-Technology/ENVI (accessed on 3 July 2019).

Ester, M., Kriegel, H. P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining* (KDD-96).

Filin, S., and Pfeifer, N. (2006). Segmentation of airborne laser scanning data using a slope adaptive neighborhood. *ISPRS Journal of Photogrammetry and Remote Sensing*, *60*(2), 71-80.

Fischler, M. A., and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381-395.

Forkuo, E. K., and King, B. (2004). Automatic fusion of photogrammetric imagery and laser scanner point clouds. *The International Archives of Photogrammetry and Remote Sensing*, 35(2004), 921-926.

Gamba, P., Dell'Acqua, F., and Lisini, G. (2009). BREC: The Built-up area RECognition tool. In: *Proceedings of the 2009 IEEE Joint Urban Remote Sensing Event*, 1-5.

Galton, A., and Duckham, M. (2006). What is the region occupied by a set of points?. In: *International Conference on Geographic Information Science*, 81-98. Springer, Berlin, Heidelberg.

Ge, Y., and Fitzpatrick, J. M. (1996). On the generation of skeletons from discrete Euclidean distance maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(11), 1055-1066. DOI: 10.1109/34.544075.

Ge, L., Cai, Y., Weng, J., and Yuan, J. (2018). Hand pointnet: 3D hand pose estimation using point sets. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8417-8426.

Gerke, M., and Xiao, J. (2014). Fusion of airborne laser scanning point clouds and images for supervised and unsupervised scene classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, *87*, 78-92. DOI: 10.1016/j.isprsjprs.2013.10.011.

Ghosh, S., Das, N., Das, I., & Maulik, U. (2019). Understanding deep learning techniques for image segmentation. *ACM Computing Surveys (CSUR)*, *52*(4), 1-35. DOI: 10.1145/3329784.

Ghosh, S., and Lohani, B. (2013). Mining LiDAR data with spatial clustering algorithms. *International journal of remote sensing*, *34*(14), 5119-5135. DOI: 10.1080/01431161.2013.787499.

Giesen, J., Miklos, B., Pauly, M., and Wormser, C. (2009). The scale axis transform. In*: Proceedings of the twenty-fifth annual symposium on Computational Geometry*, 106-115. DOI: 10.1145/1542362.1542388.

Gilani, S. A. N., Awrangjeb, M.; and Lu, G. (2016). An automatic building extraction and regularisation technique using LiDAR point cloud data and orthoimage. *Remote Sensing*, 8, 258, DOI: 10.3390/rs8030258.

Golub, G. H. (1996). CF vanLoan, Matrix Computations. *The Johns Hopkins*.

Golubiewski, N., Lawrence, G., and Fredrickson, C. (2019). Constructing Auckland: 2013 Building Outlines Update in the Urban Core and Its Periphery. Auckland Technical Report TR2019/006, Auckland Council: Auckland, New Zealand.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). Deep learning, 1(2). Cambridge: MIT press.

Griffiths, D., & Boehm, J. (2019). A review on deep learning techniques for 3D sensed data classification. *Remote Sensing*, 11(12), 1499. DOI: 10.3390/rs11121499.

Grigorishin, T., Abdel-Hamid, G., and Yang, Y. H. (1998). Skeletonisation: An electrostatic field-based approach. *Pattern Analysis and Applications*, 1(3), 163-177.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J. and Chen, T., (2018). Recent advances in convolutional neural networks. *Pattern Recognition*, *77*, 354-377. DOI: 10.1016/j.patcog.2017.10.013.

Guercke, R., and Sester, M. (2011). Building footprint simplification based on Hough transform and least squares adjustment. In: *Proceedings of the 14th Workshop of the ICA Commission on Generalisation and Multiple Representation*, Paris, France.

Guerreiro, R. F., and Aguiar, P. M. (2012). Connectivity-enforcing Hough transform for the robust extraction of line segments. *IEEE Transaction Image Processing*, *21*, 4819-4829, DOI: 10.1109/TIP.2012.2202673.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., and Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, *187*, 27-48.

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., and Bennamoun, M. (2020). Deep learning for 3d point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* DOI: 10.1109/TPAMI.2020.3005434.

Haala, N., Brenner, C., and Stätter, C. (1997). An integrated system for urban model generation. *The International Archives of Photogrammetry and Remote Sensing*, 32, 96-103.

Han, W., Wang, R., Huang, D., and Xu, C. (2020). Large-scale ALS data semantic classification integrating location-context-semantics cues by higher-order CRF. *Sensors*, *20*(6), 1700. DOI: 10.3390/s20061700.

Hana, X. F., Jin, J. S., Xie, J., Wang, M. J., and Jiang, W. (2018). A comprehensive review of 3D point cloud descriptors. arXiv preprint arXiv:1802.02297.

Haunert, J. H., and Sester, M. (2008). Area collapse and road centerlines based on straight skeletons. *GeoInformatica*, 12(2), 169-191. DOI: 10.1007/s10707-007-0028-x.

Hensman, P., and Masko, D. (2015). The impact of imbalanced training data for convolutional neural networks. *Degree Project in Computer Science, KTH Royal Institute of Technology*.

Herout, A., Dubská, M., and Havel, J. (2013). Variants of the Hough transform for straight line detection. *Real-Time Detection of Lines and Grids*, Springer, London, UK, 17-34.

Herout, A., Dubská, M., and Havel, J. (2013). Review of Hough transform for line detection. In: *Real-Time dtection of lines and grids*; 3–16, Springer: London, UK. DOI: 1007/978-1-4471-4414-4_2.

Hinz, S., Baumgartner, A., and Ebner, H. (2001). Modeling contextual knowledge for controlling road extraction in urban areas. In: *IEEE/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas,* 40-44.

Hoehle, J. (2017). Generating topographic map data from classification results. Remote Sensing, 9, 224, DOI: 10.3390/rs9030224.

Höfle, B., and Rutzinger, M. (2011). Topographic airborne LiDAR in geomorphology: A technological perspective. *Zeitschrift für Geomorphologie, Supplementary Issues*, 55(2), 1-29. DOI: 10.1127/0372-8854/2011/0055S2-0043.

Hollaus, M., Wagner, W., Molnar, G., Mandlburger, G., Nothegger, C., and Otepka, J. (2010). Delineation of vegetation and building polygons from full-waveform airborne LIDAR data using OPALS software. In: *Proceedings of the geospatial data and geovisualization: Environment, security, and society, special joint symposium of ISPRS technical commission IV and AutoCarto*.

Holt, C. M., Stewart, A., Clint, M., and Perrott, R. H. (1987). An improved parallel thinning algorithm. *Communications of the ACM*, 30(2), 156-160. DOI: 10.1145/12527.12531.

Horwath, J. P., Zakharov, D. N., Mégret, R., and Stach, E. A. (2020). Understanding important features of deep learning models for segmentation of high-resolution transmission electron microscopy images. *npj Computational Materials*, 6(1), 1-9. DOI: 10.1038/s41524-020-00363-x.

Hough, P. V. (1959). Machine analysis of bubble chamber pictures. In: *Proceedings of the International Conference on High Energy Accelerators and Instrumentation, Sept. 1959*, 554-556.

Hu, X., Li, Y., Shan, J., Zhang, J., and Zhang, Y. (2014). Road centerline extraction in complex urban scenes from LiDAR data based on multiple features. *IEEE Transactions on Geoscience and Remote Sensing*, *52*(11), 7448-7456. DOI: 10.1109/TGRS.2014.2312793.

Hui, Z., Hu, Y., Jin, S., and Yevenyo, Y. Z. (2016). Road centerline extraction from airborne LiDAR point cloud based on hierarchical fusion and optimization. *ISPRS Journal of Photogrammetry and Remote Sensing*, 118, 22-36.

Huang, H., Wu, S., Cohen-Or, D., Gong, M., Zhang, H., Li, G., and Chen, B. (2013). L1-medial skeleton of point cloud. *ACM Transaction on Graph*ics, 32(4), 65-1. DOI: 10.1145/2461912.2461913.

Huang, R., Yang, B., Liang, F., Dai, W., Li, J., Tian, M., and Xu, W. (2018). A top-down strategy for buildings extraction from complex urban scenes using airborne LiDAR point clouds. *Infrared Physics Technology*, *92*, 203-218, DOI: 10.1016/j.infrared.2018.05.021.

Idris, R., Abd Latif, Z., Jaafar, J., Rani, N. M., and Yunus, F. (2012). Quantitative assessment of LiDAR dataset for topographic maps revision. In: *2012 International Conference on System Engineering and Technology* (ICSET), 1-4.

Illingworth, J., and Kittler, J. (1988). A survey of the Hough transform. *Computer Vision Graphic Image Processing*, *44*, 87–116, DOI: 10.1016/S0734-189X(88)80033-1.

Indrajit, A., Van Loenen, B., and Van Oosterom, P. (2019). Assessing spatial information themes in the spatial information infrastructure for participatory urban planning monitoring: Indonesian cities. *ISPRS International Journal of Geo-Information*, 8(7), 305. DOI: 10.3390/ijgi8070305.

International Cartographic Association. (1996). ICA News 26. Abderdeen, UK.

Ish-Horowicz, J., Udwin, D., Flaxman, S., Filippi, S., and Crawford, L. (2019). Interpreting deep neural networks through variable importance. arXiv preprint arXiv:1901.09839.

ISPRS. Vaihingen Results: Urban Object Detection in Area 2. Available online: http://www2.isprs.org/commissions/comm3/wg4/results/a2_detect.html (accessed on 11 December 2018).

Jacobs, L., Weiss J., and Dolan, D. (2013). Object tracking in noisy radar data: comparison of Hough transform and RANSAC. *IEEE International Conference on Electro-Information Technology* (EIT), 1-6. DOI: 10.1109/EIT.2013.6632715.

Jarvis, R.A. (1973). On the identification of the convex hull of a finite set of points in the plane. *Information Processing Letters*, 2(1), 18-21. DOI: 10.1016/0020-0190(73)90020-3.

Jarzabek-Rychard, M. and Maas, H-G. (2017). Geometric refinement of LAS-data derived buildings models using monoscopic aerial images. *Remote Sensing*, 9(3), 282. DOI: 10.3390/rs9030282.

Jiang, X. Y., Bunke, H. and Meier, U., 1996. Fast range image segmentation using high-level segmentation primitives. In: *Proceedings of the 3rd IEEE Workshop on Applications of Computer Vision* (WACV '96), 83.

Johnson, J. M., and Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1), 27. DOI: 10.1186/s40537-019-0192-5.

Kabolizade, M., Ebadi, H., and Ahmadi, S. (2010). An improved snake model for automatic extraction of buildings from urban aerial images and LiDAR data. *Computers, Environment and Urban Systems*, *34*(5), 435-441. DOI: 10.1016/j.compenvurbsys.2010.04.006.

Kadaster and Geonovum. Publieke Dienstverlening Op de Kaart (PDOK). Available online: https://www.pdok.nl/ (accessed on 24 August 2018).

Kanevski M, Foresti L, Kaiser C, Pozdnoukhov A, Timonin V, and Tuia D. (2009) Machine learning models for geospatial data. Handbook of theoretical and quantitative geography. University of Lausanne, Lausanne, 175-227.

Kang, Z. and Yang, J. (2018). A probabilistic graphical model for the classification of mobile LiDAR point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 143, 108-123. DOI: 10.1016/j.isprsjprs.2018.04.018.

Kelm, A. P., Rao, V. S., and Zolzer, U. Object Contour and Edge Detection with RefineContourNet. (2019). *arXiv* **2019**, arXiv:1904.13353.

Kröger, M., Sauer-Greff, W., Urbansky, R., Lorang, M., and Siegrist, M. (2016, September). Performance evaluation on contour extraction using Hough transform and RANSAC for multi-sensor data fusion applications in industrial food inspection. In: *2016 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, 234-237. DOI: 10.1109/SPA.2016.7763619.

Kustra, J., Jalba, A., & Telea, A. (2014). Robust segmentation of multiple intersecting manifolds from unoriented noisy point clouds. In: *Computer Graphics Forum*, 33(1), 73-87. DOI: 10.1111/cgf.12255.

Kustra, J., Jalba, A., and Telea, A. (2016). Computing refined skeletal features from medial point clouds. Pattern Recognition Letters, 76, 13-21. DOI: 10.1016/j.patrec.2015.05.007.

LASTools. Efficient LiDAR Processing Software (Version 141017, Academic). Available online: http://rapidlasso.com/LAStools (accessed on 6 October 2018).

Lach, S. R, and Kerekes, J. P. (2008). Robust extraction of exterior building boundaries from topographic LiDAR data. *IEEE International Geoscience and Remote Sensing Symposium, 2*, 85–88. DOI: 10.1109/IGARSS.2008.4778933.

Landrieu, L., Raguet, H., Vallet, B., Mallet, C., and Weinmann, M. (2017). A structured regularization framework for spatially smoothing semantic labelings of 3D point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, *132*, 102-118. DOI: 10.1016/j.isprsjprs.2017.08.010.

Landrieu, L., and Simonovsky, M. (2018). Large-scale point cloud semantic segmentation with SuperPoint graphs. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4558-4567.

Leavers, V. F. (1992). The dynamic generalized Hough transform: Its relationship to the probabilistic Hough transform and an application to the concurrent detection of circles and ellipses. *CVGIP Image Understanding*, *56*, 381–398. DOI: 10.1016/1049-9660(92)90049-9.

Lee, D. T. (1982). Medial axis transformation of a planar shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (4), 363-369. DOI: 10.1109/TPAMI.1982.4767267.

Lee, J. W., and Kweon, I. S. (1997). Extraction of line features in a noisy image. *Pattern Recognition*, *30*, 1651–1660, DOI: 10.1016/S0031-3203(96)00185-9.

Leymarie, F. F., and Kimia, B. B. (2007). The medial scaffold of 3D unorganized point clouds. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(2), 313-330. DOI: 10.1109/TPAMI.2007.44.

Li, L., Yang, F., Zhu, H., Li, D., Li, Y., and Tang, L. (2017). An improved RANSAC for 3D point cloud plane segmentation based on normal distribution transformation cells. *Remote Sensing*, *9*(5), 433.

Li, Y., Bu, R., Sun, M., Wu, W., Di, X., & Chen, B. (2018). PointCNN: Convolution on x-transformed points. In *Advances in neural information processing systems*, 820-830.

Li, Y., Wu, H., An, R., Xu, H., He, Q., and Xu, J. (2013). An improved building boundary extraction algorithm based on fusion of optical imagery and LIDAR data. *Optik*, *124*, 5357–5362.

Lin, T. Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In: *Proceedings of the IEEE international conference on computer vision*, 2980-2988.

Liu, Y., Cheng, M., Hu, X., Bian, J., Zhang, L., Bai, X., Tang, J. (2017). Richer convolutional features for edge detection. *IEEE Transaction Pattern Analysis and Machine Intelligence*, *41*, 1939–1946, DOI: 10.1109/TPAMI.2018.2878849.

Liu, Y., Fan, B., Xiang, S., and Pan, C. (2019). Relation-shape convolutional neural network for point cloud analysis. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8895-8904.

Lu, T., Ming, D., Lin, X., Hong, Z., Bai, X., and Fang, J. (2018). Detecting building edges from high spatial resolution remote sensing imagery using richer convolution features network. *Remote Sensing*, *10*, 1496.

Lucas, C., and van Tilburg, T. (2019). Retrieved Jan 16, 2020, from https://github.com/Geodan/building-boundary.

Ma, J., Bae, S. W., and Choi, S. (2012). 3D medial axis point approximation using nearest neighbors and the normal field. *The Visual Computer*, 28(1), 7-19. DOI: 10.1007/s00371-011-0594-7.

Manno-Kovacs, A., and Sziranyi, T. (2015). Orientation-selective building detection in aerial images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 108, 94-112. DOI: 10.1016/j.isprsjprs.2015.06.007.

Makhoul, J., Kubala, F., Schwartz, R., & Weischedel, R. (1999, February). Performance measures for information extraction. In: *Proceedings of DARPA broadcast news workshop*, 249-252.

Máttyus, G., Luo, W., Urtasun, R. (2017). DeepRoadMapper: extracting road topology from aerial images. In: *The IEEE International Conference on Computer Vision* (ICCV), 3438-3446.

McAllister, M., and Snoeyink, J. (2000). Medial axis generalization of river networks. *Cartography and Geographic Information Science*, 27(2), 129-138. DOI: 10.1559/152304000783547966.

Mennecke, B. E. (1997). Understanding the role of geographic information technologies in business: Applications and research directions. *Journal of Geographic Information and Decision Analysis*, 1(1), 44-68.

Microsoft GitHub Repository. Available online: https://github.com/Microsoft/USBuildingFootprints (accessed on 19 May 2019).

Miller, J. (2016). Building Extraction from LiDAR Using Edge Detection. *PhD Thesis*, Department of Geoinformatics and Geospatial Intelligence, Shenandoah University, Winchester, VA, USA.

Moreira, A., and Santos, M. Y. (2007). Concave hull: A k-nearest neighbours approach for the computation of the region occupied by a set of points. In: *Proceedings of the International Conference on Computer Graphics Theory and Applications* (GRAPP), 61-68.

Morgan, M., and Habib, A. (2002). Interpolation of LiDAR data and automatic building extraction. In: *Proceedings of the ACSM-ASPRS Annual Conference*, 432-441

Mukhopadhyay, P., and Chaudhuri, B. B. (2015). A survey of Hough Transform. *Pattern Recognition*, *48*, 993–1010. DOI: 10.1016/j.patcog.2014.08.027.

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., and Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, *2*(1), 1. DOI: 10.1186/s40537-014-0007-7.

Nardinocchi, C., Forlani, G., and Zingaretti, P. (2003). Classification and filtering of laser data. *The International Archives of Photogrammetry and Remote Sensing*, *34*(3/W13).

National Research Council. (1983). Procedures and standards for a multipurpose Cadastre. *Washington DC: The National Academic Press*. DOI: 10.17226/11803.

Nguyen, A., and Le, B. (2013). 3D point cloud segmentation: A survey. In: *2013 6th IEEE conference on robotics, automation and mechatronics (RAM)*, 225-230.

Niblack, C. W., Gibbons, P. B., and Capson, D. W. (1992). Generating skeletons and centerlines from the distance transform. *CVGIP: Graphical Models and Image Processing*, 54(5), 420-437. DOI: 10.1016/1049-9652(92)90026-T.

Niemeyer, J., Rottensteiner, F., and Soergel, U. (2012). Conditional random fields for lidar point cloud classification in complex urban areas. *The ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *3*, 263-268.

Nyerges, T. L., & Jankowski, P. (2009). Regional and urban GIS: a decision support approach. *Guilford Press*.

Oehler, B., Stueckler, J., Welle, J., Schulz, D., and Behnke, S. (2011). Efficient multi-resolution plane segmentation of 3D point clouds. In: *International Conference on Intelligent Robotics and Applications*, 145-156. Springer, Berlin, Heidelberg.

Oesau, S. (2015). Geometric Modeling of Indoor Scenes from Acquired Point Data. *PhD Thesis*, Université Nice Sophia Antipolis, Nice, France

Parzen, E. (1962). On estimation of a probability density function and mode. *Annals of Mathematical Statistics,* 33(3), 1065-1076. DOI: 10.1214/aoms/1177704472.

Patterson, J., and Gibson, A. (2017). Deep learning: A practitioner's approach. *O'Reilly Media, Inc.*

Pavlidis, T. (1978). A review of algorithms for shape analysis. *Computer graphics and image processing*, 7(2), 243-258. DOI: 10.1016/0146-664X(78)90115-6.

Pavlidis, T. (1980). A thinning algorithm for discrete binary images. *Computer graphics and image processing*, 13(2), 142-157. DOI: 10.1016/S0146-664X(80)80037-2.

Peters, R. (2018). Geographical point clouds modelling with the 3D medial axis transform. *PhD dissertation*, Dept. of Urbanism, Delft University of Technology, the Netherlands. ISBN: 978-94-6186-899-2.

Petrie, G., and Toth, C.K. (2009). Introduction to laser ranging, profiling, and scanning, Chapter 1. In: Shan, J., and Toth C.K. (eds) Topographic Laser Ranging and Scanning. CRC Press, Taylor and Francis, 19-45.

Pohl, M., and Feldmann, D. (2016). Generating Straight Outlines of 2D Point Sets and Holes using Dominant Directions or Orthogonal Projections. In: *VISIGRAPP* (1: GRAPP), 59-71.

Poliyapram, V., Wang, W., and Nakamura, R. 2019. A point-wise LiDAR and image multimodal fusion network (PMNet) for aerial point cloud 3D semantic segmentation. *Remote Sensing*, 11(24), 2961. DOI: 10.3390/rs11242961

Poullis C., A (2013). Framework for automatic modeling from point cloud data. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 35(11), 2563-2575. DOI: 10.1109/TPAMI.2013.64.

Poux, F., Neuville, R., Hallot, P., and Billen, R. (2016). Smart point cloud: Definition and remaining challenges. *The ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *4*(W1), 119-127.

Princen, J., Illingworth, J., and Kittler, J. (1992). A formal definition of the Hough transform: Properties and relationships. *Journal of Mathematical Imaging and Vision*, 1(2), 153-168. DOI: 10.1007/BF00122210.

Pu, S., and Vosselman, G. (2009). Knowledge based reconstruction of building models from terrestrial laser scanning data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 64(6), 575-584. DOI: 10.1016/j.isprsjprs.2009.04.001.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 652-660. DOI: 10.1109/CVPR.2017.16

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017). PointNet++: Deep hierarchical feature learning on point sets in a metric space. *In: Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*, 5105–5114.

Qi, C. R., Liu, W., Wu, C., Su, H., and Guibas, L. J. (2018). Frustum pointnets for 3D object detection from RGB-D data. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 918-927.

Rabbani, T., and Van Den Heuvel, F. (2005). Efficient Hough transform for automatic detection of cylinders in point clouds. *The International archives of photogrammetry, remote sensing and spatial information sciences,* WG 5/1, 60-65.

Rabbani, T., Van Den Heuvel, F., and Vosselmann, G. (2006). Segmentation of point clouds using smoothness constraint. *The International archives of photogrammetry, remote sensing and spatial information sciences*, 36(5), 248-253.

Raschka, S. (2014). An overview of general performance metrics of binary classifier systems. arXiv preprint arXiv:1410.5330.

Reddy, J. M., and Turkiyyah, G. M. (1995). Computation of 3D skeletons using a generalized Delaunay triangulation technique. *Computer-Aided Design*, 27(9), 677-694. DOI: 10.1016/0010-4485(94)00025-9.

Regnauld, N., and Mackaness, W. A. (2006). Creating a hydrographic network from its cartographic representation: a case study using Ordnance Survey MasterMap data. *International Journal of Geographical Information Science*, 20(6), 611-631. DOI: 0.1080/13658810600607402.

Regulation of Information Geospatial no.11/2018 about Technical Analysis of Implementation Geospatial Information.

Regulation of Head of Geospatial Information Agency no. 15/2014 about Technical Guidelines for Base Map Accuracy.

Reniers, D., Van Wijk, J., and Telea, A. (2008). Computing multiscale curve and surface skeletons of genus 0 shapes using a global importance measure. *IEEE Transactions on Visualization and Computer Graphics*, 14(2), 355-368. DOI: 10.1109/TVCG.2008.23.

Richter, R. (2018). Concepts and techniques for processing and rendering of massive 3D point clouds. *PhD thesis*, University of Potsdam, Faculty of Digital Engineering, Hasso Plattner Institute.

Robinson, Arthur H., Joel L. Morrison, Phillip C. Muehrcke, A. Jon Kimerling and Stephen C. Guptill. (1995). *Elements of Cartography*, Fifth Edition. New York, NY: John Wiley and Sons, Inc. ISBN 0-471-55579-7.

Rottensteiner, F., and Briese, C. (2002). A new method for building extraction in urban areas from high-resolution LIDAR data. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34, 295-301.

Rottensteiner, F., and Clode, S. (2009). Building and road extraction by LiDAR and imagery, Chapter 9. In: Shan, J., and Toth C.K., (eds) Topographic Laser Ranging and Scanning. CRC Press, Taylor and Francis, 463-496.

Rutzinger, M., Rottensteiner, F., and Pfeifer, N. (2009). A comparison of evaluation techniques for building extraction from airborne laser scanning. *IEEE Journal on Selected Topic Applied Earth Observation and Remote Sensing*, 2, 11-20, DOI: 10.1109/JSTARS.2009.2012488.

Saha, P. K., Borgefors, G., and di Baja, G. S. (2016). A survey on skeletonization algorithms and their applications. *Pattern Recognition letters*, 76, 3-12. DOI: 10.1016/j.patrec.2015.04.006.

Saha, P. K., Borgefors, G., and di Baja, G. S. (2017). Skeletonization and its applications – a review. In: *Skeletonization*, 3-42. Academic Press. DOI: 10.1016/b978-0-08-101291-8.00002-x.

Salomons, E. M., and Pont, M. B. (2012). Urban traffic noise and the relation to urban density, form, and traffic elasticity. *Landscape and Urban Planning*, 108, 2-16. DOI: 10.1016/j.landurbplan.2012.06.017.

Sampath, A., and Shan, J. (2009). Segmentation and reconstruction of polyhedral building roofs from aerial LiDAR point clouds. *IEEE Transactions on geoscience and remote sensing*, *48*(3), 1554-1567. DOI: 10.1109/TGRS.2009.2030180.

Sampath, A., and J. Shan. (2007). Building boundary tracing and regularization from airborne LIDAR point clouds. *Photogrammetric Engineering and Remote Sensing*, 73(7), 805-812. DOI: 10.14358/PERS.73.7.805.

Savitzky, A., and Golay, M. J. (1964). Smoothing and differentiation of data by simplified least squares procedures. *Analytic Chemistry*, 36, 1627-1639, DOI: 10.1021/ac60214a047.

Schenk, T., and Csathó, B. (2002). Fusion of LIDAR data and aerial imagery for a more complete surface description. *The International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, *34*(3/A), 310-317.

Schmohl, S., and Sörgel, U. 2019. Submanifold sparse convolutional networks for semantic segmentation of large-scale ALS point clouds. *The ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, ISPRS Geospatial Week, Enschede, the Netherlands.

Shan, J., and Toth, C. K. (2009). Topographic laser ranging and scanning: principles and processing. *CRC Press*.

She, F. H., Chen, R. H., Gao, W. M., Hodgson, P. H., Kong, L. X., and Hong, H. Y. (2009). Improved 3D thinning algorithms for skeleton extraction. In: *2009 Digital Image Computing: Techniques and Applications*, 14-18. DOI: 10.1109/DICTA.2009.13.

Shen, W. (2008). Building boundary extraction based on LiDAR point clouds data. *The International Archives Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37(B3b), 157-161.

Shen, J., Liu, J., Lin, X., and Zhao, R. (2012). Object-based classification of airborne light detection and ranging point clouds in human settlements. *Sensor Letters*, *10*(1-2), 221-229. DOI: 10.1166/sl.2012.1826.

Shen, W., Wang, X., Wang, Y., Bai, X., Zhang, Z. (2015). Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3982–3991.

Shih, F. Y., and Cheng, S. (2005). Automatic seeded region growing for color image segmentation. *Image and Vision Computing*, 23(10), 877-886. DOI: 10.1016/j.imavis.2005.05.015.

Siddiqui, F. U., Teng, S. W., Awrangjeb, M., and Lu, G. (2016). A robust gradient based method for building extraction from LiDAR and photogrammetric imagery. *Sensors*, *16*, 1110, DOI: 10.3390/s16071110.

Singla, S., Wallace, E., Feng, S., & Feizi, S. 2019. Understanding impacts of high-order loss approximations and features in deep learning interpretation. arXiv preprint arXiv:1902.00407.

Sithole, G. (2005). Segmentation and classification of airborne laser scanner data. *Publications on Geodesy, 59*.

SNI 8205:2015. Indonesian National Standard for Base Map Accuracy (Ketelitian Peta Dasar). Available online: http://sispk.bsn.go.id/SNI (accessed on 25 August 2018).

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

Stock, K., and Guesgen, H. (2016). Geospatial reasoning with open data. In: *Automating Open Source Intelligence,* 171-204. Syngress. DOI: 10.1016/B978-0-12-802916-9.00010-5.

Soilán Rodríguez, M., Lindenbergh, R., Riveiro Rodríguez, B., and Sánchez Rodríguez, A. 2019. PointNet for the automatic classification of aerial point clouds. *The ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, ISPRS Geospatial Week, Enschede, the Netherlands.

Spiller, M., and Agudelo, C. 2011. Mapping diversity of urban metabolic functions - a planning approach for more resilient cities. In: *Proceedings of the 5th AESOP Young Academics Network Meeting*, the Netherlands.

Su, Y.T., Bethel, J., and Hu, S. (2016). Octree-based segmentation for terrestrial LiDAR point cloud data in industrial applications. *ISPRS Journal of Photogrammetry and Remote Sensing*, 113, 59-74. DOI: 10.1016/j.isprsjprs.2016.01.001.

Susetyo, D. B., Hidayat, F., and Hariyono, M.I. (2018). Automatic building model extraction using LiDAR data. In: *Proceedings of the Asian Conference and Remote Sensing*, Kuala Lumpur, Malaysia, 15–19 October 2018.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. (2016). Inception-v4, inception-resnet and the impact of residual connections on learning. arXiv preprint arXiv:1602.07261.

Tagliasacchi, A., Delame, T., Spagnuolo, M., Amenta, N., and Telea, A. (2016, May). 3D skeletons: A state-of-the-art report. In: *Computer Graphics Forum*, 35(2), 573-597. DOI: 10.1111/cgf.12865.

Tarsha-Kurdi, F., Landes, T., and Grussenmeyer, P. (2007). Hough-transform and extended ransac algorithms for automatic detection of 3d building roof planes from LiDAR data. In: *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007*, 407-412.

Taylor, J. R., and Lovell, S. T. (2012). Mapping public and private spaces of urban agriculture in Chicago through the analysis of high-resolution aerial images in Google Earth. *Landscape and urban planning*, 108(1), 57-70. DOI: 10.1016/j.landurbplan.2012.08.001.

TerraScan. Available online: http://www.terrasolid.com/products/terrascanpage.php (accessed on 3 July 2019).

Tharwat, A. (2020). Classification assessment methods. *Applied Computing and Informatics*. ISSN: 2634-1964.

Thomas, H., Qi, C. R., Deschaud, J. E., Marcotegui, B., Goulette, F., and Guibas, L. J. (2019). Kpconv: Flexible and deformable convolution for point clouds. In: *Proceedings of the IEEE International Conference on Computer Vision*, 6411-6420.

Tran, T. N., Drab, K., and Daszykowski, M. (2013). Revised DBSCAN algorithm to cluster data with dense adjacent clusters. *Chemometrics and Intelligent Laboratory Systems*, *120*, 92-96.

Tsogkas, S., and Dickinson, S. (2017). Amat: Medial axis transform for natural images. In: *Proceedings of the IEEE International Conference on Computer Vision*, 2708-2717. DOI: 10.1109/ICCV.2017.295

Tobler, W. R. (1959). Automation and cartography. *Geographical Review*, 49(4), 526-534. DOI: 10.2307/212211.

Tonkin, T. (1994). Business geographics impacts corporate America. *Business Geographics*, 2(2), 27-28.

United Nations. (2015). Integrating Geospatial Information into the 2030 Agenda for Sustainable Development. 20th UN Regional Cartographic Conference for Asia and the Pacific, 6-9 October 2015, Jeju, South Korea. Retrieved from: https://unstats.un.org/unsd/geoinfo/RCC/docs/rccap20/IP1_UNRCC-AP%20Paper%20G%20Scott.pdf, retrieved on August, 2020.

United Nations Global Geospatial Information Management (UNGGIM). (2018). Integrated Geospatial Information framework: A strategic guide to develop and strengthen national geospatial information management report. Retrieved from: https://reliefweb.int/sites/reliefweb.int/files/resources/Part%201-IGIF-Overarching-Strategic-Framework-24July2018-1.pdf, retrieved on June 2020.

Varytimidis, C., Rapantzikos, K., Avrithis, Y., and Kollias, S. (2016). α-shapes for local feature detection. *Pattern Recognition*, 50, 56-73. DOI: 10.1016/j.patcog.2015.08.01.

Visvalingam, M. (1989). Cartography, GIS and maps in perspective. *The Cartographic Journal*, *26*(1), 26-32. DOI: 10.1179/caj.1989.26.1.26.

Visvalingam, M. (1990). Trends and concerns in digital cartography. *Computer-Aided Design*, *22*(3), 115-130. DOI: 10.1016/0010-4485(90)90070-S.

Visvalingam, M., and Whyatt, J. D. (1993). Line generalisation by repeated elimination of points. *The Cartographic Journal*, 30(1), 46-51. DOI: 10.1179/000870493786962263.

Vosikis, G., and Jansa, J. (2008). Advantages and disadvantages of the Hough transformation in the frame of automated building extraction. In: *Proceedings of the XXI ISPRS Congress*, 37, 719-724.

Vosselman, G. (1999). Building reconstruction using planar faces in very high density height data. *The International Archives of Photogrammetry and Remote Sensing*, *32*(3), 87-94.

Vosselman, G. (2013). Point cloud segmentation for urban scene classification. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *1*, 257-262.

Vosselman, G., Coenen, M., and Rottensteiner, F. (2017). Contextual segment-based classification of airborne laser scanner data. *ISPRS Journal of Photogrammetry and Remote Sensing*, *128*, 354-371. DOI: 10.1016/j.isprsjprs.2017.03.010.

Vosselman, G., Gorte, B. G., Sithole, G., and Rabbani, T. (2004). Recognising structure in laser scanner point clouds. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 46(8), 33-38.

Vosselman, G., Kessels, P., and Gorte, B. (2005). The utilisation of airborne laser scanning for mapping. *International Journal of Applied Earth Observation and Geoinformation*, *6*(3-4), 177-186. DOI: 10.1016/j.jag.2004.10.005.

Vosselman, G. and Maas, H. G. (2010). Airborne and terrestrial laser scanning. CRC Press.

Wacker, A. C., and Landgrebe, D. A. (1970). Boundaries in multispectral imagery by clustering. In: *1970 IEEE Symposium on Adaptive Processes (9th) Decision and Control*, 114-114. DOI: 10.1109/SAP.1970.269984.

Wang, R., Peethambaran, J., and Chen, D. (2018). LiDAR point clouds to 3-D Urban Models: a review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(2), 606-627. DOI: 10.1109/JSTARS.2017.2781132.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., and Solomon, J. M. (2019). Dynamic graph CNN for learning on point clouds. *ACM Transactions on Graphics*, 38(5), 1-12. DOI: 10.1145/3326362

Wang, Y., Zhao, X., Li. Y, and Huang, K. (2019). Deep crisp boundaries: From boundaries to higher-level tasks. *IEEE Transaction Image Processing*, 28, 1285–1298, DOI: 10.1109/TIP.2018.2874279.

Wei, S. (2008). Building boundary extraction based on LiDAR point clouds data. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *37*, 157-161.

Weidner, U., and Förstner, W. (1995). Towards automatic building extraction from high-resolution digital elevation models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 50(4), 38-49. DOI: 10.1016/0924-2716(95)98236-S.

Weng, Q. (2012). Remote sensing of impervious surfaces in the urban areas: Requirements, methods, and trends. *Remote Sensing of Environment*, *117*, 34-49. DOI: 10.1016/j.rse.2011.02.030.

Wicaksono, S. B., Wibisono, A., Jatmiko, W., Gamal, A., and Wisesa, H. A. (2019). Semantic Segmentation on LiDAR Point Cloud in Urban Area using Deep Learning. *In: IEEE 2019 International Workshop on Big Data and Information Security (IWBIS)*, 63-66. DOI: 10.1109/IWBIS.2019.8935882.

World Bank. (2019). Retrieved from the website: https://www.worldbank.org/en/topic/land/brief/geospatial-technology-and-information-for-development on August, 2020.

Xiaoxu, L., Chang, D., Tian,T., and Cao, J. (2019). Large-margin regularized softmax cross-entropy loss. *IEEE Access*, 19572-19578. DOI: 10.1109/ACCESS.2019.2897692.

Xie, Y., Tian, J., and Zhu, X. (2020). Linking Points with labels in 3D: A review of point cloud semantic segmentation. *IEEE Geoscience and Remote Sensing Magazine*. DOI: 10.1109/MGRS.2019.2937630.

Xie, L., Zhu, Q., Hu, H., Wu, B., Li, Y., Zhang, Y., and Zhong, R. (2018). Hierarchical regularization of building boundaries in noisy aerial laser scanning and photogrammetric point clouds. *Remote Sensing*, 10, 1996. DOI: 10.3390/rs10121996.

Xiu, H., Poliyapram, V., Kim, K. S., Nakamura, R., and Yan, W. (2018). 3D semantic segmentation for high-resolution aerial survey derived point clouds using deep learning. *In: Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 588-591. DOI: 10.1145/3274895.3274950.

Xu, P., and Poullis, C. (2019). Delineation of Road Networks Using Deep Residual Neural Networks and Iterative Hough Transform. In: *International Symposium on Visual Computing*, 32-44. Springer, Cham. DOI: 10.1007/978-3-030-33720-9_3.

Yan, Y. (2018). Medial axis approximation and regularization. *PhD dissertation*, Engineering and Applied Science Theses & Dissertations, 336, Washington University.

Yirci, M., Brédif, M., Perret, J., and Paparoditis, N. (2013). 2D arrangement-based hierarchical spatial partitioning: An application to pedestrian network generation. In: *Proceedings of the Sixth ACM SIGSPATIAL International Workshop on Computational Transportation Science*, 31-36. DOI: 10.1145/2533828.2533843.

Yang, J., Price, B., Cohen, S., Lee, H., and Yang, M. (2016). Object contour detection with a fully convolutional encoder-decoder network. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 193–202.

Yu, L., Li, X., Fu, C.W., Cohen-Or, D., and Heng, P.A. (2018). EC-Net: An edge-aware point set consolidation network. In: *Proceedings of the European Conference on Computer Vision* (ECCV), 386–402.

Yu, Y., Liu, X., and Buckles, B. P. (2010). A cue line based method for building modeling from LiDAR and satellite imagery. In: *2010 Second International conference on Computing, Communication and Networking Technologies,* IEEE, 1-8.

Zai, D., Li, J., Guo, Y., Cheng, M., Lin, Y., Luo, H., and Wang, C. (2017). 3D road boundary extraction from mobile laser scanning data via supervoxels and graph cuts. *IEEE Transactions on Intelligent Transportation Systems*, 19(3), 802-813. DOI: 10.1109/TITS.2017.2701403.

Zeng, C., Wang, J., and Lehrbass, B. (2013). An evaluation system for building footprint extraction from remotely sensed data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6(3), 1640-1652. DOI: 10.1109/JSTARS.2013.2256882.

Zhao, Z., Duan, Y., Zhang, Y., and Cao, R. (2016). Extracting buildings from and regularizing boundaries in airborne LiDAR data using connected operators. *International Journal Remote Sensing*, *37*, 889–912, DOI: 10.1080/014311611.2015.1137647.

Zhu, X. X., Tuia, D., Mou, L., Xia, G.-S., Zhang, L., Xu, F., and Fraundorfe, F. (2017). Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine*, 5(4), 8-36. DOI: 10.1109/MGRS.2017.2762307.

Zhang, J. Duan, M. Yan, Q. and Lin, X. (2014). Automatic vehicle extraction from airborne LiDAR data using an object-based point cloud analysis method. *Remote Sensing*, 6(9), 8405-8423. DOI: 10.3390/rs6098405.

Zhang, J, Lin, X., and Ning, X. (2013). SVM-based classification of segmented airborne LiDAR point clouds in urban areas. *Remote Sensing*, 5(8), 3749-3775. DOI: 10.3390/rs5083749.

Zhang, J., Zhao, X., Chen, Z., and Lu, Z. (2019). A Review of Deep Learning-Based Semantic Segmentation for Point Cloud. *IEEE Access*, *7*, 179118-179133. DOI: 10.1109/ACCESS.2019.2958671.

Zhang, T. Y., & Suen, C. Y. (1984). A fast parallel algorithm for thinning digital patterns. Communications of the ACM, 27(3), 236-239. DOI: 10.1145/357994.358023.

Zhang, Q., Yang, L. T., Chen, Z., and Li, P. (2018). A survey on deep learning for big data. *Information Fusion*, *42*, 146-157. DOI: 10.1016/j.inffus.2017.10.006.

Zhang, W., Su, S., Wang, B., Hong, Q., and Sun, L. (2020). Local k-NNs pattern in Omni-Direction graph convolution neural network for 3D point clouds. *Neurocomputing*, *413*, 487-498. DOI: 10.1016/j.neucom.2020.06.095.

Zhang, Y., and Webber, R. (1996). A windowing approach to detecting line segments using Hough transform. *Pattern Recognition*, *29*, 255–265, DOI: 10.1016/0031-3203(95)00083-6.

Zhao, Z., Duan, Y., Zhang, Y., & Cao, R. (2016). Extracting buildings from and regularizing boundaries in airborne LiDAR data using connected operators. *International Journal of Remote Sensing*, 37(4), 889-912. DOI: 10.1080/014311611.2015.1137647.

Zhou, L., Zhang, C., Liu, F., Qiu, Z., & He, Y. (2019). Application of deep learning in food: A review. *Comprehensive Reviews in Food Science and Food Safety*, *18*(6), 1793-1811. DOI: 10.1109/72.991427

Zhu, Q., Li, Y., Hu, H., and Wu, B. (2017). Robust point cloud classification based on multi-level semantic relationships for urban scenes. *ISPRS Journal of Photogrammetry and Remote Sensing*, 129, 86-102. DOI: 10.1016/j.isprsjprs.2017.04.022.

About the Author

Elyta Widyaningrum took a major in Geodesy engineering at Gadjah Mada University, Yogyakarta, Indonesia from 1999 to 2003. She has over 16 years experience in the remote sensing and geospatial data industry including one year at the Indonesian National Institute of Aeronautics and Space (LAPAN) from 2003 to 2004. She works at the Indonesian Geospatial Information Agency (BIG) since 2005. Her main task in BIG is to perform topographical base map production using photogrammetry and remote sensing data.

From 2007 to 2009, Elyta studied at Technical University of Munich. There, she obtained her Master of Science degree in Land Management and Land Tenure. Her MSc thesis entitled "*Tsunami evacuation planning using geoinformation technology considering land management aspects, case study: Cilacap, Central of Java*" was supported by the German Aerospace Center (DLR) and contributed to the German-Indonesian Tsunami Early Warning System project.

From 2016 to 2021, Elyta conducted her PhD research on "*Automation object extraction from airborne laser scanning point clouds for digital base map production"*, supervised by Roderik Lindenbergh and Ramon Hanssen at the Department of Geoscience and Remote Sensing – Delft University of Technology and funded by the Indonesian Endowment Fund for Education (LPDP). During her PhD research, Elyta received several awards: (i) the best young author award at the ISPRS - TC II Symposium 2018, Riva del Garda, Italy; (ii) the best poster award at the ISPRS - TC IV Symposium 2018, Delft, the Netherlands; and (iii) the best paper (*Shunji Murai*) award at the 40th Asian Conference on Remote Sensing (ACRS) 2019, Daejeon, South Korea.