Western Graduate&PostdoctoralStudies

Western University
Scholarship@Western

Electronic Thesis and Dissertation Repository

8-22-2018 2:00 PM

Effects of Aerial LiDAR Data Density on the Accuracy of Building Reconstruction

Peter Crawford, *The University of Western Ontario*

Supervisor: Wang, Jinfei, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Geography

© Peter Crawford 2018

Follow this and additional works at: https://ir.lib.uwo.ca/etd

 Part of the Remote Sensing Commons

Recommended Citation

Crawford, Peter, "Effects of Aerial LiDAR Data Density on the Accuracy of Building Reconstruction" (2018). *Electronic Thesis and Dissertation Repository*. 5552.
https://ir.lib.uwo.ca/etd/5552

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlswadmin@uwo.ca.

Abstract

Previous work has identified a positive relationship between the density of aerial LiDAR input for building reconstruction and the accuracy of the resulting reconstructed models. We hypothesize a point of diminished returns at which higher data density no longer contributes meaningfully to higher accuracy in the end product. We investigate this relationship by subsampling a high-density dataset from the City of Surrey, BC to different densities and inputting each subsampled dataset to reconstruction using two different reconstruction methods. We then determine the accuracy of reconstruction based on manually created reference data, in terms of both 2D footprint accuracy and 3D model accuracy. We find that there is no quantitative evidence for meaningfully improved output accuracy from densities higher than 4 p/m^2 for either method, although aesthetic improvements at higher point cloud densities are noted for the 2.5D Dual Contouring method.

Keywords

LiDAR, laser scanning, building reconstruction, building extraction, 3d urban modeling, SPHARM, 3d model accuracy assessment, 2.5D dual contouring, point cloud

Acknowledgments

I would like to personally thank my supervisor Jinfei Wang and my colleagues at GITA Lab in the Department of Geography at the University of Western Ontario. Special thanks are due to Dr. Qian-Yi Zhou, currently of Intel, for making his urban modelling system available for use as part of my research, as well as his doctoral supervisor Professor Ulrich Neumann. My work, and the work of numerous others in my lab, would not be possible without the data graciously provided to the public by the municipal government of Surrey, BC. My research was supported by funding from Natural Sciences and Engineering Research Council of Canada (NSERC).

Table of Contents

Abst ..... Error! Bookmark not defined.

Abstract i

Acknowledgments ii

List of Tables v

List of Figures vi

List of Appendices x

List of Abbreviations xi

1 Introduction 1

1.1 Aerial LiDAR Technology 1

1.2 Building Extraction and Reconstruction 2

1.3 Research Question and Objectives 3

1.4 Organization of this Thesis 4

2 Literature Review 5

2.1 Conceptualization of Buildings 6

2.2 Building Footprint Extraction 7

2.3 3D Building Reconstruction 9

2.4 Previous Research on the Effects of Point Cloud Density on Extraction and Reconstruction 13

2.5 Assessment of Extraction and Reconstruction Accuracies 14

2.6 Characteristics of Polygonal 3D Models 20

3 Data and Study Area 22

3.1 Surrey BC 22

3.2 LiDAR And Supplementary Data 22

3.3 Sampling Regions 25

3.3.1 Bridgeview 27
3.3.2 North Whalley 27
3.3.3 Central Whalley 28
3.3.4 Cindrich 28
3.4 Point Cloud Density 29
4 Methods 31
4.1 Point Cloud Subsampling 31
4.2 Reference Data Creation 34
4.2.1 Reference Footprint Digitization 38
4.2.2 Reference Building Model Creation 39
4.3 2.5D Dual Contouring Pipeline 40
4.4 ENVI+TIN Modelling 43
4.5 Footprint Generation 45
4.6 Footprint Accuracy Analysis 45
4.7 3D Model Accuracy Analysis 49
5 Results 51
5.1 2D Accuracy 54
5.2 3D Accuracy 61
6 Discussion and Conclusions 68
Bibliography 79
Appendix A: Calibration of Dual Contouring Parameters 88
Appendix B: ArcGIS Python Script 91
Appendix C: Reference Building List 104
Curriculum Vitae 109

List of Tables

Table 3-1: Study Area Characteristics 25

Table 4-1: Point Cloud Density Per Subsample 34

Table 4-2: Dual Contouring Parameters 43

Table 4-3 Sample SPHARM Coefficients 50

Table 5-1: Overall 2D Accuracy 54

Table 5-2 Number of Buildings Extracted per Method and Subset 56

List of Figures

Figure 2-1 A diagrammatic explanation of classification accuracy. The blue circle represents real building area in the input, the red circle represents building area in the classification. True Negative area is represented by the area outside either circle. 15

Figure 2-2: An illustration of the SPHARM process. The polygonal model (a) is voxelized and mapped to the unit sphere using a heat diffusion algorithm illustrated by (b) and (c), with the final mapping shown by (d). The building model as approximated by SPHARM coefficients of up to degree 80 is shown by (e). Reproduced with permission from Zeng, Wang (2014). 18

Figure 3-1: Original LiDAR point cloud data, coloured by elevation, representing a hotel and nearby surroundings in the Bridgeview study area. 23

Figure 3-2: Locations of the study areas in relation to each other are shown with an orthoimage background. 24

Figure 3-3: Buildings footprints for each study area are shown coloured according to size class. Clockwise from top: Bridgeview, Cindrich, North Whalley, Central Whalley. 26

Figure 3-4: Point cloud density for the full-density subsets of all four study areas, higher point cloud densities indicated with lighter colours. Clockwise from top: Bridgeview, Cindrich, North Whalley, Central Whalley. Completely black areas indicate areas outside of the study area, data for which has been removed to accelerate processing. 30

Figure 4-1: Point clouds at different subsampling levels are shown with a transparent reference model as background. In reading order: n=1, n=2, n=5, n=10, n=20, n=30. 33

Figure 4-2: Reference data, with manually created models floating above reference footprints, is shown for the Cindrich study area. 35

Figure 4-3: Reference data, with manually created models floating above reference footprints, is shown for the North Whalley study area. 36

Figure 4-4: Reference data, with manually created models floating above reference footprints, is shown for the Bridgeview study area. 37

Figure 4-5: Reference data, with manually created models floating above reference footprints, is shown for the Central Whalley study area. 38

Figure 4-6 The manually-created reference footprint data and municipally-sourced building footprint data, overlaid on LiDAR-derived DSM elevation data. Note both the displaced outline of the large building and the absence of the smaller one in the municipal data. 40

Figure 4-7: A simplified diagram of the 2.5D dual contouring pipeline. 41

Figure 4-8: Part of a LiDAR point cloud classified into building, non-building and ground points using ENVI. 44

Figure 4-9: Key steps of the ENVI+TIN modeling process are shown. From left to right: the ground TIN, the roof TIN, and the building models generated by extruding between the two, all for the full resolution dataset of the Cindrich study area. 44

Figure 4-10: A diagrammatic representation of the ENVI+TIN building modelling process. Note that the detected building footprints are re-used for footprint analysis, since due to the nature of the TIN to TIN extrusion process they exactly represent the 2D extents of the models produced. 45

Figure 4-11: A diagrammatic representation of ArcScript, the python script used to automate the workflow for ENVI+TIN model creation, 2D accuracy assessment, and model position calculation. 48

Figure 5-1: Examples of models of the same building produced by the ENVI+TIN (left) and Dual Contouring (centre) methods from original density (n=1) LiDAR data are shown with the reference model (right) of the same building. 52

Figure 5-2: An area of the Central Walley area reconstructed from full density data using the Dual Contouring method. 52

Figure 5-3: The Cindrich neighbourhood, as reconstructed using the ENVI+TIN method from the full density dataset, is shown floating above building footprints detected by ENVI. 53

Figure 5-4: Mean completeness for buildings of different sizes is plotted, with buildings grouped based on point cloud density into 1 p/m^2 bins. 58

Figure 5-5: As in Figure 5-2, but for the ENVI+TIN method. 58

Figure 5-6 Mean footprint completeness is shown for all footprint size classes at point cloud densities from 3.9 to 0.5, rounded to the nearest 0.1 60

Figure 5-7 Mean completeness for footprints in 1 p/m^2 density bins is plotted, both overall and for only buildings larger than 50 m^2 in area. Bins with fewer than 10 extracted footprints are omitted. 60

Figure 5-8 Mean completeness of models produced by dual contouring is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown. 62

Figure 5-9 Mean completeness of models produced by ENVI+TIN method is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown. 62

Figure 5-10: Mean Q of models produced by dual contouring is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown. . 63

Figure 5-11: Mean Q of models produced by ENVI+TIN method is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown. . 63

Figure 5-12: Mean SPHARM RMSD of models produced by dual contouring is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown. 64

Figure 5-13: Mean SPHARM RMSD of models produced by ENVI+TIN method is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown. 65

Figure 5-14: The mean Q of models generated by either method from subsampled point clouds compared to those generated by the corresponding method from the full-resolution point clouds is plotted. 66

Figure 5-15: The mean SPHARM RMSD of models generated by either method from subsampled point clouds compared to those generated by the corresponding method from the full-resolution point clouds is plotted. 66

Figure 5-16: SPHARM RMSD plotted against Q for all comparisons, including both reference-to-reconstruction and reconstruction-to-reconstruction. Outlier SPHARM RMSD values (those >100, n=3) are omitted, as are comparisons in which one or both models lacked a valid SPHARM representation. 67

Figure 6-1: Overall 2D accuracy in terms of both completeness and correctness are plotted for both methods..... 68

Figure 6-2: A reference model of a large house. 70

Figure 6-3: The products of DC reconstruction of the same large house shown in Figure 6-2 are shown for four datasets, clockwise from top right: n=1, n=5, n=15, n=10. Note the loss of fine details associated with increasing initial grid length, as well as the agglomeration of a smaller, non-building structure onto the model in the n=1 model. 71

Figure 6-4: Products of reconstruction using the ENVI+TIN method are shown, on the same house and from the same datasets as in Figure 6-3. Note minor omission of building volume in the n=10 reconstruction and major omission in the n=15 volume..... 72

Figure 6-5: 3D completeness of dual contouring models is plotted against point cloud density for only models where 3D and 2D completeness are within 50%..... 74

Figure 6-6: 3D completeness of ENVI+TIN models is plotted against point cloud density for only models where 3D and 2D completeness are within 50%. 74

List of Appendices

Appendix A: Calibration of Dual Contouring Parameters 88

Appendix B: ArcGIS Python Script 91

Appendix C: Reference Building List 104

List of Abbreviations

A list of abbreviations follows, with explanations provided where the meaning of the term in question is not commonly known or apparent from the unabbreviated form.

ALS: Airborne Laser Scanning

DC: Dual Contouring – A class of surface reconstruction algorithms

DEM: Digital Elevation Model

DSM: Digital Surface Model

ENVI: Environment for Visualizing Images – A remote sensing software package

FN: False Negative

FP: False Positive

GIS: Geographic Information Systems

IBC: International Building Code

ISPRS: International Society for Photogrammetry and Remote Sensing

LAS: LASer file format – A common file format for processed LiDAR data

LGPL: Lesser GNU Public License – An open source software license

LiDAR: Light Detection and Ranging

QEF: Quadratic Error Function

RMSD: Root Mean Square Distance

SVM: Source Vector Machine - A class of machine learning algorithms

SPHARM: Spherical HARMonic

TIN: Triangulated Irregular Network – A GIS surface data format

TN: True Negative

TP: True Positive

1 Introduction

Automated 3D building reconstruction encompasses a broad range of techniques which are applicable to remote-sensed data from a variety of sources. Applications of 3D building models reconstructed from remote-sensed data include not only visualization and urban planning but also problems environmental monitoring such as air and noise pollution monitoring and heat transfer modeling, as well as modeling the propagation of telecommunications signals through the urban environment (Hron and Halounová 2015) and damage assessment for disaster response. When reconstructing building models for large urban areas, the automated nature of such techniques is a major benefit, as manual modeling requires a large time investment for even neighbourhood-sized areas. Data sources include aerial and satellite imagery, as well as Light Detection and Ranging (LiDAR) scanning. The latter has several advantages, principal among which is that depth information is inherent to the sensing technology and is collected at time of sensing as opposed to being the product of post-collection analysis (Musialski et al. 2013) – in other words, LiDAR data is inherently 3-dimensional. LiDAR scan data is however expensive to collect at high density levels and for large areas, which poses a challenge for widespread adoption for urban modelling. This thesis presents a novel methodology for assessing the relationship between data density and accuracy for 3D building reconstruction algorithms using real-world LiDAR data. We seek to provide clarity as to what level of density is adequate for the purposes of building reconstruction so that those commissioning data collection for this purpose can make informed decisions as to their requirements.

1.1 Aerial LiDAR Technology

LiDAR encompasses a variety of remote sensing technologies, all relying on the sensing of light emitted by a laser and reflected by subject phenomena. LiDAR scanners may be stationary, or ground vehicle based, or mounted to aircraft, in what is referred to as Airborne Laser Scanning (ALS). ALS's main advantage over other modes of LiDAR

remote sensing is the high speed of data collection over large areas, particularly when compared to stationary terrestrial scanners. Most airborne laser scanning technology uses ‘time-of-flight’ to determine distance; that is, they measure the time between emission and reception of a pulse of light and calculate the distance traveled by that pulse using the known speed of light in the atmosphere (Vosselman and Maas 2010). Combined with the known angle at which the laser pulse is emitted and the known position of the scanner itself, as determined by a combination of GPS and inertial measurement systems, the position of the reflecting surface can be computed and stored as one or more return(s). Several returns per pulse are possible in the case of semi-transparent or permeable surfaces such as tree canopy, since the laser beam has an areal footprint and can therefore shine ‘through’ thin objects such as tree branches and power lines. The spatial accuracy and precision of a return varies, not only with footprint of the beam but also with the type and texture of the surface it represents (El Hakim et al. 2008). In addition to discrete returns, an increasing body of research has formed using full-waveform techniques, which measures the reflection of emitted light in much greater temporal detail; this technique is more common in applications such as forestry and surface topography (Cheng et al. 2017) than for building extraction and modelling.

The spatial resolution of discrete-return LiDAR data is typically characterised in terms of the number of return points per unit of horizontal area, usually points per square meter. Mean distance between points is also sometimes used, though more often the distance between points is used as a basis for subsampling than as a density metric. The spatial resolution of aerial LiDAR data is dependent on several factors including the specifications of the sensor, the height of flight of the airborne platform carrying the sensor, and the platform’s speed. In general, these factors combine to produce a trade-off between speed of collection and spatial density for any given sensor system. An aircraft flying higher and faster will collect the same number of points over a larger area than an aircraft flying slower and lower, producing lower-resolution data.

1.2 Building Extraction and Reconstruction

The meaning of the terms ‘extraction’ and ‘reconstruction’ in the context of building identification and modeling varies across the literature, so it is useful to define them as

conceptualized here. ‘Building extraction’ for our purposes refers to the process of identifying areas and/or data corresponding to building features, including the production of 2-dimensional outlines (‘footprints’) representing the locations and shapes of buildings and potentially also the extraction of point cloud subsets corresponding to building surface returns, in some methods. ‘Building reconstruction’ here refers to the production of 3D models from LiDAR data or other remotely sense data, with building extraction processes potentially but not necessarily serving as a by-product or intermediate step to full 3D reconstruction. Although simply extruding extracted building footprints to an average height can be considered a form of building reconstruction and has been used to check the 3D accuracy of footprint extraction (Wang et al. 2016) we do not consider this to qualify as building reconstruction as roof shape and within-footprint height differences are not respected. Another process which may be characterised as intermediate between building extraction and building reconstruction is boundary line extraction, which identifies both the overall building footprint and within-footprint breaks in elevation. Such processes are often part of a full building reconstruction workflow but are also sometimes developed in a stand-alone context (Tseng and Hung 2016).

1.3 Research Question and Objectives

The overall goal of this study is to determine the relationship between point density in input LiDAR data and the accuracy of 3D building models extracted from that data. Based on previous research examining the effects of LiDAR data attributes on building model extraction (Lohani and Singh 2008), we expect a point of diminished returns for point cloud density, above which reconstruction accuracy does not significantly vary. Below this level of density, however, reconstruction accuracy is expected to deteriorate with decreasing point cloud density; we aim to both establish the level of diminished return and to characterize rate of deterioration for point cloud densities below it.

City-scale LiDAR datasets can be expected to contain a range of building sizes and architectures, and there is reason to expect that reconstruction performance will not be uniform across them for a given point cloud density. At the very least, small buildings are considered more difficult to correctly identify than larger ones (Soininen 2016). It may be the case that the point of diminished returns on the reconstruction accuracy of detached

houses, for example, is different than for large commercial outlets. Establishing the degree of difference in the relationships between point cloud density and accuracy for buildings of various sizes and purposes is another research objective.

In summary, our research questions are as follows:

1) Is there a point of diminishing returns on accuracy for increasing density of input data for building reconstruction from aerial LiDAR data?

2) Does the relationship between point cloud density and accuracy as assessed based on 2D footprints differ from the relationship with accuracy as assessed by 3D similarity metrics?

3) Does the relationship between accuracy and point cloud density vary between the two 3D reconstruction methods tested?

4) How does building size affect the accuracy of 3D reconstruction, and does the relationship between accuracy and point cloud density vary with building size?

1.4 Organization of this Thesis

The most general background information relevant to this thesis has been presented in the above sections of the introduction. Chapter 2 presents the literature review, which describes previous work in 2D extraction and 3D reconstruction of buildings from LiDAR data as well as methods of analyzing the accuracy of extraction and reconstruction. The literature review also details what has already been published on the question of the effect of point cloud density on extraction and reconstruction accuracy, and gives background information on polygonal 3D models as relevant to our method of 3D model comparison. Chapter 3 details our methods for reference data collection, point cloud thinning, building extraction and reconstruction, and accuracy assessment. Chapter 4 presents our results, which Chapter 5 discusses in depth. Finally, Chapter 6 presents our conclusions.

2 Literature Review

Relatively little literature exists on the effect of point cloud density on 2D building extraction, and virtually none on its effect on 3D reconstruction accuracy. What information is available often takes the form of informal advice; the documentation for LiDAR software TerraSolid, for example, suggests good performance for all buildings at densities above 2 p/m^2 (Soininen 2016). Point clouds used for building reconstruction range from as thin as 0.1 p/m^2 (Rottensteiner and Briese 2002) to as dense as 225 p/m^2 (Truong-Hong and Laefer 2015).

A thorough study of the effectiveness of building reconstruction at different point cloud densities requires an understanding of both reconstruction itself and of the ways in which a reconstructed building model may be assessed for accuracy. Building reconstruction must be understood as closely related to building extraction, the identification of building footprints from remote-sensed data. Reconstruction of a building model in 3D inevitably involves identifying its footprint in 2D, either explicitly as a part of the modelling process or implicitly as the external boundary of a planform projection of the 3D model. Therefore, the accuracy of building model reconstruction has both 3D and 2D components, representing respectively the accuracy of the 3-dimensional model itself and the accuracy of that model's 2-dimensional footprint. Assessment of the building footprint identified during reconstruction allows for the effectiveness of the footprint to be gauged and analyzed in relation to 3D accuracy. Although 2D and 3D accuracy are expected to be strongly correlated, measuring the strength of the relationship gives insight into how strongly the post-identification reconstruction process influences 3D accuracy. Even more crucial to the research question is 3D model accuracy assessment, for which several metrics have been developed. To use them, it is necessary to ensure that the 3D models used are topologically valid, the criteria for validity varying depending on the method used.

2.1 Conceptualization of Buildings

Conceptual uncertainty, in the context of remote sensing, arises from vagueness in the definition of what is to be studied or classified. Conceptual uncertainty in remote sensing of buildings arises from the arbitrary and culturally variable way in which humans differentiate buildings from other permanent or semi-permanent artificial structures. Kuhn (2002) defines a building as "a structure that has a roof and walls and stands more or less permanently in one place". While this definition fits with the popular conception of a building, it is not adequate to accurately classify every possible building and non-building structure that might be encountered by remote sensing technology. Consider, for example, a geodesic dome, which has no distinct walls but would nevertheless be classified as a building by most observers, or a derelict RV, which satisfies all of Kuhn's criteria but would not be considered by most to be a building.

The International Building Code (IBC) also defines buildings broadly, as "any structure used or intended for supporting any use or occupancy" (International Code Council 2011) Such a definition could be liberally interpreted as including mobile structures such as vehicles, which the scope of the IBC does not include, and to structures not inhabited by humans such as freestanding antennae or storage tanks, which are within the defined scope of the IBC. The objective of a building code to set out legally binding requirements to ensure public safety and accessibility (Potworowski, Murray-Choudhary, and Losfeld 2010) and the latter buildings fit within that scope, but such structures are not 'buildings' in the conventional sense as they have no intention of even occasional occupancy by humans. The IBC's definition is therefore not suitable for this study's purposes without modification.

Given the concerns raised for the above definitions, it is useful to devise a comprehensive definition of 'building' for the purposes of this study. That definition is composed of the following three criteria:

a) Permanence: A building is designed to remain in its constructed form for at least a year and is not intended to be regularly disassembled and reassembled, as (for example) a tent is.

b) Immobility: A building cannot move under its own power or be moved by towing, and performs its intended function while stationary. This first part of this criterion excludes both land vehicles and trailers as well as floating objects such as ships and barges; the second excludes objects such as cargo containers whose primary function is to contain cargo while in transport. It also excludes machinery such as dockyard cranes which move during normal function. Neither part excludes semi-movable buildings such mobile homes or portable classrooms, which must be loaded onto a vehicle for transport and fulfill their primary function only when unloaded.

c) Habitation: The structure must be intended for human occupancy for at least part of its operating life post-construction. This excludes tanks for fluids as well as electrical substations and other pieces of stationary outdoor machinery, but not for storage buildings such as barns, sheds and warehouses which must admit human occupants for the purposes of loading and unloading or maintenance. It also does not exclude most types of grain silo, namely those that are designed to accommodate human entry as part of maintenance.

2.2 Building Footprint Extraction

Most building footprint extraction methods for LiDAR data are more specifically roofline extraction methods; they detect the 2D outline of roofs, not walls (Potůčková and Hofman 2016). The reason for this can be readily intuited from the appearance of most aerial LiDAR point clouds; the vast majority of points represent sky-facing surfaces such as roofs and the ground, with very few points representing vertical features such as walls. Although aerial datasets with large amounts of wall and façade points do exist, these are rare and high-density datasets gathered with the specific intent of maximizing vertical surface data (Truong-Hong and Laefer 2015). Footprint extraction methods encompass a diverse array of processes and algorithms ranging from varieties of edge detection algorithms (Sajadian and Arefi 2014, Zhang et al. 2017) to shape-based contouring (Yari et al. 2014) and neural networks (Silván-Cárdenas and Wang 2011, Liu et al. 2013). Early approaches frequently rely on rasterization of range data into a DSM, filtering to produce a DEM, then algorithmically separating buildings and vegetation from a normalized DSM

(Brunn and Weidner 1997); conversely, buildings may be extracted from a DSM for the purposes of creating a building-free DEM (Priestnall, Jaafar, and Duncan 2000). Although DEM-related techniques and applications clearly drive much of the early work on building extraction, the applications of LiDAR data to 3D city modeling were also realized relatively early in the maturation of aerial LiDAR sensing (Haala and Brenner 1999).

Data fusion approaches are popular, frequently involving high-resolution multispectral imagery (e.g. Li et al. 2013), another common data source for building extraction. Fusion of LiDAR and imagery is common enough to be considered a third approach, complementing LiDAR-only and imagery-only building extraction methods, integrating height and return-intensity information with texture and edge information from optical images (Lee, Lee, and Lee 2008). Some integration techniques use imagery only to address the problem of vegetation removal, but more in-depth data fusion methodologies use both LiDAR and imagery to detect footprints as well (Awrangjeb, Ravanbakhsh, and Fraser 2010).

The work of Sohn and Dowman (2006), which fuses high resolution satellite imagery with low-density LiDAR data, has been identified as particularly influential in terms of citation by other authors (Tomljenovic et al. 2015). Most techniques that fuse LiDAR with other data do so with some other remote sensing technology, but non-remote-sensed data such as address information has also been used (Jarzabek-Rychard, 2012).

Techniques which work directly on the point cloud, such as ‘marked point’ methods, have the advantage of skipping an intermediate rasterization stage (Yang, Xu, and Dong 2013) but as a consequence cannot leverage well-developed raster-based algorithms. Methods that rely on separating the point cloud into subsets representing individual buildings are often associated with 3D building reconstruction, but it is equally possible to use such techniques to delineate building boundaries in 2D, for footprint extraction (Sampath and Shan 2007). Fully 3D methods such as plane detection, frequently applied to 3D roof reconstruction, have also been applied productively to footprint extraction (Varghese, Shajahan, and Nath 2016). Such ‘reverse’ applications of reconstruction

methods to the problem of footprint extraction illustrate the interrelatedness of these two problems, and the way in which developments in one field can contribute to advancements in the other.

Extraction methods often seek to regularize the produced footprints in some way in order to reduce noise and produce visually ‘clean’ boundaries. Established methods include the traditional Douglas-Peucker line simplification algorithm, which generalizes lines by reducing the number of vertices, Model Hypothesis-Verification algorithms that generate sets of simplified line segments and select the best fitting candidate, least-squares adjustment of initial footprint lines, and rule-based regularization processes (Jwa et al. 2008). Frequently such methods rely on the detection of ‘principle directions’; since most buildings have rectilinear footprints (i.e. footprints composed of straight lines at right angles to each other), automatic processes may be employed to estimate these directions and ‘snap’ lines composing the footprint to them (Yunfan and Hongchao, 2011) Alternative regularization methods employing shape-fitting strategies such as minimum bounding rectangles (Yunfan and Hongchao 2011) have also been developed.

2.3 3D Building Reconstruction

3D building reconstruction consists of the generation of 3D models from remote-sensed data. Applications of 3D building data include visualization for urban planning, real estate and entertainment but also in the provision of data for analysis of the urban environment, such as for solar energy assessment (Martinez Rubio et al. 2016). As with building footprint extraction, several remote sensing techniques have been leveraged for building reconstruction, including RADAR and orthophotography, but our review confines itself to method using aerial LiDAR, one of the most common methods found in literature (Wang 2013).

3D building reconstruction from LiDAR has, since early in the development of the field, been approached from two distinct classes of technique: data-driven methods and model-driven methods (Maas and Vosselman 1999). Data-driven methods seek to construct building models by reconstructing shapes such as lines and planes apparent in the LiDAR point cloud. Model-driven surfaces, on the contrary, seek to approximate the shape of

buildings by fitting parametric 3D shapes to the point cloud, often in a ‘building-block’ approach. Certain authors (e.g. Jarząbek-Rychard and Borkowski 2016) term these approaches ‘bottom-up’ and ‘top-down’, respectively. Both approaches can achieve comparable accuracy; model-driven approaches rely on having suitable models to match to the data, while data-driven approaches are more ‘universal’ in terms of geometry but can produce deformed models, particularly where the point cloud is unevenly distributed or too sparse in relation to building feature size (Tarsha-Kurdi et al. 2007)

Model driven approaches reconstruct buildings by fitting pre-defined objects or primitives to the data. Frequently, such approaches attempt to match objects in a library to features in the dataset; the contents of the library constrain the range of possible output geometries, which can be advantageous if the library is compiled with knowledge of the architecture present in the study area (Taillandier and Deriche 2004). An overly-limited library of primitives, however, will not be able to accurately reconstruct a wide range of architectural forms. Even for study areas with homogenous architectural characteristics, model-based methods must have some way to vary the shape and scale of individual primitives in order to adequately represent differently sized and proportioned buildings. Such parameters may include not only scale and rotation but also specifications of roof symmetry, slope, and other information (Lafarge et al. 2010). Building primitives may be prescribed 3D shapes such as prisms or polygons, but a high degree of flexibility can be achieved by conceptualizing primitives as topological models that may be combined to build an overall model of roof structure (Xiong et al. 2015). Model-driven approaches may also be integrated into data-driven reconstruction in order to recover details too small to be reconstructed using a purely data-driven approach (Cao et al. 2017).

Data-driven approaches frequently work on the raw LiDAR point cloud, but DSM-based methods have also been developed (Yan et al. 2017). The characteristic strategy of data-driven approaches is to construct models from shapes that approximate the distribution of building points in the point cloud; key to this process is segmentation, the grouping of points representing individual surfaces. A very common method of doing this is plane-fitting; using algorithms to detect planar sets of points and construct planes to represent them. Once roof points are segmented, roof models may thereby be assembled by

agglomerating nearby roof planes, with building models optionally generated by extruding walls to the detected ground level along the outer perimeter. Random Sample Consensus (RANSAC) algorithms are frequently employed for plane-fitting in building reconstruction (Sun and Salvaggio 2013) and indeed in point cloud-based surface reconstruction in general. Multiple RANSAC plane-fitting algorithm variants have been developed, all of which involve generating multiple random planes to fit a set of points and selecting the plane that fits the most points (Qian and Ye 2014). Of course, not all points in a LiDAR point cloud necessarily represent planar building features, or indeed features at all; a plane-based segmentation approach must have some method of excluding such points (Sampath and Shen 2010).

Region-growing is another commonly encountered segmentation approach (e.g. Xu et al. 2017). Region growing algorithms operate by selecting a point in the dataset and grouping nearby points with it in an iterative process, until no suitable nearby points can be found. An important difference from plane-fitting is that iterative growing; RANSAC methods are also iterative but iterate one surface at a time, rather than iterating multiple times as a single surface is identified. Optimization strategies deployed for region-growing algorithms in this context include filtering to select ideal 'seed points' to initiate region growing as well as voxel-based aggregation of multiple points (Vo et al. 2015). Some criterion or criteria must be used to ensure that grouped points represent the same surface; using estimated surface normal vectors is one popular solution (Sampath and Shen 2010, Chen et al. 2014). As with plane-fitting methods, there must also be a means of separating vegetation and ground points from those that represent buildings; this may be done prior to segmentation or accomplished by separating ground and/or vegetation segments from those that represent buildings; in the former case, the difference in spatial distribution between vegetation and building/ground points may be used to filter vegetation points in advance of segment identification (Zhou and Neumann 2008).

The 3D Hough transform was developed from a 2D process used for signals processing and has been adapted for matching 3D shapes such as planes (Tarsha-Kurdi, Landes, and Grussenmeyer 2007). The process relies on representing a set of points in a different mathematical space, one which facilitates the detection of the desired primitive. The

primitive detected is often a plane, but detection of lines and 3d primitives such as cylinders is also possible (Rabbani and van den Heuvel 2005). The Hough transform may be applied on its own or used in combination with region growing methods by grouping the initial products of the transform into regions using a region-growing algorithm (Leng, Xiao, and Wang 2016). Optimizations such as Randomized Hough Transform modify the algorithm to work more efficiently by reducing the effective number of points that must be processed without compromising the thoroughness of plane detection (Maltezos and Ioannidis 2016).

One of two methods used to generate 3D building models in this study is an existing implementation of the 2.5D dual contouring method developed by Zhou and Neumann (2010), henceforth referred to as dual contouring or DC, which has been the subject of recent further development by Orthuber and Avbelj (2015). Dual contouring is a data-driven process initially developed as a fully 3D method for use with high-density point clouds (Fiocco et al. 2005), as a relative of the Marching Cubes algorithm, a common surface reconstruction approach (Fuhrmann, Kazhdan, and Goesele 2015). The details of the specific implementation used in this case is described in detail in Zhou and Neumann, (2010) but a basic overview can be given as follows: First, individual roof points are segmented into roof layers using a region growing algorithm. Then, a two-dimensional square grid called a *quadtree* is created, a grid subdivided into cells in which each atomic 'leaf' or cell is one of four children of a parent cell, which is in turn a child of a larger parent cell, and so on. The spacing l_g of the atomic subtree cells is an important parameter in the DC modeling process. Each node, or center point, on the grid is assigned an elevation and surface normal based on the mean of the nearest *n*th data points. These nodes form 'hermite data', which in this context are sets of datapoints in which both a value and its derivatives are known (Ju et al. 2002, Fuhrmann Kazhdan and Goesele 2015). In this case, the value is elevation *z* with the derivative *z'* with respect to *x* and *y* calculated via covariance analysis (Zhou and Neumann 2010). Boundary points are also estimated, representing points on edges between different roof layers, separated by vertical walls (Zhou 2012). Each quadtree cell contains at a 'hyperpoint', a point with a single *x* and *y* coordinates but with (potentially) multiple z coordinates, the number depending on whether it represents an intersection of two roof polygons or the edge of a

vertical wall (Orthuber and Avbelj, 2015). The *x*, *y* and all *z* coordinates of the hyperpoint are calculated by minimizing a 2.5D quadratic error function (QEF) involving all sample and boundary points in the given hyperpoint's cell. Once QEFs have been calculated for all atomic-level quadtree cells, said cells can be amalgamated into larger and larger cells, each amalgamated cell having a QEF composed of its child cells' QEFs and a hyperpoint defined by the error-minimizing solution of that QEF. Subtree collapse halts when no four child cells can be collapsed without the residual of that prospective cell's QEF exceeding a given threshold δ, which controls the amount of simplification-induced error tolerated in the final model. The end product is realized by connecting the final set of hyperpoints into a crack-free mesh. Additional safeguards to preserve sharp features and ensure topological correctness also constrain subtree collapse, and a principle-direction snapping feature adjusts model features to better align with those of its neighbours.

2.4 Previous Research on the Effects of Point Cloud Density on Extraction and Reconstruction

Previous research has established building detection as requiring relatively high spatial data density relative to other applications; while Pirotti and Tarolli (2010) find that 0.2 ppm^2 point clouds are sufficient for relief mapping, building extraction in existing literature is typically performed using data at least an order of magnitude denser. Research using simulated LiDAR data has established point cloud density as the only attribute of LiDAR data to have a significant impact on reconstruction accuracy (Lojani and Singh 2008). One study using a raster object-based extraction algorithm reports noticably reduced accuracy, on the order of 20% loss of completeness, for point clouds below 18 ppm^2, and fewer than 50% of buildings detected at point cloud densities below 7 ppm^2 (Tomjlenovic and Roussel 2014). In contrast, a different method which included reconstruction of simple roofs found area correctness in excess of 90% for point cloud densities as low as 2 ppm^2 (Lohani and Singh 2008). The large discrepancy in reported performance may have more to do with different metrics than different methods; the former study uses per-building completeness and correctness metrics while the latter reports in terms of area.

Apart from studies looking specifically at the question of building identification and reconstruction as varying in accuracy with point cloud density, it is possible to gain some insights from research looking at these topics more generally. There is general agreement that building size is an important determinant in whether or not a building can be detected using a given extraction method, both theoretically (Kodors and Kangro 2016) and in practice. Some authors quantify rates of reconstruction based on absolute building area (Rutzinger, Rottensteiner and Pfeifer 2009) but others suggest quantifying building area relative to point cloud density for the purposes of reconstruction quality assessment (Potůčková and Hofman 2016).

2.5 Assessment of Extraction and Reconstruction Accuracies

Of the two processes, accuracy metrics for building extraction are better established than those for building reconstruction. A variety of assessment techniques exist, and no consensus on a standard set of performance metrics has as of yet been reaches (Avbelj, Muller and Bamler 2015). Disagreement between methods stems from the choice of entity to form the basis of comparison, the definition of what is and is not a building to be detected, and the method of comparison itself (Rutzinger, Rottensteiner, and Pfeifer 2009). In many cases, extraction results are either not assessed numerically or assessed without a full description of the parameters used, making comparing the performance of different published methods challenging (Potůčková and Hofman 2016).

In early literature, the rate of detection is the most common gauge of accuracy (Song and Haithcoat 2005), which gives no indication of geometric correctness. Measurement of detection rate entails both quantifying the number of buildings detected and identifying detected building objects with objects in reference data, in order to distinguish between true and false positives. More developed accuracy assessments generally assess accuracy in areal terms as at least part of their analyses. Building extraction is at its core a classification process; one in which building footprint areas are distinguished from non-building ‘background’ land covers and identified as separate based on non-contiguity. It is therefore unsurprising that many of the most widely reported accuracy metrics are classification accuracy measurements, which assess producer and user accuracy and related metrics such as quality percentage (Shufelt 1999), completeness, and correctness

(Rutzinger, Rottensteiner, and Pfeifer 2009), the latter two being particularly common (Tomljenovic et al. 2015). Overall assessment of extraction in this manner may conducted on a pixel-by-pixel basis (Ekhtari et al. 2009), very much like traditional land cover classification assessment. Such metrics classify pixels as being True Positive (TP), which are accurately classified as buildings; False Positives (FP), where a building is 'detected' where none actually exists; and False Negatives (FN), where the detection method fails to identify a building where one exists; see Figure 2-1 for a visual representation.

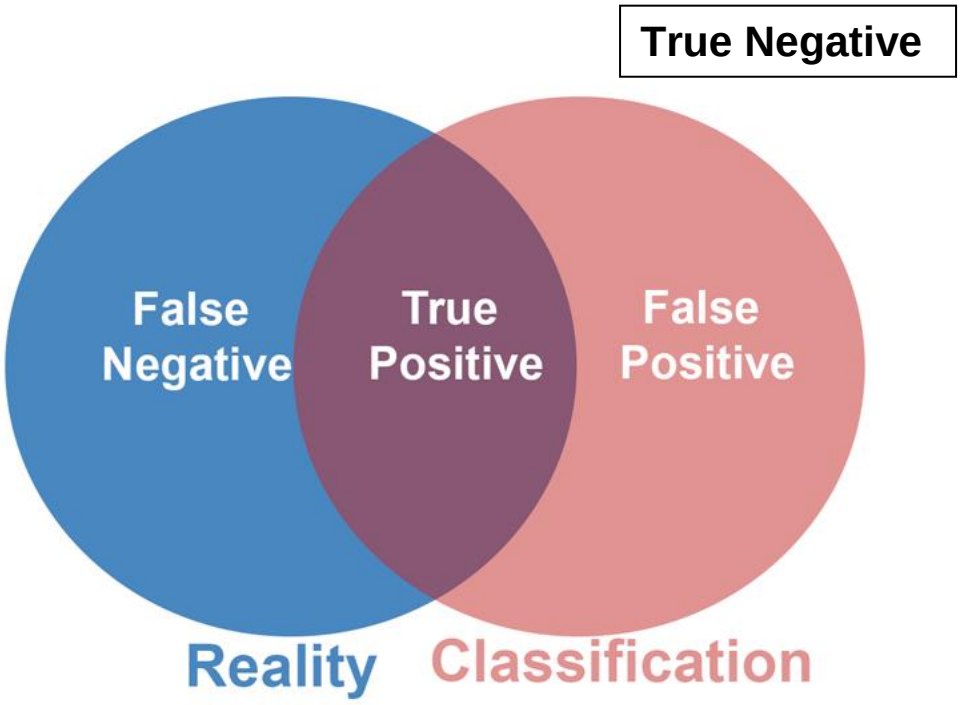


Figure 2-1 A diagrammatic explanation of classification accuracy. The blue circle represents real building area in the input, the red circle represents building area in the classification. True Negative area is represented by the area outside both circles.

From these classifications, metrics of accuracy can be calculated. Completeness measures the proportion of existing buildings (by area) that are identified in the output and is calculated by Equation (1).

$$\text{Completeness} = \frac{TP}{TP+FN} \quad (1)$$

The counterpart to completeness is correctness, which measures the proportion of area identified in the output that is in reality building area. It is calculated by Equation (2).

$$\text{Correctness} = \frac{TP}{TP+FP} \quad (2)$$

Finally, Q combines both metrics, providing an overall measure of how well buildings are classified. It is calculated by Equation (3).

$$Q = \frac{TP}{TP+FP+FN} \quad (3)$$

Note that although true negative area is not explicitly measured, it is accounted for in both correctness and Q, as any misclassification of non-building area as building area results in a false positive, reducing both Q and correctness.

Object-based analyses however are common in building extraction and allow for building-specific as well as overall accuracy assessment, including comparisons of building shape in addition to areal overlap (Zeng, Wang, and Lehrbass 2013). Object-level quality assessments are particularly useful since not all buildings are extracted to an equal level of accuracy (Avbelj and Muller 2014), and the reliability of extraction for different types of buildings is frequently of interest. Proposed shape similarity metrics include distance between matched check points (Song and Haithcoat, 2005). Such comparisons frequently require one-to-one associations between extracted and reference buildings, which can create problems when, for example, a single building is erroneously extracted as multiple separate structures. For that reason, such methods usually require some degree of manual intervention to split or merge reference footprints for comparison, although in most cases metrics such as overlap threshold and principle-direction matching can establish matches for most structures automatically (Wang et al. 2016). The

way in which erroneously merged or split buildings are evaluated introduces further variability in accuracy metrics across studies. Researchers may, for example, only count an extracted polygon as correct if and only if it represents a single building (Tomljenovic and Roussel 2014), and rates of over- or under-segmentation may or may not be reported (Siddiqui et al. 2016).

Methods of accuracy assessment for 3D building models are less developed than those for building footprints, but a variety of methods do exist to assess model accuracy and quality. One approach compares the generated model with the original point cloud data, measuring the distance between building points and the resulting surface (Oude Elberink and Vosselman 2011). Another approach is analogous to that taken by traditional footprint accuracy assessment, measuring the intersecting and non-intersecting volume of the model with a reference (Mohammed et al. 2013). In doing so, it is possible to extend the familiar principles of classification accuracy assessment to the assessment of 3D models; along with the familiar metrics of correctness, completeness and quality. Another method of accuracy assessment for 3D models involves comparing the distance between select points on model and reference (Gómez-Gutiérrez et al. 2015), but such methods are more challenging in LiDAR reconstruction relative to photogrammetric methods where control points are readily available for such comparisons. In addition to point and volume-based comparisons, it is also possible to compare building surfaces. Direct comparisons, which may measure Euclidean distance between paired surfaces (Akca et al. 2010) are simple conceptually but require some way of matching surfaces between the model and reference. Surface matching is computationally intensive for highly detailed or noisy models composed of millions of polygons.

An alternative is to parameterize the model, transforming it into a form that is more readily comparable. The SPHARM (Spherical Harmonic) method of shape comparison involves mapping the surface of a closed 3D object onto a unit sphere, then parameterizing that mapping using spherical harmonics, producing a set of parameters which may be compared numerically (Brechbühler, Gerig and Kübler 1995). To begin with, the surface of the subject shape must be mathematically projected onto the surface of a unit sphere, such that every point on the sphere has values x, y, and z, equal to the

spatial coordinates of the corresponding point on the surface of the original object. Each spherical harmonic function $Y_\ell^m(\theta, \varphi)$ is of a given positive integer degree ℓ and integer order m. A spherical harmonic function of degree ℓ for every value of m such that $|m| \leq \ell$; so for example there are three spherical harmonic functions for degree one (m=-1,0,1), five for degree two (m=-2,-1,0,1,2), et cetera. To represent the spherical mapping of an object, each function is assigned a set of three coefficients (one for each spatial dimension) referred to as its parameters. The values of these functions for any one point on the surface of the sphere equals, when multiplied by the relevant parameter and summed, the spatial coordinates of the corresponding point on the original shape. Thus, the shapes of two objects may be compared quantitatively by comparing the parameters of their SPHARM representations.

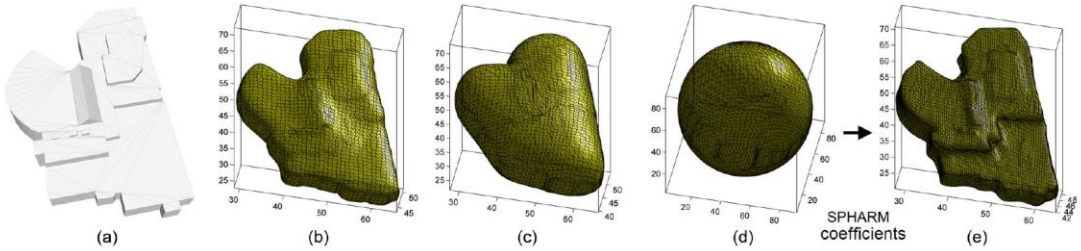


Figure 2-2: An illustration of the SPHARM process. The polygonal model (a) is voxelized and mapped to the unit sphere using a heat diffusion algorithm illustrated by (b) and (c), with the final mapping shown by (d). The building model as approximated by SPHARM coefficients of up to degree 80 is shown by (e). Reproduced with permission from Zeng, Wang, and Lehrbass (2013).

Lower-degree harmonics represent low spatial frequency (coarser) patterns, whereas high-degree harmonics represent finer details (Chung, Dalton, and Davidson 2008). There are an infinite number of spherical harmonic functions, so in practice spherical harmonic representations are computed up to a desired degree; the higher the degree, the closer the approximation of the original spherical mapping. As the three-dimensional extension of Fourier analysis, the spherical harmonic transform allows for the analysis of

geometry as a problem of frequency (Benseddik et al. 2016); crucially, the produce of the transform may be analyzed using descriptors that are positionally and rotationally invariant (Brechbühler, Gerig and Kübler 1995), meaning no surface-matching process is required so long as model and reference can be matched. SPHARM-based comparison methodologies are popular in medical imaging analysis, where they allow for shape comparison of 3D-scanned organs and tissues (Thompson et al. 2013, Paniagua et al. 2011). The scale, rotation and positional invariance of SPHARM makes it highly applicable to the problem of building shape comparison as well (Zeng, Zhao, and Wang 2014). One downside of traditional SPHARM is that lower-degree spherical harmonics tend to encode most of the shape information of the original object, while higher degree harmonics are very noisy; this can be addressed by including a weighting function such that higher-degree parameters are weighted less when the SPHARM representation is computed (Chung, Dalton, and Davidson 2008).

Accuracy assessment of any type usually requires accurate reference data for comparison, the source of which varies from study to study; researchers often create reference data manually by digitization, often of the same dataset used for extraction; this option is particularly viable where LiDAR and optical data are used in conjunction, and where the study area is relatively small (Uzar and Yastikli 2013). Use of the source dataset for accuracy assessment is recommended by some authors, since deriving reference data from an external dataset introduces additional uncertainty stemming from inaccuracies in the reference (Zhang and Geng 2006). Since LiDAR data itself contains spatial error of variable magnitude, it is possible for reference data to be ‘too’ accurate, resulting in measured errors that are not attributable to the method in question (Shen 2008). That being said, most studies derive their reference data from external sources (Potůčková and Hofman 2016). In some cases, pre-existing data is used from sources such as cadastral maps (Tomljenovic et al. 2015). An increasing number of studies make use of established benchmark datasets such as the ISPRS (International Society for Photogrammetry and Remote Sensing) benchmarks of Vaihingen, Germany and Toronto, Canada (Rottensteiner et al. 2012); doing so enables standardisation of input data across multiple studies at the expense of limiting study to one of a few pre-selected sites. Accuracy assessment can be carried out without external reference data by comparison between the

extracted building outline and manually selected points in the LiDAR cloud (Seo, Lee and Kim 2014) but such methods is not frequently used compared to those that make use of reference data of some type.

2.6 Characteristics of Polygonal 3D Models

3D model comparison methods impose certain requirements on the topology of the subjects if the comparison is to be valid. Polygonal models, often referred to as 'meshes', can have geometric properties that are not found in real objects; surfaces, for example, can self-intersect, and models may not necessarily enclose a finite volume. The comparison methods used in this study demand models with certain properties for both algorithmic and conceptual reasons; the most obvious example of the latter being that volume comparisons require both compared models to enclose a finite fixed volume. The terminology used when discussing polygonal model topology does not necessarily strictly adhere to formal mathematical usage and can become confusing since they may be applied both to the real objects being represented and the collections of vertices, edges and faces that represent them digitally; here I follow the conventions of Ju (2008). It is common, especially in informal contexts, to refer to geometrically incorrect models as being non-manifold, or more precisely, non-2-manifold. 2-Manifold meshes are those in which every point is locally homomorphic to a disc; informally, this means that the surfaces of these meshes can be divided up into pieces that, when sufficiently small, resemble flat or creased two-dimensional discs (Botsch et al. 2007). In practice, not all unbounded 2-manifold surfaces are fit for comparison, as not all enclose a volume. To do so, a surface must be orientable; it must be possible to determine a consistent surface normal and thereby an inner volume. Non-orientable shapes such as the Klein bottle do not have defined surface normals because they are one-sided, and as a consequence cannot enclose a finite volume as they lack an 'inside' space to measure. Orientability is also important in those comparisons of mesh point accuracy that require surface normal information to match points (as in Zeng 2014). Note that in general, orientability does not guarantee two-sidedness, and vice versa, but in the Euclidean (R^3) space in which 3D objects are modeled an orientable surface is necessarily two-sided, and likewise a non-orientable surface one-sided (White 2009).

To enclose a volume, a 2-manifold must also be ‘closed’, it must have no open faces or ‘holes’ in the mesh. Even if a mesh has no holes, it may not necessarily be 2-manifold; non-manifold features occur in cases where the local topology is not disc-like (Ju 2008). These include complex edges, edges which border on three or more faces (Chang and Ho 2001), as well as non-manifold vertices where the surface of the model ‘pinches’ to a point of zero thickness (Botsch et al. 2017). In extreme cases, reconstructed surfaces from scanned point clouds can produce models so riddled with gaps, holes and non-manifold elements that they are referred to as a ‘polygon soup’ and must be subject to considerable re-processing before use (Jin, Tai and Zhang 2009). Even for manually created models, significant effort must be expended to ensure that meshes are topologically correct and boundaryless.

SPHARM comparison methods place a further requirement on the topology of input models; they must be topologically equivalent to a sphere (Zeng 2014) meaning that in addition to the above topological criteria they must also be of genus-0, lacking the three-dimensional holes that some authors refer to as ‘topological handles’ (Ju 2008). This means that shapes such as the genus-1 torus and their topological equivalents cannot be subjected to SPHARM comparison without modification.

3 Data and Study Area

3.1 Surrey BC

Surrey, British Columbia is a city located in the Vancouver Metro Area, with a population of roughly half a million people as of the latest census distributed over a 316 square kilometer area (Statistics Canada 2017). The city contains numerous suburban neighbourhoods dominated by detached single-family housing as well as a mix of commercial and industrial land uses and denser residential housing. There is also a large amount of agricultural land in the south and east, and patches of forested parkland. Surrey's roof-level landscape is representative of post-WWII developments across Canada and the temperate US; in suburban neighbourhoods hipped or gabled roof shapes are punctuated by tree canopy, whereas flat roofs predominate in largely treeless industrial and commercial centres.

3.2 LiDAR And Supplementary Data

The LiDAR data used for this study is publicly available via the City of Surrey, which contracted Airborne Imaging, a private remote sensing company, to acquire LiDAR and other data for the entire area of the municipality. The LiDAR data was acquired in April of 2013 via manned fixed wing aircraft flying at 1 kilometer above ground level. The resulting point cloud data had a mean density of 25 points per square meter across the entire covered area; see Figure 3-1 for an example. The producers assessed the 95^{th} percentile of horizontal accuracy at 15cm and a 95^{th} percentile vertical accuracy of 8.2cm for flat, hard surfaces (Airborne Imaging 2016).

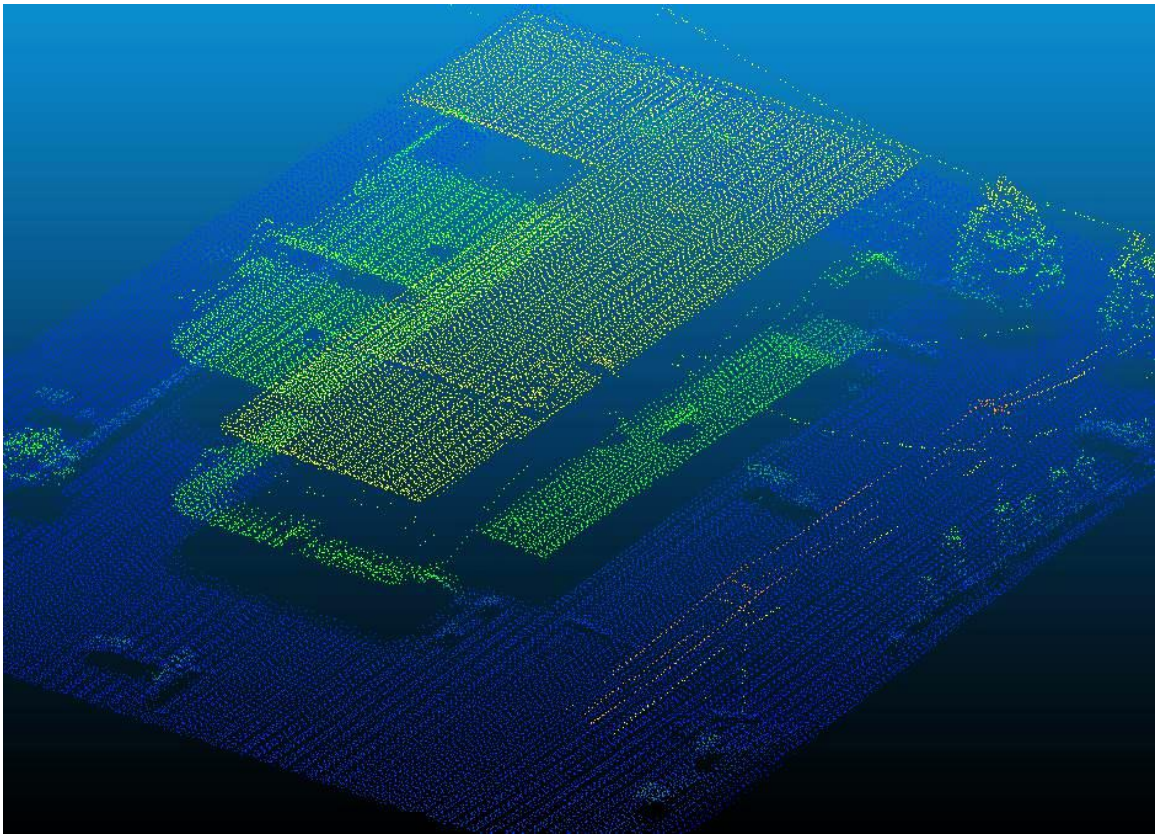


Figure 3-1: Original LiDAR point cloud data, coloured by elevation, representing a hotel and nearby surroundings in the Bridgeview study area.

Since 50% of the scan area overlaps, actual point cloud density is spatially variable; at full resolution, flat areas have a point cloud density of around 30 points per square meter in areas with overlap and 15 points per square meter in areas without overlap. Point cloud density is markedly higher in areas with tree canopy, vertical walls, power lines, and other vertical features.

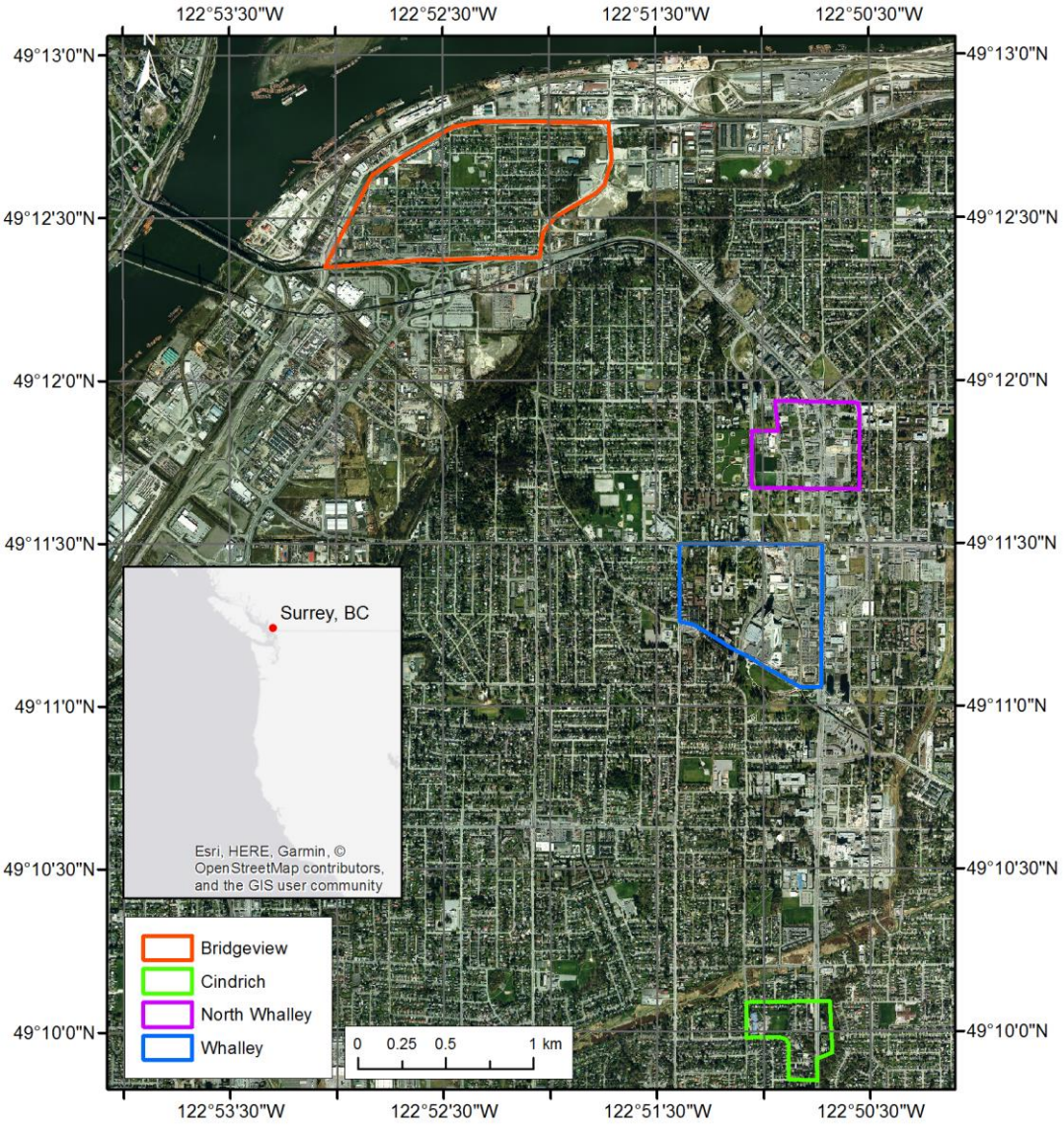


Figure 3-2: Locations of the study areas in relation to each other are shown with an orthoimage background.

Orthoimage data was collected as part of the same project that gathered LiDAR data, and is publicly available via the municipal government of Surrey. The imagery was collected on March 30th, 2013, in true colour with a ground resolution of 10 cm. Other data including hyperspectral imagery and building footprints was also available but was not used, the latter because it was not entirely consistent with the LiDAR and image data.

3.3 Sampling Regions

Time and processing power constraints render it impractical to perform reconstruction and analysis on the entire City of Surrey dataset. Instead, four neighbourhoods were selected for analysis, representing a combined area of 1.9 square kilometers out of the city's 316.4 square kilometer extent. The neighbourhoods were selected with the goal of including a range of architectures, building sizes, and building purposes, and were delineated such that their boundaries always lay on roadways or unbuilt terrain to avoid buildings being only partially included in the analysed data. The location of each neighbourhood relative to the others is shown in Figure 3-2. Figure 3-3 shows buildings footprints coloured by size. Table 3-1 shows area, built area, and total number of buildings in each study area and overall, as well the number of buildings in each study area selected for modeling. The numbers in parentheses show the sampling quota for buildings selected: 'S:' gives the number of buildings selected for their size; for example, the 30 largest buildings in Bridgeview were selected, the 10 largest in North Whalley. 'H:' gives the number of buildings selected by height; the five tallest buildings in Central Whalley were selected for this reason, before any others. 'R:' gives the number selected at random, after the other two quotas had been filled.

Table 3-1: Study Area Characteristics

Region Name	Area (m^2)	Built area (m^2)	Number of Buildings	Buildings Selected for 3D Modeling
1 - Bridgeview	930,553	121,433	907	70 (S:30, R:40)
2 - North Whalley	279,537	54,268	77	40 (S: 10, R: 30)
3 - Central Whalley	527,539	160,263	135	40 (S:20, R:15, H:5)
4 - Cindrich	146,563	25,291	113	50 (L:10, R:40)
Total	1,884,193	361,255	1,232	200 (S:60, R:85, H:5)

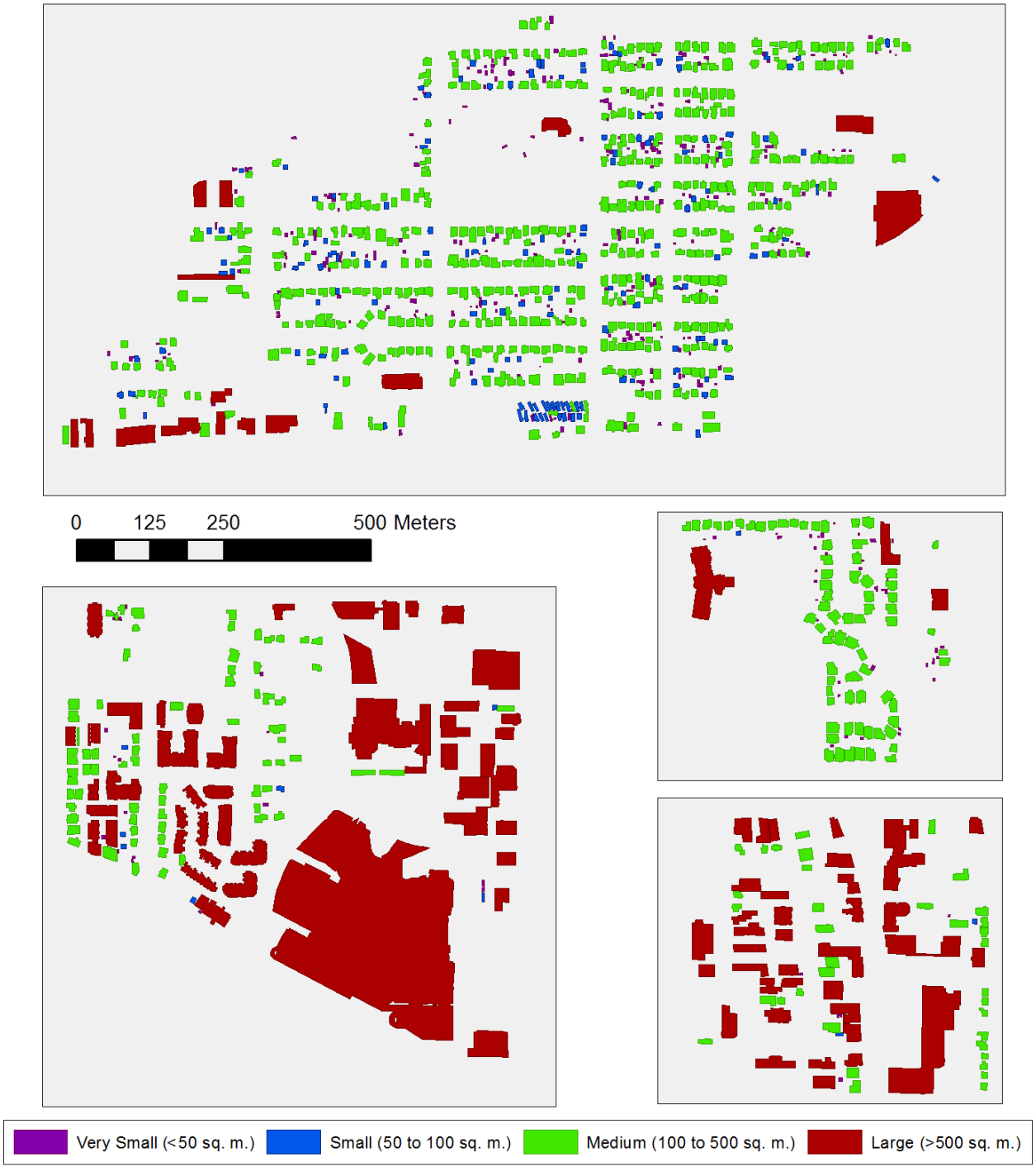


Figure 3-3: Buildings footprints for each study area are shown coloured according to size class. Clockwise from top: Bridgeview, Cindrich, North Whalley, Central Whalley.

3.3.1 Bridgeview

The neighbourhood of Bridgeview is located close to the northern limit of the City of Surrey on the Fraser River. For our purposes, it is bounded on the north and west by the South Fraser Perimeter Road (SFPR), a four-lane highway; to the south by the King George Boulevard, and to the east by Bridgeview Drive. It is the largest study area by a large margin, being 0.93 square kilometers in area. It is predominantly a detached-housing residential neighbourhood, with light industrial establishments along the western edge and various, mostly automotive related commercial establishments on the southern edge. The largest building in the area is a brewery on the eastern limits of the study area, but other large buildings include an elementary school, a community centre, a block of attached auto-repair workshops, and a small office building. Most buildings present in this study area are detached houses with gabled or hipped roofs or small structures such as sheds associated with the former. In addition to the permanent housing, there is also a block of approximately 30 mobile homes in the southern-central part of the neighbourhood. Non-building area includes patches of wooded land, open grass parks associated with the community centre and school, and vacant grass lots, as well as sand and gravel lots. The latter are in use for storage of tractor trailers and scrap metal. Potential challenges for building reconstruction identified in the Bridgeview study area include trees and smaller vegetation mixed in and sometimes overhanging residential buildings as well as the collections of non-building objects such as vehicles, piled scrap metal, and other objects found in association with the industrial sites.

3.3.2 North Whalley

North Whalley is the second and smaller study area located in the Whalley neighbourhood, with an area of 0.28 square kilometers. It is bounded by 108 Avenue to the north, University Drive to the east, 105a Avenue to the south, and Whalley Boulevard to the east. It is mainly composed of flat-roofed commercial and service buildings with a smaller number of detached houses along the eastern edge. Trees are scattered throughout the neighbourhood but there are no major areas of contiguous canopy cover; there are however large areas of grass, gravel and paved lots. Notable large buildings include a

9,800 square meter shopping center and a recreational center with an unusual curving roof profile.

3.3.3 Central Whalley

The Whalley neighbourhood is located roughly 2 km south-east of Bridgeview and is the location of two study areas. Central Whalley is the southern of the two and is comprised of high-density residential structures in its western portion and large commercial establishments to the east. It is bounded by 104 Avenue to the north, 123 Street to the west, Old Yale Road and 100 Avenue to the south, and King George Boulevard to the east, with an area of 0.53 square kilometers. The area is dominated by a combination university campus, shopping center, office high-rise and parking garage, which has a total area of roughly 77,000 square meters. Other large buildings of note include a recreation center and attached elevated light rail station as well as two high-rise apartment buildings. Also present are smaller commercial establishments, low-rise and townhouse-style housing, and a small number of detached houses. Trees are common in the residential area, where unbuilt lots are grassy or wooded. In the commercial area, trees sparser but still present, and parking lots are the predominating non-building land cover. Of special note is the elevated rail line that bisects the study area east of the largest structure; this was not categorized as a building and was identified as a potential source of false positive detections.

3.3.4 Cindrich

Cindrich is a residential neighbourhood with a typical subdivision layout and architectural style and is the smallest study area designated, at 0.15 square kilometers. It is located 2 kilometers directly south of central Whalley and is bounded by 134 Street and 135a Street to the east, 90 Avenue and 88a Avenue to the south, the wooded Quibble Creek to the east, and 91 Avenue to the north. Most buildings are detached houses with dormered or hipped roofs; some of the latter are highly complex with dozens of distinct roof facets. The largest building in Cindrich is an elementary school, other large buildings include a strip mall and a small office building. There is substantial tree cover,

both in the form of wooded patches and in front and back yards. The school's parking lot and yard form the only large continuous unbuilt and un-wooded area in the neighbourhood.

3.4 Point Cloud Density

Each study area has its own pattern of point cloud density. Figure 3-4 shows the density of the original point clouds for all four study areas on a common scale. Note that there is substantial internal variability in the density of the point cloud; this is the product of overlapping coverage by the aircraft-borne sensor. Bridgeview's point cloud is sharply divided into high and low-density segments in a 'striped' pattern. These stripes of high-density data are present in North Whalley and Cindrich datasets as well, but are wider in Bridgeview, being roughly equal in width to the lower-density stripes. Central Whalley's point cloud is the most variable of the four in terms of point cloud density; in addition to the north-south banding found in the other three study areas, there is a noticeable difference in density between the northern and southern portions of the neighbourhood. This is the product of an east-west overflight of the southern part of the neighbourhood incidental to the mainly north-south flight pattern of the scanning aircraft. As a result, the southern portion of Central Whalley is covered by variously, two or three overlapping LiDAR scans, as opposed to one or two overlapping scans for the remaining dataset.

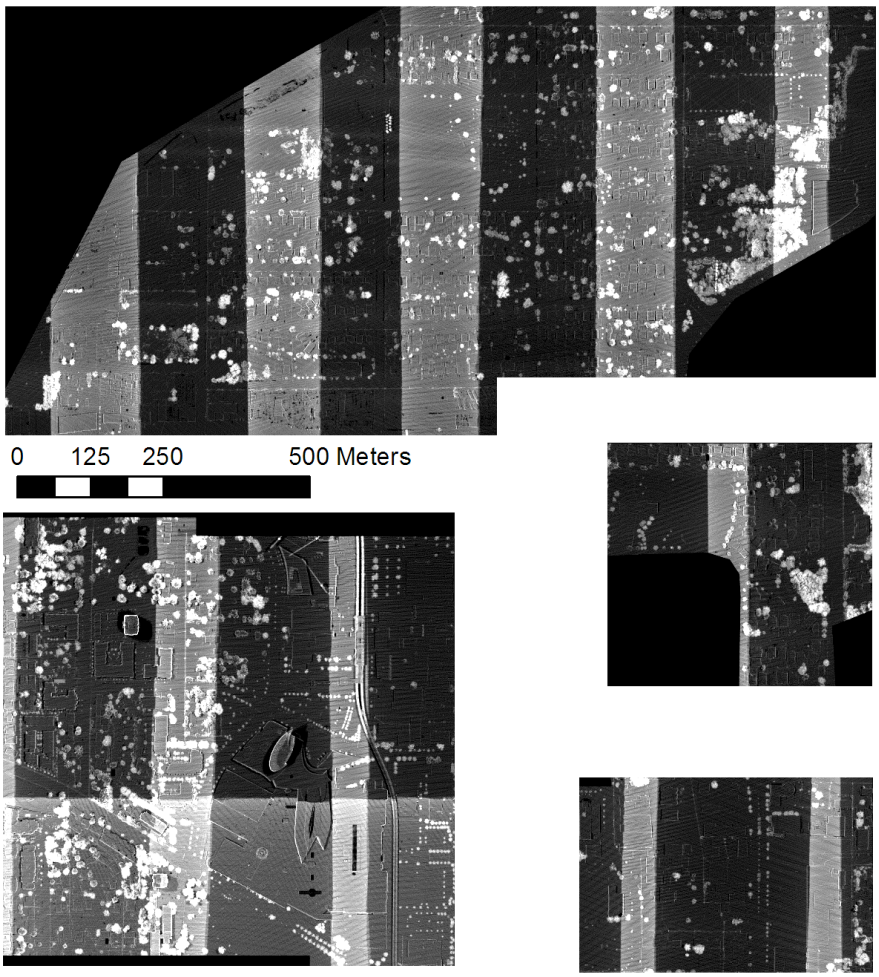


Figure 3-4: Point cloud density for the full-density subsets of all four study areas, higher point cloud densities indicated with lighter colours. Clockwise from top: Bridgeview, Cindrich, North Whalley, Central Whalley. Completely black areas indicate areas outside of the study area, data for which has been removed to accelerate processing.

4 Methods

The study methodology consists of several stages. First, multiple point clouds of varying density were generated from the existing LiDAR dataset, from building models were then reconstructed using two independent methods. The generated model footprints are then compared to reference building footprint data. Select building models are also subject to accuracy assessment by comparison to reference building models. Finally, the results of these comparisons were analyzed to determine what relationships exist between detection and reconstruction accuracies with point cloud density, building size, and building type.

4.1 Point Cloud Subsampling

Point cloud data for each preselected region was collected into a single LAS format file, which lists each point's location and properties. This is then subsampled using las2las.exe, a free (LGPL licence) component of Rapidlasso GmbH's LAStools software package to produce a less dense derivative point cloud. We refer to each of these thinned point clouds as a 'subset', as they are composed of a subset of the points making up the original point cloud. Subsampling was conducted on an every-nth-point basis, e.g. every second point sampled, every third point, et cetera, as in Pirotti and Tarolli (2010). Subsampling every nth point, versus a grid-based resampling method, has the advantage of preserving the spatial variability in point density of the point cloud, such that areas of relatively high density in the original dataset will also be of high density compared to the dataset as a whole in the subsampled datasets. Subsampling using a grid, in which one point per grid unit is sampled, produces point clouds of unrealistically uniform density, eliminating the effects of flight line overlap and vegetation. As implemented in las2las.exe, a counter starts at one and increments every time the program reads a point in the source LAS file; if the counter equals the sampling ration *n*, that point is retained and the counter resets to one for the next point.

The overall point density d_s of the resulting point cloud is given by Equation (4)

$$d_s = \frac{\frac{(p - p \bmod n)}{n}}{a} \quad (4)$$

where p represents the number of points in the original dataset, a represents the size of the area of interest of the point cloud in units of area, and n represents the subsampling factor, ex. 2 when sampling second point, 3 when sampling every third. In practical cases, $p \gg n$ and a is the same as the original for all subsets, meaning the overall point cloud density d_s for a subset of a cloud with a density of d_o can be closely approximated by the much simpler Equation (5).

$$d_s \approx \frac{d_o}{n} \quad (5)$$

The every-nth-point method has one major disadvantage for the purposes of this study: gradations in the subsampling density are coarse across low values of n, since n is always an integer. For example, the difference in mean point density between the highest point density dataset (the original) and the second highest (produced by sampling every second point) is twice that of the difference between the second and third highest density set. Table 4-1 shows mean point cloud densities for each study area and overall, while Figure 4-1 shows the same area of the point cloud for several different subsets, illustrating the effect of thinning.

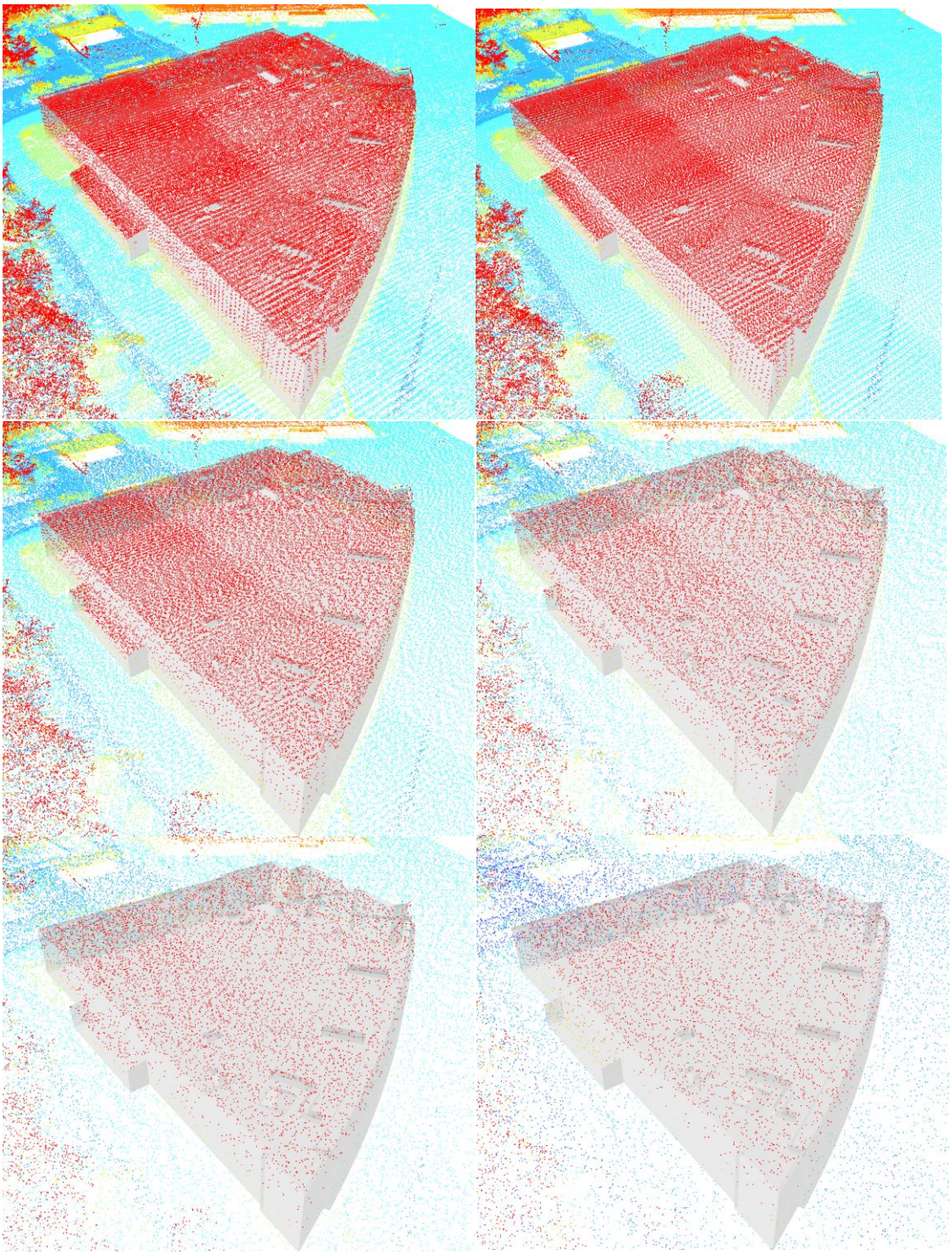


Figure 4-1: Point clouds at different subsampling levels are shown with a transparent reference model as background. In reading order: n=1, n=2, n=5, n=10, n=20, n=30.

Table 4-1: Point Cloud Density Per Subsample

Subsampling Factor *n*	Mean Point Cloud Density (p/m^2)				
	1 - Bridgeview	2 - North Whalley	3 - Central Whalley	4 - Cindrich	**Overall**
1	25.38	23.07	26.24	21.96	25.05
2	12.69	11.53	13.12	10.98	12.53
3	8.46	7.69	8.75	7.32	8.35
4	6.34	5.77	6.56	5.49	6.26
5	5.08	4.61	5.25	4.39	5.01
6	4.23	3.84	4.37	3.66	4.18
7	3.63	3.30	3.75	3.14	3.58
8	3.17	2.88	3.28	2.75	3.13
10	2.54	2.31	2.62	2.20	2.51
15	1.69	1.54	1.75	1.46	1.67
20	1.27	1.15	1.31	1.10	1.25
30	0.85	0.77	0.87	0.73	0.84

4.2 Reference Data Creation

Reference data representing 2D building footprints and 3D building shape were created for comparison to the results of building footprint extraction and 3d reconstruction, respectively. As reference models are more time consuming to create than footprints, the latter where created first, and then used to select a sample of buildings for which 3D models were created. Figures 4-2 through 2-5 show all reference data created for each study area.

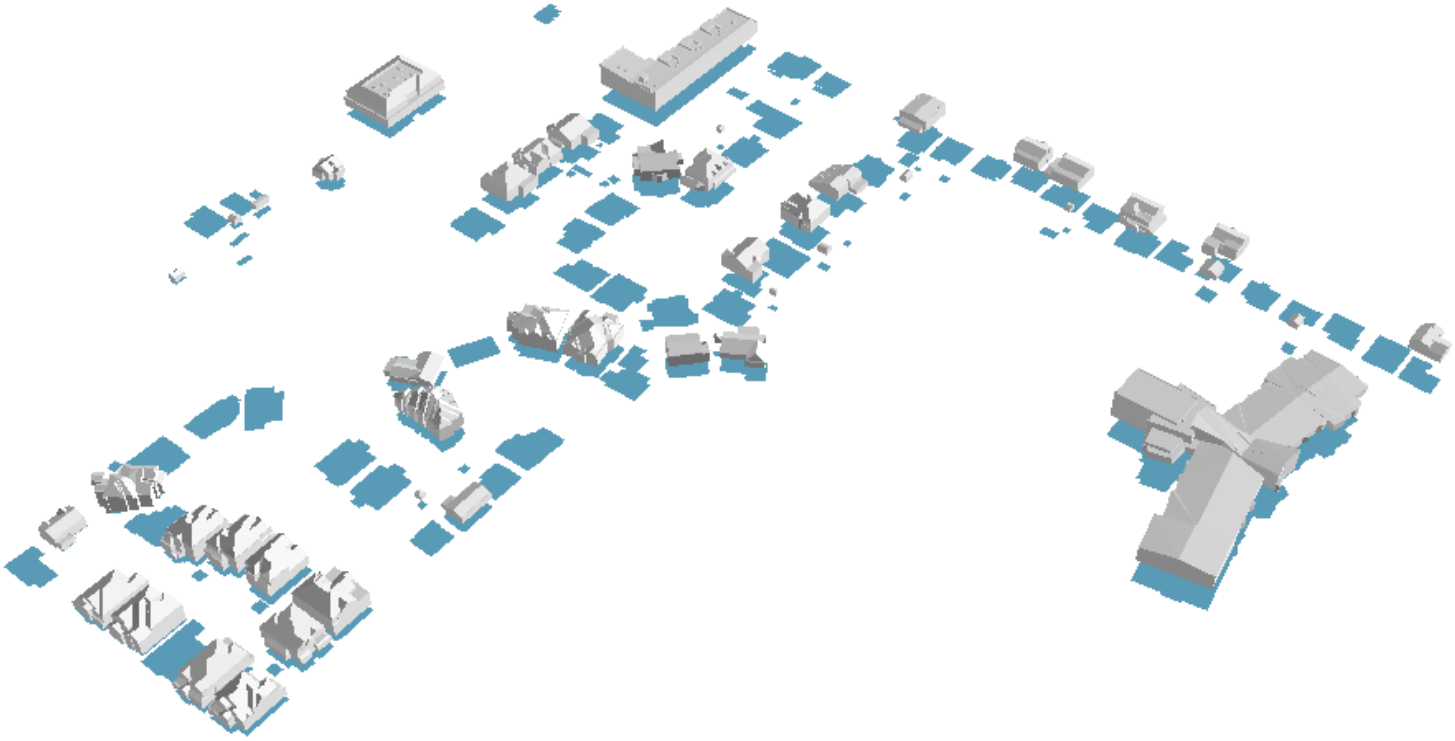


Figure 4-2: Reference data, with manually created models floating above reference footprints, is shown for the Cindrich study area.

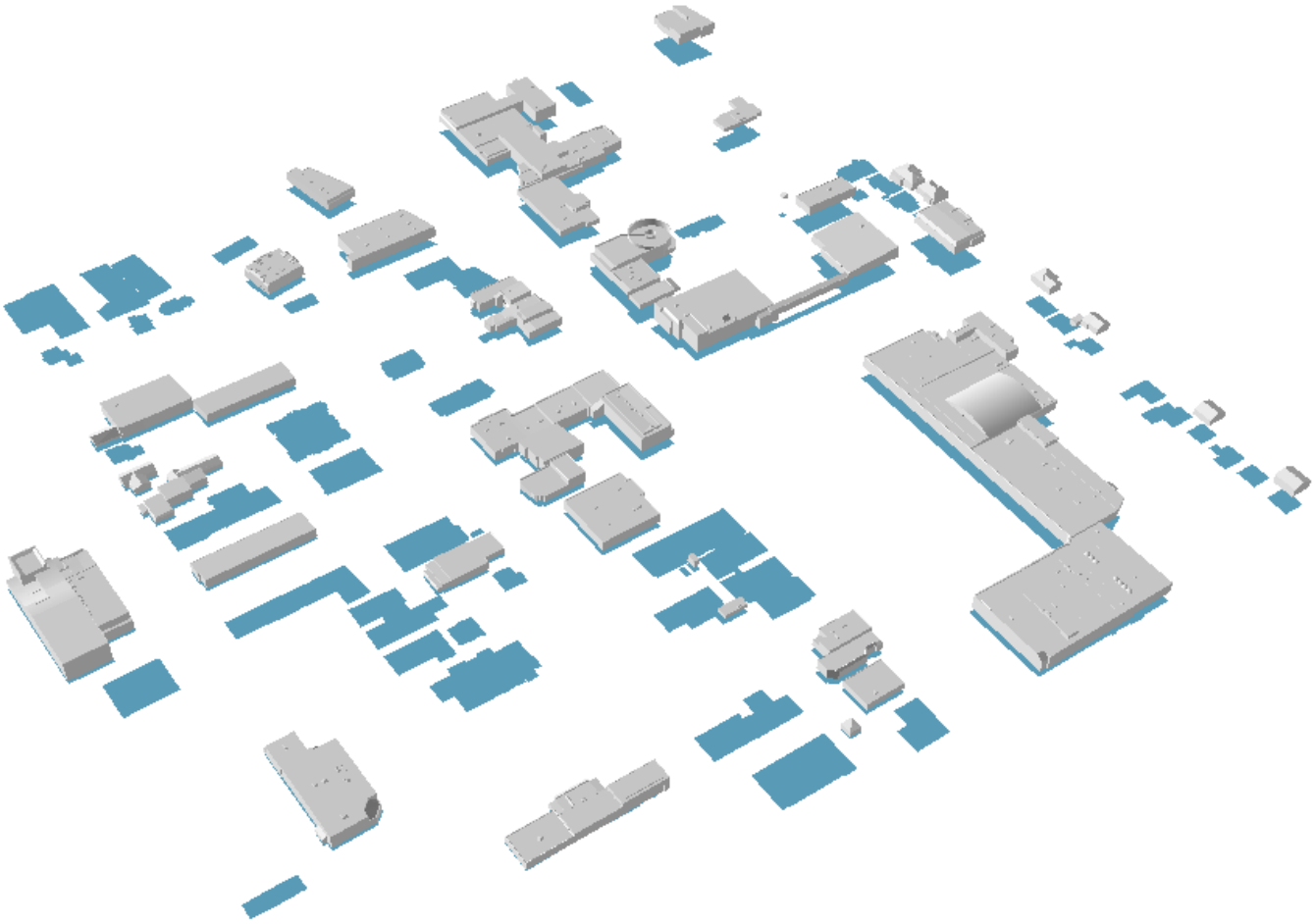


Figure 4-3: Reference data, with manually created models floating above reference footprints, is shown for the North Whalley study area.

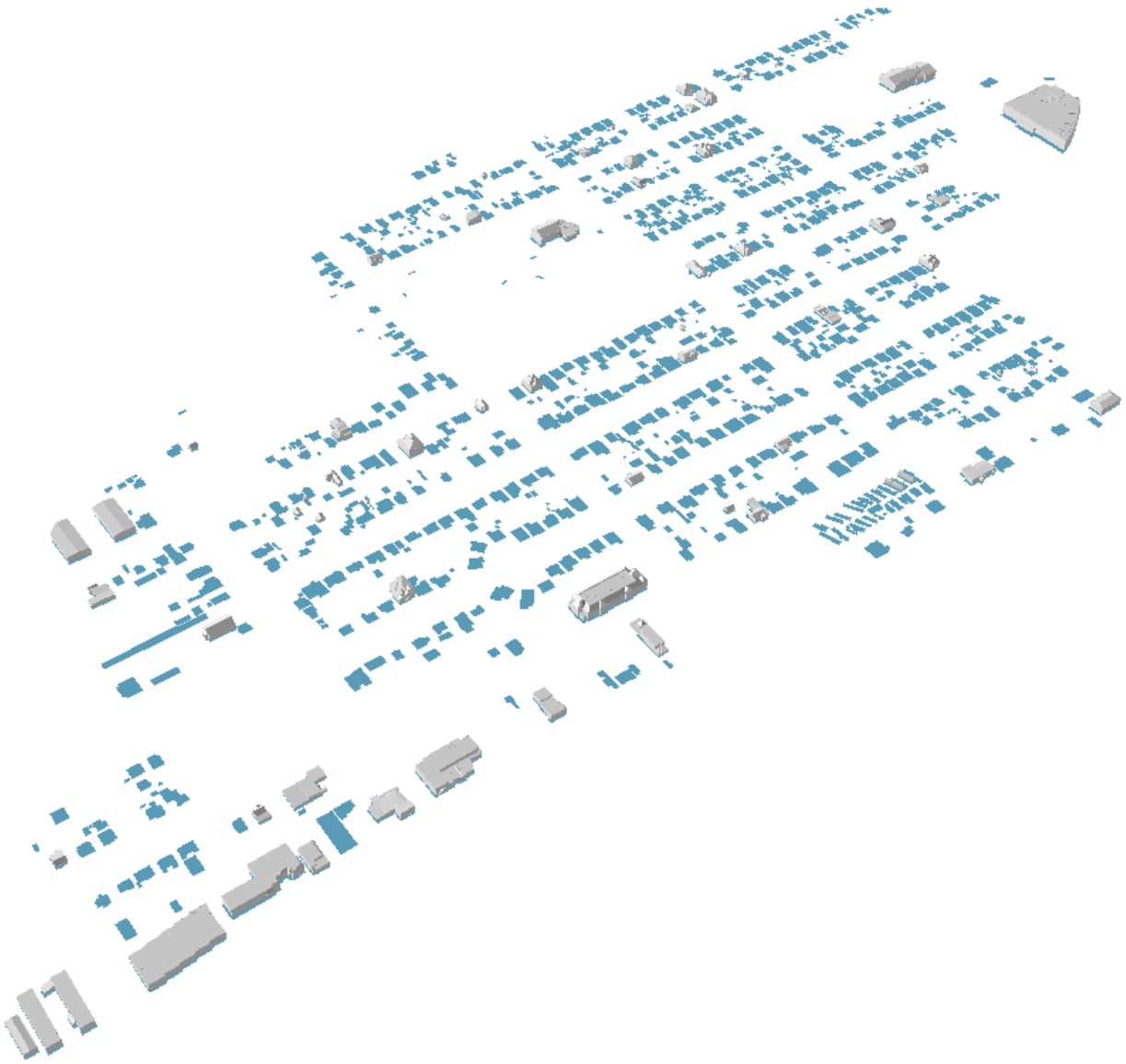


Figure 4-4: Reference data, with manually created models floating above reference footprints, is shown for the Bridgeview study area.

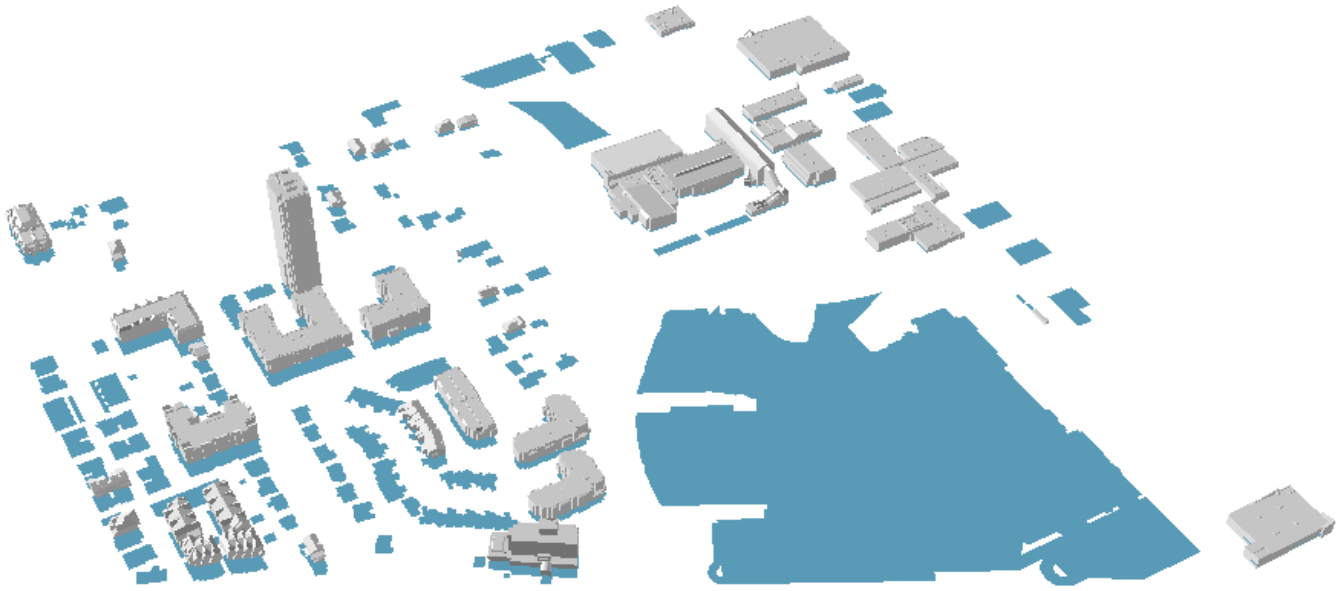


Figure 4-5: Reference data, with manually created models floating above reference footprints, is shown for the Central Whalley study area.

4.2.1 Reference Footprint Digitization

Building footprint data were created (digitized) manually for each study area based on that area's digital surface model, as derived from the full-density lidar data, with reference to orthoimage data to assist in differentiating between small buildings such as sheds and similar non-building objects such as vehicles and cargo containers. Footprint was defined as per the IBC standard, meaning it includes both the entire enclosed area of a building plus the space directly under any overhanging roof elements (International

Code Council 2011). The reference footprint data therefore represent the maximum horizontal extent of the building at any level, rather than a strictly ground-level footprint or floorplan. The reference footprint data as produced by the above mention differs from the City of Surrey building footprint data in some key respects. In the reference data, building sections are considered part of the same building if they are adjacent, with no

non-building space (i.e. space with no building object at any point on vertical axis) separating them. This is because the distinction between separate but adjacent buildings, though potentially in a legal or engineering context, is indiscernible using the building identification methods chosen. Distinguishing between contiguous building areas is considered difficult and it is therefore generally advised to collect reference footprints as unified blocks and separate them if necessary (Potůčková and Hofman 2016). By virtue of its source, the reference building data is also truer to the LiDAR data than the city building data. Buildings present in the LiDAR data may be absent in the City footprints due to being derived from data collected at different times or created using a different set of criteria for what qualifies a building. Those buildings that are present are frequently spatially out-of-alignment with the lidar data, presumably because the City footprints were derived by digitization from aerial imagery and therefore subject to the same geometric error as the source imagery. Figure 4-3 shows illustrative examples of both issues. Comparison with the manually-created reference data shows that the municipal data has a completeness rate of 89.8%, a correctness rate of 92.7% and a Q rate of 83.8%.

4.2.2 Reference Building Model Creation

Reference building models were created for some, but not all, buildings in each study area, based on digitized reference footprints. A two-part sampling strategy was used: first, the n_l largest buildings by footprint area were selected for analysis. Next, n_r random buildings were selected from the remaining un-sampled buildings. Both sets exclude buildings with shapes of genus >0, as these cannot be analyzed using SPHARM. The reference models were created manually using Trimble SketchUp modeling software, based on imported full-density point cloud data for each building and with reference to orthoimagery. In the

case of especially large buildings importing full-resolution data into the modeling software resulted in performance issues. In these cases, full resolution data was subsampled and a rough model created, after which details could be modeled by importing full-resolution data in smaller regions. Building reference models were created using the LOD 2.2 level-of-detail standard, meaning they included both overall roof shape as well as smaller roof substructures such as dormers (Biljecki, Ledoux, and Stoter

2016). Small substructures such as chimneys and rooftop air conditioning units were included in the model if their horizontal footprint was of a certain size, specifically if their footprint's bounding rectangle had a diagonal of two or more meters. This contrasts with the footprint area specification used in Biljecki, Ledoux and Stoter; the diagonal length was used in this study because of the difficulty in tracing out and measuring exact footprint geometry for roof substructures in the discontinuous LiDAR data. Overhangs were not modeled; like the generated building models, the reference models were 2.5D, with roof edges ending in vertical walls.

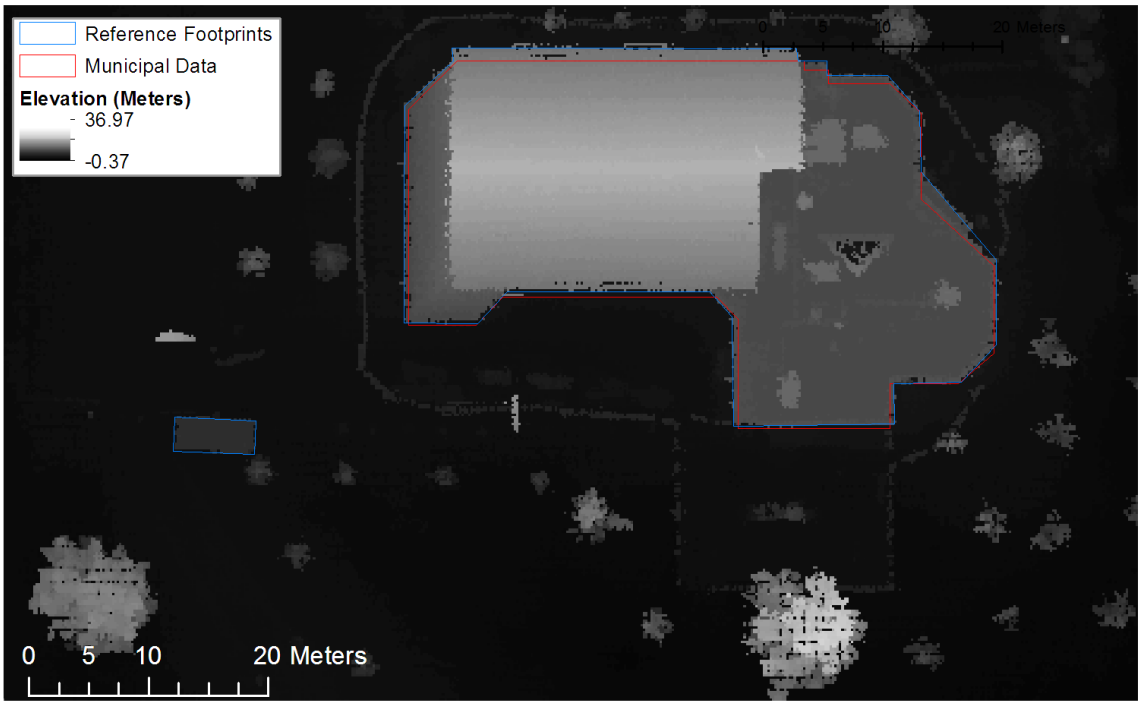


Figure 4-6 The manually-created reference footprint data and municipally-sourced building footprint data, overlaid on LiDAR-derived DSM elevation data. Note both the displaced outline of the large building and the absence of the smaller one in the municipal data.

4.3 2.5D Dual Contouring Pipeline

The first, and most complex modeling method used in this study is the 2.5D Dual Contouring (DC) method developed by Qian-yi Zhou as part of a PhD dissertation (2012). This method is a complete pipeline, shown in outline in Figure 4-4, which

performs all steps necessary to generate 3D building models from an unclassified point cloud. Zhou's implementation, used for this study, is open-source (under the MIT licence) and freely available as of writing on GitHub (Zhou 2017).

Figure 4-7: A simplified diagram of the 2.5D dual contouring pipeline.

As implemented, the DC method consists of several stages, the key stages being classification, segmentation and 2.5D dual contouring itself. Classification is performed by applying a Support Vector Machine (SVM) classifier to a set of five features, each a metric representing the spatial distribution of a given point and its neighbours (Zhou 2012). Points are discarded as noise if they do not have a required number of neighbours n_r within a given distance; otherwise, they are classified by the SVM into vegetation and non-vegetation classes. The classification process takes advantage of the difference in spatial arrangement between parts of the point cloud that represent continuous opaque surfaces and those that represent semi-transparent ones such as bushes and tree canopy. The value of n_r has a strong influence on the quality of overall classification since, if too large for a given point cloud density, large numbers of points will be erroneously classified as noise (Zhou 2012).

Once the points have been classified as vegetation, noise or non-vegetation, non-vegetation points are split into segments in the segmentation stage. Points within a certain distance d_n are assigned to the same segment based on a region-growing algorithm, with the largest segment then identified as ground and the remainder as buildings (Zhou 2012). The process takes advantage of the fact that LiDAR data rarely captures points on building sides, meaning building roofs and ground tend to be assigned to separate patches. d_n has a crucial effect on whether a patch will be correctly segmented; too low and points on the same surface will be assigned to different segments, too high and roof and ground points will be assigned to the same segment. d_n must be determined empirically for each subsampled dataset; lower data densities should have larger values, in keeping with the increased sparsity of points. Once split, point patches are assigned as

either ground or building via a region-growing algorithm. Patches are classified by number of points into large and small patches using the threshold t_{LP}; different (hard-coded) distance thresholds are used to classify these as building or non-building patches based on their distance to already detected ground patches.

2.5D contouring itself, as detailed in Section 2.2 is the final part of the core pipeline, in which roof geometry is generated based on point data and connected to the ground level by vertical polygons. An important parameter for dual contouring is the grid length g_l, which determines the lowest-level cell width of the quadtree. Model features can be no finer, spatially, than g_l, but setting g_l too low will result in holes in the model, rendering it both inaccurate and unusable for SPHARM parameterization. As with n_r and d_n, g_l must be determined empirically for each level of subsampling, as all optimal values for all three depend on average point cloud density; Table 4.2.3-1 shows their values. In our case, parameters were determined experimentally by repeatedly running DC modelling and footprint analysis for a subsection of the Bridgeview neighbourhood, targeting a footprint correctness rate of 95% and as high a completeness rate as could be obtained. The final set of parameters achieved a completeness ratio of between 80% and 85% for n 1 through 4, 70-80% for n 5 through 10, and 30-50% for n 15 through 40.

Once generated, building models must be processed using a simple hole-filling algorithm (“3D Printing Toolbox” 2013) in the 3D modeling software Blender (Blender Foundation 2018) to produce watertight final models, as by default they contain no bottom face and therefore have no enclosing volume. Models are then georeferenced, with those models representing buildings selected for 3D analysis are labeled for comparison and re-exported. After export, they are translated back into the regional local coordinate system, spatially aligning them with the reference data and rendering them suitable for comparison.

Table 4-2: Dual Contouring Parameters

Subsampling Factor *n*	Classification Neighbourhood Requirement n_r	Plane Splitting Neighborhood Distance d_n (m)	Large Patch Threshold t_{LP} (points)	Dual Contouring Grid Length l_g (m)
1	10	0.56	800	0.75
2	10	0.73	400	1.50
3	10	0.81	267	1.50
4	10	0.97	200	2.00
5	10	1.17	160	2.00
6	10	1.35	133	2.25
7	8	1.40	114	2.5
8	8	1.45	100	2.5
10	5	1.50	80	2.5
15	3	2.15	53	3
20	3	2.60	40	3
30	2	3.90	27	3.5

4.4 ENVI+TIN Modelling

The second modeling method used in this study is based on TIN (Triangulated Irregular Network) modelling from rasterized, classified LiDAR data. Harris Geospatial's 'ENVI LiDAR' (2015) software package is used to classify subsampled point clouds (see Figure 4-8), identifying which points represent buildings and extracting building footprint data as well as a Digital Terrain Model, all using proprietary algorithms. ENVI's classification parameters do not require adjustment for point cloud density, although a minimum building area is required: this was set to 5 square meters in all cases. Using ArcGIS, the resulting LiDAR point cloud is filtered to exclude non-building points, and a raster image representing building surface elevation is generated. This raster image is then filtered to smooth the elevational noise inherent in the LiDAR data, and converted to a TIN, a 2.5D surface composed of triangles. A second TIN is then generated from the DEM raster, and building models are extruded between the two TINs in areas where ENVI detected building footprints; Figure 4-8 shows key steps in this process, which is represented graphically in Figure 4-10.

The resulting models are then exported and georeferenced using the same translation method used for the DC models.

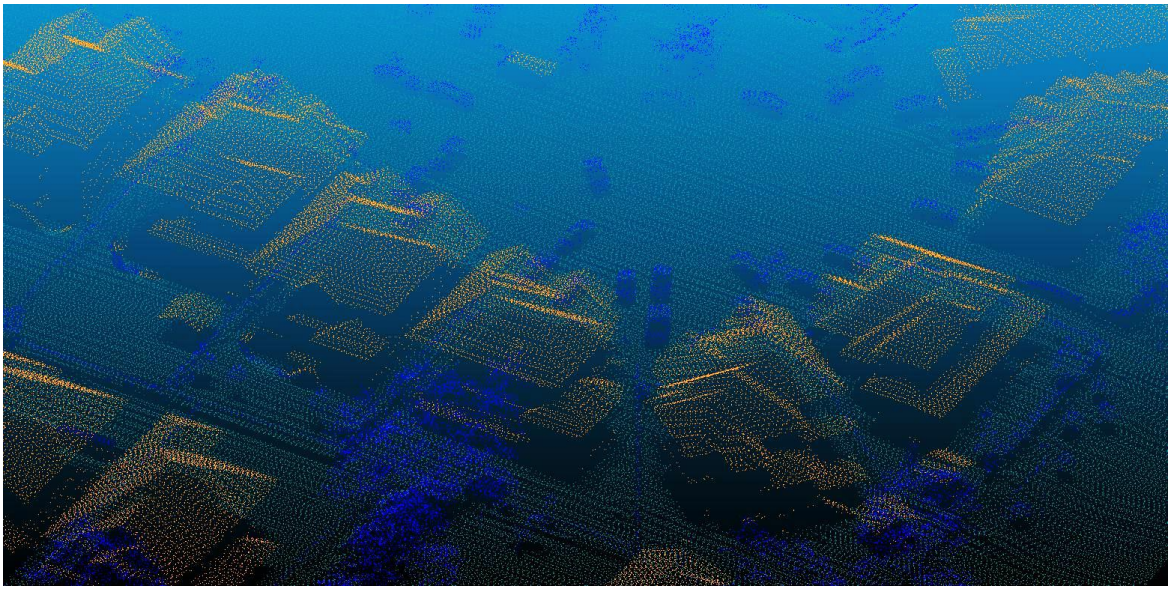


Figure 4-8: Part of a LiDAR point cloud classified into building, non-building and ground points using ENVI.

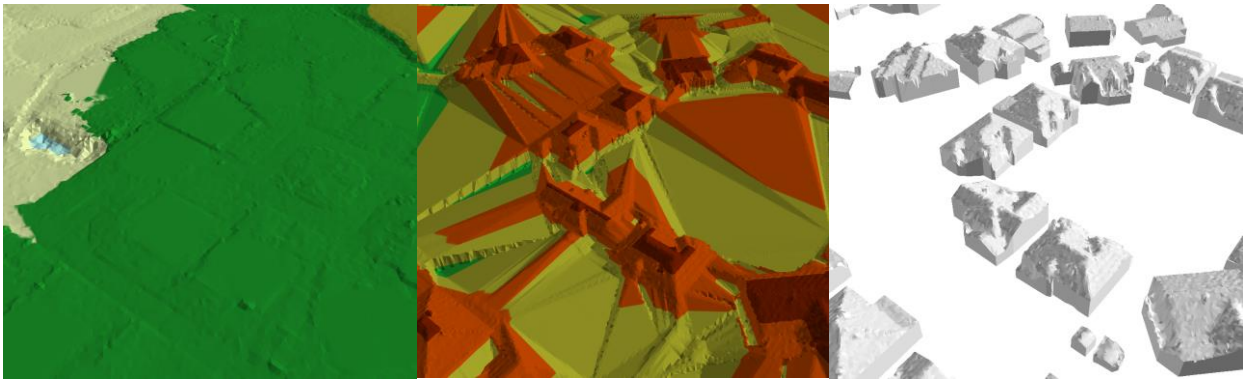


Figure 4-9: Key steps of the ENVI+TIN modeling process are shown. From left to right: the ground TIN, the roof TIN, and the building models generated by extruding between the two, all for the full resolution dataset of the Cindrich study area.

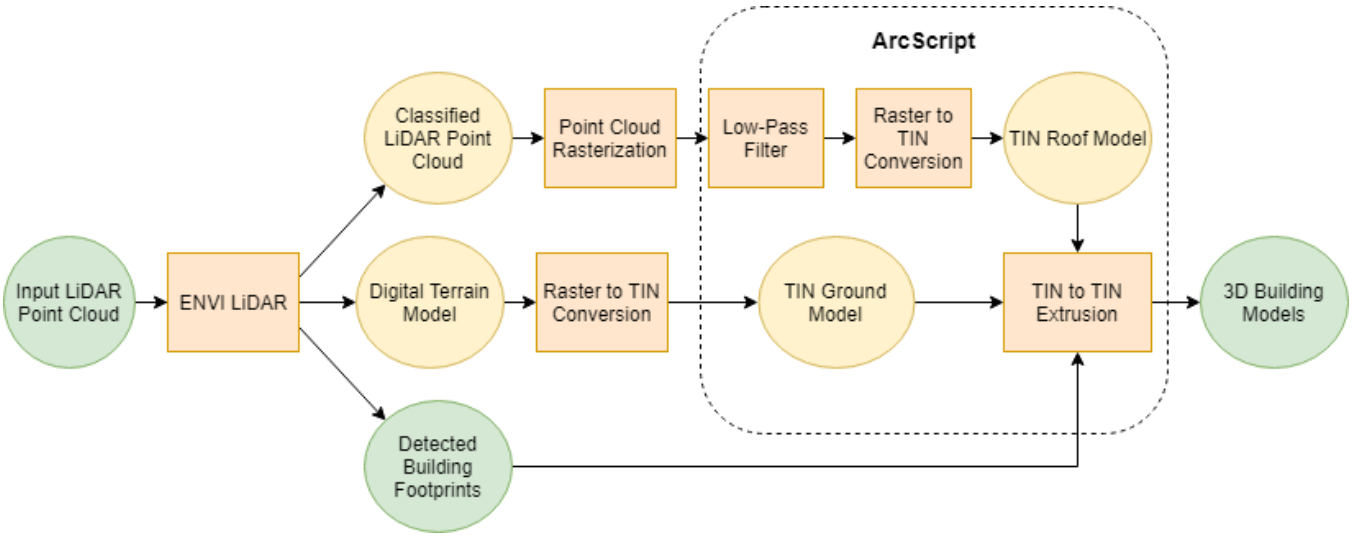


Figure 4-10: A diagrammatic representation of the ENVI+TIN building modelling process. Note that the detected building footprints are re-used for footprint analysis, since due to the nature of the TIN to TIN extrusion process they exactly represent the 2D extents of the models produced.

4.5 Footprint Generation

Building footprints are generated by ENVI Lidar (Harris Geospatial, 2015) as an intermediate step in the ENVI+TIN model generation process. Since these footprints are identical in 2D extent to the models generated using that method, they may be used to analyze models' 2D accuracy without further processing. The Dual Contouring method, in contrast, does not produce footprint data until after model creation and post-processing; output models must be analyzed using ArcGIS to extract 2D representations of their footprints, which is also a necessary step in registering generated models with their corresponding reference building models.

4.6 Footprint Accuracy Analysis

Due to the large number of iterations and individual buildings involved in the study, an object-by-object analysis of footprint extraction would be too time consuming to be practical, since manual intervention would be required to ensure a one-to-one relationship

between reference and detected building polygons. Instead, extracted footprints were analyzed on the basis of overall accuracy and by per-building completeness. The former approach examines the rate of false positives and false negatives in relation to true positives, while the latter measures how much of each building's 2D area was successfully recognized.

The area analysis looks at the study area as a whole, using a simple raster-based method to produce a map of classification accuracy for buildings in each study area, for each subset of LiDAR data. The analysis is conducted using python scripting in ArcGIS, as part of a script that also handles the ENVI-based 3D modeling and model identification processes. Extracted and reference buildings are rasterized at a 0.1m by 0.1 m resolution, producing rasters that have a value of 0 where there is no building and 1 where there is a building. A comparison raster is then created by multiplying the reference raster by a factor of two and adding its value to that of the raster representing detected buildings. The resulting raster has a value of 3 for pixels that represent true positives (TP), 2 for pixels representing false negatives (FN, buildings not detected where one exists), 1 representing false positives (FP, buildings detected where none exists) and 0 for true negatives (TN). Overall counts may be corrected to remove 'true negative' pixels that exist outside of the study area but are included in processing due to ArcGIS's raster analysis implementation. The main metric used to judge overall accuracy was q, as given by Equation (3), but Completeness and Correctness, given in Equations (1) and (2), were also calculated.

The above analysis produces an overall picture of reconstruction accuracy, with a single *q* score representing accuracy over the entire study area. This is insufficient for establishing a relationship between point cloud density and extraction accuracy because, as noted in section 3.2, point cloud density varies widely within the study areas due to flight line overlap.

Another method of measuring extraction accuracy is completeness, which measures how much of a reference building or buildings are identified by the extraction process. This does not give any indication of the rate of false positives (correctness) but does not

require intervention to match merged or split buildings to the appropriate building in the reference data. The advantage of using completeness compared to overall accuracy assessment is that the completeness of each reference building can be calculated independently, allowing for analysis of completeness rates by building characteristics such as overall area. This is important since detection rates are expected to be different for small buildings than for large ones. It is also helpful to give some indication of the 2D accuracy of buildings subject to 3D accuracy assessment, allowing for a measurement of how footprint completeness relates to model completeness. As with the overall analysis, local point cloud density is important to measure when analyzing the completeness of individual footprints. It can be assessed by extracting per-footprint point counts, then dividing by the area of the footprint. Correctness, the proportion of an extracted building footprint that is also classified as building area in the reference data, can also be analyzed in a similar fashion.

Footprint accuracy analysis, along with footprint extraction and the latter part of ENVI+TIN modelling, is performed using a python script in ArcGIS. The script, referred to as ‘ArcScript’, also calculates the centroid of each 3D model so that it may be translated into a local coordinate space for comparison. Automation using ArcScript was an important part of our method, as the many processing steps necessary to process and analyze building data would be extremely time-consuming and error-prone if conducted manually; its development represents a significant portion of the work conducted in this study. An overview of the ArcScript’s function is shown in Figure 4-11, while the script itself is presented in full in Appendix B.

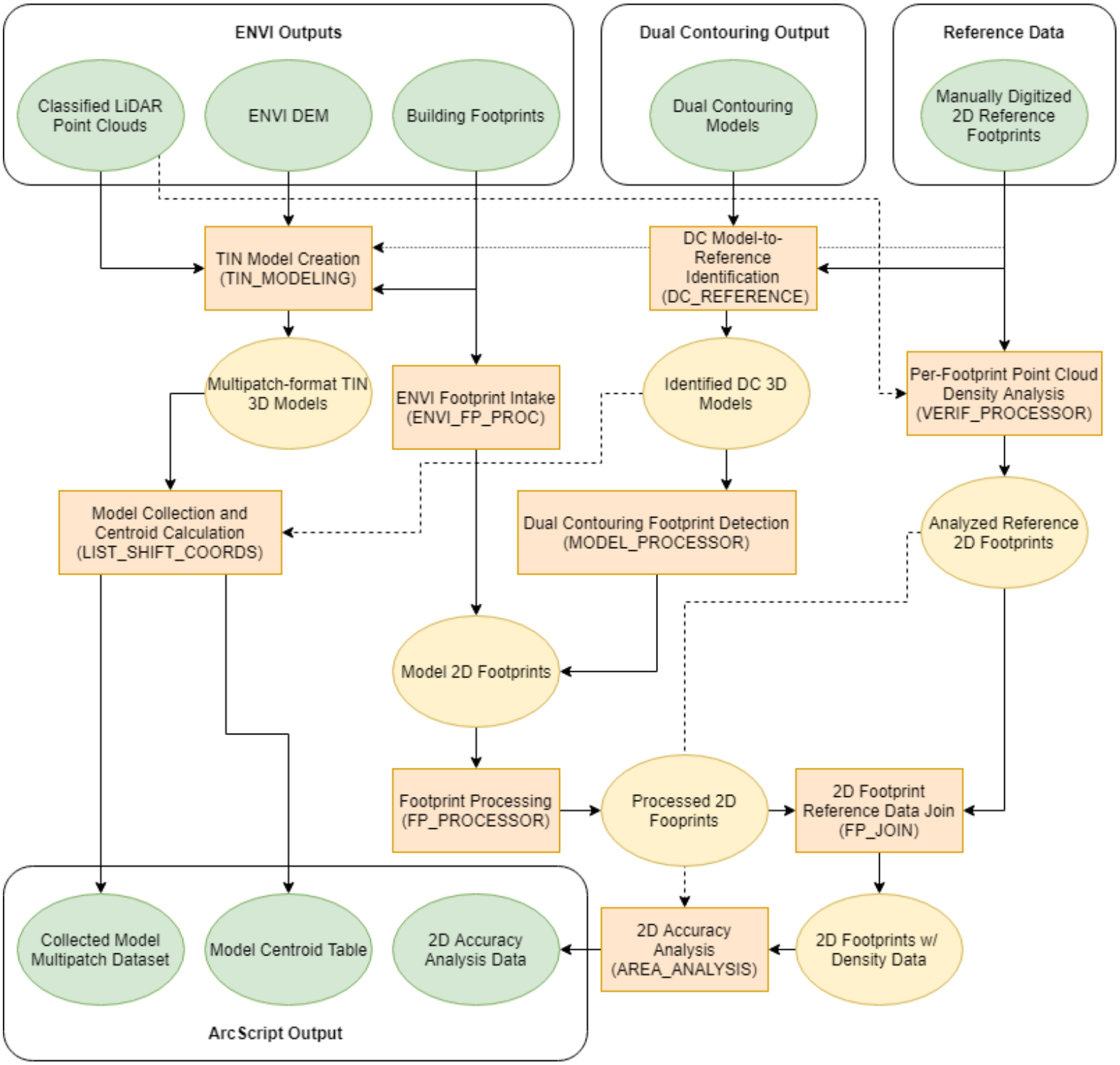


Figure 4-11: A diagrammatic representation of ArcScript, the python script used to automate the workflow for ENVI+TIN model creation, 2D accuracy assessment, and model position calculation.

4.7 3D Model Accuracy Analysis

The 3D model analysis procedure used in this study is that of Zeng, Wang and Lehrbass (2014) and analyzes the similarity of a generated model to its corresponding reference in terms of volume, point and surface similarities.

Volume accuracy is assessed via a Monte-Carlo algorithm that first places 2000 points at random locations inside an axis-aligned bounding cuboid fitted to both reference and sample models. The algorithm then checks each point to determine if it is inside either, neither, or both input shapes. Completeness, Correctness and Quality metrics may then be calculated by counting points inside both shapes as true positives, points inside only the reference as a false negative, points inside only the subject as false positives, and points outside both as true negatives. All of these metrics are proportional to area; a model of a shed and a model of a shopping mall with the same completeness score will have correctly identified the same *ratio* of their respective total volumes, even though the volumes identified will be vastly different in absolute terms. One notable requirement of the volume comparison process which also applies to the point comparison process as well is that subject and reference model must be in identical coordinate systems.

Point accuracy is assessed by determining the Euclidean distance between matched corner points in the reference and subject models. Corners are discovered by searching in each model for vertices between three near-perpendicular polygon faces. Matches between eligible corner points are made based on shortest distance. The final per-building metrics produced the scalar average distance between matched points and a vector representing the mean shift in each spatial dimension.

Surface accuracy is measured by comparing the parameterized weighted Spherical Harmonic (SPHARM) representation of subject and reference models. First, a voxel representation of each model is generated, in this case at a resolution of 50 by 50 voxels. This representation is then subject to weighted SPHARM transformation, producing a set of spherical harmonics by mapping the surface of the model onto a sphere, in our case using an isotropic heat diffusion model (Chung Dalton and Davidson 2008). This mapping is then decomposed into a set of basic functions referred to as spherical

harmonics. The weighting, as governed by parameter σ, has the effect of smoothing the SPHARM isosurface at each stage, reducing both the processing time required and the ringing artifacts known as the Gibbs phenomenon (Chung et al. 2007). Once a SPHARM representation has been computed, its Fourier coefficient matrices, of which there are one for each spatial dimension, may be vectorized. The root mean square distance (RMSD) between the set of SPHARM coefficients for each building may then be calculated by Equation (6), as given in Zeng (2014),

$$\text{RMSD} = \sqrt{4\pi \sum_{l=0}^{\inf} \sum_{m=-l}^{l} \left\| c_{1,l}^{m} - c_{2,l}^{m} \right\|^2} \quad (6)$$

where *l* is the degree of the SPHARM representation and *m* is the order. A percent score may be produced by dividing the SPHARM RMSD by a reference RMSD, such as one produced by comparing a unit sphere and unit cube (Zeng 2014).

Table 4-3 Sample SPHARM Coefficients

		m						
		-3	-2	-1	0	1	2	3
l	0	0	0	0	95.61397	0	0	0
	1	0	0	3.018421	-16.6051	-0.18695	0	0
	2	0	1.854371	2.156973	0.070593	0.46308	-0.13101	0
	3	0.470993	0.588065	-1.55056	3.52027	0.01401	-0.0043	0.209342

SPHARM representations are calculated to a degree *l*, which governs the number of basic functions into which the spherical mapping is decomposed. Each degree *l* has (non-zero) coefficients for order *m=-l* to *m=l*, for a total of *l*2+1* coefficients, for each spatial dimension (Chung, Dalton and Davidson, 2008). For the SPHARM representations of degree 80 used in this study, this means there were a total of 19,683 coefficients for each model analyzed. Table 4-3 shows all non-zero SPHARM coefficients for l=0 to l=3 for the x dimension of an example building in the Bridgeview study area, as an illustration of SPHARM representation. Note that this represents only a small fraction of the SPHARM coefficients that make up the representations used in this study, which were of degree 80. Note also the trend of decreasing absolute value as l increases.

5 Results

2D and 3D reconstruction accuracy are closely linked; not only does the accuracy of extraction affect what parts of a building's planform area are modeled, the completeness of building detection also determines whether an existing building is modeled at all. The correctness of the 2D footprint is also important: areas of a detected footprint that do not actually represent building area will produce defects in the resulting model. Figure 5-1 shows an example building as reconstructed by both methods, plus the reference 3D model for comparison. Figure 5-2 shows a representative reconstruction via the DC method, from full sensity data, while Figure 5-3 shows an overview of the Cindrich neighbourhoods with ENVI generated footprints and ENVI+TIN models, both also from full density data. In general, models produced by the Dual Contouring method had a 'cleaner' asthetic as compared to those produced by the ENVI+TIN method, which suffered from noisy roof surfaces due to being derived from rasterized LiDAR data. The noise issue is not purely aesthetic; ENVI+TIN models contained large numbers of polygons and thus consumed more space to store and took more time to load and process. The ENVI+TIN building models also suffer from artefacts around roof edges, caused by interpolation between non-adjacent building areas when the roof surface rasters were produced. This results in a 'chipped' appearance when a roof edge's nearest external

neighbouring building surface is a lower-elevation roof surface, or a raised edge in cases where the nearest neighbour building surface is higher.

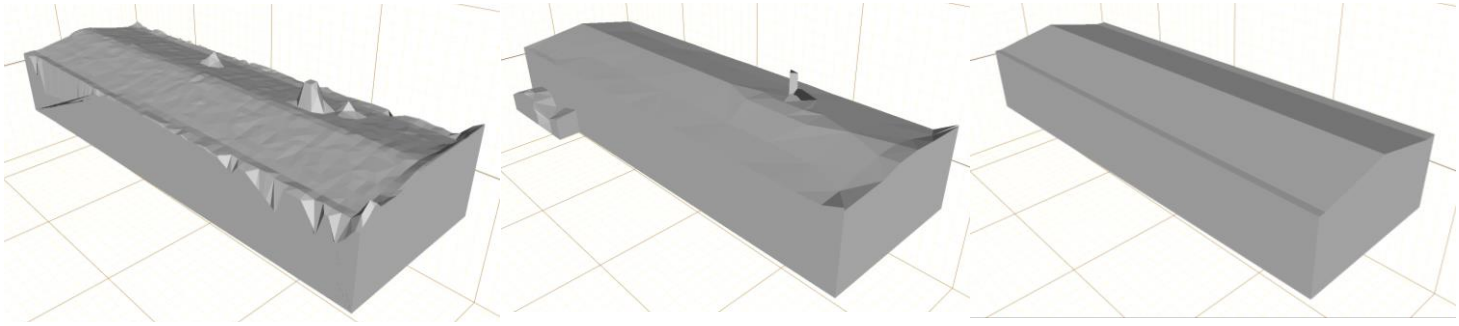


Figure 5-1: Examples of models of the same building produced by the ENVI+TIN (left) and Dual Contouring (centre) methods from original density (n=1) LiDAR data are shown with the reference model (right) of the same building.

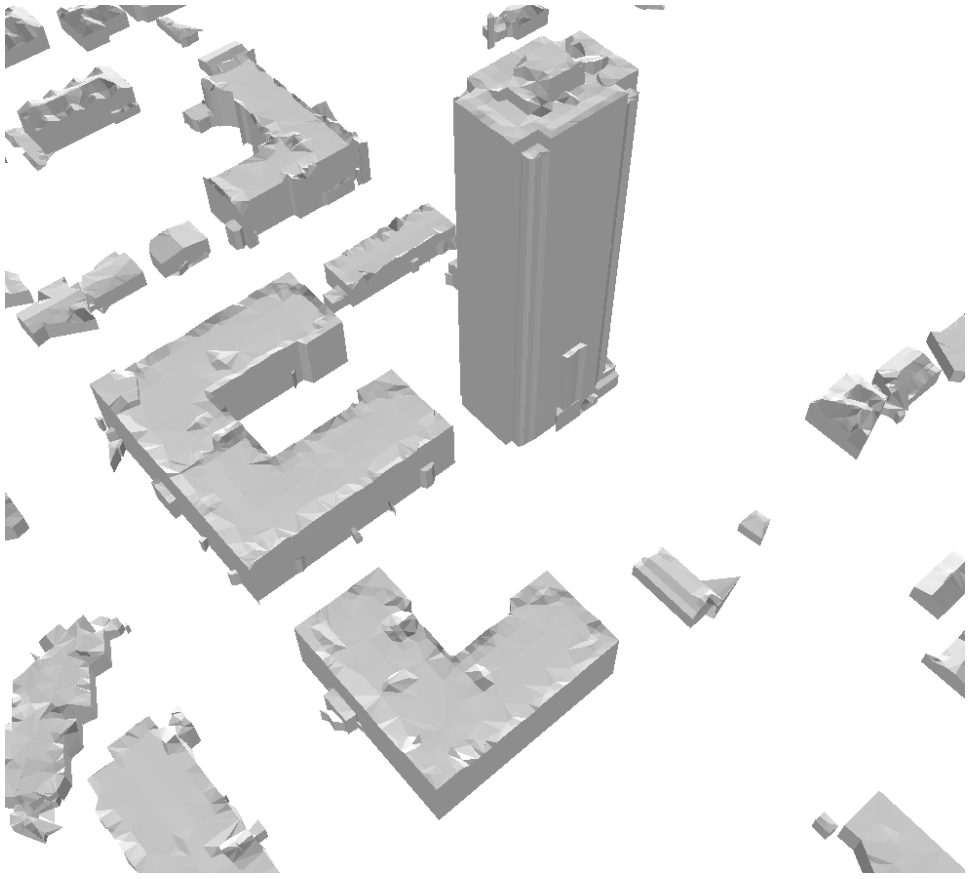


Figure 5-2: An area of the Central Walley area reconstructed from full density data using the Dual Contouring method.

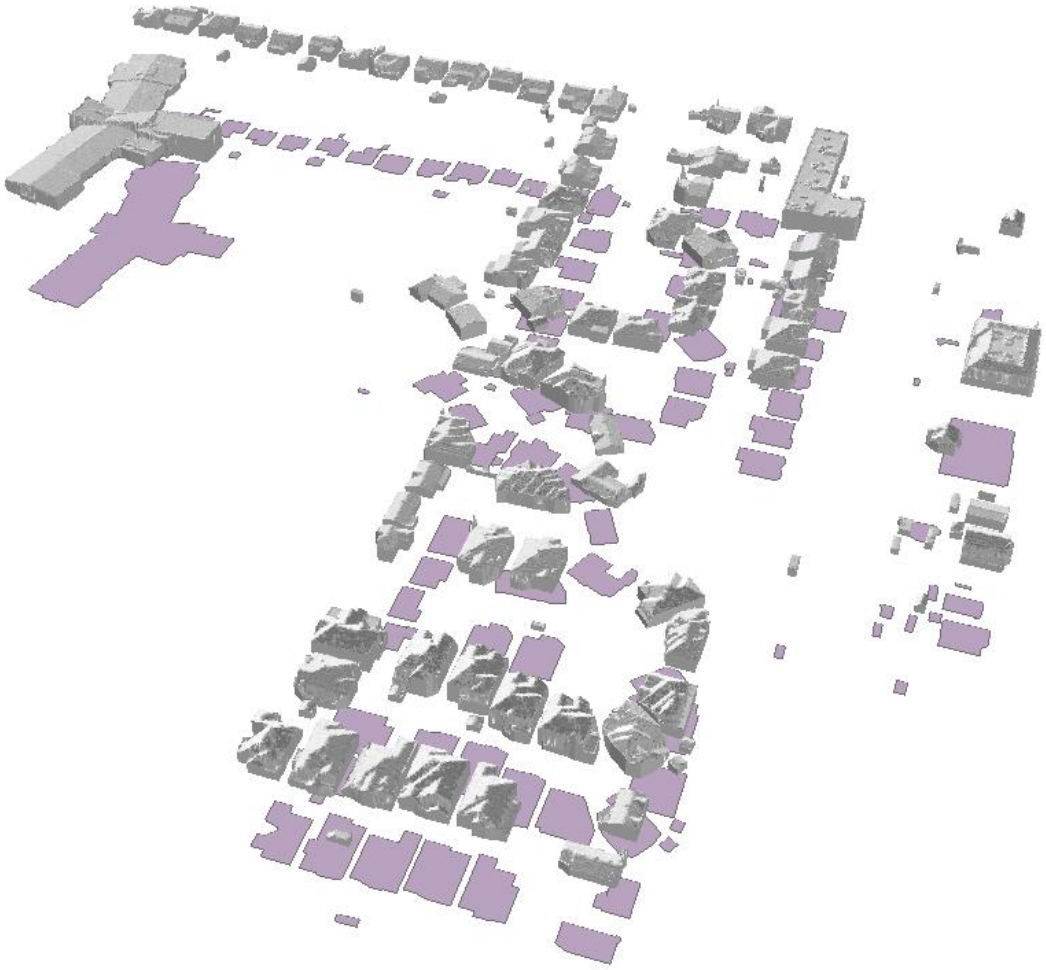


Figure 5-3: The Cindrich neighbourhood, as reconstructed using the ENVI+TIN method from the full density dataset, is shown floating above building footprints detected by ENVI.

5.1 2D Accuracy

Overall performance for each subset for all four regions and both methods is summarized in Table 5-1. Metrics for Completeness, Correctness and Q are calculated as per the formulae in section 4.6 for the output of both methods from each subset of the LiDAR data. Overall mean point cloud density is also shown for reference. As predicted, there is a negative relationship between overall accuracy indicated by Q and the degree of subsampling; thinner point clouds produce less accurate classifications of building and non-building area. For both methods, there is a noticeable drop in accuracy from n=10 to n=15, corresponding to a decrease in mean overall point cloud density from 2.20 to 1.46 points per square meter. Apart from being generally more accurate, the ENVI+TIN method's classification, produced by ENVI Lidar, shows a more consistent level of accuracy from n=1 to n=8, but with a sharper decrease for thinner point clouds.

Table 5-1: Overall 2D Accuracy

Subsampling Factor *n*	**Mean Point Cloud Density (Pt. m^{-2})**	**DC**			**TIN**		
		Completeness	**Correctness**	**Q**	**Completeness**	**Correctness**	**Q**
1	22.97	67.09%	91.04%	62.94%	85.39%	94.94%	81.67%
2	10.99	69.54%	90.70%	64.91%	83.25%	95.60%	80.17%
3	7.32	69.92%	91.13%	65.46%	83.12%	96.07%	80.39%
4	5.49	67.51%	91.78%	63.66%	82.60%	95.99%	79.85%
5	4.39	63.05%	93.35%	60.34%	78.03%	95.80%	75.45%
6	3.66	57.74%	93.05%	55.35%	80.06%	95.95%	77.44%
7	3.14	58.97%	92.66%	56.34%	79.94%	96.28%	77.54%
8	2.75	54.61%	92.75%	52.38%	77.43%	96.55%	75.35%
10	2.20	59.78%	93.22%	57.29%	67.64%	96.98%	66.25%
15	1.46	41.14%	92.77%	39.86%	30.44%	97.07%	30.16%
20	1.10	36.30%	91.52%	35.12%	18.19%	98.14%	18.13%
30	0.73	20.96%	87.85%	20.37%	1.84%	98.14%	1.84%

Due to the spatially variable density of the Surrey point cloud, a per-subset accuracy assessment fails to give a complete picture of the relationship; local point cloud density often varies substantially from the average. One way of accounting for the variability of point cloud density in the data is to instead look at the relationship between the point cloud density within an extracted building's footprint area and the accuracy of that footprint. Due to the very large number of extracted building footprints, it is not possible to manually establish a one-to-one relationship between detected and reference building footprints; instead, the accuracy of extraction must be judged by two separate criteria: one measuring how much of a reference footprint is correctly identified as a building (completeness) and one measuring how much of an extracted building footprint represent actual building area (correctness).

With a combined total of 1232 buildings in the four study areas, and 12 data subsets per extraction method, there are 14,784 possible partial or full building footprint detections per method. Of these only a fraction (7,402 for DC, 9,477 for TIN) have any part of their area classified as a building in the output. Table 5-2 shows the number of buildings in the study area with non-zero completeness in the output for each method and subset, indicating at least partial detection. Also shown is the mean proportion complete and the standard deviation of completeness for all footprints, detected or not.

Table 5-2 Number of Buildings Extracted per Method and Subset

Method	Subset	Study Area				Total	Proportion Identified	Mean Completeness	St. Dev of Completeness
		Bridgeview	Cindrich	Central Whalley	North Whalley				
Reference	n/a	909	113	135	77	1234	n/a	n/a	n/a
DC	1	598	80	122	70	870	70.62%	60.02%	40.67%
	2	579	76	117	66	838	68.02%	58.61%	41.86%
	3	613	77	115	65	870	70.62%	59.77%	40.69%
	4	553	80	108	62	803	65.18%	54.70%	42.16%
	5	470	74	91	61	696	56.49%	47.01%	42.36%
	6	461	72	78	56	667	54.14%	43.27%	41.33%
	7	437	72	68	57	634	51.46%	40.69%	41.40%
	8	404	72	62	54	592	48.05%	37.40%	41.35%
	10	465	69	78	61	673	54.63%	42.71%	41.00%
	15	247	65	49	39	400	32.47%	24.85%	37.31%
	20	188	53	44	46	331	26.87%	18.18%	32.24%
	30	3	1	14	10	28	2.27%	6.45%	20.18%
TIN	1	823	99	129	76	1127	91.48%	84.26%	27.33%
	2	805	95	128	76	1104	89.61%	81.61%	29.38%
	3	767	87	122	72	1048	85.06%	76.83%	33.77%
	4	735	81	122	70	1008	81.82%	74.04%	36.25%
	5	727	77	119	73	996	80.84%	72.90%	37.01%
	6	696	76	117	69	958	77.76%	69.57%	39.02%
	7	672	75	115	71	933	75.73%	66.97%	39.40%
	8	634	74	115	69	892	72.40%	62.86%	40.32%
	10	528	67	112	64	771	62.58%	49.66%	41.06%
	15	288	12	65	29	394	31.98%	23.46%	36.53%
	20	153	9	39	17	218	17.69%	10.53%	24.24%
	30	3	1	14	10	28	2.27%	0.51%	4.11%

Figures 5-4 and 5-5 shows the mean completeness of extracted building footprints with point cloud densities relative to the reference of between one and ten, for a set of 9 building size classes. Each reference footprint is assigned a completeness value and point cloud density for each combination of subset and method. Footprint completeness is calculated for each reference footprint and each subset and represents the proportion of area of the reference footprint represented by a building in the extracted output. Point cloud density is calculated for the entire reference footprint area, from the subset point cloud from which a given output was extracted, by counting all points within the 2-dimensional area of the footprint and then dividing the total by the area in meters. The relationship between mean point cloud area and accuracy is more easily shown if footprints or models with similar mean point cloud density values are grouped together in a bin and their accuracy metrics represented as a mean. To this end, results are displayed by rounding individual footprint point cloud density values to the nearest integer, in effect creating 1 p/m^2 wide bins. We combine data on detection and non-detection across all 12 subsets; thus for the point cloud density-based analyses each building is represented in twelve different bins.

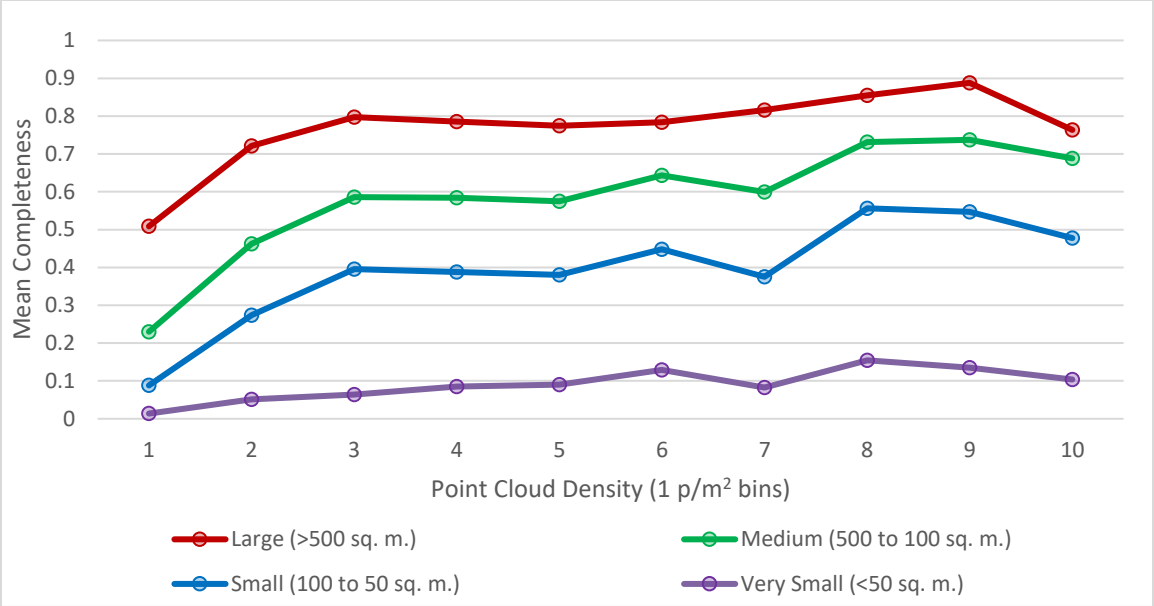


Figure 5-4: Mean completeness for buildings of different sizes is plotted, with buildings grouped based on point cloud density into 1 p/m^2 bins.

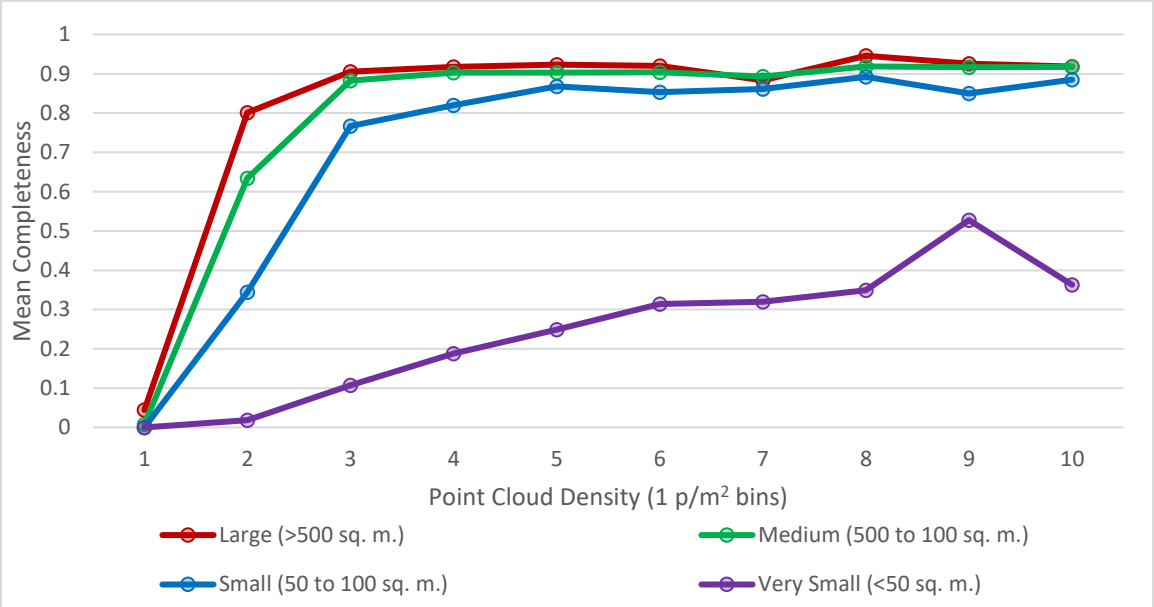


Figure 5-5: As in Figure 5-2, but for the ENVI+TIN method.

A similar relationship is found to that demonstrated in the overall accuracy assessment for most size classes: completeness remains fairly across high and mid-density levels before degrading sharply at lower (1-2 p/m^2) densities. For the very small size class however, overall accuracy is low regardless of local point cloud density for both

methods. For the other size classes, performance differs noticeably between the two methods. For the ENVI+TIN method, mean completeness is high at and above the 2.5-3.5 p/m^2 bin, with a sharp degradation in accuracy below that level. For the DC method, the three larger size classes also experience a sharp drop in performance from a stable level below the 2.5-3.5 p/m^2 bin, but the stable level of mean completeness varies; the Large size class shows the highest level of mean completeness, the small size class shows the lowest of the three. Very Small buildings have poor mean completeness at all point cloud density bins with substantial data; for the ENVI+TIN method, mean completeness peaks for 8.5-9.5 p/m^2 and degrades steadily below that level, but for DC mean completeness is well below 20% for all density bins.

Figure 5-6 shows mean footprint completeness for footprints grouped by mean point cloud density, rounded to the nearest 0.1, for footprints with very low point cloud density. The transition from relatively high to near zero mean completeness for the ENVI+TIN method is clearly visible, as is the Dual Contouring method's more gradual decline. Although the nature of the subsampling method means most building footprints are extracted from relatively low-density point clouds, enough data is available to reliably measure extraction performance at higher point cloud densities. Even with a 1 p/m^2 wide binning method, many bins above 10 p/m^2 have too few members to be a reliable indicator of extraction performance at their respective density levels; the multi-modal nature of distribution of point cloud density in the study areas means that certain bins will have many more member footprints than others. Figure 5-7 shows a plot of mean completeness of footprints in 1 p/m^2 bins, with low population bins ($n<10$) omitted, for both methods, both omitting and including buildings with a reference footprint area below 50 m^2. The performance characteristics of both methods can be readily intuited. For buildings greater than 50 m^2 in area, extraction by ENVI shows consistently high performance (~90% completeness) for all bins above 4 p/m^2, while the DC method shows more variable performance with mean completeness ranging from 60% to just above 80%, for point cloud densities greater than 5 p/m^2. As one would expect based on the size class analysis, overall performance is consistently worse than performance for only buildings larger than 50 m^2.

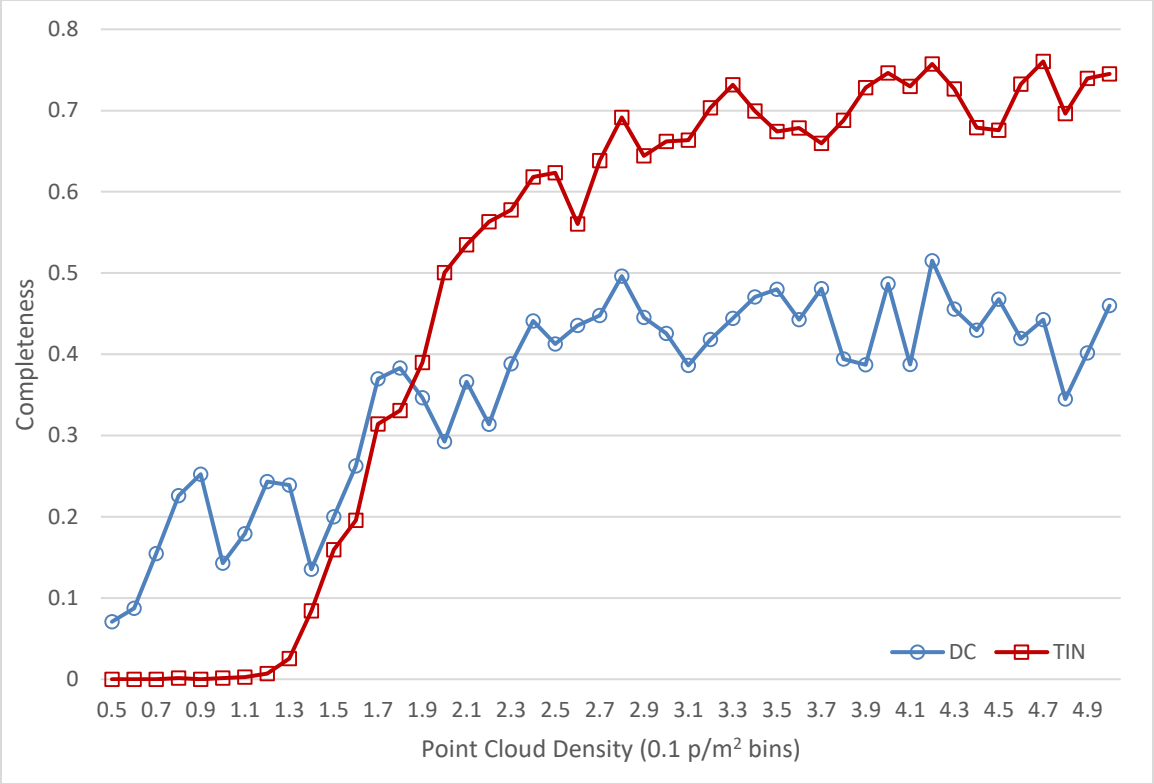


Figure 5-6 Mean footprint completeness is shown for all footprint size classes at point cloud densities from 3.9 to 0.5, rounded to the nearest 0.1

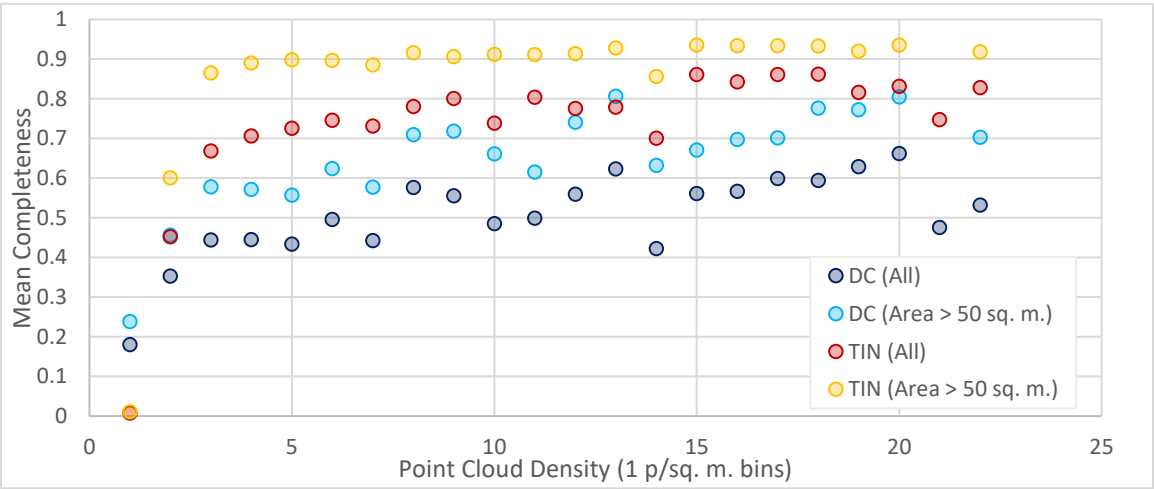


Figure 5-7 Mean completeness for footprints in 1 p/m^2 density bins is plotted, both overall and for only buildings larger than 50 m^2 in area. Bins with fewer than 10 extracted footprints are omitted.

5.2 3D Accuracy

While the accuracy of footprint extraction is assessed based on reference data for the entirety of each study area, time constraints necessitated that 3D reference data be created only for a subset of buildings; information on sample selection is given in Chapter 3. To show reconstruction performance on the basis of point cloud density, we use a binning method identical to that used for 2D footprint accuracy, where extracted models are placed in 1 p/m^2 bins based on mean per-building point cloud density, with those bins that have fewer than 10 members omitted. Note that one key difference separating the 2D footprint comparisons from the 3D model comparisons is that a comparison between reference and extracted model can only be made if at least part of the building in question has been extracted and identified. If no corresponding building is extracted from a given subsample, it will not contribute to the assessed accuracy of extraction; this differs from the 2D assessment where a completeness and correctness score of 0 would be assigned to said building's footprint for that subset. 3D assessment is therefore an indication of the quality of models produced by reconstruction, not a metric of overall performance.

Figures 5-8 and 5-9 show accuracy for both methods as measured by completeness, the three-dimensional counterpart to the 2D completeness metric used to assess footprints earlier. Figures 5-10 and 5-11 show accuracy as measured by Q, which as discussed in section 2.4 describes both the completeness and correctness of the output model.

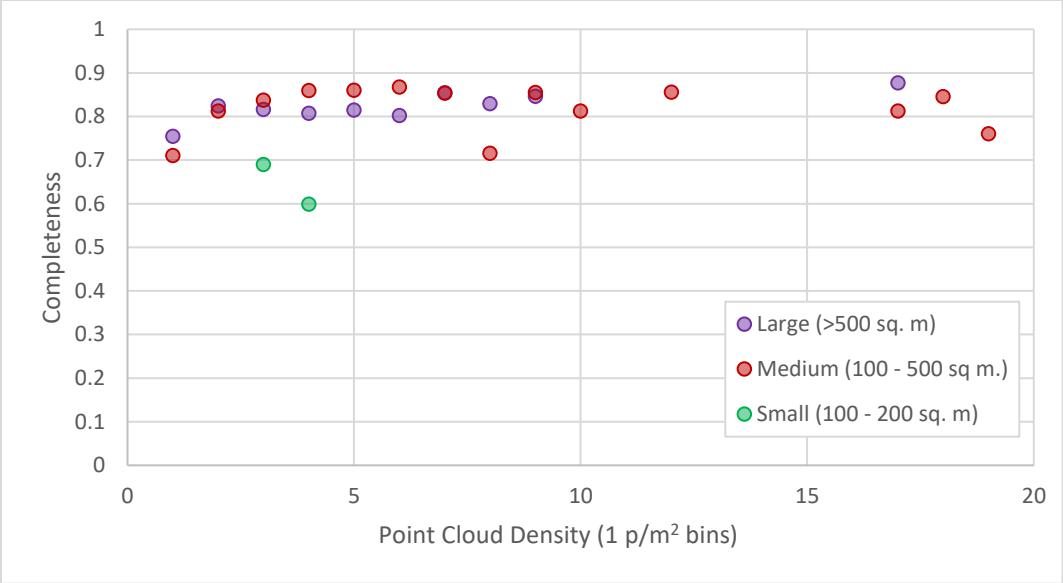
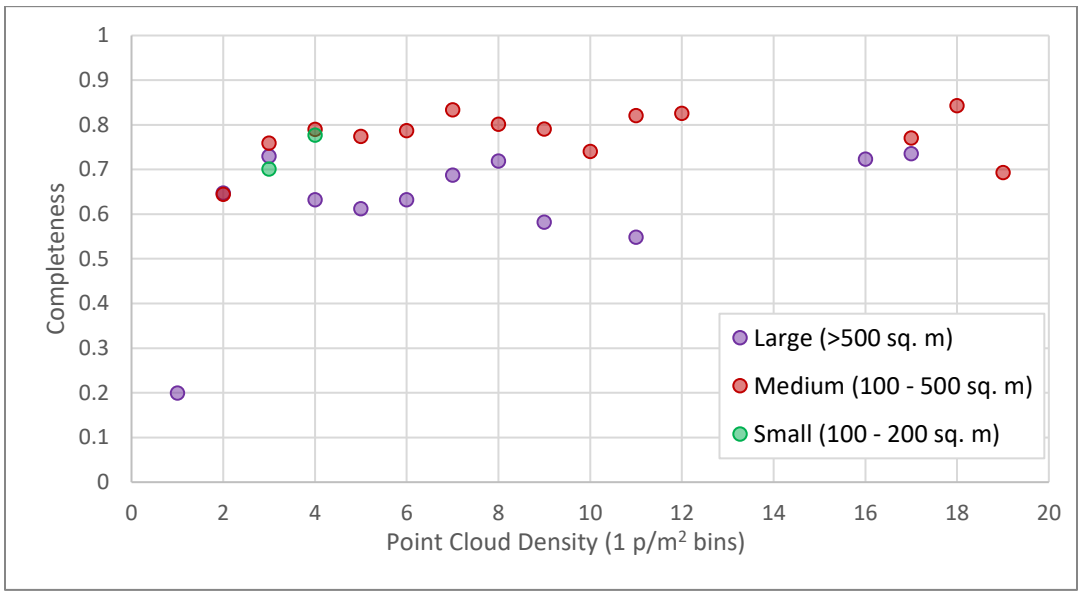


Figure 5-8 Mean completeness of models produced by dual contouring is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown.

Figure 5-9 Mean completeness of models produced by ENVI+TIN method is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown.



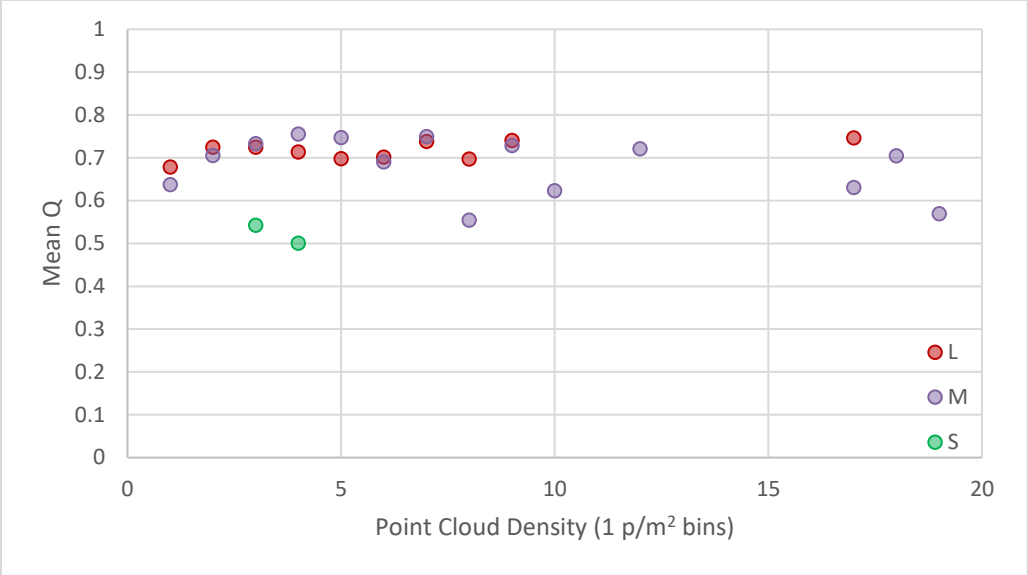
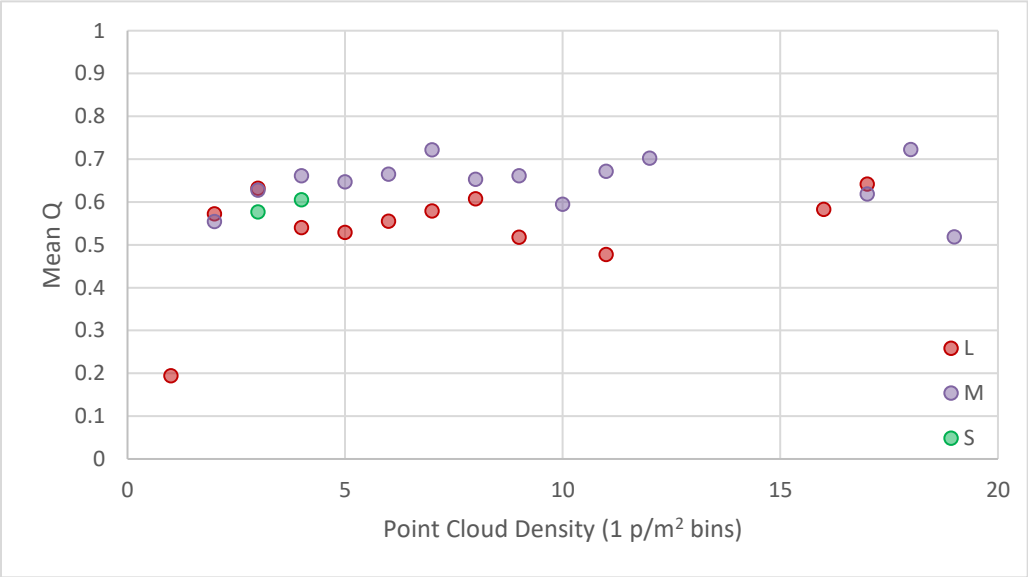


Figure 5-10: Mean Q of models produced by dual contouring is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown.

Figure 5-11: Mean Q of models produced by ENVI+TIN method is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown.



Figures 5-12 and 5-13 show the accuracy of produced models on the basis of SPHARM RMSD for the Dual Contouring and ENVI+TIN method, respectively. Note that unlike with completeness and Q, higher SPHARM RMSD values indicate less similarity found in comparison, and therefore lower accuracy. Note also that the SPHARM comparison process produced a handful of outlier RMSD values (>100) that have been omitted as they strongly skew the binned mean RMSD values.

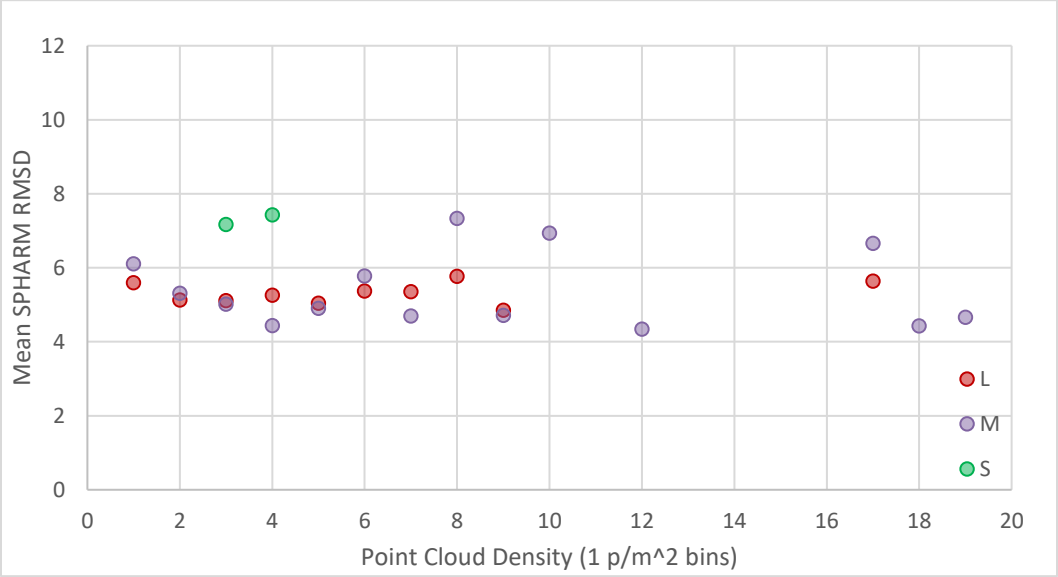


Figure 5-12: Mean SPHARM RMSD of models produced by dual contouring is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown.

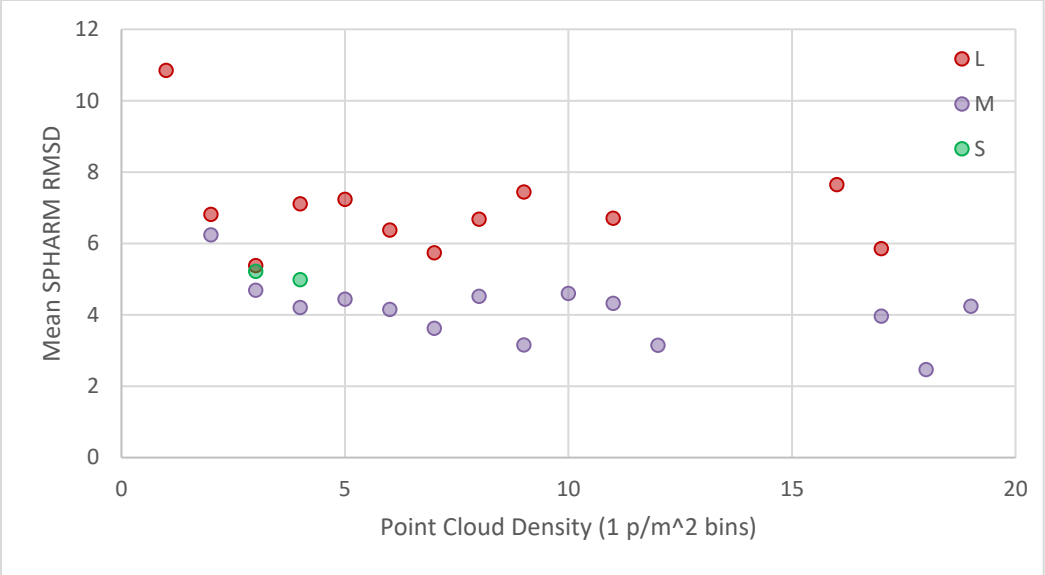


Figure 5-13: Mean SPHARM RMSD of models produced by ENVI+TIN method is plotted, binned by mean point cloud density, for three size classes. Only bins with 10 or more members are shown.

Besides comparing generated models to a manually created reference, it is also possible to compare them to each other. By comparing the models generated from the full resolution point cloud data to those from subsampled datasets, it is possible to measure how the result of automatic reconstruction changes based on point cloud density. Figures 5-14 and 5-15 show the results of these comparisons in terms of both Q and SPHARM RMSD.

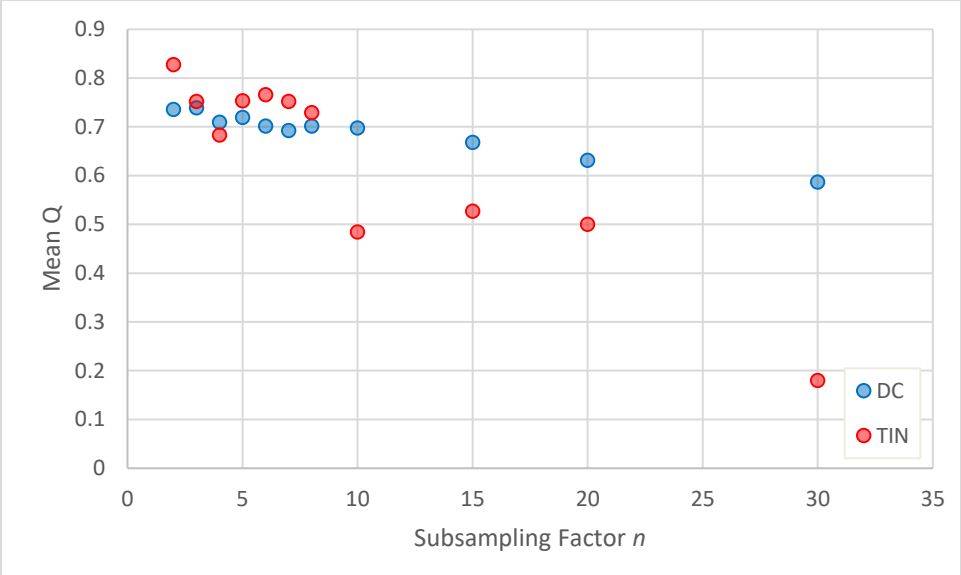


Figure 5-14**:** The mean Q of models generated by either method from subsampled point clouds compared to those generated by the corresponding method from the full-resolution point clouds is plotted.

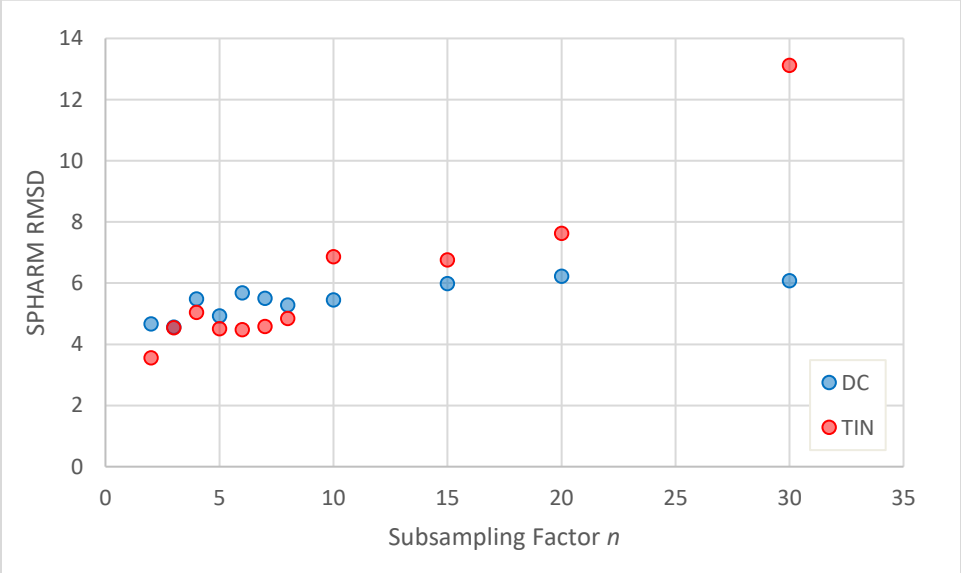


Figure 5-15: The mean SPHARM RMSD of models generated by either method from subsampled point clouds compared to those generated by the corresponding method from the full-resolution point clouds is plotted.

Q and SPHARM RMSD measure related but distinct properties; shared volume and shape similarity, respectively. Two identical models have a Q of 1, indicating 100% of volume is shared by both models, and an RMSD of 0, indicating an identical spherical harmonic representation. Higher values of SPHARM RMSD indicate difference in shape while lower values of Q indicate a lower proportion of shared area. As one would expect, the two measures are negatively correlated (see figure 6-4).

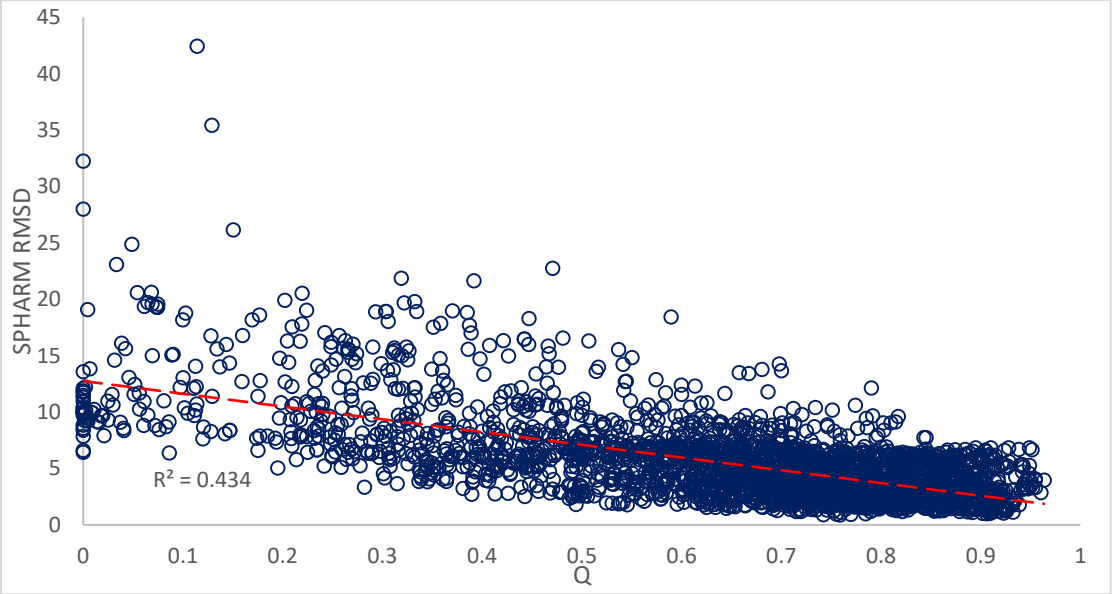


Figure 5-16: SPHARM RMSD plotted against Q for all comparisons, including both reference-to-reconstruction and reconstruction-to-reconstruction. Outlier SPHARM RMSD values (those >100, n=3) are omitted, as are comparisons in which one or both models lacked a valid SPHARM representation.

6 Discussion and Conclusions

6.1 Discussion

Overall assessment of 2D accuracy shows higher performance under the ENVI method than for the Dual Contouring for all but the lowest-density subsets, which is unsurprising given that the former is a mature, commercially developed system and the latter an academically-developed method focused on 3D reconstruction. The most important lesson drawn from the overall accuracy assessment is that, relative to completeness, correctness is quite high for both methods and all subsets, being above 90% for all but one combination of the two. This is critically important since the per-footprint assessments can measure completeness only and would thus be a poor measure of performance were false positives a large contributor to overall inaccuracy. Fortunately, this does not appear to be the case, and there is little apparent relation between subset factor *n* (and thus overall mean point cloud density, as the relationship is shown in Figure 6-1) and correctness.

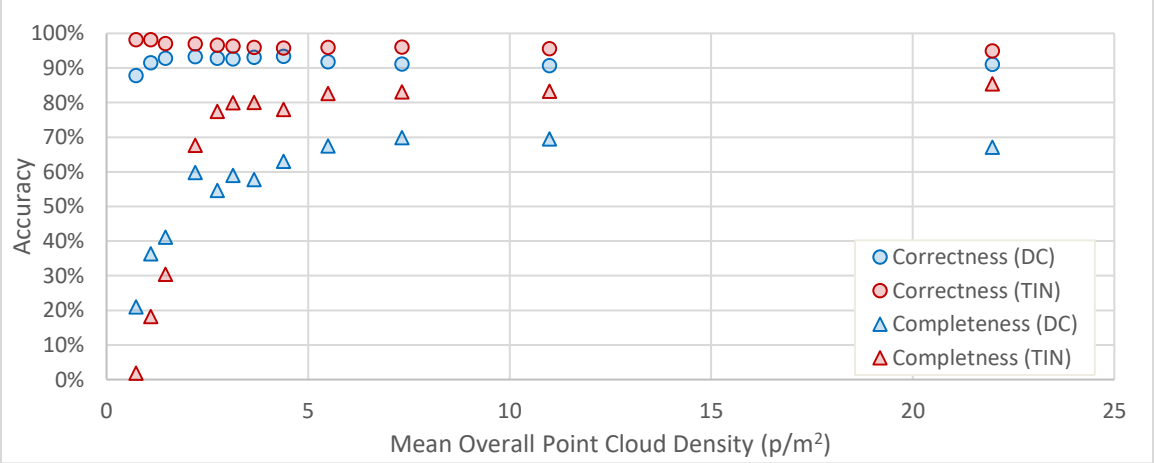


Figure 6-1: Overall 2D accuracy in terms of both completeness and correctness are plotted for both methods.

The relationship between point cloud density and 2D building extraction accuracy is clearly positive, as expected. Both methods show a clear decline in completeness as mean point cloud density drops below 3 p/m^2. This drop is clearly visible for buildings of all

size classes except for those buildings below 50 m^2 in area, which is either very low overall (for the DC method) or declines continuously below 6 p/m^2 mean density (for the ENVI method).

Figure 5-7 best illustrates overall footprint completeness across the entire range of point cloud density levels available in our data. There is little evidence of a stable positive or negative relationship between completeness and point cloud density above 3 p/m^2 except for this latter trend. For all other size classes, mean completeness is consistently between 85% and 95% for all density bins above 3 p/m^2. Mean completeness is both lower in general and less consistent for the DC method than for ENVI above 3 p/m^2. Completeness at low density values is shown in higher detail in Figure 5-4 which shows mean completeness for footprints in point cloud density bins 0.1 p/m^2 wide. For both methods, footprint completeness declines below 3 p/m^2, but the nature of the decline is not the same between methods. The completeness of footprints produced by the ENVI method declines very sharply between 2.5 and 1.2 p/m^2 from a range of 60-70% to near zero. For the DC method, the decline is more gradual and less pronounced, from 30-40% mean completeness at the high end to 10-20% at the low end. Note that the mean completeness values quoted here are for all buildings; as Figure 5-7 shows, completeness is typically substantially higher when buildings smaller than 50 square meters are excluded.

In terms of 3D reconstruction accuracy, the relative performances of the two methods are reversed; the ENVI+TIN method creates less accurate models than the Dual Contouring, particularly for large buildings. It is difficult to identify a trend in accuracy with point cloud density using any of the three metrics shown in Section 5.2, particularly in the higher point cloud density bins where sample size is small. Both methods show lower accuracy at the lowest density bin (1 p/m^2, corresponding to mean densities between 0.5 and 1.5 p/m^2) for all metrics compared to the 2, 3 and 4 p/m^2 bins, suggesting there may be a major effect on reconstruction accuracy at very low density levels. Overall, accuracy as assessed by 3D metrics appears to be less sensitive to decreases in point cloud density than when assessed based on footprints. This is not to say that 3D reconstruction process is unaffected by point cloud density: there are noticeable differences in models of the

same building generated from different point cloud subsets (as shown in Figures 5-13 and 5-14). The difference between models generated using the full resolution dataset and those created with subsampled datasets appears to increase with increasing values of n (thinner point clouds), a trend that is more pronounced for the ENVI+TIN models than those generated using dual contouring. These differences do not appear to reflect an overall trend in 3D accuracy, however. It is notable that subjectively, there is less apparent detail in DC models created from heavily thinned datasets than from the original or less thinned (n = 1,2,3) datasets. This appears to be due to the larger values of the grid spacing parameter l_g necessary to generate hole-free models from sparse datasets; grid cells that contain no points will be reconstructed as pits in the output model. Since cells in the dual contouring grid cannot by smaller than l_g, this parameter acts as a lower bound on the size of features that may be represented in the reconstructed model. Interestingly, the loss of detail does not appear to be reflected in numerical assessments of model accuracy, suggesting that other factors such as spuriously added or excluded objects have a larger effect on both volume and shape accuracy metrics. See figures 6-2, 6-3 and 6-4 for an illustrative example of how reconstruction output changes with decreased point cloud density from both methods.

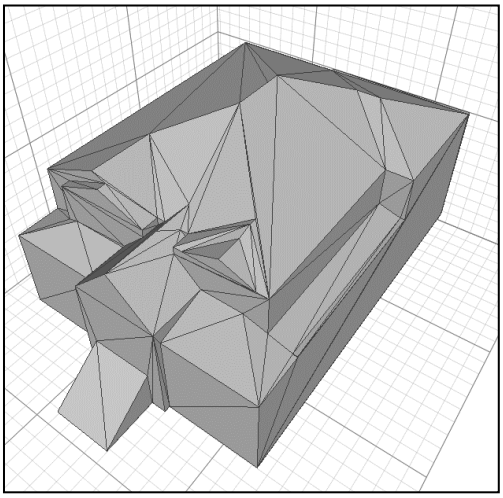


Figure 6-2: A reference model of a large house.

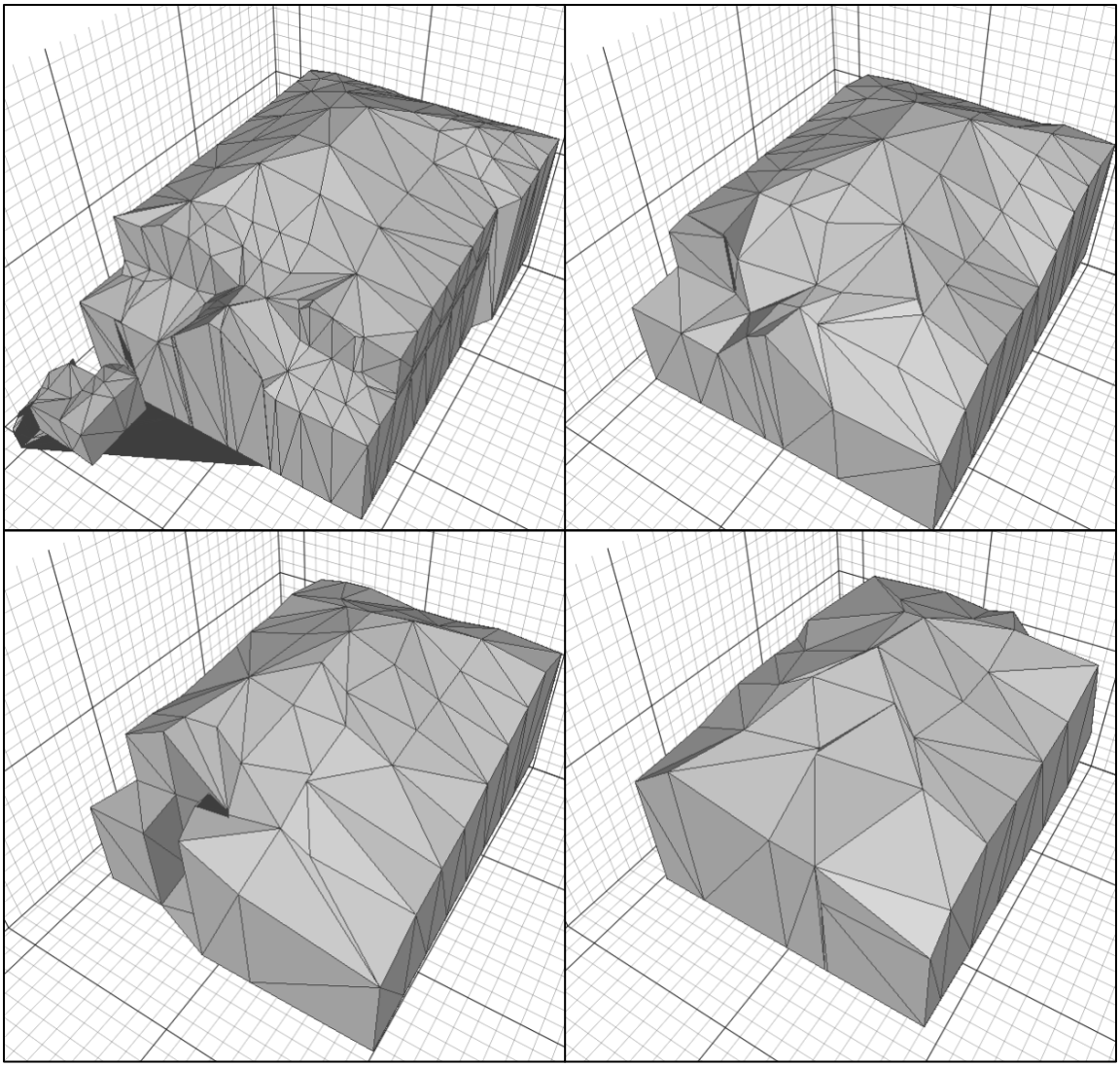


Figure 6-3: The products of DC reconstruction of the same large house shown in Figure 6-2 are shown for four datasets, clockwise from top right: n=1, n=5, n=15, n=10. Note the loss of fine details associated with increasing initial grid length, as well as the agglomeration of a smaller, non-building structure onto the model in the n=1 model.

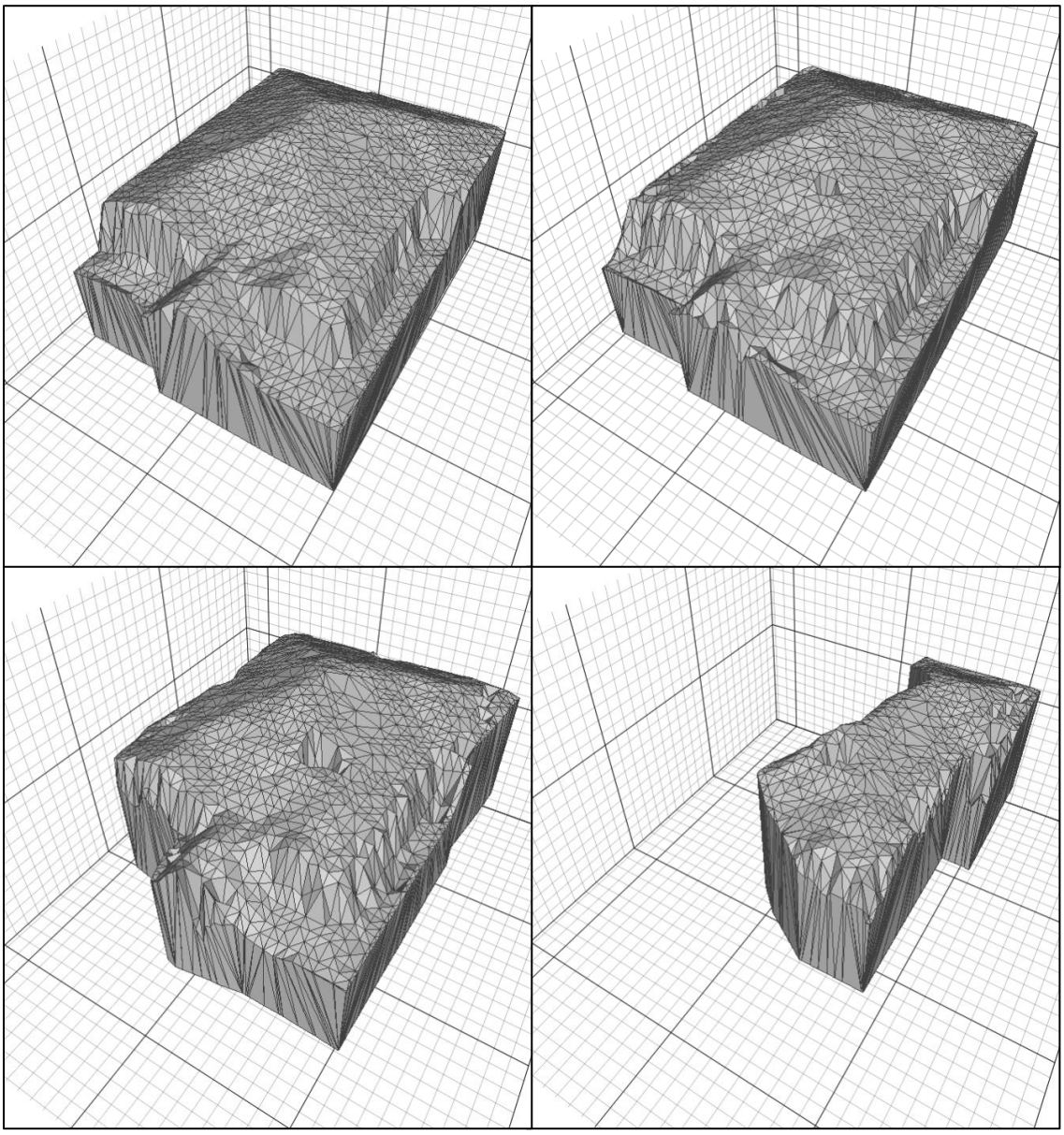


Figure 6-4: Products of reconstruction using the ENVI+TIN method are shown, on the same house and from the same datasets as in Figure 6-3. Note minor omission of building volume in the n=10 reconstruction and major omission in the n=15 volume.

Interestingly, the 3D quality metrics show a considerable accuracy gap for the ENVI+TIN method between large and medium-sized buildings; unexpectedly, large buildings are reconstructed to noticeably worse accuracy in many cases. We have strong reason to believe that this in fact reflects a flaw in the quality assessment methodology rather than an actual difference in reconstruction performance. In general, the 3D completeness metric of a generated model tends to be close to the footprint completeness, with a mean absolute difference of 0.13. For some models however, the difference is far higher, in excess of 0.9. For high-discrepancy models, those 152 models with a footprint completeness vs. 3D completeness difference of greater than 0.5, all have a 3D (i.e. volume) completeness lower than their associated footprint completeness. On inspection, it appears that the 3D model accuracy assessment process in some cases greatly under-reports model accuracy; models that appear complete and accurately positioned are assessed as though only a tiny fraction of their volume is accurate. The problem appears to disproportionately affect large models and those generated using the ENVI+TIN method: of the 152 high-discrepancy models 44.4% are TIN models of Large (>500 sq. m) buildings, despite such models making up only 18.3% of models with valid accuracy assessments. When high-discrepancy models are eliminated from the data, the gap in accuracy between large and medium-sized TIN models disappears almost completely (see Figures 6-5 and 6-6). Fortunately, the overall point cloud density to accuracy curves are unaffected, as no correlation between discrepancy and point cloud density is apparent.

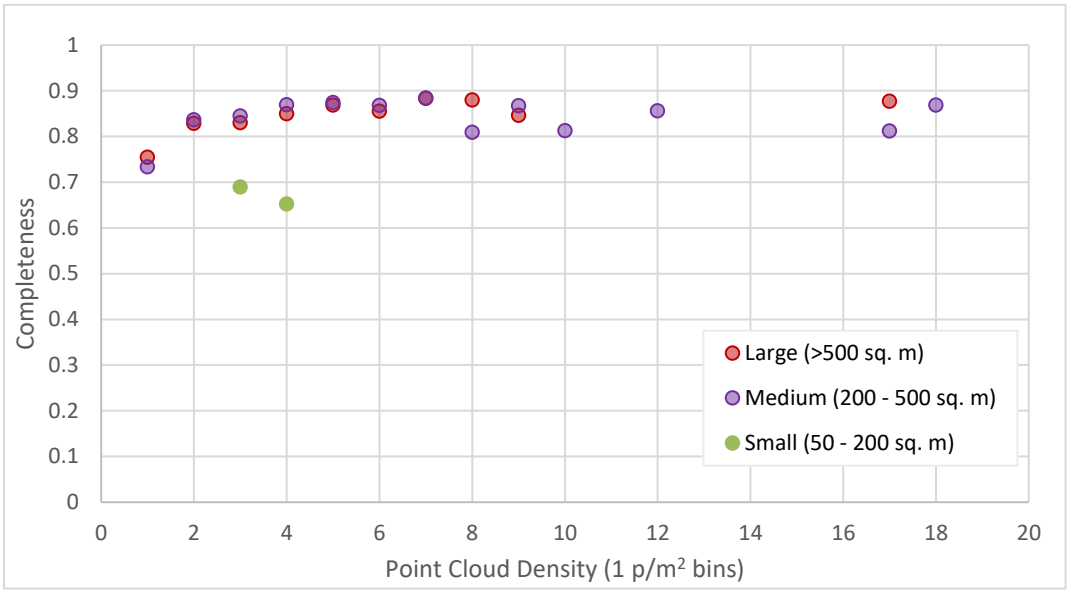


Figure 6-5: 3D completeness of dual contouring models is plotted against point cloud density for only models where 3D and 2D completeness are within 50%.

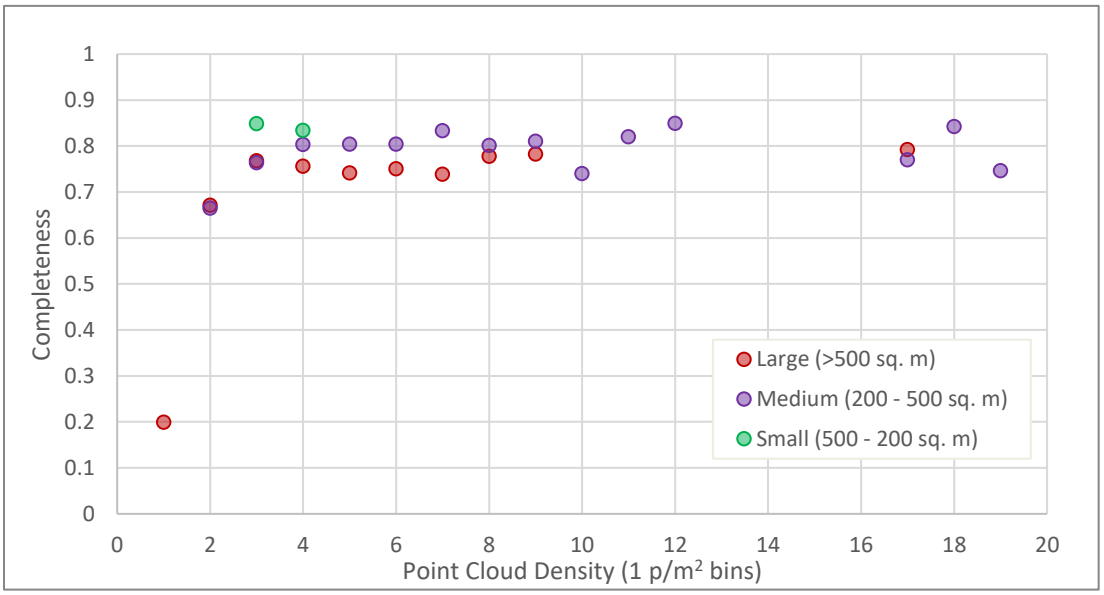


Figure 6-6: 3D completeness of ENVI+TIN models is plotted against point cloud density for only models where 3D and 2D completeness are within 50%.

More generally, it is possible to make several observations on the various metrics used to measure model accuracy. Completeness, Correctness and Q were all calculated without any apparent issues. The SPHARM RMSD calculation however sometimes produced extreme values (on the order of 1000 or higher) in rare cases, roughly one per thousand models for the reference comparisons. More frequently, models may lack a valid SPHARM representation due to topological defects, making comparison using SPHARM RMSD impossible. As described previously, there was also a distance-based metric which relied on measuring the distance between matched points and finding a mean. The distance metric was omitted from analysis as it proved vulnerable to outliers created when tall buildings were not modeled with the correct height; it was however useful in identified models that were identified with the wrong reference model or that were projected into the wrong local coordinate system.

The ENVI+TIN method is effective in identifying the two-dimensional footprint of buildings but is not always reliable when reconstructing them in 3D. In one instance, the roof TIN was generated using only a small portion of building points, meaning most buildings were either not modeled or modeled with inaccurate roofs. In most other cases the method was sufficient to create geometrically accurate building models, albeit with overly-detailed roof surfaces, resulting in large file sizes and consequentially lengthy processing times. Roof surfaces generated by the ENVI+TIN method have a rough appearance caused by noise in the LiDAR data propagating to the interpolated roof surface raster which is attenuated, but not eliminated, by the filtering process prior to the creation of the roof TIN. The DC method in contrast generated much simpler models that were nevertheless more accurate on average than those created by the ENVI+TIN method.

Our work shows the SPHARM shape comparison methodology as suitable for use on a large scale, that of hundreds or thousands of comparisons. One important drawback of this method is the considerable processing time needed to map each shape to the unit sphere and parameterize it; this was one of the factors that lead us to limit the number of

buildings subject to comparison to a fraction of those president in the study areas. A more efficient implementation of the underlying process will likely be necessary to make comparisons between methods on the scale of entire cities prohibitive without massive investments of either time or computing capacity, but on the scale of neighbourhoods the computational barrier is not insurmountable. A more fundamental limitation is the strict topological criteria imposed on models by the SPHARM process; despite post-processing it was not always possible to ensure that automatically reconstructed models fulfilled these criteria. It is advisable for researchers looking to use SPHARM-based comparisons to assess the accuracy of their own reconstruction methods to tailor their method to produce topologically appropriate models when possible.

Based on our analysis of the relationship between building extraction and reconstruction accuracy and point cloud density, we can make several recommendations and observations relevant to those looking to commission aerial LiDAR data. The first is that there is no clear evidence a trend in footprint extraction accuracy at per-building point cloud densities above 3 p/m^2. Above that level, accuracy gain from increased point cloud density appears minimal to non-existent. Practitioners should therefore be careful to ensure that data density does not fall below 3 p/m^2 for flat surfaces; vegetated areas will require higher density data to achieve the same functional density for the purposes of building extraction, due to the effect of multiple returns. It may be advisable to specify a flat-surface minimum density somewhat higher than the 3 p/m^2 minimum to provide a margin of error both in collection and potential method-specific variability in the relationship between density and extraction accuracy. We have also noted subjective differences in models created using the DC method which were attributable to the need for larger atomic grid lengths to suit sparser datasets; if high-detail models are desired it may therefore be desirable to make use of much higher density datasets than the 3 p/m^2 minimum, perhaps on the order of 10 p/m^2 or more if surface features smaller than 0.5 m^2 are to be represented. Interestingly, the loss subjective detail noted when atomic grid length is increased does not appear to produce a major loss of model accuracy as measured quantitatively, suggesting that the differences between more and less detailed models may be quantitatively small.

Although building size does affect the likelihood of extraction under both methods, in most cases there is not a noticeable difference in terms of the relationship between mean extraction completeness and point cloud density for buildings of different size. A notable exception is for buildings less than 50 square meters in size, which for the ENVI+TIN method shows a continuous decline in completeness as point cloud density declines below 6 p/m^2. This is of limited importance however as neither method was able to reliably extract buildings of this size; using full resolution data, the ENVI+TIN method achieved a mean completeness of 58% for very small buildings, while the Dual Contouring method achieved a mean completeness of only 22%.

As we find no evidence in a trend in reconstructed model accuracy above the 3 p/m^2 threshold. Below that threshold, there is evidence of a decline in accuracy for both methods, but especially for the ENVI+TIN method. There is clear evidence that models generated using subsampled data vary from those generated from the full resolution data, but only for the ENVI+TIN method is it obvious that those reconstructed from highly thinned point clouds vary more than those from higher-density subsamples. Based on these observations, we believe that any aerial LiDAR data sufficient for use in building extraction should be equally sufficient for traditional 3D reconstruction. Reconstruction of façade details would likely require much higher density data with specialized characteristics (as in Truong-Hong and Laefer 2015), but such techniques are beyond the scope of this study.

6.2 Conclusions

Returning to our principal research questions, we find the following:

1) The existence of a point of diminished returns on accuracy of extraction and reconstruction with respect to point cloud density is confirmed, with no clear quantitative improvements on accuracy for point clouds denser than 3 p/m^2 for either method.

2) The relationship between accuracy and point cloud density is similar regardless of whether accuracy is judged based on 2D footprints or 3D model similarity.

3) The character of the relationship between point cloud density and overall accuracy differs noticeably between methods; although both methods share the same point of diminished returns on accuracy, accuracy for the ENVI+TIN method decreases much more sharply below this compared to the Dual Contouring method.

4) Building size as measured by footprint area is a strong influence on the likelihood of a building being reconstructed, but except for very small (<50 m^2) buildings as extracted by the ENVI+TIN method, the relationship between point cloud density and accuracy is consistent for buildings of all sizes.

We conclude by advising a 4 p/m^2 scan density for flat surfaces as optimal for building reconstruction using either of our methods; those commissioning LiDAR data for building reconstruction, such as municipalities, should ensure that data is collected to at least this density. The 4 p/m^2 specification provides a margin for error to accommodate local variation in point cloud density and thereby assure that all areas are covered at a local point cloud density comfortably above the 3 p/m^2 point of diminishing returns. Our findings contrast with those of Lohani and Singh (2008) and Tomljenovic and Roussel (2014), who both found improvements in accuracy for point cloud densities on intervals well above 3 p/m^2. This suggests that although both methods tested in this study had similar responses to changes in point cloud density, other methods of extraction and/or reconstruction may benefit from higher densities. We advice those developing or assessing reconstruction techniques to test their process on a range of LiDAR data densities; this study's most significant contribution for building reconstruction researchers is the development and presentation of a systematic means of doing so. For those using existing techniques, our work gives guidance as to what data is fit for use for building reconstruction from aerial LiDAR, which we believe will be useful particularly as 3D visualization techniques such as virtual reality see wider and wider adoption in settings such as municipal government and real estate development.

Bibliography

"3D Printing Toolbox." 2013. Wiki Article. *Wiki.Blender.Org*. https://wiki.blender.org/index.php/Extensions:2.6/Py/Scripts/Modeling/PrintToolbox

Akca, Devrim, Mark Freeman, Isabel Sargent, and Armin Gruen. 2010. "Quality Assessment of 3D Building Data: Quality Assessment of 3D Building Data." *The Photogrammetric Record* 25 (132):339–55. https://doi.org/10.1111/j.1477-9730.2010.00598.x.

Airborne Imaging. 2013. "Final Report for Project: City of Surrey LiDAR." Calgary, AB.

Avbelj, Janja, and Rupert Muller. 2014. "Quality Assessment of Building Extraction from Remote Sensing Imagery." In , 3184–87. IEEE. https://doi.org/10.1109/IGARSS.2014.6947154.

Avbelj, Janja, Rupert Muller, and Richard Bamler. 2015. "A Metric for Polygon Comparison and Building Extraction Evaluation." *IEEE Geoscience and Remote Sensing Letters* 12 (1):170–74. https://doi.org/10.1109/LGRS.2014.2330695.

Awrangjeb, Mohammad, Mehdi Ravanbakhsh, and Clive S. Fraser. 2010. "Automatic Detection of Residential Buildings Using LIDAR Data and Multispectral Imagery." *ISPRS Journal of Photogrammetry and Remote Sensing* 65 (5):457–67. https://doi.org/10.1016/j.isprsjprs.2010.06.001.

Benseddik, Houssem-Eddine, Hicham Hadj-Abdelkader, Brahim Cherki, and Samia Bouchafa. 2016. "Direct Method for Rotation Estimation from Spherical Images Using 3D Mesh Surfaces with SPHARM Representation." *Journal of Visual Communication and Image Representation* 40 (October):708–20. https://doi.org/10.1016/j.jvcir.2016.08.010.

Biljecki, Filip, Hugo Ledoux, and Jantien Stoter. 2016. "An Improved LOD Specification for 3D Building Models." *Computers, Environment and Urban Systems* 59 (September): 25–37. doi:10.1016/j.compenvurbsys.2016.04.005.

Blender (version 2.8). 2018. Blender Foundation.

Botsch, Mario, Mark Pauly, Leif Kobbelt, Pierre Alliez, Bruno Lévy, Stephan Bischoff, and Christian Rössl. 2007. "Geometric Modeling Based on Polygonal Meshes." https://graphics.ethz.ch/Downloads/Publications/Tutorials/2007/Bot07a/SIG07Notes.pdf.

Brechbühler, C.H., G. Gerig, and O. Kübler. 1995. "Parameterization of Closed Surfaces of 3-D Shape Description." *Computer Vision and Image Understanding* 61 (2):154–70.

Brunn, Ansgar, and Uwe Weidner. 1997. "Extracting Buildings from Digital Surface Models." In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. XXXII-3-4W2. Stuttgart, Germany: International Society for Photogrammetry and Remote Sensing.

Cao, Rujun, Yongjun Zhang, Xinyi Liu, and Zongze Zhao. 2017. “3D Building Roof Reconstruction from Airborne LiDAR Point Clouds: A Framework Based on a Spatial Database.” *International Journal of Geographical Information Science* 31 (7):1359–80. https://doi.org/10.1080/13658816.2017.1301456.

Chang, Eun-Young and Yo-Sung Ho. 2001. “Three-Dimensional Mesh Simplification by Subdivided Edge Classification.” In , 1:39–42. IEEE. https://doi.org/10.1109/TENCON.2001.949547.

Chen, Dong, Liqiang Zhang, P. Takis Mathiopoulos, and Xianfeng Huang. 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. https://doi.org/10.1109/JSTARS.2014.2349003.

Cheng, Yang, Jie Cao, Qun Hao, Yuqing Xiao, Fanghua Zhang, Wenze Xia, Kaiyu Zhang, and Haoyong Yu. 2017. “A Novel De-Noising Method for Improving the Performance of Full-Waveform LiDAR Using Differential Optical Path.” *Remote Sensing* 9 (11):1109. https://doi.org/10.3390/rs9111109.

Chung, Moo K., Kim M. Dalton, Li Shen, Alan C. Evans, and Richard J. Davidson. 2007. “Weighted Fourier Series Representation and Its Application to Quantifying the Amount of Gray Matter.” *IEEE Transactions on Medical Imaging* 26 (4):566–81. https://doi.org/10.1109/TMI.2007.892519.

Chung, M.K., K.M. Dalton, and R.J. Davidson. 2008. “Tensor-Based Cortical Surface Morphometry via Weighted Spherical Harmonic Representation.” *IEEE Transactions on Medical Imaging* 27 (8):1143–51. https://doi.org/10.1109/TMI.2008.918338.

El Hakim, S., J.-A. Beraldin, M. Picard, and L. Cournoyer. 2008. “Surface Reconstruction of Large Complex Structures from Mixed Range Data – the Erechtheion Experience.” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XXXVII (5B):1077–82.

Ekhtari, Nima, Mohammad Javad Valadan Zoej, Mahmod Reza Sahebi, and Ali Mohammadzadeh. 2009. “Automatic Building Extraction from LIDAR Digital Elevation Models and WorldView Imagery.” *Journal of Applied Remote Sensing* 3 (1):033571. https://doi.org/10.1117/1.3284718.

ENVI Lidar. 2015. Harris Geospatial.

Fiocco, M., G. Bostrom, J.G.M. Goncalves, and V. Sequeira. 2005. “Multisensor Fusion for Volumetric Reconstruction of Large Outdoor Areas.” In , 47–54. IEEE. https://doi.org/10.1109/3DIM.2005.60.

Fuhrmann, Simon, Michael Kazhdan, and Michael Goesele. 2015. “Accurate Isosurface Interpolation with Hermite Data.” In , 256–63. IEEE. doi:10.1109/3DV.2015.36.

Gómez-Gutiérrez, Álvaro, José de Sanjosé-Blasco, Javier Lozano-Parra, Fernando Berenguer-Sempere, and Javier de Matías-Bejarano. 2015. “Does HDR Pre-Processing Improve the Accuracy of 3D Models Obtained by Means of Two Conventional SfM-MVS Software Packages? The Case of the Corral Del Veleta

Rock Glacier." *Remote Sensing* 7 (8):10269–94. https://doi.org/10.3390/rs70810269.

Haala, Norbert, and Claus Brenner. 1999. "Extraction of Buildings and Trees in Urban Environments." *ISPRS Journal of Photogrammetry and Remote Sensing* 54 (2–3):130–37. https://doi.org/10.1016/S0924-2716(99)00010-6.

Hron, Vojtěch, and Lena Halounová. 2015. "Automatic Generation of 3D Building Models from Point Clouds." In *Geoinformatics for Intelligent Transportation*, edited by Igor Ivan, Itzhak Benenson, Bin Jiang, Jiří Horák, James Haworth, and Tomáš Inspektor, 109–19. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-11463-7_8.

International Code Council, ed. 2011. *2012 International Building Code: 2012 IBC ; a Member of the International Code Family*. Country Club Hills, Ill: International Code Council.

Jarzabek-Rychard, M. 2012 "Reconstruction of Building Outlines in Dense Urban Areas Based on Lidar Data and Address Points." In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 39-B3:121–26. Melbourne, Australia: International Society for Photogrammetry and Remote Sensing.

Jarząbek-Rychard, M., and A. Borkowski. 2016. "3D Building Reconstruction from ALS Data Using Unambiguous Decomposition into Elementary Structures." *ISPRS Journal of Photogrammetry and Remote Sensing* 118 (August):1–12. https://doi.org/10.1016/j.isprsjprs.2016.04.005.

Jin, Xiaogang, Chiew-Lan Tai, and Hailin Zhang. 2009. "Implicit Modeling from Polygon Soup Using Convolution." *The Visual Computer* 25 (3):279–88. https://doi.org/10.1007/s00371-008-0267-3.

Ju, Tao, Frank Losasso, Scott Schaefer, and Joe Warren. 2002. "Dual Contouring of Hermite Data." *ACM Transactions on Graphics* 21 (3). doi:10.1145/566654.566586.

Ju, Tao. 2009. "Fixing Geometric Errors on Polygonal Models: A Survey." Journal of Computer Science and Technology 24 (1):19–29. https://doi.org/10.1007/s11390-009-9206-7.

Kodors, Sergejs, and Ilmārs Kangro. 2016. "Simple Method of LiDAR Point Density Definition for Automatic Building Recognition." In *Engineering for Rural Development*, 2016:415–24. Jelgava, Latvia: Latvia University of Agriculture. http://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=116286770&site=ehost-live.\

Lafarge, F., X. Descombes, J. Zerubia, and M. Pierrot-Deseilligny. 2010. "Structural Approach for Building Reconstruction from a Single DSM." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (1):135–47. https://doi.org/10.1109/TPAMI.2008.281.

Lee, Dong Hyuk, Kyoung Mu Lee, and Sang Uk Lee. 2008. "Fusion of Lidar and Imagery for Reliable Building Extraction." *Photogrammetric Engineering & Remote Sensing* 74 (2):215–25. https://doi.org/10.14358/PERS.74.2.215.

Leng, Xiaoxu, Jun Xiao, and Ying Wang. 2016. "A Multi-Scale Plane-Detection Method Based on the Hough Transform and Region Growing." *The Photogrammetric Record* 31 (154):166–92. https://doi.org/10.1111/phor.12145.

Lohani, Bharat, and Rajneesh Singh. 2008. "Effect of Data Density, Scan Angle, and Flying Height on the Accuracy of Building Extraction Using LiDAR Data." *Geocarto International* 23 (2):81–94. https://doi.org/10.1080/10106040701207100.

Li, Hui, Cheng Zhong, Xiaoguang Hu, Long Xiao, and Xianfeng Huang. 2013. "New Methodologies for Precise Building Boundary Extraction from LiDAR Data and High Resolution Image." *Sensor Review* 33 (2):157–65. https://doi.org/10.1108/02602281311299699.

Liu, Chun, Beiqi Shi, Xuan Yang, Nan Li, and Hangbin Wu. 2013. "Automatic Buildings Extraction From LiDAR Data in Urban Area by Neural Oscillator Network of Visual Cortex." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 6 (4):2008–19. https://doi.org/10.1109/JSTARS.2012.2234726.

Maas, Hans-Gerd, and George Vosselman. 1999. "Two Algorithms for Extracting Building Models from Raw Laser Altimetry Data." *ISPRS Journal of Photogrammetry and Remote Sensing* 54 (2–3):153–63. https://doi.org/10.1016/S0924-2716(99)00004-0.

Mohamed, M., Landes, T., Grussenmeyer, P., and Zhang, W. 2013. "Multi-Dimensional Quality Assessment of Photogrammetric and LIDAR Datasets Based on a Vector Approach." *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XL-2/W1 (May):93–98.

Mongus, Domen, Niko Lukač, and Borut Žalik. 2014. "Ground and Building Extraction from LiDAR Data Based on Differential Morphological Profiles and Locally Fitted Surfaces." *ISPRS Journal of Photogrammetry and Remote Sensing* 93 (July):145–56. https://doi.org/10.1016/j.isprsjprs.2013.12.002.

Musialski, P., P. Wonka, D. G. Aliaga, M. Wimmer, L. van Gool, and W. Purgathofer. 2013. "A Survey of Urban Reconstruction: A Survey of Urban Reconstruction." *Computer Graphics Forum* 32 (6): 146–77. https://doi.org/10.1111/cgf.12077.

Tomljenovic, Ivan, and Adam Rousell. 2014. "Influence of Point Cloud Density on the Results of Automated Object-Based Building Extraction from ALS Data." In *Connecting a Digital Europe through Location and Place: Proceedings of the AGILE'2014 International Conference on Geographic Information Science*. Castellón, Spain: Association of Geographic Information Laboratories in Europe.

Orthuber, E., and J. Avbelj. 2015. "3d Building Reconstruction from Lidar Point Clouds by Adaptive Dual Contouring." *ISPRS Annals of Photogrammetry, Remote*

Sensing and Spatial Information Sciences II-3/W4 (March): 157–64. doi:10.5194/isprsannals-II-3-W4-157-2015.

Oude Elberink, Sander, and George Vosselman. 2011. "Quality Analysis on 3D Building Models Reconstructed from Airborne Laser Scanning Data." *ISPRS Journal of Photogrammetry and Remote Sensing* 66 (2):157–65. https://doi.org/10.1016/j.isprsjprs.2010.09.009.

Paniagua, Beatriz, Lucia Cevidanes, David Walker, Hongtu Zhu, Ruixin Guo, and Martin Styner. 2011. "Clinical Application of SPHARM-PDM to Quantify Temporomandibular Joint Osteoarthritis." *Computerized Medical Imaging and Graphics* 35 (5):345–52. https://doi.org/10.1016/j.compmedimag.2010.11.012.

Pirotti, Francesco, and Paolo Tarolli. 2010. "Suitability of LiDAR Point Density and Derived Landform Curvature Maps for Channel Network Extraction." *Hydrological Processes* 24 (9): 1187–97. doi:10.1002/hyp.7582.

Potůčková, Markéta, and Petr Hofman. 2016. "Comparison of Quality Measures for Building Outline Extraction." *The Photogrammetric Record* 31 (154):193–209. https://doi.org/10.1111/phor.12144.

Potworowski, André, Anne Murray-Choudhary, and Bénédicte Losfeld. 2010. "Making It Happen - The Transition to a Sustainable Society - Case Study: The Transformation of the National Building Code: From Prescriptions to Objectives." Telfer School of Management, University of Ottawa.

Priestnall, G., J. Jaafar, and A. Duncan. 2000. "Extracting Urban Features from LiDAR Digital Surface Models." *Computers, Environment and Urban Systems* 24 (2):65–78. https://doi.org/10.1016/S0198-9715(99)00047-2.

Qian, Xiangfei, and Cang Ye. 2014. "NCC-RANSAC: A Fast Plane Extraction Method for 3-D Range Data Segmentation." *IEEE Transactions on Cybernetics* 44 (12):2771–83. https://doi.org/10.1109/TCYB.2014.2316282.

Rabbani, Tahir, and Frank van den Heuvel. 2005. "Efficient Hough Transform for Automatic Detection of Cylinders in Point Clouds." In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36:60–65. Enschede, the Netherlands: International Society for Photogrammetry and Remote Sensing. http://www.isprs.org/proceedings/XXXVI/3-W19/papers/060.pdf.

Rottensteiner, F., and Ch. Briese. 2002. "A New Method for Building Extraction in Urban Areas from High-Resolution LIDAR Data." *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 32.

Rottensteiner, F., Sohn, G., Jung, J., Gerke, M., Baillard, C., Benitez, S., and Breitkopf, U. 2012. "The Isprs Benchmark on Urban Object Classification and 3d Building Reconstruction." In . Melbourne, Australia: International Society for Photogrammetry and Remote Sensing. http://www2.isprs.org/tl_files/isprs/wg34/docs/ISPRS_Test_Urban_Objects_joint_Melbourne_2012.pdf.

Rutzinger, M., F. Rottensteiner, and N. Pfeifer. 2009. "A Comparison of Evaluation Techniques for Building Extraction From Airborne Laser Scanning." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 2 (1):11–20. https://doi.org/10.1109/JSTARS.2009.2012488.

Sampath, Aparajithan, and Jie Shan. 2007. "Building Boundary Tracing and Regularization from Airborne Lidar Point Clouds." *Photogrammetric Engineering & Remote Sensing* 73 (7):805–12. https://doi.org/10.14358/PERS.73.7.805.

Sampath, A., and Jie Shan. 2010. "Segmentation and Reconstruction of Polyhedral Building Roofs From Aerial Lidar Point Clouds." *IEEE Transactions on Geoscience and Remote Sensing* 48 (3):1554–67. https://doi.org/10.1109/TGRS.2009.2030180.

Seo, Suyoung, Jeongho Lee, and Yongil Kim. 2014. "Extraction of Boundaries of Rooftop Fenced Buildings From Airborne Laser Scanning Data Using Rectangle Models." *IEEE Geoscience and Remote Sensing Letters* 11 (2):404–8. https://doi.org/10.1109/LGRS.2013.2263575.

Shen, Wei. 2008. "Building Boundary Extraction Based on LIDAR Point Clouds Data." In *ISPRS Archives*, XXXVII Part 3b:157–62. Beijing, China: International Society for Photogrammetry and Remote Sensing. http://www.isprs.org/proceedings/XXXVII/congress/3b_pdf/37.pdf.

Shufelt, J.A. 1999. "Performance Evaluation and Analysis of Monocular Building Extraction from Aerial Imagery." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21 (4):311–26. https://doi.org/10.1109/34.761262.

Siddiqui, Fasahat, Shyh Teng, Mohammad Awrangjeb, and Guojun Lu. 2016. "A Robust Gradient Based Method for Building Extraction from LiDAR and Photogrammetric Imagery." *Sensors* 16 (7):1110. https://doi.org/10.3390/s16071110.

Silván-Cárdenas, José Luis, and Le Wang. 2011. "Extraction of Buildings Footprint from LiDAR Altimetry Data with the Hermite Transform." In *Pattern Recognition: Third Mexican Conference, MCPR 2011, Cancun, Mexico, June 29 - July 2, 2011. Proceedings*, edited by José Francisco Martínez-Trinidad, Jesús Ariel Carrasco-Ochoa, Cherif Ben-Youssef Brants, and Edwin Robert Hancock, 314–21. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-21587-2_34.

Sohn, Gunho, and Ian Dowman. 2007. "Data Fusion of High-Resolution Satellite Imagery and LiDAR Data for Automatic Building Extraction." *ISPRS Journal of Photogrammetry and Remote Sensing* 62 (1):43–63. https://doi.org/10.1016/j.isprsjprs.2007.01.001.

Arttu Soininen. 2016. *TerraScan User's Guide*. Terrasolid. https://www.terrasolid.com/download/tscan.pdf.

Statistics Canada. 2017. "Focus on Geography Series, 2016 Census." Ottawa, Ontario

Sun, Shaohui, and Carl Salvaggio. 2013. "Aerial 3D Building Detection and Modeling From Airborne LiDAR Point Clouds." *IEEE Journal of Selected Topics in*

Applied Earth Observations and Remote Sensing 6 (3):1440–49. https://doi.org/10.1109/JSTARS.2013.2251457.

Taillandier, Franck, and R Deriche. 2004. "Automatic Buildings Reconstruction from Aerial Images: A Generic Bayesian Framework." In *Proceedings of the ISPRS Congress*. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.184.1775&rep=rep1&type=pdf.

Tarsha-Kurdi, F., T. Landes, and P. Grussenmeyer. 2007. "Hough-Transform and Extended Ransac Algorithms for Automatic Detection of 3d Building Roof Planes from Lidar Data." In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36:407–12. International Society for Photogrammetry and Remote Sensing. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.221.9952&rep=rep1&type=pdf

Tarsha-Kurdi, Fayez, Tania Landes, Pierre Grussenmeyer, and Mathieu Koehl. 2007. "Model-Driven and Data-Driven Approaches Using LIDAR Data : Analysis and Comparison." In *ISPRS Workshop, Photogrammetric Image Analysis (PIA07)*, 87–92. Munich, Germany. https://halshs.archives-ouvertes.fr/halshs-00264846.

Tomljenovic, Ivan, and Adam Rousell. 2014. "Influence of Point Cloud Density on the Results of Automated Object-Based Building Extraction from ALS Data." In *Connecting a Digital Europe through Location and Place: Proceedings of the AGILE'2014 International Conference on Geographic Information Science*. Castellón, Spain: Association of Geographic Information Laboratories in Europe.

Tomljenovic, Ivan, Bernhard Höfle, Dirk Tiede, and Thomas Blaschke. 2015. "Building Extraction from Airborne Laser Scanning Data: An Analysis of the State of the Art." *Remote Sensing* 7 (4):3826–62. https://doi.org/10.3390/rs70403826.

Thompson, Deanne K., Christopher Adamson, Gehan Roberts, Nathan Faggian, Stephen J. Wood, Simon K. Warfield, Lex W. Doyle, Peter J. Anderson, Gary F. Egan, and Terrie E. Inder. 2013. "Hippocampal Shape Variations at Term Equivalent Age in Very Preterm Infants Compared with Term Controls: Perinatal Predictors and Functional Significance at Age 7." *NeuroImage* 70 (April):278–87. https://doi.org/10.1016/j.neuroimage.2012.12.053.

Truong-Hong, Linh, and Debra F. Laefer. 2015. "Quantitative Evaluation Strategies for Urban 3D Model Generation from Remote Sensing Data." *Computers & Graphics* 49 (June):82–91. https://doi.org/10.1016/j.cag.2015.03.001.

Tseng, Yi-Hsing, and Hsiao-Chu Hung. 2016. "Extraction of Building Boundary Lines from Airborne Lidar Point Clouds." *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLI-B3 (October):957–62. https://doi.org/10.5194/isprs-archives-XLI-B3-957-2016.

Uzar, Melis, and Naci Yastikli. 2013. "Automatic Building Extraction Using LiDAR and Aerial Photographs." *Boletim de Ciências Geodésicas* 19 (2):153–71. https://doi.org/10.1590/S1982-21702013000200001.

Varghese, Viji, Dimple A. Shajahan, and Aneesh G Nath. 2016. “Building Boundary Tracing and Regularization from LiDAR Point Cloud.” In , 1–6. IEEE. https://doi.org/10.1109/ICETT.2016.7873645.

Vo, Anh-Vu, Linh Truong-Hong, Debra F. Laefer, and Michela Bertolotto. 2015. “Octree-Based Region Growing for Point Cloud Segmentation.” *ISPRS Journal of Photogrammetry and Remote Sensing* 104 (June):88–100. https://doi.org/10.1016/j.isprsjprs.2015.01.011.

Vosselman, George and Hans-Gerd Maas, eds. 2010. “Laser Scanning Technology.” In *Airborne and Terrestrial Laser Scanning*. Dunbeath, Scotland, UK: Whittles Publishing.

Wang, Ruisheng. 2013. “3D Building Modeling Using Images and LiDAR: A Review.” *International Journal of Image and Data Fusion* 4 (4): 273–92. https://doi.org/10.1080/19479832.2013.811124.

Wang, Ruisheng, Yong Hu, Huayi Wu, and Jian Wang. 2016. “Automatic Extraction of Building Boundaries Using Aerial LiDAR Data.” *Journal of Applied Remote Sensing* 10 (1):016022. https://doi.org/10.1117/1.JRS.10.016022.

White, Brian. 2009. “Which Ambient Spaces Admit Isoperimetric Inequalities for Submanifolds?” *Journal of Differential Geometry* 83 (1):213–28. https://doi.org/10.4310/jdg/1253804356.

Xiong, B., M. Jancosek, S. Oude Elberink, and G. Vosselman. 2015. “Flexible Building Primitives for 3D Building Modeling.” *ISPRS Journal of Photogrammetry and Remote Sensing* 101 (March):275–90. https://doi.org/10.1016/j.isprsjprs.2015.01.002.

Xu, Yusheng, Wei Yao, Ludwig Hoegner, and Uwe Stilla. 2017. “Segmentation of Building Roofs from Airborne LiDAR Point Clouds Using Robust Voxel-Based Region Growing.” *Remote Sensing Letters* 8 (11):1062–71. https://doi.org/10.1080/2150704X.2017.1349961.

Yan, Yiming, Fengjiao Gao, Shupei Deng, and Nan Su. 2017. “A Hierarchical Building Segmentation in Digital Surface Models for 3D Reconstruction.” *Sensors* 17 (2):222. https://doi.org/10.3390/s17020222.

Yang, Bisheng, Wenxue Xu, and Zhen Dong. 2013. “Automated Extraction of Building Outlines From Airborne Laser Scanning Point Clouds.” *IEEE Geoscience and Remote Sensing Letters* 10 (6):1399–1403. https://doi.org/10.1109/LGRS.2013.2258887.

Yari, Diako, Mehdi Mokhtarzade, Hamid Ebadi, and Salman Ahmadi. 2014. “Automatic Reconstruction of Regular Buildings Using a Shape-Based Balloon Snake Model.” *The Photogrammetric Record* 29 (146):187–205. https://doi.org/10.1111/phor.12060.

Yunfan, Li, and Ma Hongchao. 2011. “A New Method for Estimating Building Dominant Directions in LiDAR Data Based on Straight Skeleton.” In , 205–8. IEEE. https://doi.org/10.1109/ICCSN.2011.6014035.

Zeng, Chuiqing, Jinfei Wang, and Brad Lehrbass. 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):1640–52. https://doi.org/10.1109/JSTARS.2013.2256882.

Zhang, Xiaohong 张小红, and Jianghui Geng 耿江辉. 2006. 用不变矩从机载激光扫描测高点 云数据中重建规则房屋 [“Building Reconstruction from Airborne Laser Altimetry Points Cloud Data Set Based on Invariant Moments.”] 武汉大学学报信息科学版[*Geomatics and Information Science of Wuhan University*]31 (2):168–71.

Zhou, Qian-Yi, and Ulrich Neumann. 2008. “Fast and Extensible Building Modeling from Airborne LiDAR Data.” In , 1. ACM Press. https://doi.org/10.1145/1463434.1463444.

Zhou, Qian-Yi, and Ulrich Neumann. 2010. “2.5D Dual Contouring: A Robust Approach to Creating Building Models from Aerial LiDAR Point Clouds.” In *Computer Vision – ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part III*, edited by Kostas Daniilidis, Petros Maragos, and Nikos Paragios, 115–28. Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-642-15558-1_9.

Zhou, Qian-Yi. 2012. “3D Urban Modeling from City-Scale Aerial LiDAR Data.” PhD Dissertation, University of Southern California. http://qianyi.info/docs/papers/usc_thesis.pdf.

Zhou, Qian-Yi. 2017. Automatic 3D Urban Modeling Software. Windows. University of Southern California. https://github.com/qianyizh/UrbanReconstruction.

Appendix A: Calibration of Dual Contouring Parameters

The 2.5D Dual Contouring method used in this study (see section 4.3) has numerous user-defined parameters, of which a few must be set to appropriate values for a given input point cloud to maximize output accuracy. Three parameters, neighbourhood requirement n_r, neighbourhood distance d_n, and large patch threshold t_{LP}, are relevant to point cloud classification. The fourth, dual contouring grid length l_g, controls the initial grid length for building reconstruction once the point cloud has been classified.

Neighbourhood requirement n_r sets the number of neighbouring points used to calculate per-point neighbourhood metrics during the initial classification pass, which uses each point's n_r closest neighbours. These neighbourhood metrics contribute to a factor analysis which determines whether a given point likely lies on a surface with its neighbours, in which case it is on the ground or a building; if it lies in a diffuse cloud of points, in which case it represents vegetation, or if it is isolated from its neighbours, in which case it is classified as noise. The neighbourhood requirement must be set low enough that only a point's immediate neighbours are used to calculate the metrics; set too high, distant neighbours will contribute to the metrics used for point classification, resulting misclassification of points. At the same time, n_r must be set high enough that a sufficient number of points are included in the metric calculations so as to well represent the neighbourhood of the point in question. The neighbourhood requirement is thus set highest for high-density point clouds, and decreases as subsampling factor *n* increases, to a minimum of 2 for the *n=30* subset. Since near neighbours are required for accurate point classification, the neighbourhood-based point classification method imposes a lower bound on the density of point clouds used for this method; below a certain density, the immediate neighbourhood of each point is so sparse that even the nearest neighbor frequently lies on an entirely different surface plane, making calculating accurate neighbourhood metrics impossible. Appropriate values for each subset were determined experimentally; this was relatively straightforward since n_r's effect on output accuracy is not strong except at low density levels.

The Large Patch Threshold t_{LP} gives the number of points necessary for a point patch to be classified as 'large'. Such patches are treated differently when the algorithm separates building patches from ground patches; the threshold for a large point patch to be classified as part of the ground based on its distance to a ground patch is shorter than for small point patches. Since t_{LP} is a number of points it can be roughly equated to area for point clouds of uniform density. This allows us to set it for each subsample such that it corresponds to an area of constant size by dividing the value for the full-resolution dataset, 800 (corresponding to an area of 40 sq. m), by the subsampling factor n, as shown in Equation (7)

$$T_{LP_n} = \frac{800}{n} \qquad (7)$$

The neighbourhood distance d_n controls the effect of distance on the region growing algorithm used to separate building and ground points into patches. It is highly sensitive to point cloud density and must therefore be carefully calibrated for each point cloud subsample. Of the three sets of DC parameter tuning experiments performed, the last primarily concerned d_n, optimal values three parameters having been (for the most part) having been found previously. Calibration of d_n was performed iteratively by repeatedly reconstructing a spatial subset of the Bridgeview region using different values of d_n for each subset, then comparing the results of reconstruction using 2D accuracy assessment.

Three sets of calibration experiments were performed. The first set established preliminary values of d_n using an iterative process aimed at optimizing the value of Q for the output of each subset. The second set of experiments involved various parameters and determined their effect, or non-effect, on the products of the DC process. The third set of experiments, like the first, aimed to optimize values of d_n, neighbourhood distance having been established as the most density-sensitive parameter of those tested. Unlike the first series, output quality in the third series was judged using all three metrics (completeness, correctness and quality), and used the values for other parameters determined over the course of the second series. Table A shows nine of the ten rounds of the third series of calibration experiments. Note that Round 4, an experiment which reduced the area corresponding to t_{LP} from 40 m^2 to 30 m^2 is omitted as did not result in any detectable difference from Round 3. Also notable

is that the neighbourhood requirement for subset n=7 was lowered from 10 to 8 for Round 8; the change resulted in slightly improved accuracy and so was retained for subsequent rounds and the final parameter set. Of the values of d_n shown in Round 10, all but one were used as the final values for their corresponding subset. The exception was for n=30, for which d_n was set to 3.9, between the values for Round 6 and Round 9 where highest completeness scores were achieved. Note the presence of a subset n=40; data from this subset's extraction was discarded due to poor performance in both methods, but particularly for the ENVI+TIN method, which produced no buildings from that subset.

Table A

	Round 1				**Round 2**				**Round 3**			
N	ND	Correctness	Completeness	Q	ND	Correctness	Completeness	Q	ND	Correctness	Completeness	Q
1	0.6	96.97%	79.80%	77.86%	0.55	94.77%	82.52%	78.93%	0.58	96.71%	80.17%	78.05%
2	0.71	93.89%	85.98%	81.42%	0.75	95.44%	80.53%	77.55%	0.73	94.99%	81.51%	78.15%
3	0.78	94.68%	85.74%	81.80%	0.8	94.82%	83.63%	79.98%	0.81	95.74%	83.58%	80.58%
4	0.9	91.10%	83.53%	77.23%	1	96.16%	77.76%	75.42%	0.95	94.42%	81.70%	77.94%
5	1	90.85%	83.83%	77.30%	1.1	93.48%	79.84%	75.63%	1.05	93.61%	81.62%	77.31%
6	1	89.36%	83.54%	75.98%	1.2	92.15%	80.65%	75.46%	1.15	90.97%	81.63%	75.51%
7	1.5	90.57%	76.32%	70.70%	1.4	89.54%	77.23%	70.84%	1.3	89.23%	77.23%	70.64%
8	1.7	97.85%	65.28%	64.35%	1.6	97.86%	66.73%	65.77%	1.5	97.15%	68.53%	67.18%
10	1.25	86.80%	82.71%	73.47%	1.8	98.29%	64.86%	64.13%	1.6	97.22%	69.89%	68.52%
15	1.45	71.31%	84.05%	62.81%	2	96.02%	64.19%	62.53%	1.8	91.43%	74.77%	69.87%
20	2	84.28%	69.74%	61.71%	2.2	89.43%	63.25%	58.85%	2	84.28%	69.74%	61.71%
30	2.5	57.66%	59.69%	41.50%	2.7	56.76%	56.96%	39.72%	2.2	55.53%	59.65%	40.37%
40	2.5	41.02%	67.86%	34.35%	3	43.89%	51.41%	31.02%	2.5	41.02%	67.86%	34.35%
	Round 5				**Round 6**				**Round 7**			
N	ND	Correctness	Completeness	Q	ND	Correctness	Completeness	Q	ND	Correctness	Completeness	Q
1	0.56	95.11%	82.72%	79.34%	0.56	95.11%	82.72%	79.34%	0.56	95.11%	82.72%	79.34%
2	0.72	94.56%	82.97%	79.19%	0.73	95.00%	82.18%	78.78%	0.73	95.00%	82.18%	78.78%
3	0.82	95.01%	84.55%	80.95%	0.81	95.18%	84.49%	81.02%	0.81	95.18%	84.49%	81.02%
4	0.96	94.59%	81.58%	77.95%	0.97	95.20%	81.12%	77.93%	0.97	95.20%	81.12%	77.93%
5	1.07	93.00%	81.03%	76.38%	1.2	95.65%	74.90%	72.43%	1.17	95.50%	76.10%	73.46%
6	1.1	90.99%	83.20%	76.86%	1.35	95.27%	77.15%	74.30%	1.35	95.27%	77.15%	74.30%
7	1.15	87.91%	80.57%	72.54%	1.7	90.13%	75.13%	69.42%	1.4	89.13%	77.50%	70.80%
8	1.4	95.59%	70.89%	68.64%	1.5	96.91%	68.66%	67.19%	1.45	95.67%	70.64%	68.45%
10	1.5	95.89%	72.56%	70.37%	1.5	95.89%	72.56%	70.37%	1.5	95.89%	72.56%	70.37%
15	2	96.01%	64.27%	62.60%	2.15	95.52%	51.00%	49.81%	2.15	95.52%	51.00%	49.81%
20	2.5	94.29%	55.44%	53.64%	2.6	94.29%	54.73%	52.97%	2.55	94.29%	54.73%	52.97%
30	3	74.92%	42.03%	36.84%	4	84.34%	35.74%	33.52%	3	74.92%	42.03%	36.84%
40	4	49.77%	42.37%	29.68%	6	13.24%	0.29%	0.29%	3.5	47.09%	45.92%	30.29%
	Round 8				**Round 9**				**Round 10**			
N	ND	Correctness	Completeness	Q	ND	Correctness	Completeness	Q	ND	Correctness	Completeness	Q
1	0.56	95.11%	82.72%	79.34%	0.56	95.11%	82.72%	79.34%	0.56	95.11%	82.72%	79.34%
2	0.73	95.00%	82.18%	78.78%	0.73	95.00%	82.18%	78.78%	0.73	95.00%	82.18%	78.78%
3	0.81	95.18%	84.49%	81.02%	0.81	95.18%	84.49%	81.02%	0.81	95.18%	84.49%	81.02%
4	0.97	95.20%	81.12%	77.93%	0.97	95.20%	81.12%	77.93%	0.97	95.20%	81.12%	77.93%
5	1.17	95.50%	76.10%	73.46%	1.17	95.50%	76.10%	73.46%	1.17	95.50%	76.10%	73.46%
6	1.35	95.27%	77.15%	74.30%	1.35	95.27%	77.15%	74.30%	1.35	95.27%	77.15%	74.30%
7	1.4	95.18%	76.25%	73.42%	1.4	95.18%	76.25%	73.42%	1.4	95.18%	76.25%	73.42%
8	1.45	95.67%	70.64%	68.45%	1.45	95.67%	70.64%	68.45%	1.45	95.67%	70.64%	68.45%
10	1.5	95.89%	72.56%	70.37%	1.5	95.89%	72.56%	70.37%	1.5	95.89%	72.56%	70.37%
15	2.15	95.52%	51.00%	49.81%	2.15	95.52%	51.00%	49.81%	2.15	95.52%	51.00%	49.81%
20	2.7	93.29%	45.50%	44.06%	2.4	88.13%	60.54%	55.98%	2.6	94.29%	54.73%	52.97%
30	4.5	81.60%	30.45%	28.49%	3.75	83.44%	36.27%	33.83%	4.3	81.82%	30.96%	28.97%
40	5	19.07%	0.77%	0.75%	3.8	49.10%	43.97%	30.20%	4.5	49.97%	41.32%	29.23%

Appendix B: ArcGIS Python Script

The script used to convert ENVI footprint and classified point cloud outputs into 3D models, perform 2D accuracy assessment on the outputs of both methods, and create the table of model centroids used to position each model in local coordinates for comparison.

```
import arcpy
import os
import time
import csv
import datetime

rootFolder = arcpy.GetParameterAsText(0) #Root folder for LiDAR subsets as exported from ENVI
buildingfootprintsIn=arcpy.GetParameterAsText(1) #Input verification footprints
buildingfootprintsOut=arcpy.GetParameterAsText(2) #Output feature class with information on footprint matches and corresponding point densities
dcmodelgdb = arcpy.GetParameterAsText(3) #Input GDB containing DC MultiPatches
tinmodelgdb = arcpy.GetParameterAsText(4) #Input GDB containing TIN MultiPatches
modelfolder = arcpy.GetParameterAsText(5) #folder to contain output DAE files
outgdb = arcpy.GetParameterAsText(6) #Output GDB
SectorCode = arcpy.GetParameterAsText(7) #Sector code, e.g. 'BVW' for Bridgeview
DCmodelOrigCoordSys = arcpy.GetParameterAsText(8) #
DCModelRootFolder = arcpy.GetParameterAsText(9) #Folder in which input Dual Contouring models are located
BoundaryPoly = arcpy.GetParameterAsText(10) #Polygon boundary within which processing will be constrained
deleteflag = arcpy.GetParameter(11) #If Deleteflag is True, the contents of the destination files will be deleted before processing starts (Note: may not work for excel files)
testflag = arcpy.GetParameter(12) #testflag indicates that certain loops should terminate after the first run, making debugging faster if True
#Raster area analysis settings
rootworkspace = os.path.dirname(outgdb)
arcpy.env.workspace=rootworkspace
arcpy.env.scratchWorkspace = rootworkspace
endmessages = [] #Creates an empty array to which warning messages may be appended for display at the end of processing
rasterRes = 0.1 #raster analysis resolution
startTime = time.time() # mark off start time for messages
startdatetime = datetime.datetime.today() #sets start datetime for log file

#start storing log messages, starting with the initialization settings
logmessages = []
initsettings = ["---INIT SETTINGS---","rootFolder: {}".format(rootFolder), "buildingfootprintsIn: {}".format(buildingfootprintsIn), "buildingfootprintsOut: {}".format(buildingfootprintsOut), "dcmodelgdb: {}".format(dcmodelgdb), "tinmodelgdb: {}".format(tinmodelgdb), "modelfolder: {}".format(modelfolder), "outgdb: {}".format(outgdb), "SectorCode: {}".format(SectorCode),"DCmodelOrigCoordSys: {}".format(DCmodelOrigCoordSys), "DCModelRootFolder: {}".format(DCModelRootFolder)]
logmessages.extend(initsettings)
logmessages.extend([" ","---SCRIPT START---"])

def LIST_SHIFT_COORDs(dcGdb, tinGdb, outmodelfolder):
#ArcGIS outputs COLLADA files in metric with a local coordinate system with the origin at the centroid of the multipatch. Therefore we need to identify the coordinates of the centroid of each model and then translate them to the correct position in the LCS using a script in Blender. We do so by using the calculate geometry feature to calc the coordinates, then exporting a .csv file with a row for each model.
#WARNING: Sometimes this script fails at the CalculateField stage, probably because of problems with the input geometry that RepairGeometry does not appear to solve. ArcGIS provides no facility to handle errors like this, so in this case the user will have to run the 'Calculate Geometry' tool themselves in the allMPCs table and then export it after the script crashes.
        tempTinMPCs = os.path.join(outgdb,arcpy.ValidateTableName('TinMPCs',outgdb))
```

```
        tempDCMPCs =  os.path.join(outgdb,arcpy.ValidateTableName('DCMPCs',outgdb))
        allMPCs = os.path.join(outgdb,arcpy.ValidateTableName('AllMPCs',outgdb))
        oldworkspace = arcpy.env.workspace
        #list and merge DC MPCs in preparation for final merge
        arcpy.env.workspace=dcGdb
        dcMPCs = arcpy.ListFeatureClasses(feature_type="Multipatch")
        arcpy.Merge_management(dcMPCs,tempDCMPCs)
        #list and merge TIN MPCs in preparation for final merge
        arcpy.env.workspace=tinGdb
        tinMPCs = arcpy.ListFeatureClasses(feature_type="Multipatch")
        arcpy.Merge_management(tinMPCs,tempTinMPCs)
        arcpy.env.workspace=oldworkspace
        #Need to create field mappings so that the name of each modeled MPC is properly represented in the same field
        fm_modelName = arcpy.FieldMap()
        fMaps = arcpy.FieldMappings()
        fm_modelName.addInputField(tempTinMPCs,'FullModelName')
        fm_modelName.addInputField(tempDCMPCs,'ModelNameField')
        model_name = fm_modelName.outputField
        model_name.name='ModelName'
        fMaps.addFieldMap(fm_modelName)
        #Merge TIN and DC MPCs into a single Multipatch feature class (MPC). Need to have merged into 1 DC MPC and 1 TIN MPC previously for the field mappings to work properly.
        oldcoordsys = arcpy.env.outputCoordinateSystem
                arcpy.env.outputCoordinateSystem = DCmodelOrigCoordSys #Set the merged MPCs to use the DC models' original coordinate system, ensuring the calculated centroid coordinates are in the desired LCS
        arcpy.Merge_management([tempTinMPCs,tempDCMPCs],allMPCs,fMaps)
        #Now dispose of temporary MPCs
        arcpy.Delete_management(tempTinMPCs)
        arcpy.Delete_management(tempDCMPCs)
        #Now add coordinate fields
        arcpy.AddField_management(allMPCs,'CentroidX','FLOAT')
        arcpy.AddField_management(allMPCs,'CentroidY','FLOAT')
        arcpy.AddField_management(allMPCs,'CentroidZ','FLOAT')
        #Calculate the centroid coordinates
        arcpy.RepairGeometry_management(allMPCs)
        arcpy.CalculateField_management(allMPCs,'CentroidX','!SHAPE.CENTROID.X!','PYTHON_9.3')
        arcpy.CalculateField_management(allMPCs,'CentroidY','!SHAPE.CENTROID.Y!','PYTHON_9.3')
        arcpy.CalculateField_management(allMPCs,'CentroidZ','!SHAPE.CENTROID.Z!','PYTHON_9.3')
        arcpy.env.outputCoordinateSystem = oldcoordsys #reset coordinate system
        shiftCoordsFile=os.path.join(outmodelfolder,'daeshiftcoords.csv')
        with open(shiftCoordsFile,'wb') as csvfile:
                csvwriter=csv.writer(csvfile,delimiter=',')
                with arcpy.da.SearchCursor(allMPCs,['FullModelName','CentroidX','CentroidY','CentroidZ']) as MPCTable:
                        for MPCrow in MPCTable:
                                if MPCrow[0] != 'None': #Skip models with no name assigned

        csvwriter.writerow([MPCrow[0],MPCrow[1],MPCrow[2],MPCrow[3]])
                                        if testflag:
                                                MESSAGE_USER("Model {} with X coord {}, Y coord {}, and Z coord {}".format(MPCrow[0],MPCrow[1],MPCrow[2],MPCrow[3]))

def DC_REFERENCE(modelRootFolder,refFootprints,modelOrigCoordSys,OutputRootFolder,outgdb):
        DCfolderList = os.listdir(modelRootFolder)
        refFootprintsPt = os.path.join(dcmodelgdb,arcpy.ValidateTableName('refFPPoints',dcmodelgdb))
        arcpy.FeatureToPoint_management(refFootprints, refFootprintsPt, 'INSIDE') #generates centre(ish) points for footprint matching
    #Exact algorith is a black box; setting 'CENTROID' generates points outside of the polygon in question for some polygons. Fortunately, ArcGIS's model output function exports models with the centroid as the origin.
        MESSAGE_USER('DEBUG: REF_FPPTs: {}'.format(refFootprintsPt))
        #so we need to use 'INSIDE', which appears to constrain the calculated
```

```
		#Note: Keeping the reference point feature class so it can be inspected
		if testflag:
			DCfolderList = DCfolderList[-6:-5]
		for DCfolder in DCfolderList:
			DCfolder = os.path.join(modelRootFolder,DCfolder)
			foldDesc = arcpy.Describe(DCfolder)
			MESSAGE_USER("Now processing DC models in folder: {}".format(foldDesc.baseName))
			DatasetCode = foldDesc.baseName[-2:]
			arcpy.AddMessage(modelRootFolder)
			arcpy.AddMessage('test')
			arcpy.env.workspace=outgdb
			arcpy.AddMessage(DCfolder)
			outMP =
os.path.join(outgdb,arcpy.ValidateTableName(os.path.basename('D'+DatasetCode+'_'+SectorCode), outgdb))
			arcpy.Delete_management(outMP) #delete MP in case it already exists
			arcpy.Import3DFiles_3d(in_files=DCfolder, out_featureClass=outMP,
root_per_feature="ONE_ROOT_ONE_FEATURE", spatial_reference=modelOrigCoordSys, y_is_up="Z_IS_UP",
file_suffix="*", in_featureClass="", symbol_field="")
			outMPdesc=arcpy.Describe(outMP)
			MESSAGE_USER("MultiPatch feature class created: {}".format(outMPdesc.baseName))
			tempfootprints = arcpy.ValidateTableName('tempfootprints',outgdb)
			tempjoin = arcpy.ValidateTableName('tempjoin',outgdb)
			arcpy.MultiPatchFootprint_3d(outMP,tempfootprints)
			#MESSAGE_USER("Multipatch footprint {} created".format(outMPdesc.baseName))
			#MESSAGE_USER("DEBUG: OUTGDB {}".format(outgdb))
			#MESSAGE_USER("DEBUG: TEMPFOOTRINTS {}".format(tempfootprints))
			#MESSAGE_USER("DEBUG: tempjoin {}".format(tempjoin))

		arcpy.SpatialJoin_analysis(tempfootprints,refFootprintsPt,tempjoin,'JOIN_ONE_TO_ONE','KEEP_ALL',
'#','CLOSEST','2')
			#the above Spatial Join attempts to match temp footprint polygons to the closest footprint point
within 2 meters of its boundary. The 5m threshold prevents the algorithm from matching distant, unrelated footprint
centers in the case of a false positive. In the typical case for a correct detection, the reference point is inside the subject
footprint and the distance is calculated as zero. A search radius is necessary however because the feature-to-point
algorithm may place the reference point on the boundary of it's respective polygon in cases where the actual centroid
lies outside the polygon.
			arcpy.JoinField_management(outMP,'OID',tempjoin,'TARGET_FID','ModelNameSuffix')
			arcpy.AddField_management (outMP, 'ModelNameField', 'TEXT')
			arcpy.CalculateField_management(outMP,
'ModelNameField','"D{0}_"+!ModelNameSuffix!'.format(DatasetCode),'PYTHON')
			arcpy.Delete_management(tempfootprints)
			arcpy.Delete_management(tempjoin)
			tempSelectExport = 'outTemp'
			arcpy.Select_analysis(outMP,tempSelectExport,'ModelNameSuffix IS NOT NULL')

		arcpy.MultipatchToCollada_conversion(tempSelectExport,OutputRootFolder,'PREPEND_NONE','ModelNa
meField')
			MESSAGE_USER("Multipatch features in {} exported to
COLLADA".format(outMPdesc.baseName))
			arcpy.Delete_management(tempSelectExport)
		arcpy.env.workspace = oldworkspace

def TIN_MODELING(ENVIFolderList,refFootprints,TINOutGDB,DAEOutFolder):
		refFootprintsPtTwo = os.path.join(tinmodelgdb,arcpy.ValidateTableName('refFPPoints',tinmodelgdb))
		arcpy.FeatureToPoint_management(refFootprints, refFootprintsPtTwo, 'INSIDE') #generates centre(ish)
points for footprint matching
		for ENVIFolder in ENVIFolderList:
			ENVIFoldDesc = arcpy.Describe(ENVIFolder)
			DatasetCode = ENVIFoldDesc.baseName[-2:]
			MESSAGE_USER("ENVI folder {} IDed as containing subset with code
{}".format(ENVIFolder,DatasetCode))
			perimPath=os.path.join(os.path.abspath(ENVIFolder),r'Products\buildings_perimeter.shp')
```

```
                    if not arcpy.Exists(perimPath):
                            MESSAGE_USER("No building perimeters found for folder {}".format(ENVIFolder))
                            break
                    tempLASD1 =
os.path.join(DAEOutFolder,arcpy.CreateUniqueName('templasd1.lasd',DAEOutFolder))
                    tempLASD2 =
os.path.join(DAEOutFolder,arcpy.CreateUniqueName('templasd2.lasd',DAEOutFolder))
                    tempLASFolder = 'tempLASFolder'
                    arcpy.CreateFolder_management(DAEOutFolder,tempLASFolder)
                    tempLASFolder = os.path.join(DAEOutFolder,tempLASFolder)
                    sourceLAS = filename = os.path.join(ENVIFolder,r'Products\PointClouds\pointCloud_000.las')
                    CREATE_LASD(sourceLAS,tempLASD1)
                    MESSAGE_USER("LAS dataset loaded")
                    laslayer = arcpy.CreateUniqueName('laslayer',DAEOutFolder)
                    arcpy.MakeLasDatasetLayer_management(tempLASD1, laslayer, 6)
                    MESSAGE_USER("LAS dataset filtered")
                    arcpy.ExtractLas_3d (laslayer, tempLASFolder, perimPath, perimPath, '', '', '', '', '', tempLASD2)
                    MESSAGE_USER("LAS point data within detected footprints extracted")
                    arcpy.management.Delete(laslayer)
                    arcpy.Delete_management(tempLASD1)
                    unAgRast=arcpy.CreateUniqueName("RoofUnAgTemp")
                    AgRastName=arcpy.CreateUniqueName("RoofAgTemp")
                    arcpy.LasDatasetToRaster_conversion (tempLASD2, unAgRast, "ELEVATION", "BINNING
MAXIMUM LINEAR", "FLOAT", "CELLSIZE", "0.1", "1")
                    MESSAGE_USER("Roof points converted to raster")
                    AgRast=arcpy.sa.Aggregate(unAgRast, 5, "MAXIMUM")
                    MESSAGE_USER("Roof elevation raster aggregation complete")
                    arcpy.Delete_management(unAgRast)
                    AgRast.save(AgRastName)
                    RoofTIN = os.path.join(DAEOutFolder,arcpy.CreateUniqueName('roofTIN'))
                    arcpy.RasterTin_3d (AgRastName, RoofTIN, 0.01)
                    MESSAGE_USER("Roof TIN generated.")
                    #arcpy.Delete_management(AgRastName)
                    DEMPath=os.path.join(ENVIFolder,r'Products\dem.tif')
                    if not arcpy.Exists(DEMPath):
                            DEMPath=os.path.join(ENVIFolder,r'Products\dem.dat')
                    GroundTIN=os.path.join(DAEOutFolder,arcpy.CreateUniqueName('groundTIN'))
                    arcpy.RasterTin_3d(DEMPath,GroundTIN,0.01)
                    MESSAGE_USER("Ground TIN generated.")
                    outMP = os.path.join(tinmodelgdb,'T'+DatasetCode+'_'+SectorCode)
                    arcpy.Delete_management(outMP) #delete MP in case it already exists
                    arcpy.ExtrudeBetween_3d(RoofTIN, GroundTIN, perimPath, outMP)
                    MESSAGE_USER("Models extruded to multipatch feature
class".format('T'+DatasetCode+'_'+SectorCode))
                    #arcpy.Delete_management(GroundTIN)
                    #arcpy.Delete_management(RoofTIN)
                    oldworkspace=arcpy.env.workspace
                    arcpy.env.workspace=TINOutGDB
                    tempfootprints = arcpy.ValidateTableName('tempfootprints',TINOutGDB)
                    tempjoin = arcpy.ValidateTableName('tempjoin',TINOutGDB)
                    arcpy.MultiPatchFootprint_3d(outMP,tempfootprints)

            arcpy.SpatialJoin_analysis(tempfootprints,refFootprintsPtTwo,tempjoin,'JOIN_ONE_TO_ONE','KEEP_CO
MMON', '#','CLOSEST','2')
                    arcpy.JoinField_management(outMP,'OBJECTID',tempjoin,'TARGET_FID','ModelNameSuffix')
                    arcpy.AddField_management (outMP, 'FullModelName', 'TEXT')
                    arcpy.CalculateField_management(outMP,
'FullModelName','"T{0}_"+!ModelNameSuffix!'.format(DatasetCode),'PYTHON')
                    MESSAGE_USER("TIN model referencing complete for models in feature class
{}".format('T'+DatasetCode+'_'+SectorCode))
                    arcpy.Delete_management(tempfootprints)
                    arcpy.Delete_management(tempjoin)
```

```
                tempSelectExport = 'outTemp'
                arcpy.Select_analysis(outMP,tempSelectExport,'ModelNameSuffix IS NOT NULL')
                try:

        arcpy.MultipatchToCollada_conversion(tempSelectExport,DAEOutFolder,'PREPEND_NONE','FullModelN
ame')
                except:
                        MESSAGE_USER("COLLADA export error on T{}, check outputs
manually".format(DatasetCode))
                        endmessages.append("COLLADA export error on T{}".format(DatasetCode))
                MESSAGE_USER("TIN models in MP feature class {} exported to
COLLADA".format('T'+DatasetCode+'_'+SectorCode))
                arcpy.Delete_management(tempSelectExport)
                arcpy.env.workspace = oldworkspace

def DELETE_EXISTING():
        oldworkspace = arcpy.env.workspace
        arcpy.env.workspace = rootworkspace
        walk = arcpy.da.Walk(arcpy.env.workspace)
        for dirpath, dirnames, filenames in walk:
                for filename in filenames:
                        #MESSAGE_USER("Attempting to delete file: {}".format(filename))
                        arcpy.Delete_management(os.path.join(dirpath,filename))
                        #MESSAGE_USER("{} deleted".format(filename))
        for root, dirs, files in os.walk(modelfolder):
                for file in files:
                        os.remove(os.path.join(root,file))

def MESSAGE_USER(message): #quick funtion to send messages with a timestamp, and store them for writing in the
log file
        currtime = time.time()-startTime
        fullmessage = ('%09.2fs: %s' % (currtime, message))
        arcpy.AddMessage(fullmessage)
        logmessages.append(fullmessage)

def CREATE_LASD(fname,outLASDat): #using this function simplifies lasdat creation by supplying a default
argument for projection files
        arcpy.CreateLasDataset_management (fname, outLASDat, 'NO_RECURSION', '', '', '', 'NO_FILES')

def FP_IDENTIFY(fpIn): #a simple function to id the series and subset codes of a given model footprint feature class
        fpDescribe = arcpy.Describe(fpIn) #create a describe object for the fp
        fpSerCode = fpDescribe.baseName[0:3] #cut out the part of the basename corresponding to the series code
(e.g. 'D02')
        fpSubCode = fpSerCode[1:3] #get the sub code from the last two chars in the series code (e.g. '02' from
'D02')
        return (fpSerCode,fpSubCode) #return both values, series code first

def FOLDER_IDENTIFY(folderin): #similar to FP_IDENTIFY, but returns only one value, the subset code
        folderDescribe=arcpy.Describe(folderin)
        folderSubCode = folderDescribe.baseName[-2:]
        return folderSubCode;

#MODEL_PROCESSOR makes and names model footprints and attaches their area, then appends each FP file to a list.
Runs only for DC, we use ENVI_FP_PROC for ENVI footprints
def MODEL_PROCESSOR(inWorkspace, fplist):
        modelist = []
        DCWalk = arcpy.da.Walk(inWorkspace)
        for dirpath, dirnames, filenames in DCWalk:
                for filename in filenames:
                        MESSAGE_USER(filename)
                        modelist.append(os.path.join(inWorkspace, filename))
                        MESSAGE_USER("DEBUG - APPENDED {}".format(filename))
```

```
	for modelfc in modelist:
		modelfcdesc = arcpy.Describe(modelfc)
		modelfcname = modelfcdesc.baseName
		MESSAGE_USER("DEBUG - MODEL_PROCESSOR ITERATING ON
".format(modelfcdesc.baseName))
		if modelfcdesc.shapeType == "MultiPatch":
			modelfcname = modelfcdesc.baseName
			MESSAGE_USER("Processing multipatch feature set
{}".format(modelfcdesc.baseName))
			outFPName = os.path.join(outgdb,modelfcname[0:3] + '_FP')
			outFPNameTemp = os.path.join(outgdb,modelfcname[0:3] + '_FPTEMP')
			arcpy.MultiPatchFootprint_3d(modelfc, outFPNameTemp)
			arcpy.Clip_analysis(outFPNameTemp,BoundaryPoly,outFPName) #need to clip out any
detected buildings not in the study area
			arcpy.Delete_management(outFPNameTemp)
			fpareafieldname = 'M_'+modelfcname[0:3]+'_A'
			arcpy.AddField_management(outFPName,fpareafieldname,'DOUBLE')

	arcpy.CalculateField_management(outFPName,fpareafieldname,'!shape.area@SquareMeters!','PYTHON')
			fplist.append(outFPName)
			MESSAGE_USER("Processing complete for multipatch feature set
{}".format(modelfcname))
		else:
			modelfcdesc.basename
			MESSAGE_USER("DEBUG - FC {} classified as non
MultiPatch".format(modelfcname))

def ENVI_FP_PROC(fplist, foldlist): #need to process ENVI footprints seperately since we want all of them, not just
those selected for modeling
	for folder in foldlist:
		subCode = FOLDER_IDENTIFY(folder)
		fpPath = os.path.join(folder,r'Products\buildings_perimeter.shp')
		if arcpy.Exists(fpPath):
			fpDest = os.path.join(outgdb,'T'+subCode+'_FP') #create naming convention conforming
FP dataset name in output GDB
			fpDestTemp = os.path.join(outgdb,'T'+subCode+'_FPTEMP')
			arcpy.CopyFeatures_management(fpPath,fpDestTemp) #copy the features to GDB
			arcpy.RepairGeometry_management(fpDestTemp) #The building perimeter polygons
output by ENVI sometimes have invalid topological qualities (e.g. self intersection)
			#Running RepairGeometry corrects these and allows the boundary clip operation to
procede without errors.
			arcpy.Clip_analysis(fpDestTemp,BoundaryPoly,fpDest)
			fplist.append(fpDest) #append the FP to the fp list for analysis
			MESSAGE_USER("ENVI footprints for subset {} identified and moved to
{}".format(subCode,fpPath))

#FP_PROCESSOR calculates point count transfers it to a new, permanent field, as well as calculating local point cloud
density using the built in Shape.Area field. Note that arcpy.LasPointStatsByArea_3d works by appending the specified
field onto the subject polygon table directly, overwriting if one already exists. This necessitates adding a new field for
each LAS dataset.
def FP_PROCESSOR(fplist,folder_list):
	MESSAGE_USER("Per-MFP point cloud density processing initialized")
	for fp in fplist:
		fpDesc = arcpy.Describe(fp)
		(fpSeriesCode,fpSubsetCode) = FP_IDENTIFY(fp) #call fp identify to get the needed identifier
codes
		MESSAGE_USER("Target FP {} identified as having series code {} and subset code
{}".format(fpDesc.baseName,fpSeriesCode,fpSubsetCode))
		foundFolder = False
		for folder in folder_list:
			folderdesc=arcpy.Describe(folder)
			folderPCDens = FOLDER_IDENTIFY(folder)
```

```
                                if folderPCDens == fpSubsetCode: #check if is same subset as fp; fortunately we only
need one LAS dataset this time
                                        fieldname = 'PTS_'+str(fpSeriesCode)
                                        arcpy.AddField_management(fp,fieldname,'DOUBLE')
                                        filename = folder+r'\Products\PointClouds\pointCloud_000.las'
                                        MESSAGE_USER("Target MFP {} matched to point cloud in folder {}
".format(fpDesc.baseName,folderdesc.baseName))
                                        tempLASD = os.path.join(modelfolder,'templasd'+fpSubsetCode+'.lasd')
                                        CREATE_LASD(filename,tempLASD)
                                        arcpy.LasPointStatsByArea_3d(tempLASD,fp,'POINT_COUNT')
                                        MESSAGE_USER("Point density information for MFP {}
extracted".format(fpDesc.baseName))
                                        densfieldname = 'PTD_' + str(fpSeriesCode)
                                        arcpy.AddField_management(fp,densfieldname,'DOUBLE')
                                        arcpy.CalculateField_management(fp,fieldname,'!PointCount!','PYTHON')

                arcpy.CalculateField_management(fp,densfieldname,'!PointCount!/!Shape.Area@SquareMeters!','PYTHON'
)
                                        MESSAGE_USER("Point density information for MFP {} appended to MFP
table".format(fpDesc.baseName))
                                        foundFolder = True
                                        break #break out of the loop once the correct folder has been found
                        if not foundFolder:
                                MESSAGE_USER("No match found for FP {}".format(fpSeriesCode))
                        arcpy.DeleteField_management(fp,'POINT_COUNT') #delete remnant POINT_COUNT field to
avoid any conflicts later
                MESSAGE_USER("Per-MFP point cloud density processing completed")

#FP_JOIN does two jobs
#               1 - flags verif polygons if they have a match in each of the generated polygons
#               2 - attaches calculated point counts and densities for each corresponding generated polygon (not that only
first join value is attached)
def FP_JOIN(fplist,buildingfootprintsOut):
                MESSAGE_USER("Footprint join process initiated.")
                VerifPolysTempCopy=os.path.join(outgdb,arcpy.ValidateTableName('cleanVFP_temp',outgdb)) #create a
'clean' backup VFP feature class to avoid a feedback loop in which attributes joined to the main VFP carry over through
the spatial join and are re-added each loop
                arcpy.CopyFeatures_management(buildingfootprintsOut,VerifPolysTempCopy)
                for fp in fplist:
                        #VerifPolysTempCopy2=VerifPolysTempCopy=os.path.join(outgdb,'cleanVFP_temp2')
                        #arcpy.CopyFeatures_management(buildingfootprintsOut,VerifPolysTempCopy2) need to create a
-second- clean VFP as a source for the spatial join, otherwise feedback loop still exists
                        fpdesc=arcpy.Describe(fp)
                        MESSAGE_USER("MFP file {} selected for analysis".format(fpdesc.baseName))
                        tempSJoin= os.path.join(outgdb,arcpy.ValidateTableName('temp_SJoin'+fpdesc.baseName))

                arcpy.SpatialJoin_analysis(VerifPolysTempCopy,fp,tempSJoin,'JOIN_ONE_TO_ONE','KEEP_COMMON',"
,'INTERSECT') #only keep matches
                        #first task is to flag verif fp as detected if footprint fp contains at least one spatial match
                        anyintersectname = arcpy.ValidateFieldName('AnyIntrsct_'+fpdesc.baseName[0:3], outgdb)
#create a field for anyintersect for the coresponding generated model set
                        arcpy.AddField_management(buildingfootprintsOut,anyintersectname,'TEXT')

                        targetfields = ['OBJECTID',str(anyintersectname)]
                        with arcpy.da.UpdateCursor(buildingfootprintsOut,targetfields) as fpcursor:
                                with arcpy.da.SearchCursor(tempSJoin,['TARGET_FID']) as sjcursor: #look only at
target id to check if it is same as fp OID
                                        for fprow in fpcursor:
                                                sjcursor.reset() #IMPORTANT: inner cursor 'remembers' its position
and therefore must be reset for each iteration of the outer; otherwise it will resume where it left off, which may be at the
end of the table
                                                fprow[1]='FALSE' #initialize anyintersect val as false
```

```
                                        for sjrow in sjcursor:
                                                if int(sjrow[0]) == int(fprow[0]): #check if TID is same as
fp OID for each row of sjcursor
                                                        fprow[1] = 'TRUE'
                                                        break #stop looking for TID matches
                                        fpcursor.updateRow(fprow) #update verif fp entry, flagging as
detected if anyintersect = 'TRUE'S, 'FALSE' if not
                MESSAGE_USER("MFP to VFP matching completed for MFP {}".format(fpdesc.baseName))
                #this part joins local point cloud count and density for each footprint
                joinFields=arcpy.ListFields(tempSJoin)
                #MESSAGE_USER("DEBUG fields listed")
                fieldstojoin = [] #initialize empty list of fields to join
                #MESSAGE_USER("DEBUG empty list created")
                for field in joinFields: #go through list of fields and check if they have one of the desired prefixes
(from FP_PROCESSOR)
                        #MESSAGE_USER("DEBUG checking "+field.name)
                        if field.name[0:4]=='PTS_':
                                fieldstojoin.append(field.name)
                                MESSAGE_USER("Field name appended: "+field.name)
                        elif field.name[0:4]=='PTD_':
                                fieldstojoin.append(field.name)
                                MESSAGE_USER("Field name appended: "+field.name)
                if fieldstojoin !=[]: #join only if there are field matches

        arcpy.JoinField_management(buildingfootprintsOut,'OBJECTID',tempSJoin,'TARGET_FID',fieldstojoin)
                        MESSAGE_USER("Field join performed for {}".format(fpdesc.baseName))
                MESSAGE_USER("Footprint join process finished for MFP {}".format(fpdesc.baseName))
        arcpy.Delete_management(VerifPolysTempCopy) #delete the temporary VFP FC after we're done with it
        MESSAGE_USER("Footprint join process completed.")

#This function calculates point counts and density for each verif footprint, from each LAS subset (based off of ENVI
subsets since DC subsets are translated in XY plane to make them more manageable in 3D modeling software (coords
have too many sig. figures when in UTM)
def VERIF_PROCESSOR(folderlist,buildingfootprintsOut):
        MESSAGE_USER("Verification footprint processing initiated")
        for folder in folderlist:
                foldesc=arcpy.Describe(folder)
                MESSAGE_USER("Folder IDed as :" + foldesc.file)
                ptcSuffix = foldesc.file[-2:]
                MESSAGE_USER('Point Cloud Code IDed as: {}'.format(ptcSuffix))
                filename = folder+r'\Products\PointClouds\pointCloud_000.las'
                #LASDesc=arcpy.Describe(filename)
                MESSAGE_USER('Point Cloud at {} IDed'.format(filename))
                tempLASD = os.path.join(modelfolder,'LASD_' + ptcSuffix + '.lasd')
                CREATE_LASD(filename,tempLASD)
                tempLASDesc = arcpy.Describe(tempLASD)
                MESSAGE_USER('LASD named as: {}'.format(tempLASDesc.file))
                MESSAGE_USER("Calculating VFP point counts for subset: {}".format(ptcSuffix))
                arcpy.LasPointStatsByArea_3d(tempLASD,buildingfootprintsOut,'POINT_COUNT')
                MESSAGE_USER("Point point counts calculated")
                fieldname = 'VPPointCount_'+str(ptcSuffix)
                arcpy.AddField_management(buildingfootprintsOut,fieldname,'LONG')
        arcpy.CalculateField_management(buildingfootprintsOut,fieldname,'!PointCount!','PYTHON')
                densfieldname = 'VPPointDens' + str(ptcSuffix)
        arcpy.AddField_management(buildingfootprintsOut,densfieldname,'DOUBLE')
        arcpy.CalculateField_management(buildingfootprintsOut,densfieldname,'!PointCount!/!Shape.Area@Square
Meters!','PYTHON')
                arcpy.DeleteField_management(buildingfootprintsOut,'PointCount')
                arcpy.Delete_management(tempLASD)
                MESSAGE_USER("VFP Point count and density calculated for subset: {}".format(ptcSuffix))
        MESSAGE_USER("Verification footprint processing initiated")
```

```
def AREA_ANALYSIS(folderlist, verifFPs, fplist):
	MESSAGE_USER("Area Analysis initiated.")
	oldSnap = arcpy.env.snapRaster #store old snap raster, will restore later
	def POLY_RASTERIZE(inPoly,outRast):
	#Simple function to produce the desired identity raster from a given input polygon dataset.
	#Takes a polygon dataset, produces a raster with a value of '1' where there is a building and '0' where there isn't.
		defaultZFlagVal = arcpy.env.outputZFlag
		defaultMFlagVal = arcpy.env.outputMFlag
		arcpy.env.outputZFlag="Disabled"
		arcpy.env.outputMFlag="Disabled"
		predispoly=os.path.join(outgdb,arcpy.ValidateTableName('TempFP',outgdb))
		arcpy.CopyFeatures_management(inPoly,predispoly)
		inpolyName = arcpy.Describe(inPoly).baseName
		MESSAGE_USER("Rasterizing polygon {}".format(inpolyName))
		disPoly = os.path.join(outgdb,arcpy.ValidateTableName('polygon_dissolve'))
		arcpy.Dissolve_management(predispoly, disPoly) #dissolve the input polygons
		tempRast=os.path.join(outgdb,arcpy.ValidateTableName('TEMPRAST'))
		arcpy.PolygonToRaster_conversion(disPoly,'OBJECTID',tempRast,"","",rasterRes)
		outRastobj = ~arcpy.sa.IsNull(tempRast) #isnull returns 0 where the raster isnt null and 1 where it is, the ~ operator flips this
		outRastobj.save(outRast)
		#delete temporary files
		arcpy.Delete_management(tempRast)
		arcpy.Delete_management(disPoly)
		arcpy.Delete_management(predispoly)
		MESSAGE_USER("Rasterization of polygon {} complete".format(inpolyName))
		arcpy.env.outputZFlag=defaultZFlagVal
		arcpy.env.outputMFlag = defaultMFlagVal
	#rasterize verif polys first
	MESSAGE_USER("Rasterizing VFPs")
	verifRast = os.path.join(outgdb,'VRT')
	POLY_RASTERIZE(verifFPs,verifRast)
	MESSAGE_USER("VFPs rasterized")
	arcpy.env.extent = verifRast
	arcpy.env.snapRaster = verifRast #set verif rast as the snap raster, ensuring the other rasters are aligned to it
	csvpath = os.path.join(modelfolder,'classificationAreas.csv')
	csvfile = open(csvpath,'wb') #create a csv file to which classification accuracy stats may be copied
	csvwriter=csv.writer(csvfile,delimiter=',',quotechar='|',quoting=csv.QUOTE_MINIMAL)
	csvwriter.writerow(['Sector_Code','Series_Code','Subset_Code','True_Positive','False_Negative','False_Positive','True_Negative'])
	for folder in folderlist:
		folderSubCode = FOLDER_IDENTIFY(folder)
		MESSAGE_USER("Folder for subset {} identified".format(folderSubCode))
		tempLASD = os.path.join(modelfolder,'TEMP_ARAN_'+folderSubCode+'.lasd') #name temp lasd
		filename = folder+r'\Products\PointClouds\pointCloud_000.las'
		CREATE_LASD(filename,tempLASD) #Want to generate LASD stats at folder level since we can use the same one for each subsampling level
		lasStatsRast = os.path.join(outgdb,arcpy.ValidateTableName('TARANLS'+folderSubCode))
		MESSAGE_USER("Calculating 1x1m point density raster for subset {}".format(folderSubCode))
		arcpy.LasPointStatsAsRaster_management(tempLASD, lasStatsRast, 'POINT_COUNT', 'CELLSIZE', '1')
		LasStatsObj = arcpy.Raster(lasStatsRast)
		lasstatsCorRastObj = arcpy.sa.Con(arcpy.sa.IsNull(LasStatsObj),0, LasStatsObj) #For some reason the above function returns cells with no points as NoData instead of 0. This throws off the average, so you need to set NoData to zero with this.
		lasstatsCorRastObj = arcpy.sa.Con(lasstatsCorRastObj,0,lasstatsCorRastObj,"VALUE = -1") #Make sure any empty cells are set to a value of 0, not negative 1, which would throw off analysis.
	lasstatsCorRastObj.save(os.path.join(modelfolder,'DLASSTATS'+folderSubCode+'.tif'))
		lasstatsCorRastObj.save(lasStatsRast)
		MESSAGE_USER("Point density raster for subset {} calculated".format(folderSubCode))
		arcpy.Delete_management(tempLASD)#we can delete the LASD right away, before the fp loop
```

```
                for fp in fplist: #iterate through footprints,
                        (fpSerCode,fpSubCode) = FP_IDENTIFY(fp)
                        if folderSubCode == fpSubCode: #check for a match between subset code and folder.
                                #This process should run twice per LAS file, once for each reconstruction
method
                                MESSAGE_USER("Match identified for FP with series code
{}".format(fpSerCode))
                                FPRast= os.path.join(outgdb,arcpy.ValidateTableName('AFPR_'+fpSerCode))
                                POLY_RASTERIZE(fp,FPRast)
                                MESSAGE_USER("FP with series code {} rasterized".format(fpSerCode))
                                vrast = arcpy.Raster(verifRast)
                                AreaRast = (2*vrast) + arcpy.Raster(FPRast)
                                MESSAGE_USER("Area raster for FP with series code {}
calculated".format(fpSerCode))
#AreaRast has the following values:
# 3 - building in both fp and verif rasters (true positive)
# 2 - building in verifraster but not fp (false negative)
# 1 - building in fp but not verifraster (false positive)
# 0 - building in neither verifraster nor fp (true negative)
                        arcpy.BuildRasterAttributeTable_management (AreaRast)# Make sure that AreaRast has an
attribute table if it didn't get one automatically
                        perFPTableBaseName = fpSerCode+'_AreaAnalysis'
                        perFPTableName = os.path.join(outgdb,arcpy.ValidateTableName(perFPTableBaseName,outgdb))
                        perFPProportionTablename =
os.path.join(outgdb,arcpy.ValidateTableName('AreaTab'+fpSerCode,outgdb))
                        arcpy.sa.ZonalStatisticsAsTable(verifFPs, "Verif_UID", FPRast, perFPProportionTablename,
'DATA', 'MEAN')
                        #Since FPrast represents detected building as 1 and non-building as 0, the mean per-footprint will
be equal to the proportion of footprint detected as a building
                        ProportionFieldName = arcpy.ValidateFieldName("FCompR_"+fpSerCode) #FCR - Footprint
Completeness Ratio, ratio of area in verif footprint correctly identified
                        arcpy.AlterField_management (perFPProportionTablename, 'MEAN', ProportionFieldName)
#change calculated field name
                        arcpy.JoinField_management(verifFPs, "Verif_UID", perFPProportionTablename, "Verif_UID",
[ProportionFieldName])
                        #New per-fp correctness analysis here
                        perFPCorTableBaseName = fpSerCode+'_CorArea'
                        perFPCorTableName =
os.path.join(outgdb,arcpy.ValidateTableName(perFPCorTableBaseName,outgdb))
                        arcpy.sa.TabulateArea(fp,"OBJECTID",verifRast,"Value",perFPCorTableName)
                        arcpy.AlterField_management (perFPCorTableName, 'VALUE_0', 'FalsePositiveArea')
                        arcpy.AlterField_management (perFPCorTableName, 'VALUE_1', 'TruePositiveArea')
                        arcpy.JoinField_management(fp, "OBJECTID", perFPCorTableName, "OBJECTID",
['FalsePositiveArea','TruePositiveArea',])
                        arcpy.AddField_management(fp,"CorRatio","DOUBLE")

arcpy.CalculateField_management(fp,"CorRatio",'!TruePositiveArea!/(!FalsePositiveArea!+!TruePositiveArea!)',"PYT
HON")
                        #New per-fp correctness analysis ends
                        perFPAreaPerPDensTableName =
os.path.join(outgdb,arcpy.ValidateTableName(fpSerCode+"_APerPtDAn",outgdb))
                        perFPTablePath = os.path.join(modelfolder,perFPTableBaseName+'.xls')
                        perFPAreaPerPDensTablePath = os.path.join(modelfolder,perFPAreaPerPDensTableName+'.xls')
                        geomTable = arcpy.ValidateTableName('ARAN_GEOM_TEMP')
                        AreaRast_name = os.path.join(modelfolder,fpSerCode+'_BuildingAreas.tif')
                        arcpy.sa.ZonalStatisticsAsTable(AreaRast, "Value", lasStatsRast, perFPTableName, "DATA",
"ALL")
                        arcpy.sa.TabulateArea(lasStatsRast, "Value", AreaRast, "Value", perFPAreaPerPDensTableName,
1)
                        MESSAGE_USER("DEBUG: Alterfield Tablename:
{0}".format(perFPAreaPerPDensTableName))
                        arcpy.AlterField_management(perFPAreaPerPDensTableName,"Value","PtDensity")
```

```
                    arcpy.AlterField_management(perFPAreaPerPDensTableName,"Value_0","TrueNegative")
                    arcpy.AlterField_management(perFPAreaPerPDensTableName,"Value_1","FalsePositive")
                    arcpy.AlterField_management(perFPAreaPerPDensTableName,"Value_2","FalseNegative")
                  arcpy.AlterField_management(perFPAreaPerPDensTableName,"Value_3","TruePositive")
arcpy.sa.ZonalGeometryAsTable(AreaRast,"Value",geomTable)
arcpy.JoinField_management(perFPTableName,"Value",geomTable,"Value",["Area"])
                    try:
                        arcpy.TableToExcel_conversion(perFPTableName,perFPTablePath)
                        MESSAGE_USER("Area table for FP with series code {} exported as
{}".format(fpSerCode,perFPTablePath))
                    except:
                        MESSAGE_USER("Per-FP Table to Excel conversion failure for table {} to path
{}".format(perFPTableName,perFPTablePath))
                    try:
arcpy.TableToExcel_conversion(perFPAreaPerPDensTableName,perFPAreaPerPDensTablePath)
                        MESSAGE_USER("Area per PT Density table for FP with series code {} exported as
{}".format(fpSerCode,perFPAreaPerPDensTablePath))
                    except:
                        MESSAGE_USER("Per-FP Area Per Pt Dens Table to Excel conversion failure for table {} to
path {}".format(perFPAreaPerPDensTableName,perFPAreaPerPDensTablePath))
                    AreaRast.save(AreaRast_name)
                    TPArea='nul'
                    FNArea='nul'
                    FPArea='nul'
                    TNArea='nul'
areaRastReader=arcpy.da.SearchCursor(AreaRast_name,['VALUE','COUNT'])
                    for row in areaRastReader:
                        if row[0]==0:
                            TNArea=(row[1]/100) #Arearasts is 1x1cm resolution, so divide by 100 to get area in sq.
m
                        elif row[0]==1:
                            FPArea=(row[1]/100)
                        elif row[0]==2:
                            FNArea=(row[1]/100)
                        elif row[0]==3:
                            TPArea=(row[1]/100)
                    if testflag:
                        MESSAGE_USER("DEBUG: {0} TN {1} sq. m.".format(fpSerCode,TNArea))
                        MESSAGE_USER("DEBUG: {0} FP {1} sq. m.".format(fpSerCode,FPArea))
                        MESSAGE_USER("DEBUG: {0} FN {1} sq. m.".format(fpSerCode,FNArea))
                        MESSAGE_USER("DEBUG: {0} TP {1} sq. m.".format(fpSerCode,TPArea))
                    csvwriter.writerow([SectorCode,fpSerCode,fpSubCode,TPArea,FNArea,FPArea,TNArea])
                    try:

arcpy.TableToExcel_conversion(fp,os.path.join(modelfolder,(arcpy.Describe(fp).name+".xls")))
                    except:
                        MESSAGE_USER("FP Excel export failed.")
                    MESSAGE_USER("Area raster for FP with series code {} saved as
{}".format(fpSerCode,AreaRast_name))
            arcpy.env.snapRaster = oldSnap #reset the snap raster
            csvfile.close() #close csv after we're done
            MESSAGE_USER("Area analysis complete")

#Main sequence of the script starts here: functions defined above are called in sequence.
MESSAGE_USER("Script started")
oldworkspace = arcpy.env.workspace
arcpy.env.workspace=rootFolder
folderlist = arcpy.ListWorkspaces()
MESSAGE_USER("Folders recognized:")
for folder in folderlist:
            foldesc = arcpy.Describe(folder)
            MESSAGE_USER("            "+foldesc.name)
```

```
arcpy.env.workspace = oldworkspace
if deleteflag:
	DELETE_EXISTING()
	MESSAGE_USER("Pre-existing files deleted")
if testflag:
	folderlist = folderlist[-6:-5]#truncate the subset list if testflag is true to save time (bigger subsets take much longer),truncating ensures a smaller subset is picked first
	MESSAGE_USER("Folderlist truncated to:")
	for folder in folderlist:
		MESSAGE_USER("		"+folder)
	MESSAGE_USER("WARNING: TEST FLAG SET TO TRUE")


NewInBFPs = os.path.join(outgdb,'InFPs') #copy input features just to make sure they aren't modified by accident
arcpy.CopyFeatures_management(buildingfootprintsIn,NewInBFPs)
buildingfootprintsIn=NewInBFPs
arcpy.CopyFeatures_management(buildingfootprintsIn,buildingfootprintsOut)
MESSAGE_USER("DC Referencing started")
DC_REFERENCE(DCModelRootFolder,buildingfootprintsIn,DCmodelOrigCoordSys,modelfolder,dcmodelgdb)
MESSAGE_USER("DC referencing finished")
MESSAGE_USER("Beginning TIN Modeling")
TIN_MODELING(folderlist,buildingfootprintsIn,tinmodelgdb,modelfolder)
MESSAGE_USER("TIN Modeling complete")
#First, calculate verif poly point counts/densities for each LAS subset
MESSAGE_USER("Calculating point densities for verification footprints")
VERIF_PROCESSOR(folderlist,buildingfootprintsOut)
MESSAGE_USER("Point density information acquired for verification footprints")
footprintlist = [] #initialize empty list of model footprints to be filled by MODEL_PROCESSOR
#Next, process models from each method, storing the resulting footprints
MESSAGE_USER("DC Model processing initiated.")
MODEL_PROCESSOR(dcmodelgdb, footprintlist)
MESSAGE_USER("DC Modeling complete")
MESSAGE_USER("Envi footprint processing initiated.")
ENVI_FP_PROC(footprintlist, folderlist)
MESSAGE_USER("ENVI footprints processed")
#Process the resulting footprints
MESSAGE_USER("Model footprint processing initiated")
FP_PROCESSOR(footprintlist,folderlist)
MESSAGE_USER("Model footprints processed")
#Finally, join the desired data to the verification footprints
MESSAGE_USER("Footprint joining initiated")
FP_JOIN(footprintlist,buildingfootprintsOut)
MESSAGE_USER("Footprint joining complete")
MESSAGE_USER("Commencing area analysis")
AREA_ANALYSIS(folderlist,buildingfootprintsOut, footprintlist)
MESSAGE_USER("Area Analysis complete")
MESSAGE_USER("Extracting centroid coordinates")
try:
	LIST_SHIFT_COORDs(dcmodelgdb, tinmodelgdb, modelfolder)
except:
	MESSAGE_USER("WARNING: Centroid calculation failed, user must run calculation manually.")
MESSAGE_USER("KML coordinates extracted to table in {}".format(outgdb))
if endmessages != []:
	arcpy.AddMessage("Ouput errors detected:")
	for endmessage in endmessages:
		arcpy.AddMessage(endmessage)
MESSAGE_USER("Process complete")
sdt=datetime.datetime.now()
logfilename=os.path.join(os.path.dirname(outgdb),"logfile_{}-{}-{}_{}-{}-{}.txt".format(sdt.year,sdt.month,sdt.day,sdt.hour,sdt.minute,sdt.second))
with open(logfilename, 'w') as logfile:
	for logmessage in logmessages:
```

```
                    logfile.write(logmessage+'\n')
if testflag:
        MESSAGE_USER("WARNING: TEST FLAG SET TO TRUE")
```

Appendix C: Reference Building List

A table showing basic information on all reference building models is shown below. Coordinates are in meters using the Web Mercator (WGS84) projection.

Name	Study Area	Centroid X (m)	Centroid Y (m)	Centroid Z (m)	Centroid Height (m)	Footprint Area (m²)	Minimum Z (m)	Maximum Z (m)	Maximum Height
REF_BVW_L01	Bridgeview	-13679292.62	6309978.72	4.69	2.68	308.78	2.02	7.67	5.66
REF_BVW_L02	Bridgeview	-13679269.61	6309983.53	5.48	3.32	545.80	2.16	8.80	6.64
REF_BVW_L03	Bridgeview	-13679235.06	6309984.43	5.49	3.10	608.74	2.38	8.73	6.35
REF_BVW_L04	Bridgeview	-13679110.26	6309976.81	6.55	3.90	1872.89	2.65	10.96	8.31
REF_BVW_L05	Bridgeview	-13678993.32	6309998.59	6.69	3.86	1272.85	2.83	11.19	8.35
REF_BVW_L06	Bridgeview	-13678931.56	6309991.21	5.20	3.04	311.03	2.16	8.56	6.40
REF_BVW_L08	Bridgeview	-13678888.74	6310079.51	3.76	2.00	578.93	1.75	6.15	4.39
REF_BVW_L09	Bridgeview	-13678822.44	6310001.07	5.07	2.86	590.39	2.22	9.45	7.24
REF_BVW_L10	Bridgeview	-13678733.79	6310009.06	6.30	4.18	1480.33	2.11	11.86	9.74
REF_BVW_L11	Bridgeview	-13678585.34	6310014.68	5.19	3.19	361.93	2.00	9.43	7.43
REF_BVW_L12	Bridgeview	-13678419.26	6310028.70	5.66	3.15	419.85	2.51	9.65	7.14
REF_BVW_L13	Bridgeview	-13678419.18	6310115.83	8.03	5.35	1656.54	2.68	14.95	12.28
REF_BVW_L15	Bridgeview	-13678854.00	6310356.88	5.68	3.77	340.56	1.90	10.39	8.48
REF_BVW_L16	Bridgeview	-13678951.00	6310498.37	3.75	1.96	345.22	1.78	6.23	4.45
REF_BVW_L17	Bridgeview	-13678942.20	6310604.76	6.03	3.40	868.61	2.62	9.72	7.10
REF_BVW_L18	Bridgeview	-13678875.90	6310605.45	6.19	3.32	898.67	2.86	9.76	6.90
REF_BVW_L19	Bridgeview	-13678630.80	6310275.28	6.09	4.61	313.73	1.48	11.51	10.04
REF_BVW_L20	Bridgeview	-13678473.33	6310498.53	6.68	4.74	350.45	1.95	12.73	10.79
REF_BVW_L21	Bridgeview	-13678536.99	6310585.01	4.58	3.05	299.26	1.52	8.32	6.79
REF_BVW_L22	Bridgeview	-13677866.34	6310000.24	4.88	2.40	444.62	2.47	7.62	5.14
REF_BVW_L23	Bridgeview	-13677615.49	6309994.97	5.66	3.34	386.22	2.33	9.39	7.07
REF_BVW_L24	Bridgeview	-13677802.30	6310375.61	3.83	1.81	310.36	2.01	6.14	4.12
REF_BVW_L25	Bridgeview	-13677575.83	6310493.11	4.59	2.80	353.16	1.78	8.46	6.68
REF_BVW_L26	Bridgeview	-13678016.63	6310779.54	6.61	4.03	1087.11	2.58	12.46	9.88
REF_BVW_L27	Bridgeview	-13677891.63	6310575.09	3.74	1.89	299.00	1.85	6.32	4.47
REF_BVW_L28	Bridgeview	-13677572.71	6310939.99	6.39	4.72	324.45	1.66	12.12	10.45
REF_BVW_L29	Bridgeview	-13677242.28	6310784.03	7.89	5.56	1639.05	2.34	14.75	12.41
REF_BVW_L30	Bridgeview	-13677132.05	6310544.80	10.43	6.50	5533.09	3.92	18.52	14.60
REF_BVW_R01	Bridgeview	-13678949.89	6310078.99	4.16	2.41	120.61	1.75	7.45	5.70
REF_BVW_R02	Bridgeview	-13679166.80	6310158.52	3.95	1.75	116.80	2.21	6.32	4.11
REF_BVW_R03	Bridgeview	-13678667.30	6310478.57	3.14	1.41	42.96	1.73	4.88	3.14

REF_BVW_R04	Bridgeview	-13678644.91	6310494.86	2.89	1.38	12.64	1.50	4.39	2.89
REF_BVW_R05	Bridgeview	-13678593.26	6310505.48	3.49	1.89	113.52	1.60	5.77	4.16
REF_BVW_R06	Bridgeview	-13678647.34	6310451.30	3.29	1.69	41.32	1.60	5.53	3.93
REF_BVW_R07	Bridgeview	-13678579.13	6310441.56	2.95	1.06	7.14	1.88	4.18	2.30
REF_BVW_R08	Bridgeview	-13678345.09	6310508.59	5.41	3.47	89.47	1.95	9.84	7.89
REF_BVW_R09	Bridgeview	-13678720.64	6310674.92	3.70	1.99	58.46	1.71	6.30	4.59
REF_BVW_R10	Bridgeview	-13678258.66	6310505.84	5.87	3.99	216.58	1.88	11.12	9.24
REF_BVW_R11	Bridgeview	-13677998.81	6310480.19	3.02	0.99	22.75	2.02	4.13	2.10
REF_BVW_R12	Bridgeview	-13678035.91	6310423.46	4.75	3.07	170.28	1.67	8.32	6.65
REF_BVW_R13	Bridgeview	-13678257.58	6310267.26	3.96	2.31	144.92	1.64	6.85	5.21
REF_BVW_R14	Bridgeview	-13678045.56	6310198.02	4.10	2.10	148.87	2.00	6.75	4.74
REF_BVW_R15	Bridgeview	-13678189.32	6310145.85	3.75	1.70	50.46	2.04	5.88	3.83
REF_BVW_R16	Bridgeview	-13678163.81	6310121.14	4.12	2.25	297.59	1.87	7.24	5.36
REF_BVW_R17	Bridgeview	-13677976.07	6310050.22	3.85	1.62	73.96	2.23	5.65	3.41
REF_BVW_R18	Bridgeview	-13677952.54	6310051.32	3.79	1.78	57.80	2.01	5.61	3.59
REF_BVW_R19	Bridgeview	-13677642.72	6310105.16	3.61	1.48	23.83	2.13	5.39	3.25
REF_BVW_R20	Bridgeview	-13677688.13	6310129.56	4.02	1.78	17.12	2.23	6.13	3.89
REF_BVW_R21	Bridgeview	-13677642.22	6310350.92	3.32	1.41	12.25	1.91	4.81	2.90
REF_BVW_R22	Bridgeview	-13677586.92	6310377.48	6.07	4.18	219.10	1.88	11.40	9.51
REF_BVW_R23	Bridgeview	-13678287.80	6310887.76	4.99	3.24	156.98	1.75	9.00	7.26
REF_BVW_R24	Bridgeview	-13678137.36	6310918.99	2.85	1.21	46.35	1.63	4.37	2.74
REF_BVW_R25	Bridgeview	-13678101.75	6310887.01	5.04	3.35	172.49	1.70	8.82	7.12
REF_BVW_R26	Bridgeview	-13678018.72	6310977.04	3.03	1.54	25.46	1.48	4.96	3.48
REF_BVW_R27	Bridgeview	-13677843.05	6310933.17	3.30	1.65	107.61	1.64	5.27	3.63
REF_BVW_R28	Bridgeview	-13677760.09	6310964.25	3.08	1.46	13.37	1.62	4.66	3.04
REF_BVW_R29	Bridgeview	-13677799.20	6310862.60	6.04	4.50	181.86	1.54	11.60	10.05
REF_BVW_R30	Bridgeview	-13677822.84	6310810.64	4.38	2.60	111.74	1.77	7.86	6.08
REF_BVW_R31	Bridgeview	-13677800.07	6310572.29	4.23	2.49	189.62	1.74	7.58	5.84
REF_BVW_R32	Bridgeview	-13677678.04	6310816.52	6.39	4.67	212.11	1.72	12.21	10.50
REF_BVW_R33	Bridgeview	-13677585.96	6310983.26	5.32	3.88	201.00	1.45	10.50	9.06
REF_BVW_R34	Bridgeview	-13677611.96	6310934.20	3.17	1.92	97.10	1.25	5.47	4.21
REF_BVW_R35	Bridgeview	-13677470.37	6310960.67	2.99	1.63	72.73	1.35	5.00	3.65
REF_BVW_R36	Bridgeview	-13677399.63	6310954.91	2.70	0.98	6.35	1.72	3.79	2.07
REF_BVW_R37	Bridgeview	-13677293.35	6310964.25	3.20	1.32	11.67	1.87	4.67	2.80
REF_BVW_R38	Bridgeview	-13677448.34	6310583.97	3.09	1.55	54.15	1.54	5.19	3.65
REF_BVW_R39	Bridgeview	-13677417.85	6310570.95	4.72	2.92	143.92	1.79	8.52	6.73
REF_BVW_R40	Bridgeview	-13677453.75	6310488.38	3.75	1.87	207.25	1.88	5.80	3.91
REF_CND_L01	Cindrich	-13675659.54	6303257.69	57.14	6.29	4319.11	50.85	63.97	13.12
REF_CND_L02	Cindrich	-13675195.23	6303355.01	54.13	5.29	1347.07	48.84	60.30	11.46
REF_CND_L03	Cindrich	-13675063.13	6303212.62	49.90	4.59	949.54	45.31	55.25	9.94
REF_CND_L04	Cindrich	-13675322.15	6303114.46	50.45	3.88	360.74	46.57	55.68	9.11
REF_CND_L05	Cindrich	-13675340.22	6302880.39	48.69	5.27	355.92	43.42	55.44	12.02
REF_CND_L06	Cindrich	-13675287.76	6303012.49	50.09	4.42	325.05	45.67	55.85	10.18

REF_CND_L07	Cindrich	-13675270.59	6302812.23	45.72	5.44	306.89	40.28	52.77	12.49
REF_CND_L08	Cindrich	-13675291.88	6303101.41	50.64	4.55	323.70	46.09	56.86	10.77
REF_CND_L09	Cindrich	-13675251.92	6303260.98	51.76	3.65	322.60	48.11	56.70	8.59
REF_CND_L10	Cindrich	-13675324.59	6302810.94	46.14	3.85	347.97	42.29	51.55	9.26
REF_CND_R01	Cindrich	-13675349.13	6302975.19	46.81	2.05	145.32	44.76	49.29	4.53
REF_CND_R02	Cindrich	-13675188.13	6303249.50	51.07	3.56	240.04	47.51	55.73	8.22
REF_CND_R03	Cindrich	-13675360.48	6303403.35	54.08	2.90	250.66	51.18	57.58	6.40
REF_CND_R04	Cindrich	-13675282.20	6303272.83	51.86	3.28	290.08	48.58	56.31	7.73
REF_CND_R05	Cindrich	-13675386.54	6303348.23	51.30	1.47	15.13	49.83	52.98	3.15
REF_CND_R06	Cindrich	-13675380.40	6303160.05	48.58	1.11	7.62	47.47	49.84	2.37
REF_CND_R07	Cindrich	-13675299.91	6302878.22	48.56	4.58	267.01	43.98	54.73	10.75
REF_CND_R08	Cindrich	-13675184.28	6302827.71	42.74	3.31	139.42	39.43	47.03	7.60
REF_CND_R09	Cindrich	-13675580.57	6303411.28	55.70	3.30	167.99	52.40	59.83	7.43
REF_CND_R10	Cindrich	-13675185.45	6303218.71	50.29	3.42	214.21	46.87	54.70	7.83
REF_CND_R11	Cindrich	-13675526.66	6303403.56	55.02	2.96	235.73	52.06	59.00	6.94
REF_CND_R12	Cindrich	-13675188.05	6303188.99	50.11	3.65	256.43	46.46	55.09	8.63
REF_CND_R13	Cindrich	-13675646.34	6303375.35	55.10	1.64	23.06	53.46	56.99	3.53
REF_CND_R14	Cindrich	-13675169.23	6302864.38	41.43	1.84	19.47	39.59	42.99	3.40
REF_CND_R15	Cindrich	-13675191.87	6302869.36	45.05	5.03	299.68	40.02	51.68	11.66
REF_CND_R16	Cindrich	-13675355.42	6303279.10	52.86	3.65	253.81	49.21	57.88	8.67
REF_CND_R17	Cindrich	-13675341.57	6302852.00	47.50	3.80	286.89	43.70	52.81	9.11
REF_CND_R18	Cindrich	-13675248.77	6302867.64	47.25	4.81	274.85	42.44	53.65	11.21
REF_CND_R19	Cindrich	-13675485.51	6303384.57	52.40	1.17	7.26	51.23	53.74	2.51
REF_CND_R20	Cindrich	-13675718.09	6303403.46	59.06	2.59	166.33	56.47	62.45	5.98
REF_CND_R21	Cindrich	-13675077.87	6303004.42	42.93	1.67	18.81	41.26	44.95	3.69
REF_CND_R22	Cindrich	-13675351.62	6302808.72	47.30	4.49	276.71	42.81	53.58	10.77
REF_CND_R23	Cindrich	-13675272.94	6302877.57	47.70	4.67	290.96	43.03	54.00	10.97
REF_CND_R24	Cindrich	-13675584.35	6303386.65	54.00	1.94	51.12	52.06	56.40	4.34
REF_CND_R25	Cindrich	-13675324.69	6302963.87	45.71	1.16	7.49	44.55	47.06	2.51
REF_CND_R26	Cindrich	-13675255.81	6303032.61	47.24	1.87	244.25	45.37	49.58	4.21
REF_CND_R27	Cindrich	-13675372.81	6303139.35	50.23	3.38	166.88	46.85	54.33	7.48
REF_CND_R28	Cindrich	-13675384.79	6303261.76	50.07	1.34	15.73	48.73	51.52	2.79
REF_CND_R29	Cindrich	-13675243.21	6302811.10	45.61	4.01	298.95	41.60	51.14	9.54
REF_CND_R30	Cindrich	-13675185.65	6303366.00	50.35	1.24	14.31	49.11	51.69	2.58
REF_CND_R31	Cindrich	-13675398.80	6303164.82	49.92	2.63	266.97	47.29	53.81	6.52
REF_CND_R32	Cindrich	-13675382.61	6303212.35	49.01	1.15	5.86	47.86	50.26	2.40
REF_CND_R33	Cindrich	-13675355.78	6303317.01	52.19	2.85	267.79	49.34	56.01	6.67
REF_CND_R34	Cindrich	-13675262.26	6303313.89	50.35	1.19	6.75	49.16	51.67	2.51
REF_CND_R35	Cindrich	-13675354.17	6303223.35	51.74	3.61	224.01	48.13	56.29	8.16
REF_CND_R36	Cindrich	-13675067.94	6303062.06	43.48	1.53	16.21	41.95	45.21	3.26
REF_CND_R37	Cindrich	-13675085.13	6303128.86	48.03	3.10	98.23	44.93	52.08	7.15
REF_CND_R38	Cindrich	-13675440.39	6303413.27	54.65	3.25	160.62	51.40	58.82	7.42
REF_CND_R39	Cindrich	-13675468.35	6303411.67	54.87	3.14	207.80	51.73	59.12	7.39

REF_CND_R40	Cindrich	-13675071.95	6303080.96	45.38	1.30	21.80	44.08	46.72	2.64
REF_CWH_L01	C. Whalley	-13675451.31	6307076.42	89.74	8.50	10068.14	81.24	102.69	21.45
REF_CWH_L02	C. Whalley	-13675176.51	6307241.85	85.51	5.09	4926.52	80.42	92.28	11.86
REF_CWH_L03	C. Whalley	-13675173.68	6306968.64	81.97	3.36	3285.49	78.61	86.41	7.80
REF_CWH_L04	C. Whalley	-13676003.83	6307036.02	100.34	6.99	3126.35	93.35	108.02	14.67
REF_CWH_L05	C. Whalley	-13675197.53	6306267.08	82.42	5.52	2645.12	76.90	89.70	12.80
REF_CWH_L06	C. Whalley	-13675248.84	6306847.66	83.78	5.13	2225.47	78.65	91.14	12.49
REF_CWH_L07	C. Whalley	-13676145.88	6306938.40	101.34	7.24	1909.97	94.10	109.52	15.42
REF_CWH_L08	C. Whalley	-13675886.00	6307024.00	96.68	8.13	1751.34	88.55	105.36	16.81
REF_CWH_L09	C. Whalley	-13675236.87	6306933.82	81.58	3.14	1743.91	78.44	85.58	7.14
REF_CWH_L10	C. Whalley	-13676136.99	6307127.22	99.68	7.26	1711.41	92.42	108.15	15.73
REF_CWH_L11	C. Whalley	-13675815.80	6306763.93	91.65	5.56	1635.61	86.09	98.20	12.11
REF_CWH_L12	C. Whalley	-13675839.18	6306681.29	91.02	6.03	1580.57	84.99	97.99	13.00
REF_CWH_L13	C. Whalley	-13675911.93	6306614.64	100.28	10.46	1472.69	89.82	116.01	26.19
REF_CWH_L14	C. Whalley	-13675881.79	6306838.89	94.62	5.14	1429.95	89.48	100.56	11.08
REF_CWH_L15	C. Whalley	-13675286.47	6307092.20	85.85	4.30	1310.72	81.55	91.95	10.40
REF_CWH_L16	C. Whalley	-13676217.81	6307372.42	96.49	6.31	1251.88	90.18	104.83	14.65
REF_CWH_L17	C. Whalley	-13675299.35	6307015.41	87.33	5.92	1158.26	81.41	94.29	12.88
REF_CWH_L18	C. Whalley	-13676220.20	6306805.24	100.11	5.13	1017.39	94.98	106.13	11.15
REF_CWH_L19	C. Whalley	-13675288.87	6307383.64	83.34	2.72	1027.19	80.62	86.40	5.78
REF_CWH_L20	C. Whalley	-13675266.57	6307146.63	85.63	4.00	970.43	81.63	92.50	10.87
REF_CWH_R01	C. Whalley	-13676114.19	6306724.69	97.66	2.53	192.73	95.13	100.94	5.81
REF_CWH_R02	C. Whalley	-13675794.18	6307332.90	89.08	3.00	187.93	86.08	93.12	7.04
REF_CWH_R03	C. Whalley	-13676135.61	6307282.44	94.22	3.20	191.20	91.02	98.64	7.62
REF_CWH_R04	C. Whalley	-13675211.80	6306649.02	80.60	1.69	51.20	78.91	82.30	3.39
REF_CWH_R05	C. Whalley	-13675939.26	6306829.15	94.48	3.27	808.37	91.21	98.55	7.34
REF_CWH_R06	C. Whalley	-13675769.88	6307080.01	88.17	1.36	9.69	86.81	89.71	2.90
REF_CWH_R07	C. Whalley	-13675862.89	6307241.88	92.76	2.89	177.24	89.87	96.29	6.42
REF_CWH_R08	C. Whalley	-13675641.39	6307316.85	85.00	3.04	159.69	81.96	88.96	7.00
REF_CWH_R09	C. Whalley	-13675673.82	6307319.42	86.16	2.98	145.44	83.18	89.86	6.68
REF_CWH_R10	C. Whalley	-13675153.04	6307139.61	83.87	2.45	222.96	81.42	86.95	5.53
REF_CWH_R11	C. Whalley	-13676270.63	6306919.36	98.06	3.29	403.60	94.77	102.31	7.54
REF_CWH_R12	C. Whalley	-13676112.40	6307064.02	95.97	3.27	162.12	92.70	99.90	7.20
REF_CWH_R13	C. Whalley	-13676276.10	6306856.02	98.23	3.63	272.62	94.60	103.20	8.60
REF_CWH_R14	C. Whalley	-13675771.69	6307000.07	89.49	2.13	131.12	87.36	92.16	4.80
REF_CWH_R15	C. Whalley	-13675765.07	6307320.97	88.32	2.90	155.48	85.42	92.59	7.17
REF_CWH_T01	C. Whalley	-13675958.74	6307128.20	141.59	50.70	847.96	90.89	193.96	103.07
REF_CWH_T02	C. Whalley	-13676175.10	6306817.59	100.22	5.21	767.88	95.01	106.16	11.15
REF_CWH_T04	C. Whalley	-13675771.32	6306932.90	89.71	2.48	151.73	87.23	92.86	5.63
REF_CWH_T05	C. Whalley	-13676179.57	6306764.68	99.83	4.90	458.67	94.93	105.44	10.51
REF_NWH_L01	N. Whalley	-13674939.31	6308070.40	90.60	7.13	22691.07	83.47	98.44	14.97
REF_NWH_L02	N. Whalley	-13675001.55	6308559.62	91.68	4.01	10775.77	87.67	97.66	9.99
REF_NWH_L03	N. Whalley	-13674950.31	6308317.26	91.64	6.17	6445.29	85.47	98.47	13.00

REF_NWH_L04	N. Whalley	-13675520.19	6308334.07	86.92	5.39	4830.65	81.53	93.46	11.93
REF_NWH_L06	N. Whalley	-13675155.91	6308285.02	88.25	5.17	4580.94	83.08	94.84	11.76
REF_NWH_L07	N. Whalley	-13675371.31	6308466.47	86.06	3.28	3440.20	82.78	90.36	7.58
REF_NWH_L08	N. Whalley	-13675331.72	6308014.05	83.63	3.32	2404.75	80.31	88.08	7.77
REF_NWH_L09	N. Whalley	-13675016.45	6308397.16	92.42	3.58	3482.72	88.83	97.10	8.27
REF_NWH_L10	N. Whalley	-13675451.88	6308119.05	84.99	4.09	2821.56	80.90	90.03	9.13
REF_NWH_R01	N. Whalley	-13675287.59	6308222.50	84.00	2.18	1183.04	81.82	86.59	4.77
REF_NWH_R02	N. Whalley	-13674787.31	6308325.71	98.04	2.50	329.31	95.54	101.28	5.74
REF_NWH_R03	N. Whalley	-13675397.81	6308311.90	83.98	2.62	1656.30	81.36	86.78	5.42
REF_NWH_R04	N. Whalley	-13675162.20	6307970.53	83.00	1.69	80.15	81.32	85.05	3.73
REF_NWH_R05	N. Whalley	-13674786.04	6308196.14	97.20	2.60	280.80	94.60	100.41	5.81
REF_NWH_R06	N. Whalley	-13674790.15	6307950.84	94.12	3.12	280.67	91.00	98.15	7.15
REF_NWH_R09	N. Whalley	-13675431.62	6308417.80	83.64	2.04	374.47	81.60	86.06	4.46
REF_NWH_R10	N. Whalley	-13675127.47	6308020.33	86.84	3.51	1204.39	83.33	91.25	7.92
REF_NWH_R11	N. Whalley	-13674787.61	6308029.20	94.67	2.44	237.12	92.24	97.69	5.45
REF_NWH_R12	N. Whalley	-13675171.84	6308624.24	88.76	3.09	1150.72	85.67	92.30	6.63
REF_NWH_R13	N. Whalley	-13675125.31	6308411.65	88.54	3.66	1992.46	84.88	93.61	8.73
REF_NWH_R14	N. Whalley	-13675411.66	6308392.37	84.16	2.28	1176.02	81.88	87.16	5.28
REF_NWH_R16	N. Whalley	-13674790.72	6308144.58	95.75	2.31	260.86	93.44	98.74	5.30
REF_NWH_R17	N. Whalley	-13674803.54	6308285.80	97.68	3.83	1404.15	93.85	102.03	8.18
REF_NWH_R18	N. Whalley	-13674788.67	6308351.99	97.58	3.05	415.43	94.53	101.55	7.02
REF_NWH_R19	N. Whalley	-13674854.48	6308374.96	94.58	2.77	854.17	91.81	97.95	6.14
REF_NWH_R20	N. Whalley	-13674879.47	6308395.86	93.11	1.15	19.15	91.96	94.44	2.48
REF_NWH_R21	N. Whalley	-13674847.64	6308501.25	95.29	1.75	800.67	93.54	97.89	4.35
REF_NWH_R22	N. Whalley	-13674808.00	6308629.20	97.44	3.21	1308.87	94.23	101.10	6.87
REF_NWH_R23	N. Whalley	-13675251.53	6308555.66	87.40	3.70	1052.23	83.70	91.53	7.83
REF_NWH_R24	N. Whalley	-13675163.75	6308539.95	88.62	3.58	2258.48	85.04	92.64	7.60
REF_NWH_R25	N. Whalley	-13675182.83	6308275.00	86.66	3.53	883.48	83.14	90.19	7.05
REF_NWH_R26	N. Whalley	-13675198.33	6308249.75	86.04	2.95	898.51	83.09	89.82	6.73
REF_NWH_R27	N. Whalley	-13675180.83	6308201.65	86.12	3.06	2252.58	83.06	89.69	6.63
REF_NWH_R28	N. Whalley	-13675165.07	6308087.77	84.56	1.30	144.17	83.26	86.18	2.92
REF_NWH_R29	N. Whalley	-13675161.42	6308135.55	84.61	1.63	35.14	82.98	86.24	3.26
REF_NWH_R30	N. Whalley	-13675130.26	6307986.93	84.47	2.00	726.10	82.47	86.93	4.46

Curriculum Vitae

Name: Peter Crawford

Post-secondary Education and Degrees:
Queen's University
Kingston, Ontario, Canada
2011-2016 B.Sc.

The University of Western Ontario
London, Ontario, Canada
2016-2018 M.Sc.

Honours and Awards:
Dean's Honours List (GPA above 3.5/4.3): 2014, 2015, 2016
ESRI Young Scholar Award, 2016

Related Work Experience
Teaching Assistant
The University of Western Ontario
2016-2018

Research Assistant
University of Western Ontario
Summer 2017, Summer 2018

Publications:

- Potvin, Dominique, Peter Crawford, S.A. MacDougall-Shackleton, and E.A. MacDougal-Shackleton. "Song Repertoire Size, Not Territory Location, Predicts Reproductive Success and Territory Tenure in a Migratory Songbird". Canadian Journal of Zoology 93, no. 8, 627-33. doi:10.1139/cjz-2015-0039.
- Peter Crawford, and Jinfei Wang. 2017. "Effects of Point Cloud Density on the Accuracy of Building Reconstruction from LiDAR." CAGONT. Kingston, Ontario.