University College Dublin

Geometric Processing Techniques for Urban Aerial Laser Scan Data

by
Tommy Hinks

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
College of Engineering, Mathematical and Physical Sciences
School of Computer Science and Informatics

January 2011

Head of School: Dr Joe Carthy

Supervisors: Dr Hamish Carr and Dr Debra Laefer

Abstract

Current *Aerial Laser Scanning* (ALS) technology rapidly produces large amounts of accurate point data for urban regions, making it a suitable tool for city-scale geometric modeling of buildings. However, acquisition and processing of urban ALS data remains a challenge because of the geometric complexity of urban scenes. Existing techniques have focused on geometric modeling from elevation data, ignoring details on building walls. This thesis introduces several improvements and simplifications for the acquisition and processing of ALS data: urban flight path planning, scan line analysis, visualization, building extraction, and simple and robust conversion of ALS data into solid models for further processing. By applying geometric reasoning, it is shown that certain flight paths vastly improve the point data quality on building walls. Single scan line analysis then exploits latent information in the data to insert missing echoes caused by undetected pulse reflections, and to identify building wall segments in individual scan lines. Points on building wall segments are then transferred to a digital image and complete building footprints are then extracted from using innovative morphological techniques. Finally, a simple and robust method for direct conversion of point data into solid models based on volumetric subdivision rather than surface reconstruction is presented.

Acknowledgements

First of all I would like to thank my supervisors, Dr Hamish Carr and Dr Debra Laefer. This thesis would not have been possible without their guidance and wisdom. I would also like to thank Linh Troung Hong, my fellow PhD student, for the many discussions we have had and for help with simulations. Additionally, I would like to acknowledge Dr Yann Morvan and Dr Carol O'Sullivan at Trinity College Dublin for the shared work that was done.

During the years I spent in Dublin I met many friends. These people probably meant more to me than they could understand, in a time when I was separated from my family back in Sweden. I will not try to make a complete list of friends here. If you think you are on the list, then you are.

Finally, there are some people in Stockholm that deserve to be acknowledged. The Important Looking Pirates kindly lent me a desk when I was in need of one. My family has supported me in my studies ever since I was a child and continued to do so when my interests shipped me abroad. For this I am thankful and my debt will be repaid to future generations of us. However, during the writing of this thesis there is one person who above all has shared my toils. That is Tove and even though I can never fully repay her, I will try to do so in years to come.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Spatial information plays an important role in modern day society. Knowledge about our surroundings is in many ways crucial to planning and communication on all scales. In the past, geographic exploration and recording has shaped the history of mankind, enabling worldwide empires to influence every corner of the planet and astronauts to walk the surface of the moon. Today hand-drawn maps complete with sea-monsters have been replaced by computer programs where any given street corner is never more than a few clicks away.

Modern remote sensing technologies, such as digital photography and laser scanning, have changed the way our surroundings are recorded. Painstaking manual surveys are being replaced by more automated approaches. Current devices are capable of rapidly recording vast areas in great detail. As acquisition technologies have improved, increasing amounts of spatial data have become available. While new spatially aware applications are being spawned, existing applications are taking the opportunity to incorporate a spatial component. Urban planning, forest monitoring, and flood plain mapping are examples of fields that have exploited the recent surge in spatial data availability. Applications in these fields perform domain-specific tasks using geometric models of our surroundings as input. As an effective means of acquiring spatial data, remote sensing thus supports spatially aware applications by providing the raw data necessary for further processing. Also, fast turn-around times for data acquisition enable up-to-date geometric models and on-demand acquisition of missing areas.

Having emerged in recent decades, *Aerial Laser Scanning* (ALS) is a relatively new remote sensing technology. An aircraft equipped with a laser scanner flies over an area of interest, sampling points on surfaces below. Current ALS systems are capable of sampling many thousand points per second. Over time this yields enormous point sets with counts on the order of hundreds of millions of points, or more. ALS provides a (typically very large) set of points sampled on surfaces in the scanned scene. The scanned points provide a discretized representation of the scanned surfaces. However, most applications require continuous surface representations. For this reason, raw ALS data needs to be processed after acquisition in order to become usable in other applications.

Although ALS rapidly generates vast quantities of accurate positional data, automatic detection and interpretation of individual objects remains a challenge. ALS (along with other remote sensing technologies) does not associate any semantic information with the acquired data. Therefore, some sort of processing is required for the data to become meaningful. Such processing is commonly known as *filtering*. Filters are used to create interpretable representations from raw data, ideally without loss or corruption of information. Advanced filtering techniques are necessary in order to fully leverage the huge volumes of data produced by ALS. The sheer sizes of datasets make manual filtering infeasible, as it would simply be too costly and take too long. Instead, automatic or semi-automatic methods are required. Automatic methods are also desirable in that they are of a deterministic nature, making their output meaningfully comparable over time.

Large-scale three-dimensional geometric models of cities are used in a wide range of applications, including such seemingly disparate subjects as noise prediction and disaster mitigation. As an example, in the event of sudden infrastructural changes, such applications require accurate geometric models, and in a post-disaster scenario, the models must also be acquired rapidly. The work in this thesis focuses on creating such urban models using ALS. More specifically, this thesis focuses on the creating of engineering models, which are necessarily somewhat different from models used in visualizations in that they must support numerical analysis. Such models are frequently used for damage assessment and in natural disaster simulations. Creation of building models on a city-scale has benefited hugely from recent advances in ALS, since accurate and detailed spatial data for entire cities can be collected within hours. However, at present, no automatic methods exist for creating engineering models directly from ALS data. Current practices often resort to hand-modeling each building, which is both slow and often inaccurate. Additionally, such approaches are hugely expensive given the amount of manual labor involved.

Urban modeling is mostly concerned with buildings and other large permanent, man-made structures, and, therefore, the first filtering task for urban modeling is to classify points as belonging to objects of interest (e.g. buildings) or background. Similar classification problems arise in other applications of ALS, for instance in classifying bare-earth from vegetation in forest landscapes. Such classification problems are highly domain-specific, and in the case of urban landscapes architectural diversity provides significant challenges, since it is difficult to robustly define geometrically what a building is. The filtering approach presented in this thesis is based on the common assumption that buildings are bounded by vertical planes, and that the detection of these planes gives the buildings footprint, which can be used to extract points contained within.

A simple, yet effective, method for converting ALS point data into engineering models is presented in this thesis. This method is based on volumetric subdivision and allows ALS points to be assigned to distinct volumetric elements. These elements are connected to provide an approximative shape, derived directly from the ALS point data. The feasibility of this method is demonstrated using real-world ALS data, showing that the generated models are indeed usable for engineering purposes.

1.1 Outline

The contents of this thesis are divided into the following four parts:

- Part I provides the necessary background information required to clearly state the goals of this thesis. Previous work in fields related to this thesis are discussed and toward the end of this part a set of tools required to achieve the goals of this thesis is presented.
- Part II describes the tools that were developed in order to achieve the goals of this thesis.
- Part III presents results of using the tools described in the previous part.
- Part IV gives conclusions drawn from the results achieved in this thesis and presents possible directions for future work.

Part I

Background

Chapter 2

Laser Scanning

This chapter gives a brief introduction to *Laser Scanning* (LS). The material presented in this chapter serves as the basis for discussions in following chapters since it describes the data used and how such data is acquired. Laser scanning is a large field of study and, therefore, this chapter presents only the basic principles of this technology. In particular, topics relevant in coming chapters are discussed. Further details can be found in the provided references.

Surveying is the process of determining precisely the positions of points on the earth's surface. Traditionally, each point would be measured manually using optical instruments and anchored in some coordinate system using triangulation [Dug04]. The number of points that can be measured with manual surveying approaches are limited by the time each measurement takes, and also, as a consequence, economical factors. Additionally, manual surveying has focused on measuring horizontal points, with three-dimensional data being limited to a highly narrow set of applications [SB05].

Measured points are used to create models for a wide variety of applications and the quality of such models is directly dependent on the measured points. In order to incorporate fine details in models a large number of points is required, which has lead to the development of more automated approaches in earth measurement. Optical instruments are, therefore, being replaced by more automated tools, such as LS, which are capable of rapidly and accurately measuring a large number of points. The remaining parts of this chapter provide descriptions of the aspects of LS that are of importance in this thesis.

First, the basic principles of LS are explained and, thereafter, aerial and terrestrial LS are discussed. Aerial and terrestrial LS are considered separately since the differences between these two approaches have significant impact on the subsequent data processing steps. The main difference is that *Terrestrial Laser Scanning* (TLS) is done from fixed positions, whereas *Aerial Laser Scanning* (ALS) is carried out from a moving platform. Since this thesis is concerned with the modeling of buildings, ALS and TLS are considered in the context of acquiring point data for urban scenes.

2.1 Principles

LS is based on the principles of *Light Detection And Ranging* (LIDAR) [Jel92, WUM*04]. Laser scanning devices measure distances to surfaces. Knowledge of the scanner position and orientation allows distances to be converted into point samples, thereby recording the geometries of the surfaces. LIDAR is similar to radar in that they both operate using electromagnetic waves. LIDAR, however, operates at shorter wavelengths than radar, allowing smaller features to be captured, thereby providing higher levels of detail on scanned surfaces [Ree01]. The raw output of LS is points sampled on surfaces. Ideally, scanned points are spaced closely enough to allow for faithful reconstruction of the sampled surfaces.

Laser scanning is an *active* remote sensing technology in that measurements are based on emitted energy. In comparison, photography (assuming no flash is used) is a *passive* remote sensing technology since it measures existing energy, contributed by sources unrelated to remote sensing devices [UMF82]. The most common form of LIDAR is based on pulsed energy emission [PB07], and is often referred to as *time-of-flight*. This name stems from the fact that ranges are determined by measuring time delays between emissions and detections of energy pulses [UMF82]. Other variants, such as *continuous wave*, exist but are less common and not further discussed in this thesis, cf. [HW97, WL99, MRC*07]. The work in this thesis assumes that points are acquired using time-of-flight LIDAR, the principles of which are described in more detail in the following section.

2.1.1 Time-of-Flight

Time-of-flight LS operates by emitting electromagnetic pulses toward surfaces and measuring the return-times of reflected energy, as shown in Figure 2.1. Return-times can be readily converted into distances using the following relation:

$$d = \frac{ct}{2}$$

where d is the distance from scanner to surface, c is the speed of light and t is the measured time between emission and detection of reflected energy [Bal99b]. Precise knowledge of the scanner position and orientation in some coordinate system allows further conversion of distances into points on surfaces in that coordinate system. It is, therefore, possible to acquire dense point samplings of objects by emitting pulses in a large number of directions. Note that LS is limited by line-of-sight, since pulse energy cannot travel through solid objects. Further, pulse reflections depend on interactions between pulse energy and surfaces, as discussed next.

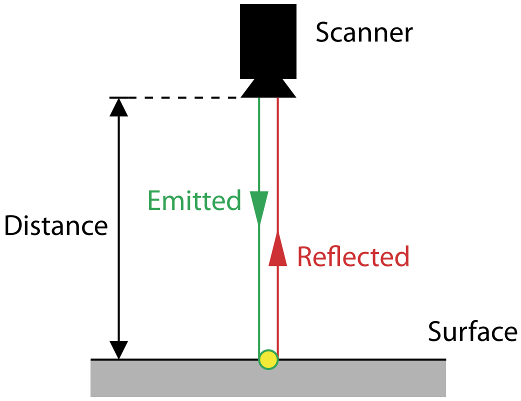


Figure 2.1: Time-of-flight laser scanning measures distances to surfaces by emitting pulses and measuring the return-times of reflected pulse energy.

2.1.2 Material Reflectance

Interactions between emitted pulses and surfaces are largely dependent on material reflectance properties [War92, MPBM03, NDM05]. Laser pulses are emitted from a scanner toward surfaces (Figure 2.2) and if a sufficiently large amount of reflected energy reaches back to the scanner the return-time can be measured.

In a *Lambertian* [Lam60] reflection model incoming energy is reflected more or less evenly in all directions on the hemisphere around the surface normal (Figure 2.2). A detailed description of such a model for LS can be found in [Bal99b]. Since Lambertian materials reflect incoming energy evenly it is likely that sufficient amounts of reflected energy reach back to the scanner, allowing return-times to be measured. Examples of materials that fit well into the Lambertian reflection model are concrete and marble.

However, not all materials fit well into the Lambertian reflection model. Highly reflective materials, such as metal, reflect most of the incoming energy in a narrow cone around a single direction (Figure 2.2). This is known as *specular* reflection, and as a consequence reflected pulse energy often cannot be detected in these cases. Hence, no return-time can be measured and no point sample is recorded for the emitted pulse. Other materials, such as glass or water, refract incoming pulse energy so that little or no reflected energy reaches back to the scanner. Thus, highly reflective or refractive materials are difficult, or impossible, to sample using laser scanning. In addition to material properties, surface orientations with respect to the scanner play an important role in pulse measurements. Surfaces not directly facing the scanner tend to reflect incoming energy away from the scanner, thus decreasing the chances of successful detection.

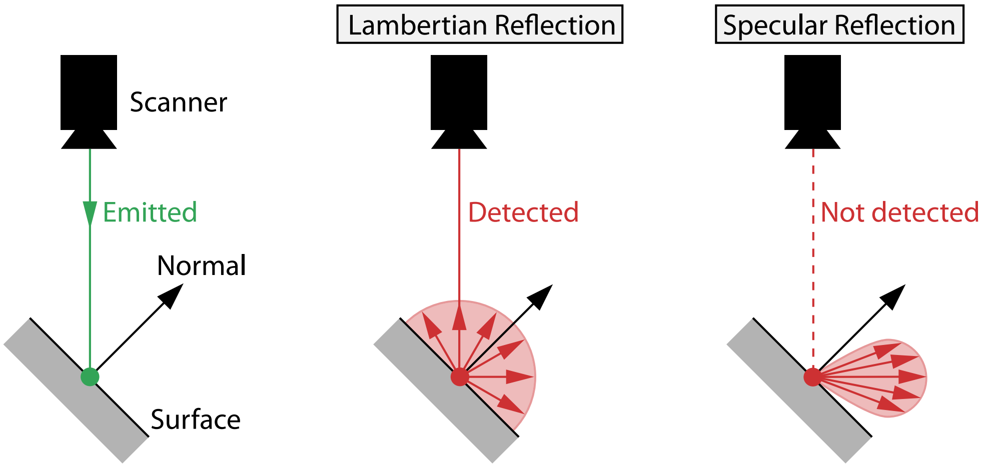


Figure 2.2: *Left*: A laser pulse is emitted from a scanner toward a surface. *Middle*: A Lambertian surface reflects pulse energy more or less evenly around the surface normal and part of the reflected energy is detected at the scanner. *Right*: A specular surface reflects pulse energy in a narrow cone and reflected energy never reaches back to the scanner.

In conclusion, complex interplay between emitted pulse energy, surface orientations, and material properties determines if reflected pulse energy is detected. However, the details of these topics are beyond the scope of this thesis. For the purposes of this thesis it is sufficient to acknowledge that points cannot be sampled on all surfaces and that not all emitted pulses produce corresponding point samples. In particular, scans of buildings typically do not contain points on glass windows, which has important consequences in the creation of building models and will be discussed in coming chapters. Having introduced the basic principles of LS, the following sections of this chapter describe how laser scanning technology is used in practice. The next section describes how LS is done from an aircraft.

2.2 Aerial Laser Scanning

This section presents topics related to point data acquisition using ALS. As additional sources of information several good overview articles on ALS exist, cf. [KHE96, Ack99, WL99, Axe99, Bal99b, Bal99c, Bal99a, Bal99d, PB07]. First attempts using aircraft-mounted laser scanners to measure the earth were made in the 1960's and the aircraft position was determined using in-flight photographs [Lin93]. However, this approach did not achieve the accuracy necessary for commercial use. Modern ALS systems achieve higher levels of accuracy by using a combination of a *Global Navigation Satellite System* (GNSS) and an *Inertial Measurement Unit* (IMU) to determine the aircraft's position and orientation, respectively. Together with the laser scanning device these two components form a multi-sensor system that is synchronized using the GNSS time measurements.

As the aircraft moves over the scene pulses are emitted toward surfaces below (e.g. bare earth, buildings, vegetation, and other objects). Modern ALS systems are capable of acquiring hundreds of thousands of points per second, providing detailed samplings of the scanned surfaces. Figure 2.3 shows an example of ALS point data acquired over an urban area, with points color-coded according to elevation from a ground plane. The points shown are a small subset of a larger data set containing $\sim$ 225 million points. Note that points are mostly acquired on horizontal surfaces and are missing for building walls.

ALS has been used extensively to acquire detailed point data over vast areas [KP98]. To date, ALS data have been used in a number of applications, including:

- Geographic Information System content generation [HBA98, SMS05],

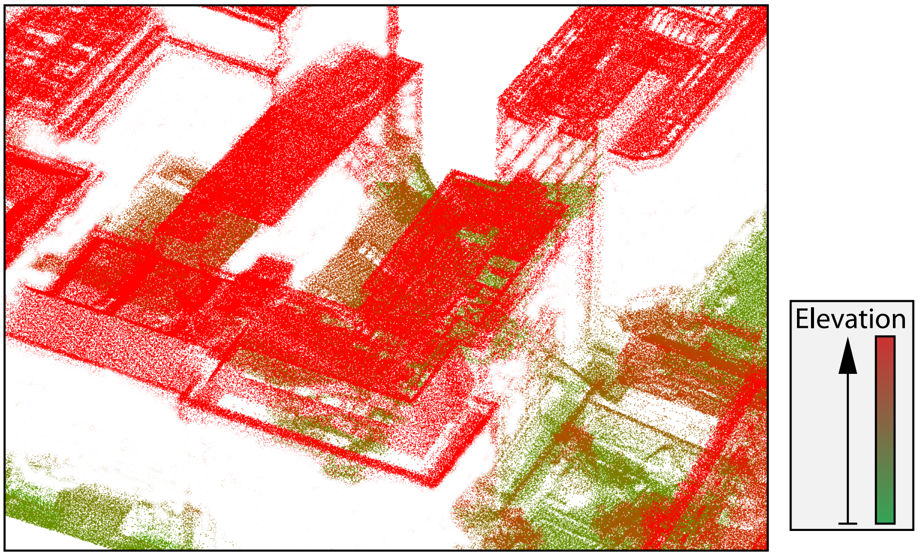


Figure 2.3: An example of ALS data taken from a larger urban scan. Points are color-coded according to elevation measured from a ground plane.

- Disaster response and damage assessment [Kev03, LP06],
- Hydrology [HWK05],
- Forestry [HHL*04, AMR05, HWEK06],
- Glacial monitoring [GLS03, ARDA06],
- Urban modeling [HB99, Vos00, VD01, EB02, Rot03, OTDS04, SMS05, FNSZ06, RTCK07, DP08, MWK09].

As evident from the listing above, ALS is a versatile tool being used in a wide range of applications. While the remainder of this section presents details on ALS in general, the work in this thesis focuses urban modeling applications using LS. As mentioned, an aircraft moves over the scene, acquiring points on surfaces below. The motion of the aircraft during scanning, often referred to as the *flight path*, is discussed next.

2.2.1 Flight Path

The term flight path is used to describe the motion of the aircraft during scanning. As mentioned, position and orientation of the aircraft are accurately measured using GNSS and IMU, respectively. During scanning aircraft position and orientation are recorded at regular time intervals and are stored as *flight path points* (FPP). Flight path points describe instantaneous aircraft position and orientation at different times and have unique time-stamps that are used to track the aircraft motion over time. Together, the flight path points recorded during an ALS mission describe the motion of the aircraft over time, as shown in Figure 2.4.

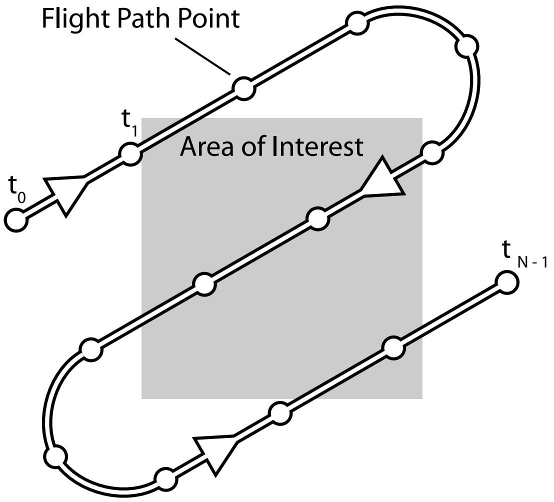


Figure 2.4: A flight path is stored as a set of flight path points.

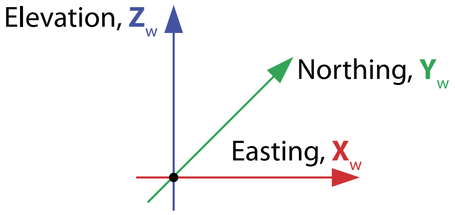


Figure 2.5: Global coordinate system.

The data for a FPP has the format shown in Table 2.1. Aircraft position is given in a global coordinate system (Figure 2.5) and is provided by a GNSS unit. Yaw, pitch and roll angles

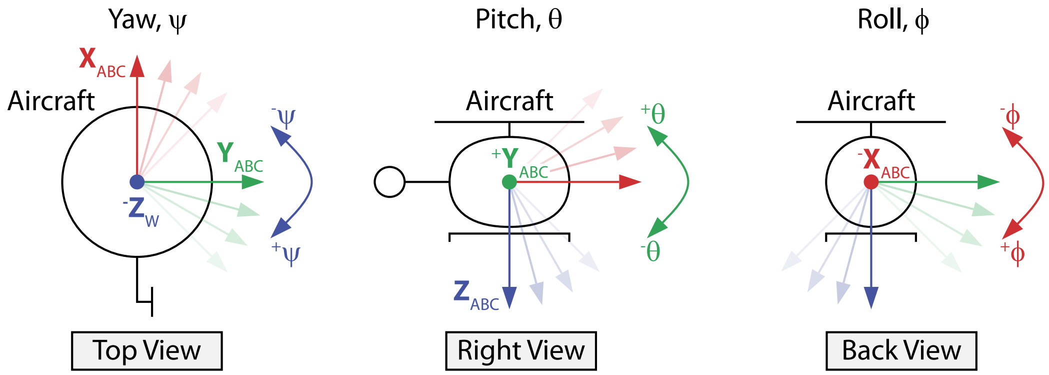


Figure 2.6: A local aircraft body coordinate system is defined as a sequence of rotations with respect to a world coordinate system (Figure 2.5). The local aircraft body coordinate system is such that $\hat{x}_{ABC}$ points in the forward direction, $\hat{y}_{ABC}$ points to the right, and $\hat{z}_{ABC}$ points downward. *Left*: The yaw angle describes rotation around $\hat{z}_w$. *Middle*: The pitch angle describes rotation around $\hat{y}_{ABC}$. *Right*: The roll angle describes rotation around $\hat{x}_{ABC}$.

describe the rotation of the aircraft as shown in Figure 2.6 and are provided by an on-board IMU.

2.2.2 Scan Pattern

As the aircraft moves, pulses are emitted out to the sides, perpendicular to the aircraft's forward movement, by rotating a mirror that deflects the laser before it exits the aircraft [Lat05]. As such, a strip on the ground beneath the aircraft is scanned and the width of this strip is known as the *swath width*, w_s, and is given by the expression $w_s = 2h\tan(\theta/2)$, where h is the aircraft altitude and θ is the *scan angle* (Figure 2.7). The instantaneous scan angle of a pulse, $\theta_i \in [-\theta/2, \theta/2]$, is measured from the aircraft body vertical axis, and the angular resolution $\Delta\theta$ is the angle between two consecutive pulses. The number of pulses emitted per second, f_p, is known as the *pulse rate*. Note that although pulses are emitted at fixed angular intervals, this does not mean that scan points are evenly spaced on surfaces. Point spacing depends on the scanned geometry, and this topic is further discussed in Chapter 8.

In this thesis it is assumed that points are acquired in roughly parallel *scan lines* (Figure 2.7). Other scan patterns exist and are described in [Bal99b]. Each scan line is acquired by emitting

Variable	Description	Unit
t	Time-stamp	[s]
x	Northing	[m]
y	Easting	[m]
z	Elevation	[m]
ψ	Yaw, $\psi \in (-180, 180]$	[degrees]
θ	Pitch, $\theta \in [-90, 90]$	[degrees]
ϕ	Roll, $\phi \in (-180, 180]$	[degrees]

Table 2.1: Flight path point format.

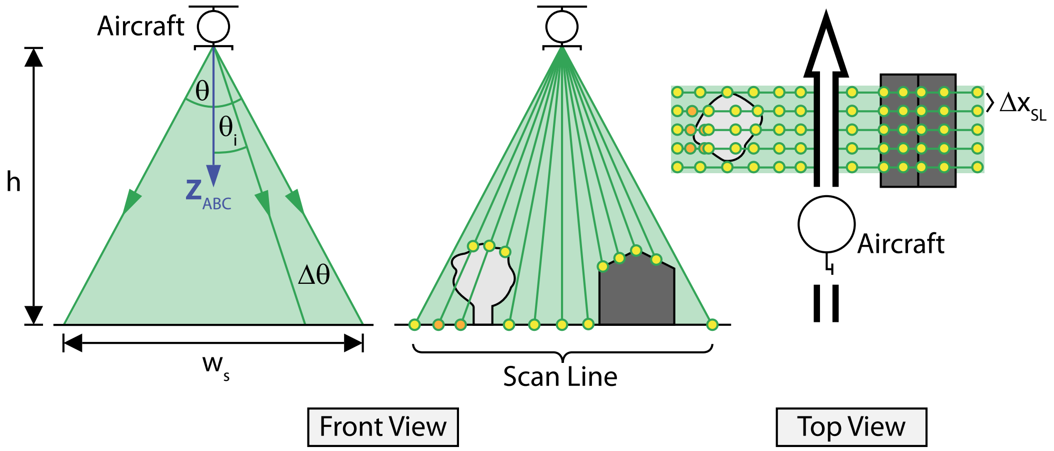


Figure 2.7: *Left*: The swath width, w_s, is dependent on aircraft altitude, h, and scan angle, θ. Instantaneous scan angle, θ_i, is measured from the aircraft body vertical axis and pulses are emitted at regular angular offsets, $\Delta\theta$. *Middle*: A scan line is a set of pulses emitted at regular angular offsets in the direction perpendicular to the aircraft movement. *Right*: Consecutive scan lines are parallel and fairly regularly spaced (Δx_{SL}) in the forward movement direction of the aircraft.

pulses at regular angular intervals in the direction perpendicular to the flight direction, commonly referred to as the *across-track* direction. The number of scan lines acquired per second, f_s is known as the *scan rate* and the distance in meters between two consecutive scan lines is $\Delta x_{SL} = v/f_s$, where v is the speed of the aircraft in meters per second. Typically, the scan rate is large compared to the speed of the aircraft and, therefore, the aircraft is often considered as being stationary during the acquisition of a single scan line.

Most often, the area of interest is wider than a single strip, which requires several passes to be flown over the area of interest to ensure that the entirety of the area is covered by at least one strip. Often strips are slightly overlapping, since this allows navigation and calibration errors to be corrected. This type of correction is called *strip adjustment* and is well described in [KHE96, VM01, Fil03, FV04]. During acquisition, aircraft trajectories together with strip data are used to correct internal systemic errors, which may occur as a result of calibration errors. The *relative* orientation between strips can be improved by minimizing plane fitting residuals [Kag04, RKM08]. As an example, gable roofs are often used for this purpose, since they provide convenient planes along with a line of intersection. Similarly, the *absolute* orientation of strips can be corrected using ground control planes. Strip adjustment and many other processing techniques operate directly on ALS point data, the format of which is described next.

2.2.3 Multiple Echoes

As pulse energy travels from the aircraft toward the ground energy spreads out, as described by wave propagation theory [AP05]. This phenomenon is known as *beam divergence* in ALS literature. Beam divergence has an averaging effect on distance measurements, which depends on the distribution of reflectivity on surfaces within the pulse cone [Bal99b]. The smallest measurable size is dependent on surface reflectance properties, since detection is based on the amount of reflected energy [WL99]. Thus, a small, highly reflective surface facing the scanner may reflect detectable amounts of energy, while a larger, less reflective surface may fail to do so.

Further, beam divergence may cause pulse energy to interact with multiple surfaces. Consider reflected energy to be a signal sampled at very small time-steps (on the order of nanoseconds). In this signal, reflections from multiple surfaces appear as separate peaks (Figure 2.8). A peak corresponds to significant reflection of emitted pulse energy from a surface. Peaks are often

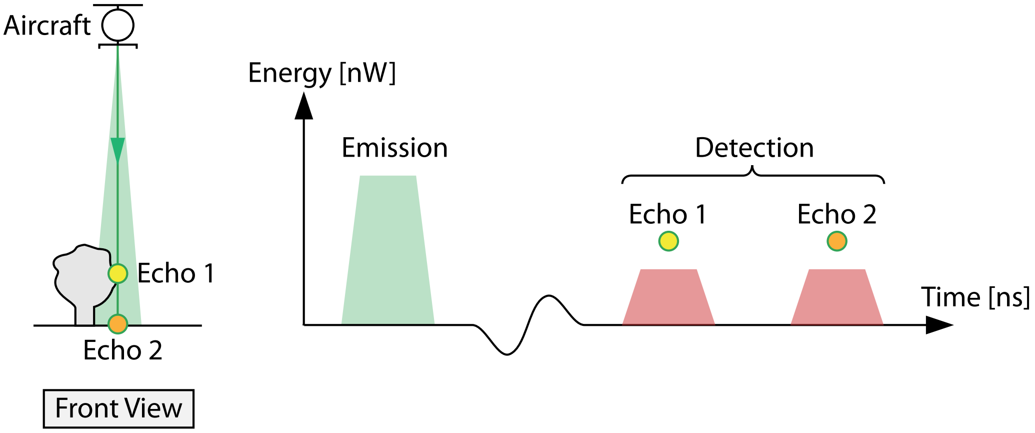


Figure 2.8: A pulse may interact with several surfaces, which results in several peaks in the detected reflection signal.

referred to as *echoes* and each echo corresponds to a point sample on a surface. As such, a single pulse may produce several point samples. For instance, it is common for multiple echoes to occur in areas with vegetation. The first echo is then detected from surfaces on the vegetation, and the following echoes from surfaces beneath or inside the vegetation.

Certain types of recent ALS systems, known as *full-waveform* [PSTA05, KAHS05, WUD*06, JS06, RHHP08], store the entire reflected signal. This alleviates the need for real-time processing (i.e. peak-detection) during acquisition. Also, the increased amounts of information present in full-waveform data allow processing beyond peak-detection, such as material identification and other classification tasks in some cases [PB07]. Since processing is not required to be real-time it becomes possible to apply more sophisticated signal processing algorithms. A disadvantage is that the requirements for data storage increase significantly.

The work in this thesis, however, does not use full-waveform ALS data and assumes that scan data is delivered in the form of discrete points. The format of these points is described next.

2.2.4 ALS Point Data

The scan point data format presented here is based on a commonly open binary format often used by ALS service providers. A complete of this format description is available in [ASP03].

Variable	**Description**	**Unit**
t	Time-stamp	[s]
x	Easting	[m]
y	Northing	[m]
z	Elevation	[m]
i	Intensity	
n	Number of echoes	
r	Echo number, $r \in [1, n]$	

Table 2.2: ALS scan point format.

The data for a scan point has the format shown in Table 2.2. A time-stamp is provided for each scan point and it will be shown further on how these time-stamps are useful for linking scan points to aircraft positions. Additionally, scan point time-stamps are not unique and typically scan points acquired in the same scan line have identical time-stamps, which provides an effective means of extracting individual scan lines from large ALS data sets.

ALS scan point positions are given in a global coordinate system, related to the aircraft position, which is determined by GNSS (Figure 2.5). Accuracy in the horizontal plane as high as $\pm 2cm$ has been reported [Vos08]. Vertical accuracy is often estimated to be around $\pm 15cm$, e.g. [RMAC03, Vos08]. It should, however, be noted that accuracy validations are difficult to carry out since there is often no ground-truth available, especially for vertical measurements [Lat02].

Further, modern scanners usually provide an intensity value for each scan point. The intensity value is proportional to the amount of reflected pulse energy detected for the echo in question. However, these intensity values are rarely used, since their interpretations vary largely between different vendors [HP07]. Moreover, each scan point has information about the pulse in which it was acquired. The total number of echoes detected for the current pulse is recorded, along with an echo index for the scan point. Finally, in some cases a camera is used in the flyover, which enables the association of colour information to the scan points [Haa99].

2.3 Terrestrial Laser Scanning

TLS is the term used for ground-based laser scanning. TLS operates on the same principles as ALS, but the scanner is stationary on the ground during scanning, as opposed to being mounted underneath a moving aircraft. Approaches using truck-mounted scanners have been proposed [FZ03, FZ04], but the most common form of TLS consists of a laser scanner mounted on a tripod. Complementary to the description of TLS in this section is the recent and detailed treatise available in [Res09].

TLS has become popular for the acquisition of architectural scenes because dense surface measurements are provided through a flexible process that allows a wide range of buildings to be recorded [CC08]. Figure 2.9 illustrates the procedure of scanning a single building using TLS. In order to acquire points on all sides of a large object, such as a building, it is necessary to position the scanner in several different locations. Note that, for reasons explained in Section 2.4, the overlap between scans is intentional. Simple, box-shaped buildings typically require at least four different scanner positions, as shown in Figure 2.9.

Repositioning of the scanner between scans is done manually and the acquisition time for a single building is on the order of several hours [MHC*08]. In contrast, large portions of a city can be scanned in the same time frame using ALS. However, TLS provides significantly higher point densities on scanned surfaces than ALS, enabling the capture of finer features. Moreover, TLS readily acquires points on vertical surfaces, which are difficult to capture with ALS (Figure 2.3).

Figure 2.10 shows an example of TLS point data acquired on a building from a single scanner position. As shown in the inset (red rectangle), small features, such as window frames, are present in the point data. Gaps in the data coverage on the building wall are caused by occluding trees and vehicles located close to the building during the scanning. Points acquired on occluding objects are typically removed, leaving only points acquired on the object of interest, in this case the building.

Further, some parts of the roof are occluded by raised windows (Figure 2.10, green circles). In general, slanted roofs can be acquired using TLS, however, as was just shown, there are exceptions to this rule. Another exception is the case where roofs consist of horizontal surfaces, which are not visible from ground level. If there are no suitable elevated vantage points looking down at the scanned building, these surfaces cannot be acquired with TLS. As such, ALS and

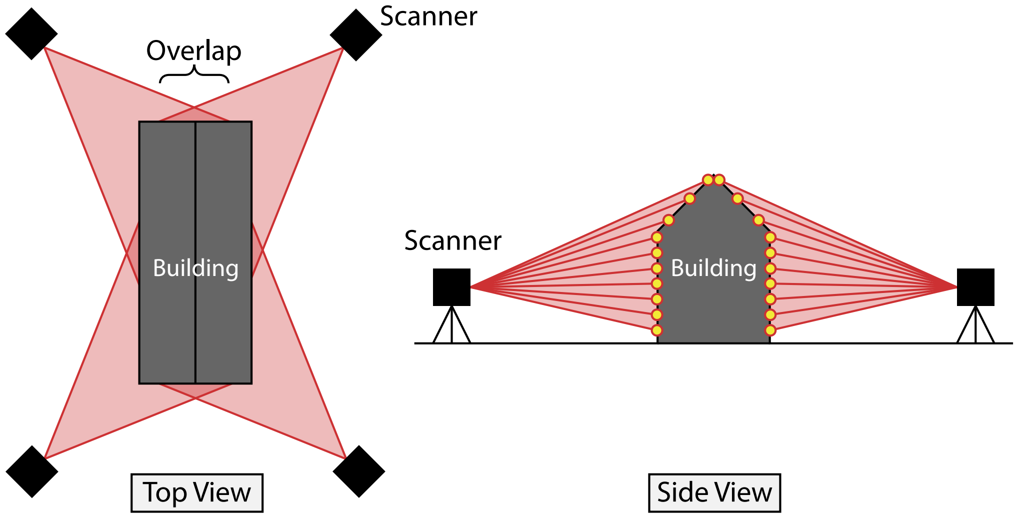


Figure 2.9: *Left*: Several scanner positions are required to acquire data for all sides of a building. Four different scanner positions are shown in the image. *Right*: With TLS it is possible to acquire detailed point data for vertical surfaces, which are difficult to acquire with ALS.

TLS are complementary in the sense that ALS is better for acquiring horizontal surfaces, while TLS is better for acquiring vertical surfaces. This relationship is illustrated through Figure 2.3 and Figure 2.10, even though these two images do not show scans of the same building.

Although ALS and TLS are conceptually similar, there are important differences between the points acquired by each method. Next, a common format for TLS points is presented and the aforementioned differences are discussed.

2.3.1 TLS Point Data

Each point acquired with TLS has a three-dimensional coordinate, in some cases accompanied by an intensity value. The format of TLS points is shown in Table 2.3. TLS points are typically provided in a coordinate system where the scanner is located at the origin. As with ALS points, the intensity value for TLS points is somewhat unreliable and is rarely used [PDHF07].

As for ALS points, colour information can be associated with each scan point. However, since the topics in this thesis are strictly concerned with geometric aspects of point data, colour information has been omitted from Table 2.3. Because TLS points are stored as local coordinates

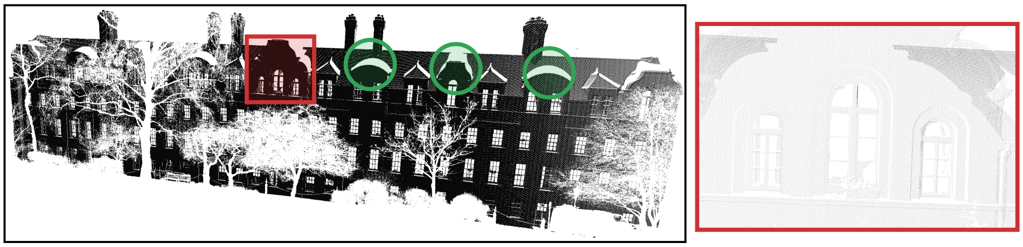


Figure 2.10: TLS points acquired on a building from a single scanner position. The inset (red rectangle) shows a close-up view where it is possible to identify individual points and small features, such as window frames. Green circles show three regions where raised windows occlude roof parts.

Variable	**Description**	**Unit**
x	Local x-coordinate	[m]
y	Local y-coordinate	[m]
z	Local z-coordinate	[m]
i	Intensity	

Table 2.3: TLS scan point format.

and several scanner positions are required for large objects, point data from multiple TLS scans must be aligned. This process is known as *registration* and is discussed next.

2.4 Registration

The procedure of fitting, or aligning, two point data sets is known as registration and comes in many variants, often tailored towards a specific application domain. The general problem statement for registration reads as follows: given two point data sets, often referred to as *model* and *data*, the goal is to find a rigid transformation that optimally positions, or registers, the *data* with respect to the *model* [GMGP05]. Complementary to the discussions on registration in this section is the material found in [SB05].

While ALS points are readily acquired in a global coordinate system, owing to the fact that the aircraft position is determined using GNSS, TLS points are typically acquired in a coordinate systems that is local to the scanner position. As mentioned, the scanner needs to be repositioned in order to acquire complete data of a large object. Thus, points are acquired in separate coordinate systems, each relating to a different scanner position. Therefore, in order to consider TLS points in the context of the scanned object, it is necessary to register the points into a common coordinate system, which is usually done by pair-wise registration of the scans [MGPG04].

There are two main classes of registration techniques and these are described separately in Section 2.4.1 and Section 2.4.2. For considering a single object, it is sufficient to determine a common coordinate system for the scans of that object. However, if the object needs to be considered in relation to other objects, or other spatial data in general, it is necessary to use a global coordinate system, as discussed in Section 2.4.3.

2.4.1 Sensor-driven Registration

Sensor-driven registration is based on determining the scanner position and orientation using additional sensor equipment such as *Global Positioning Systems* (GPS), digital compasses, and IMU. ALS (Section 2.2), for instance, uses sensor-driven registration to acquire scan points in a global coordinate system. Similarly, position and orientation of a TLS device can be established using one or more of the abovementioned sensors [WMDS10].

If the scanner position and orientation are known, the scanned points can be acquired in a global coordinate system, relating to the positional information provided by the GPS [BH05]. As such, scans from different positions are readily provided in the same coordinate system, alleviating the need for further registration. Although sensor-driven registration techniques have the advantage that points can be registered in real-time, the high-end GPS equipment required for high-accuracy measurements is expensive [SB05]. Moreover, high-accuracy (differential) GPS measurements require additional base-stations and clear views of approximately five satellites [MMH05]. Intermittent sensor interference has been reported due to canyoning problems in urban environments, where users often find themselves between tall structures [LR07]. Because sensor-driven registration may be cost-prohibitive and performs badly in urban environments, registration techniques that operate directly on the acquired point data have been proposed and are discussed next.

2.4.2 Data-driven Registration

Data-driven registration techniques are based on matching features in two point data sets. Thus, it is required that the two point data sets, at least partially, overlap spatially so that features are present in both data sets (Figure 2.9). By identifying matching features it is possible to compute the rigid transforms used to bring points into a common coordinate system [ELF97]. Where many geometric features are distinguishable, data-driven registration can be extremely accurate [PHYH06]. Moreover, data-driven registration does not require any additional sensor data to be collected during scanning, which is attractive in terms of planning and acquisition time.

Feature identification in point data sets can be difficult and artificial features, known as *targets*, can be added to the scene in order to ease the process of feature detection. Targets are easily identifiable and can often be automatically detected by registration software. However, it is not always possible to use targets because of limited accessibility to the scene. Moreover, target placement requires considerable amounts of planning and placement of targets adds significantly to the overall acquisition time. For these reasons data-driven techniques in general rely on matching features found directly in the acquired point data.

Further, most data-driven approaches require an iterative refinement step in order to achieve high accuracy. A commonly used iterative refinement algorithm is the *Iterative Closest Point*

(ICP) method [CM91, BM92, CM92, RL01]. A rigid transform is refined by alternating the steps of choosing corresponding points across the two data sets and finding the best rotation and translation that minimize an error metric based on the distance between corresponding points. ICP, however, requires the data sets involved to be roughly aligned initially in order to converge properly [MGPG04].

Fully automatic alignment techniques for the purpose of initializing an ICP process have been proposed [GMGP05]. Features based on local curvature are used to identify corresponding points on small objects, such as artefacts. For objects with unique, highly curved protrusions, these techniques are clearly effective. However, building scans are characterized by large planar features, offering few opportunities for high curvature feature matching. Hence, the rough initial alignment is often done manually in practice [GMGP05, MPD06]. In some cases it is possible to use low-cost sensors for the first, rough initial guess, but the aforementioned canyoning issues in urban regions may prohibit the use of such technology [BH05].

Up to this point registration has involved bringing point data sets into a common coordinate system. Sensor-driven and data-driven approaches have been discussed and the strengths and weaknesses of each approach have been pointed out. Appropriate use of registration is sufficient for studying a single object, such as a building or artefact. However, if this object needs to be considered in relation to other spatial data, a global coordinate system is required. The task of anchoring spatial data in a global coordinate system is discussed next.

2.4.3 Georeferencing

To *georeference* something means to establish its existence in physical space [Hil09]. Most georeferencing tasks are undertaken because two or more different data sets need to be linked because they relate to the same geographic location. In terms of laser scanning, georeferencing means that point coordinates are given in a global coordinate system. Georeferencing is required to support *sensor data fusion*, where spatial, or spatially related (e.g. color information), measurements from multiple sensors are combined in order to provide more complete data coverage of objects [BH05].

Being able to combine data from disparate sources may provide significant advantages, since various acquisition methods have strengths and weaknesses that can be exploited to suit the needs of a particular situation [Haa99, RTCK07]. However, in order to leverage the combined strengths of multiple acquisition methods a global coordinate system is required.

ALS points are readily provided in a global coordinate system, relating to the fact that GNSS is used to measure the aircraft position. TLS point data, however, is typically acquired in a local coordinate system, where the scanner is placed at the origin. Methods for linking TLS point data to a global coordinate system through the use of additional sensors, such as GPS, are expensive and perform badly in urban environments (Section 2.4.1). Because of these deficiencies, data-driven georeferencing approaches for TLS data have become popular.

It is becoming increasingly common to use an existing georeferenced point data set as the framework into which TLS point data sets are registered (e.g. [SB05]). Examples of existing georeferenced data are virtual city models (e.g. [HO03]) and ALS scans. If TLS points are acquired within an area that has existing georeferenced data, these points can be georeferenced using data-driven registration techniques.

2.5 Summary

Laser scanning is a technology based on the principles of LIDAR capable of acquiring dense point sets on surfaces. The two most common forms of laser scanning are ALS and TLS. In ALS, a scanner is mounted underneath an aircraft, acquiring points on surfaces below as the aircraft moves. TLS operates on the same principles as ALS, but the scanner is usually stationary on the ground during scanning. While ALS is used to acquire point data over large areas, TLS is more suitable for acquiring points on a single object or for a small scene.

This chapter has presented two different methods of acquiring point data through laser scanning. Points acquired with laser scanning can be used to create geometric models of objects and such models are used in a wide variety of applications, ranging from forest inventory to virtual cities. The creation of such models from point data requires significant amounts of processing and following chapters will discuss this topic in more detail. The next chapter describes digital images, which are commonly used to visualize ALS points and to identify buildings in large ALS data sets.

Chapter 3

Digital Images

Digital images are commonly used in a number of applications relating to *Aerial Laser Scanning* (ALS) point data, including both filtering and visualization. This chapter describes digital images and introduces key concepts used in digital image processing. Although digital images are often associated with devices such as monitors and printers, the concept of a digital image is much wider. The material in this chapter, therefore, is not limited to any particular device, but rather describes fundamental properties of digital images. In this context, a digital image can be thought of a bounded area in a flat plane. This plane is subdivided into regular cells, where each cell stores a value representing some quantity, such as color. Digital imaging is a huge field and the selection of techniques presented here were chosen for their relevance to the discussions in coming chapters. For material beyond the scope of this thesis relevant references are provided. The notation used in this chapter is based on the seminal work by Gonzalez and Woods [GW02], unless otherwise noted.

First, a basic mathematical definition of digital images is given, explaining the concepts of discretizing continuous functions. This basic definition is then extended to include information that makes it possible to associate point data with digital images, with applications in ALS data processing and visualization. An example using a common technique in ALS data visualization is given. The rest of this chapter presents digital image processing techniques relevant to discussions in coming chapters on ALS data filtering.

3.1 Sampling and Quantization

Images in general are often expressed as continuous, two-dimensional functions of the form $f(x, y) \mapsto \mathbb{R}^C$, where $C \geq 1$ is the number of *channels*. Continuous in this case refers to both the domain and range of f. The value of f at spatial coordinates (x, y) is some quantity, the meaning of which is determined by the interpretation of the image. The cases $C = 1$ and $C = 3$ are often referred to as *gray-scale* and *colour* images, respectively. Gray-scale image functions represent spatially varying single-valued intensities, while colour image functions represent spatially varying triple-valued intensities. The components of colour triplets are typically interpreted as red, green, and blue intensities. By varying the proportions between these intensities it is possible to reproduce a broad array of colours. The two image types described above are well-suited for visualization. However, the values of an image function need not be interpreted as colour intensities. In general, the image function can be interpreted as any spatially varying quantity.

Creating a digital representation of an image requires the continuous function $f(x, y)$ to be discretized. Spatial discretization is referred to as *sampling* and involves computing the values of f at discrete positions. Discretizing the range of f is known as *quantization.* Sampling and discretization are discussed in more detail below.

3.1.1 Sampling

The sampling process may be viewed as partitioning the xy-plane into a regular grid. The coordinates of the center of each grid cell is then a pair of (positive) integers. Hence, $f(x, y)$ is a digital image if (x, y) are integers coordinates and f is a function that assigns some value to each distinct pair of coordinates (x, y).

Assume that an image $f(x, y)$ is regularly sampled so that the resulting digital image has M rows and N columns (Figure 3.1). The values of the coordinates (x, y) are then discrete quantities and at the origin $(x, y) = (0, 0)$. The preceding definitions allow a digital image to

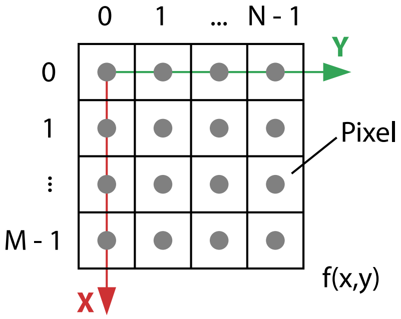


Figure 3.1: A regularly sampled digital image pixel grid.

be expressed in compact matrix form as

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \tag{3.1}$$

The right side of Equation 3.1 is then by definition a digital image. Each element of this matrix is called a picture element, or *pixel* for short (Figure 3.1). Having shown how the domain of f is discretized through sampling on a regular grid, quantization of the range of f is discussed next.

3.1.2 Quantization

In general, quantization is the procedure of mapping something from a continuous set of values (such as the real numbers) to a discrete set (such as the integers). Quantization of an image function f involves dividing the continuous range (per channel) into discrete levels and assigning each pixel one such level (per channel) based on the value of f. Thus, quantization involves discarding information in order to limit the number of possible values to a discrete set of levels. In digital images pixel values are represented as a number bits, k. The number of bits directly determines the number of possible levels, $L = 2^k$, where a larger number of bits increases the number of possible levels. Moreover, a larger number of levels enables images to contain more details since more fine-grained differences in pixel values become representable. Typically, 8 bits per channel are used, providing 256 levels per channel. The digital images shown in this thesis use 8 bits per channel, unless otherwise noted. Next the use of ALS data with digital images is further explored, describing techniques used to associate point data with pixels in a digital image.

3.2 World to Image Mapping

Digital images occur in a number of ALS applications, either as intermediate steps in filtering or as a means of visualizing point data. The regular pixel grids of digital images provide a structure that is often easier to work with than large sets of unorganized three-dimensional points. The basic mechanism driving such applications is the ability to *map* a three-dimensional position to a pixel in a digital image. By mapping points to pixels it is possible to transfer unstructured information to a grid, which simplifies further processing. This section explains how such mapping can be achieved.

First the necessary extensions to digital images, which enable three-dimensional point mapping, are described. Thereafter, a mapping approach using the global coordinate system introduced in Chapter 2 is presented. This section is concluded with an example of a visualization technique commonly used with ALS point data.

3.2.1 Global Coordinate Bounds

In order to relate pixels to a global coordinate system it is necessary to specify the image bounds in global coordinates. This is achieved by extending digital images with global coordinates at the corner pixels, as shown in Figure 3.2. Since ALS applications are the motivation for these mapping discussions the global coordinate system for ALS points (Figure 2.5) is used for convenience.

In the following discussions it is assumed that a digital image lies in the xy-plane of the global coordinate system and that the bounds in this plane are given with respect to the global coordinate system origin. In general any plane may be chosen. However, in practice most ALS applications use a horizontal image plane, motivated by the fact that the horizontal extents of ALS data are typically much larger than the vertical span. The discussions in this section could be extended to support general planes using orthogonal projections [Str88].

The values $x_{min} < x_{max}$ and $y_{min} < y_{max}$ define the (horizontal) extents of the a $M \times N$ digital image in global coordinates (Figure 3.2). The distance between pixels centers, which is

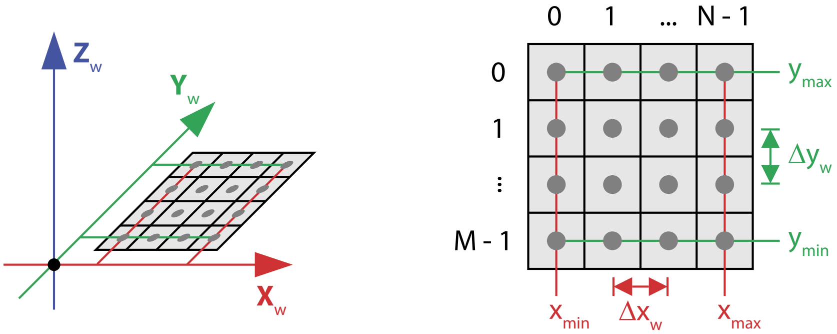


Figure 3.2: Digital image with (horizontal) global coordinate bounds.

the same as the length of a pixel side, is computed as

$$\Delta x_w = \frac{(x_{max} - x_{min})}{(N-1)}$$

$$\Delta y_w = \frac{(y_{max} - y_{min})}{(M-1)}$$

From the above expressions it is clear that a digital image must have $N \geq 2$ and $M \geq 2$ to avoid division by zero. Further, in order to avoid directional biases it is desirable to have square pixels, i.e. $\Delta x_w = \Delta y_w$, which expands to

$$\begin{aligned} \frac{(x_{max} - x_{min})}{(M-1)} &= \frac{(y_{max} - y_{min})}{(N-1)} \\ \frac{(x_{max} - x_{min})}{(y_{max} - y_{min})} &= \frac{(M-1)}{(N-1)} \end{aligned}$$

Since M and N are integers it may be impossible to choose them such that the ratio between them matches the ratio on the left-hand side. Therefore, global coordinate bounds must be chosen such that their ratio matches the pixel dimension ratio. Having extended digital images to lie in a bounded global coordinate plane, the mathematical mapping from a world space position to a pixel is presented next.

3.2.2 Global Coordinate Mapping

A mapping takes as input a position in one coordinate system and transforms it to another coordinate system. In the case of mapping ALS points to digital image pixels this corresponds to transforming a global coordinate position into a digital image pixel coordinate, as shown in Figure 3.3.

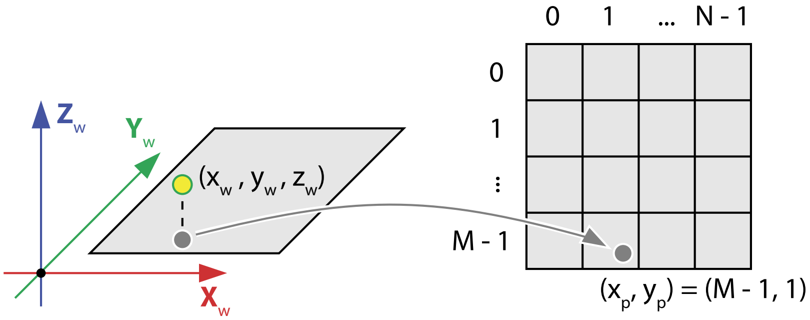


Figure 3.3: Global to pixel coordinate mapping.

A mapping from global coordinates (x_w, y_w, z_w) to pixel coordinates (x_p, y_p) may be defined in the following way

$$(x_p, y_p) = \left(\left[(M-1)\frac{(y_{max} - y_w)}{(y_{max} - y_{min})} + 0.5 \right], \left[(N-1)\frac{(x_w - x_{min})}{(x_{max} - x_{min})} + 0.5 \right] \right) \tag{3.2}$$

where the operator $[x]$ rounds x to the nearest integer toward zero and is defined as

$$[x] = \begin{cases} \lfloor x \rfloor & x \geq 0 \\ \lceil x \rceil & x < 0 \end{cases} \tag{3.3}$$

where $\lfloor x \rfloor$ is the standard *floor*-operator and $\lceil x \rceil$ is the standard *ceil*-operator. Note that the elevation (z_w) component is ignored in Equation 3.2 since the global coordinate bounds are assumed to be in the horizontal plane. Further, the x- and y-coordinates are flipped due to the orientations of the global coordinate axes with respect to the image axes. Equation 3.2 yields valid pixel coordinates

$$\begin{aligned} x_p &\in [0, M-1] \\ y_p &\in [0, N-1] \end{aligned}$$

for global coordinates in the ranges

$$x_w \in \left[x_{min} - \frac{\Delta x_w}{2}, x_{max} + \frac{\Delta x_w}{2} \right] \tag{3.4}$$

$$y_w \in \left[y_{min} - \frac{\Delta y_w}{2}, y_{max} + \frac{\Delta y_w}{2} \right] \tag{3.5}$$

The mapping of global coordinates outside the ranges in Equations 3.4–3.5 yields invalid pixel coordinates. Since the image plane is bounded in the global coordinate system, clearly not all global positions map to a valid pixel coordinate. It is, therefore, common to set up the global coordinate bounds after first examining the bounds of the global coordinate positions that will be mapped to the image, avoiding issues with positions mapping to invalid pixel coordinates. Having demonstrated how world space positions can be mapped to pixels, an application using ALS point data is presented next.

3.2.3 Elevation Images

The most common type of image created from ALS point data is an *elevation image* (Figure 3.4). An elevation image is created by mapping ALS points to pixels (Equation 3.2) and assigning intensities proportional to the largest elevation (i.e. z_w-coordinate) of mapped points. As shown in Figure 3.4 elevation images are able to convey a top-down view of ALS point data, where buildings and vegetation are recognizable by a trained human observer. In ALS literature, elevation images are often referred to as *Digital Surface Models* (DSM's) or *Digital Elevation Models* (DEM's). Additionally, elevation images created from filtered ALS data, containing only points sampled on bare-earth are often referred to as *Digital Terrain Models* (DTM's).

The regular sampling of digital images greatly simplifies many ALS filtering tasks. Besides reducing the dimensionality of filtering tasks from three dimensions to two, the grid-based nature of digital images enables a wealth of digital image processing methods, which is arguably richer than the tools available for direct processing of three-dimensional point data. An example of a filtering task where elevation images are commonly used is building extraction (e.g. [RB02, FNSZ06, FN07]).

Having introduced digital images and showed how to relate them to a global coordinate system, the rest of this chapter will focus on digital image processing techniques that are commonly used in ALS data processing. These techniques are relevant both to the work presented in this thesis and to previous work found in literature.

3.3 Pixel Connectivity

Connectivity between pixels is a fundamental concept used in numerous digital image processing techniques. Several important relationships between pixels are discussed in this section.

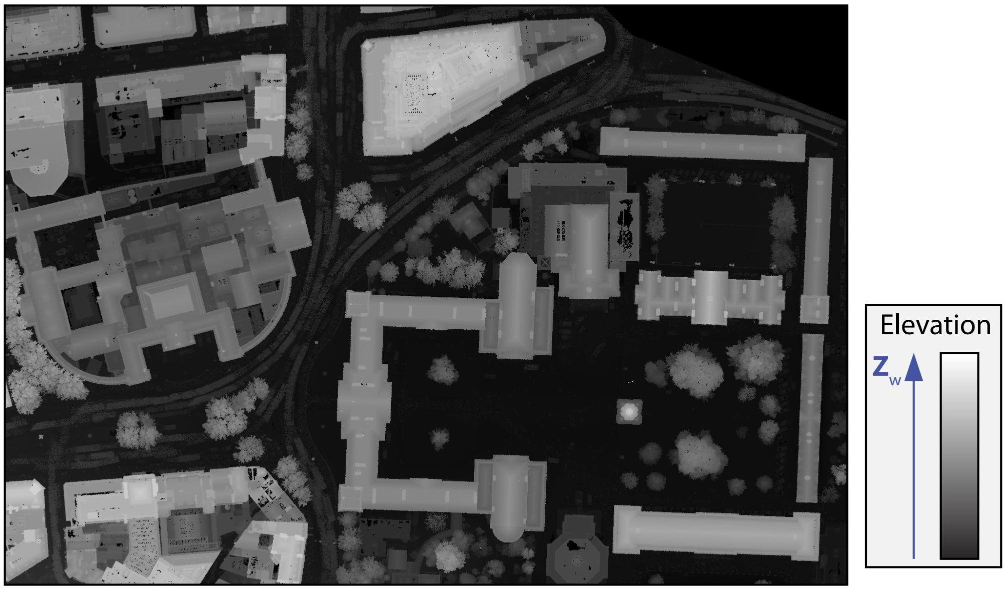


Figure 3.4: An elevation image created by mapping ALS points to an image and assigning pixel values proportional to elevation. Pixel dimensions in global coordinates are $\Delta x_w = \Delta y_w = 0.2\ m$.

3.3.1 Neighbors

A pixel p at coordinates (x, y) has in total four *horizontal* and *vertical* neighbors (Figure 3.5) whose coordinates are given by

$$(x+1, y), (x-1, y), (x, y+1), (x, y-1)$$

This set of pixels is called the *4-neighbors* of p and is denoted by $N_4(p)$. Similarly, the four *diagonal* neighbors of p have coordinates

$$(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)$$

and are denoted by $N_D(p)$. The diagonal neighbors, together with the 4-neighbors, are called the *8-neighbors* of p, denoted by $N_8(p)$. It is possible for coordinates in $N_4(p)$, $N_D(p)$ or $N_8(p)$ to lie outside the digital image if (x, y) belongs to the first or last column or row. The way outside neighbors are handled is application-specific, but failure to detect such cases leads to undefined results.

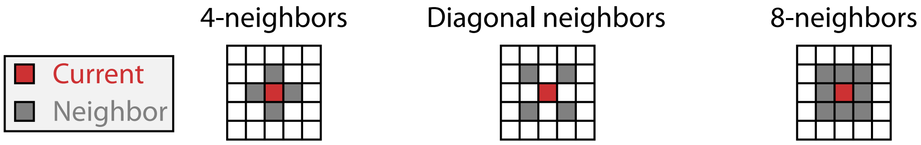


Figure 3.5: Three different pixel neighborhoods. *Left*: 4-neighbors (gray) of the current pixel (red). *Middle*: Diagonal neighbors. *Right*: 8-neighbors.

3.3.2 Adjacency

The concept of *adjacency* is an extension to the neighbor concept introduced in Section 3.3.1. Adjacency takes into account not only the spatial relationship between pixels, but also their values. For two pixels to be adjacent they must be neighbors *and* have the same value. Consider two pixels p and q that have the same value. If q is part of $N_4(p)$ the pixels are *4-adjacent*. Similarly, if q is part of $N_8(p)$ the pixels are *8-adjacent*.

The two types of adjacency just described are the most common. Other types of adjacency exist and are described in [GW02]. In summary, pixels that are not neighbors cannot be adjacent. For two pixels to be adjacent they must be neighbors *and* have the same value.

3.3.3 Paths

Paths extend the concept of adjacency to involve more than two pixels. A path is said to exist between two pixels if there is a set of adjacent pixels linking the two pixels. Thus, a *path* from pixel p with coordinates (x_p, y_p) to another pixel q with coordinates (x_q, y_q) is a sequence of distinct pixels with coordinates

$$(x_0, y_0), (x_1, y_1), ..., (x_n, y_n)$$

where $(x_0, y_0) = (x_p, y_p)$ and $(x_n, y_n) = (x_q, y_q)$ and pixels (x_i, y_i) and (x_{i-1}, y_{i-1}) are adjacent for $i \in [1, n]$ (Figure 3.6). Here, n is known as the *length* of the path. If $(x_0, y_0) = (x_n, y_n)$ the path is said to be *closed*, otherwise it is said to be *open*. The terms *4-path* and *8-path* are used depending on the definition of adjacency.

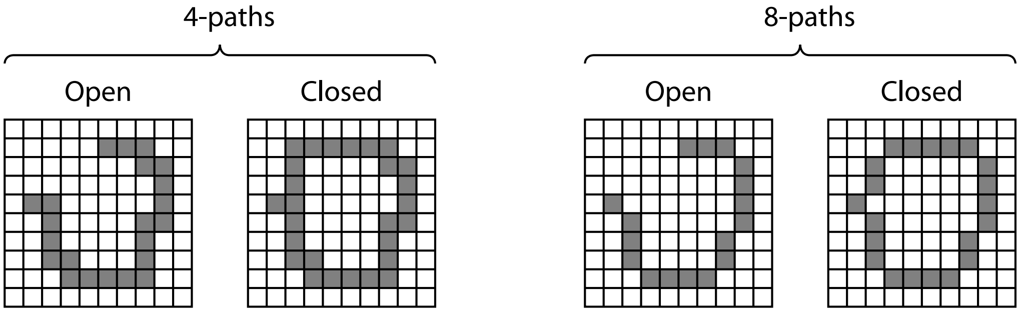


Figure 3.6: Four different paths using two different adjacency definitions. *Left*: Open and closed 4-paths. *Right*: Open and closed 8-paths.

In summary, a path is a set of pixels where each pixel is adjacent to at least one other pixel in the set. Additionally, paths can be open or closed. In a closed path the first pixel is adjacent to the last pixel, while this is not the case for an open path.

3.3.4 Connected Components

Two pixels are said to be *connected* if there exists a path between them. A set of pixels where every pixel is connected to every other pixel is called a *connected component* (Figure 3.7). The terms *4-connected component* and *8-connected component* are used depending on the path definition.

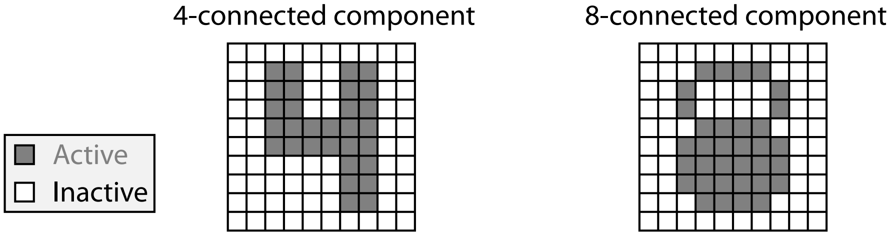


Figure 3.7: Two different connected components using two different path definitions. *Left*: 4-connected component. *Right*: 8-connected component.

The pixel connectivity concepts presented in this section are fundamental to more high-level digital image processing techniques. A few high-level digital image processing techniques are presented in the remaining sections of this chapters. These techniques are commonly used in the processing of digital images created from ALS data and are relevant to methods presented further on.

3.4 Image Morphology

Image morphology is a broad set of digital image operations for processing images based on shapes. Digital image morphology was introduced by Serra [Ser82] and later extended by Sternberg [Ste86]. Mathematical set theory is often used to describe image morphology. However, the discussions in this section will be less formal, providing more intuitive descriptions of some fundamental morphological operations. Rigorous mathematical notation can be found in [SS01] and the abovementioned works. First, a basic mechanism common to all morphological operations is introduced and, thereafter, two fundamental morphological operations are described.

3.4.1 Structural Elements

A *structural element* (SE) is a digital image of some shape and size (Figure 3.8). For practical purposes, structural elements often have an odd number of pixels in each dimension so that there is a clearly defined middle pixel, which is chosen to be the *center*. Also, the SE is typically much smaller than the input image. The pixels of a SE are either *active* or *inactive*.

Two commonly used structural elements are shown in Figure 3.8. These two structural

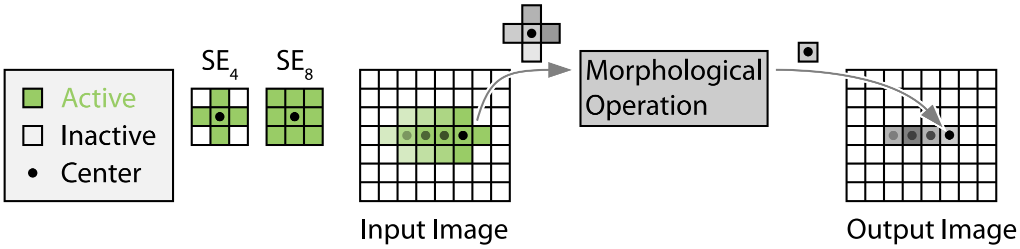


Figure 3.8: Structural elements are (small) digital images. A structural element is centered on each pixel and defines the input to a morphological operation that assigns a single value to the corresponding center pixel in an output image.

elements relate to the pixel neighborhoods discussed earlier (Section 3.3.1) and are given corresponding names: SE_4 and SE_8. These two structural elements are used in the examples below.

A morphological process involves centering a SE on each pixel of an input image (Figure 3.8). The SE acts as a mask and pixels in the input image that overlap active SE pixels become the input to a morphological operation. A morphological operation takes as input a set of pixels and computes a single value. This value is assigned to the corresponding center pixel in an output image having the same dimensions as the input image. As such, the SE defines the neighborhood of each input pixel. Since computed values are written to a separate image the order in which pixels are processed is irrelevant. Two fundamental morphological operations, *dilation* and *erosion*, are discussed next.

3.4.2 Dilation

Morphological dilation can be thought of as expansion and can be used to fill holes in shapes. Dilation is also useful for defining the neighborhood of a shape. Some criterion is used to classify pixels in an input image as *active* or *inactive* (Figure 3.9). Typically, if a pixel value exists in some set of values it is classified as active, otherwise inactive.

If the current input pixel is active the corresponding output pixel is assigned the input value. Thus, dilation does not affect active pixels. However, if the current input pixel is inactive and there is at least one input pixel in the SE that is active, the corresponding output pixel is assigned some value that identifies it as *dilated*. The size and shape of the SE determine which pixels are marked as dilated and can be varied to achieve the desired amount of dilation.

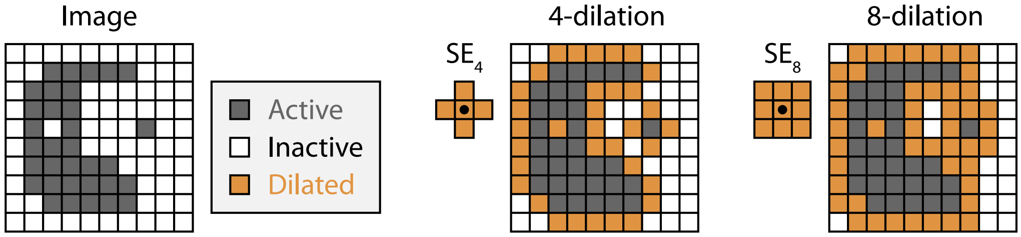


Figure 3.9: Morphological dilation using two different structural elements.

Dilations using two different structural elements are shown in Figure 3.9. Dilation using SE$_4$ is known as *4-dilation*. Similarly, dilation using SE$_8$ is known as *8-dilation*.

3.4.3 Erosion

Morphological erosion can be thought of as shrinking and can be used to eliminate thin protruding parts of shapes. The erosion process is similar to the dilation process in that input pixels are classified as *active* or *inactive* (Figure 3.10).

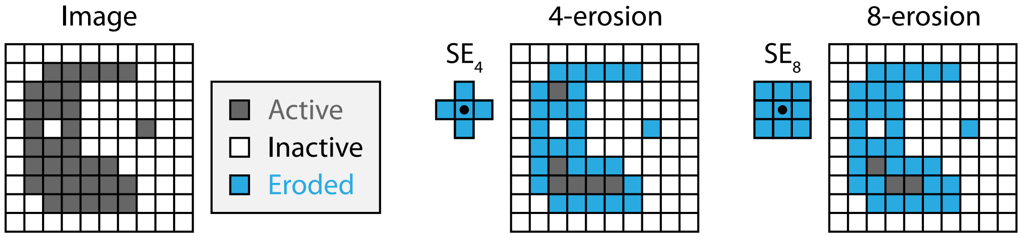


Figure 3.10: Morphological erosion using two different structural elements.

If the current input pixel is inactive the corresponding pixel in an output image is assigned the input value. Thus, erosion does not affect inactive input pixels. However, if the current pixel is active, all input pixel within the SE must be active in order for the corresponding output pixel to be active. If this is not the case the output pixel is assigned some value identifying it as *eroded*. Again, the size and shape of the SE used determine which pixels are marked as eroded. Erosion using SE$_4$ is known as *4-erosion*. Similarly, erosion using SE$_8$ is known as *8-erosion*.

3.4.4 Opening and Closing

Morphological *opening* and *closing* are two techniques that use combinations of dilation and erosion. Opening is defined as an erosion followed by a dilation. Typically, the same SE is used in both passes, but this is not strictly required. Opening has the effect of smoothing contours, removing small features, and eliminating thin protrusions. Closing is defined as a dilation followed by an erosion. While closing also has a smoothing effect on contours, it expands small features and may merge disconnected regions. Closing can be used to fill gaps or holes, but may lead to undesired connectivity changes between pixels in the digital image.

3.5 Flood Filling

The process of filling a region of a digital image with a certain value is referred to as *flood filling*, or *region filling*. It is possible to define flood filling in terms of morphological operations, using an arbitrary SE [GW02]. In practice, however, flood filling is rarely implemented in terms of morphological operations. Instead the two types of adjacency discussed in Section 3.3.2 are used.

Flood filling creates connected components of pixels with identical values in a digital image. First, a *seed* pixel is chosen (Figure 3.11). The goal is to propagate the seed pixel value to pixels that are connected to it. Pixels with values that exist in some set T are referred to as *target* pixels, all other pixels are referred to as *boundary* pixels. Target pixels that are connected to the seed pixel are assigned the seed pixel value. Note that target pixels are not necessarily connected to the seed pixel. Therefore, it is necessary to first determine which target pixels are connected to the seed pixel.

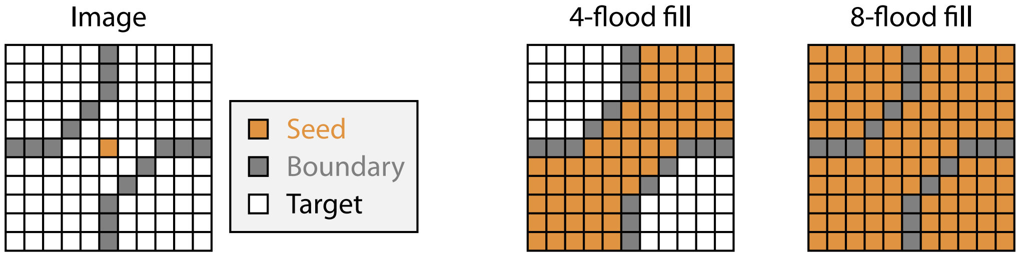


Figure 3.11: Flood filling using two different definitions of adjacency.

There are several ways of identifying which target pixels are connected to the seed pixel. A common approach is to use a queue or stack data structure to propagate adjacency outward from the seed pixel until all connected target pixels have been found. The terms *4-flood filling* and *8-flood filling* are used depending on the definition of adjacency (Figure 3.11).

In some cases it is useful to specify more than one seed pixel. Seed pixels are either specified explicitly in terms of pixel coordinates, or implicitly in terms of some value, in which case all pixels with that value become seed pixels. It is assumed in this thesis that all seed pixels have the same value. In the case of multiple seed pixels adjacency propagation is done simultaneously, yielding the same results as if performing separate flood filling passes for each seed pixel. This is intuitive, since two seed pixels are either in the same connected component or not. When they are in the same connected component that component would have been filled anyway, and when they are not, the separate adjacency propagations will never interact, by the definition of connected components.

3.6 Connected Component Labeling

While flood filling creates connected components in a digital image, *Connected Component Labeling* (CCL) identifies existing connected components. As with flood filling, CCL can be expressed in terms of morphological operations. However, for clarity, this section provides a more intuitive description. A digital image often contains more than one connected component and quite often separate connected components represents different objects in an image. The extraction of connected components is, therefore, a fundamental task in computer vision and is the first step in many automated object detection approaches (e.g. [RJ99, SG00, MCUP02]).

A label is a symbol that uniquely names an entity. While character labels are possible, positive integers are more convenient and are most often used to label the connected components of a digital image. The pixels of an input image are classified as *active* or *inactive* based on some criterion (Figure 3.12). CCL outputs an image where the value of each correspondingly active pixel is the label of its connected component. Correspondingly inactive pixels are assigned some null label. Depending on the connected component definition the terms *4-CCL* and *8-CCL* are used (Figure 3.12).

There are a number of different algorithms for performing CCL. A simple approach is to consider one label at a time and use flood filling to find the connected components of that label.

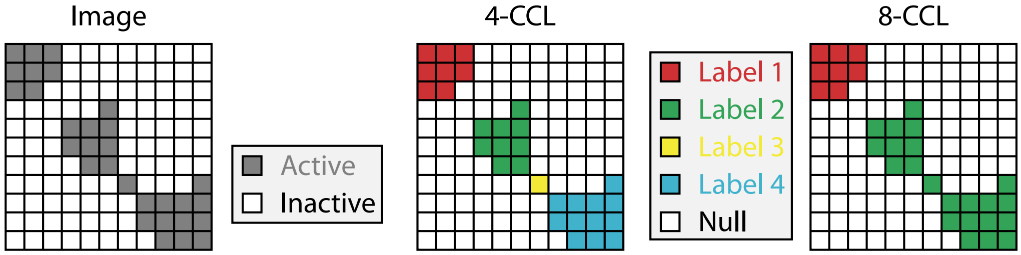


Figure 3.12: The active connected components of an image are assigned unique labels, here shown as different colours. Two different versions using different definitions of connected components are shown.

Although straight-forward to implement, this approach is highly inefficient since the number of flood fills required is the same as the number of connected components in the image. More sophisticated approaches use single or dual-pass techniques that record all labels simultaneously, using equivalence tables to merge labels progressively. For implementation details the reader is referred to the works of [RP66, ST88, DST92, SS01, SHS03].

3.7 Summary

Digital images are used to represent spatially varying data in two dimensions. Values are stored at discrete locations, referred to as pixels, on a regular grid. While digital images often store colour information at pixels, it is possible to store any type of information. Further, an image can be considered as a bounded plane in a three-dimensional coordinate system, which allows the mapping of points to pixels. Mapping is useful for computing pixel values from ALS points. The use of digital images to represent properties of ALS points has significant advantages in that the regular structure of pixel grids is easier to work with than unstructured points in three dimensions. An example of this is elevation images, where pixel values are proportional to the elevations of mapped ALS points.

Besides providing regular grid structures, there is a large body of existing processing techniques available for digital images. Image morphology is a collective name for a set of tools used to process shapes in digital images. Fundamental morphological operations like erosion and dilation are often used in digital image processing related to ALS data. Other useful image processing techniques include flood filling and connected component labeling. Flood filling is used to fill an area with a certain values, whereas connected component labeling is a technique for identifying areas with a certain value.

The following chapter discusses the field of urban modeling, where the goal is to create large-scale surface models from point data. Arguably, digital images play an important role in urban modeling and are commonly used in urban modeling processes.

Chapter 4

Urban Modeling

Before computers were widely available urban planners and civil engineers would create city models from wood, or other light-weight materials, using elaborate manual measurements. Advances in computer technology present opportunities to work with digital models of cities. Digital models enable tools for visualization and simulation beyond the capabilities of real-world miniatures. However, digital models of cities are not readily available and must be created using tools designed for the task. Manual survey measurements are impractical on a city-scale and efforts to collect spatial data for entire cities have turned toward various sensor approaches to improve efficiency. Urban models in this thesis are understood to be computer models. At present, applications revolving around urban scenarios are carried out on computers in an overwhelming majority of cases.

Large-scale models of cities are used in a wide range of applications, including such seemingly disparate subjects as noise prediction (e.g. [Ste01]) and disaster mitigation (e.g. [VMY04]). Increasingly, large-scale computational efforts are being applied to urban planning and disaster management. These range from optimizing single-incident fire department responses [Kev03] to predicting and mitigating regional flooding effects [HWK05].

Modern urban modeling approaches use one or more sensors to acquire accurate spatial data on a city-scale. The goal in urban modeling is to create digital models of cities from acquired sensor data and this chapter gives an overview of existing approaches in this field. Cityscapes are geometrically complex, containing objects of widely varying sizes and shapes. Urban geometry is dominated by buildings, which are typically the objects of interest in urban modeling, along with other large, permanent man-made features [För99, HYN03]. Other types of objects have been studied, including road networks [CR07, Elb10], power lines [MB04], and urban vegetation [RHHP08]. This thesis, however, focuses on the modeling of buildings. Several overview papers on urban modeling have been published and are complementary to the material presented in this chapter [GS00, Shi01, HYN03, VGSR04]. Additionally, the recent book by Vosselman and Maas [VM10] discusses current state-of-art techniques for urban point data acquisition, both aerial and terrestrial, and includes an entire chapter dedicated to building extraction from aerial point data [Bre10].

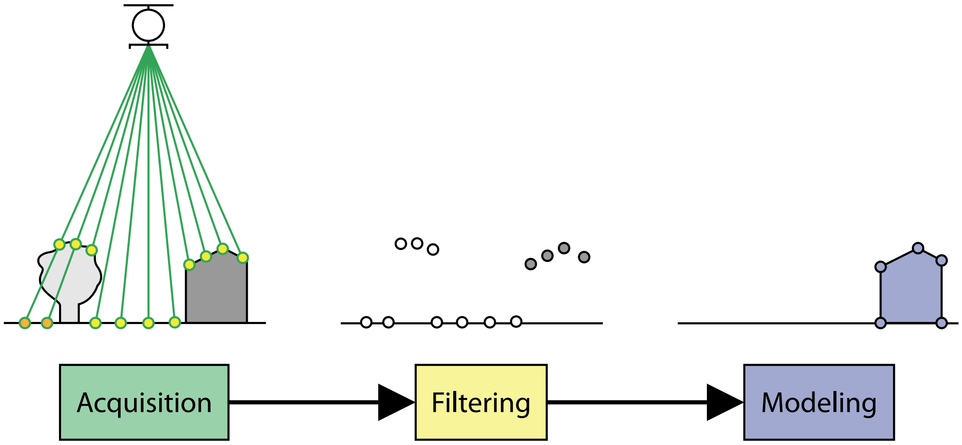


Figure 4.1: The three major steps of urban modeling. *Left*: Scan points (yellow and orange) are acquired by measuring distances to surfaces from a moving platform. *Middle*: Scan points are filtered and points on buildings (gray) are extracted and separated from non-building points (white). *Right*: A surface model is created from grouped building scan points (blue).

Urban modeling consists of three major steps (Figure 4.1). *Acquisition* is the task of collecting geometric data in some raw format on an urban scale. This thesis focuses on the use of *Aerial Laser Scanning* (ALS) for large-scale acquisition of point data in an urban environment. *Filtering* refers to the task of classifying and grouping acquired data so that objects of interest (e.g. buildings) can be modeled. High degrees of automation are required in the filtering step to process the vast amounts of data acquired in large-scale urban modeling. *Modeling* is the task of creating surface models from filtered data. Surface models are created with a specific application in mind and a suitable level of detail is chosen to meet the requirements a particular application.

For instance, visual models require high levels of detail in order to convincingly portray buildings. Other applications, used for simulation purposes may use cruder models where structural properties are more important than ornamental details. As with filtering, the modeling step requires high degrees of automation because of the large number of surface models that need to be created [Bre00, BHF01]. As well as rapidly coping with huge data sets, automated modeling approaches provide some level of uniformity in the generated models.

In urban modeling based on ALS acquisition the majority of processing time is spent in the filtering and modeling steps [Flo99, Flo01a, Flo01b]. Thus, these two steps have been the predominant targets when it comes to optimizing throughput by proposing fast, automated approaches. Acquisition, filtering, and modeling are further discussed separately in the following sections.

4.1 Acquisition

In urban modeling, acquisition is the task of acquiring geometric data on a city-scale. Acquisition of geometric data for urban landscapes is challenging because of the sizes of cities and the complexity of the geometry found therein. Acquisition of point data for a single building can be achieved with *Terrestrial Laser Scanning* (TLS), as discussed in Chapter 2. However, since it takes several hours to scan each building, this approach is severely limited by time-factors when the number of buildings becomes large. Additionally, since TLS is ground-based it is often difficult to acquire point data for roofs and other architectural features that are only visible from above. Other ground-based approaches, using truck-mounted sensors (e.g. [FZ03, FZ04]) have been proposed. However, these approaches require traveling along streets, which in the case of sudden infrastructural changes (e.g. earthquake or blast explosion), may be impossible.

At present, the most promising acquisition approaches for covering the vast areas in question have been based on aircraft-mounted sensors. However, aerial acquisition approaches have provided *elevation data*. Elevation data is characterized by the fact that points are defined as $(x, y, f(x, y))$ triplets, where the elevation (z) component is treated as a function of the horizontal components (x, y). Since a function $f(x, y) \in \mathbb{R}$ is required to evaluate to a unique value for any distinct pair (x, y), it is only possible to associate a single elevation value with each horizontal position. An important consequence of this is that elevation data cannot represent details on vertical surfaces. This severely limits the opportunities for fully three-dimensional geometric modeling of buildings. Up to this point, aerially acquired geometric data for cities has not consistently included data for vertical surfaces, which has meant that subsequent filtering and modeling approaches have focused on elevation data.

Besides *Aerial Laser Scanning* (ALS), which was described in Chapter 2, photogrammetric methods based on matching of stereo image pairs, acquired from an airborne platform, have been used to derive elevation data for large areas [Kra07, Hir08]. Elevation data produced from stereo image matching has suffered from poor accuracy in the vertical direction, leading to problems in areas with sharp discontinuities [BM99]. Additionally, photogrammetric methods do not deliver point data directly, since matching of features in stereo image pairs is required as an intermediate step. As such, ALS allows faster turn-around times for acquisition, which is critical in emergency response situations. Also, ALS points are acquired in fixed patterns, e.g. scan lines, while photogrammetric data is typically unstructured.

The colour information provided by aerial imagery may prove useful in further steps, such as building extraction, and may also be used to add textures to building models. Also, photogrammetry is often reported as been slightly cheaper than ALS [Ack99]. The main reasons for this are that flying times are shorter because photogrammetric sensors cover a significantly wider area beneath the aircraft, reducing the number of passes that have to be made over a scene. Although, it may not be the case that photogrammetry is cheaper if the post-processing of matching stereo images is included. However, photogrammetric methods are constantly being improved. Recent photogrammetric methods show results comparable to ALS, both in terms of point density and accuracy [Klu11].

As is typical in a field where there is overlap in data gathering strategies, hybrid approaches for acquisition have evolved. The most common form of hybrid method in ALS data processing is to incorporate color information that was acquired simultaneously with the laser scan data [HYNP04, RB02, RB03]. Also, multi-sensor approaches introduce the need to merge data in different formats. If fast acquisition times are required such merging is a potential bottle-neck.

Having described how geometric data can be acquired on a city-scale, the following section discusses filtering of such data.

4.2 Filtering

In urban modeling, the most common filtering task is to separate points acquired on buildings from other points. This section focuses on building extraction in ALS data. Since ALS acquires points on all surfaces beneath the aircraft it becomes necessary to distinguish points captured on buildings from other points. As it turns out this is a challenging problem. One reason for this it is difficult to find a strong definition of what a building is.

4.2.1 Automatic Building Extraction

Automatic building extraction requires only minimal manual input. This is a desirable feature, since current data sets are often extremely large. A robust method must detect any type of building, and even within the same city buildings have different sizes and proportions. Added to this, certain architectural extravagant buildings are fairly unique, making them difficult to place in building categories.

Haala and Anders [HA97, HBA98], as well as Vosselman and Dijkman [VD01], use existing ground plans to determine the locations of buildings in ALS data. However, building plans are not always available and up-to-date. The latter is almost guaranteed not to be true after a natural disaster such as an earth-quake [VMY04, LP06].

Zhou and Neumann [ZN08] use a machine learning approach to detect buildings in a large urban data set and impressive results are shown. However, training times for the machine learning part are not given.

Haala and Brenner [HB99] and Haala [Haa99] use color images and ALS elevation data to extract buildings in a rural scene. Intensity measurements along with elevation differences are used by Matikainen et al. [MHH03] to separate buildings from vegetation in a rural area. Satellite images together with elevation data acquired with ALS were used by Sohn and Dowman [SD03, SD07] to extract buildings, but their methods rely on a large number of user parameters and results are shown on a small area with clearly separated buildings.

Morgan and Tempfli [MT00] resample ALS data into an elevation image and use morphological operations to separate buildings from bare earth. Their method, however, requires a priori knowledge of building areas. Verma et al. [VKH06] use eigen-analysis of point neighborhoods to detect flat regions in a point data set. Their method, however, relies on large, flat roof shapes, which occurs mostly in rural areas. It was claimed by Maas and Vosselman [MV99] that elevation thresholding along with minimum building height and area parameters are sufficient to extract buildings for a large urban region, but they do not provide any further implementation details. Alharty and Bethel [AB02] created separate elevation images from first and last echoes and the difference between these was thresholded to detect the buildings of a university campus. Elaksher and Bethel [EB02] used minimum filters to identify building points, but demonstrate results for only two separate buildings, which are not in an urban context. Morphological opening was used by Rottensteiner and Briese [RB02] to identify buildings in an urban area. These techniques were applied to elevation images where bare-earth had been subtracted. Sithole and Vosselman [SV04] compare eight different methods for extracting bare-earth and note that these methods are unsuitable for complex urban areas. Dorninger and Pfeifer [DP08] propose an automated building extraction method based on clustering of estimated surface normals for points. Their method shows positive results for an urban area but requires an initial manual step in the building extraction.

Building features have been detected in individual scan lines. This was most recently done by Hebel and Stilla [HS08]. However, their approach is experimental and results are shown on a very small scene without much evaluation. Axelsson [Axe99] attempts to classify building points based on second derivates within scan lines. Results are shown, however, for a single building block and the applicability of the method is unclear. The profiles used by Sithole and Vosselman [SV03] are conceptually similar to scan lines. Line segments based on elevation and slope thresholding were connected to identify features in elevation data. Results are shown for very small scenes and the task of choosing the two thresholds is not evaluated for large areas.

As with most automated systems, building extraction must be verified after completion. In some cases existing ground plans can be used for this purpose. However, ground plans are not always available. Also, the format of available ground plans is not always suitable for comparison with extracted results. In practice ground plans can be estimated from visualizations of the data in the form of images. This, however, requires that buildings are clearly identifiable in the visualizations. Next, a brief discussion on semi-automatic building extraction is provided. Thereafter, building modeling is discussed.

4.2.2 Semi-automatic Building Extraction

Semi-automatic building extraction relies upon user input for the identification of buildings. The user is presented with a view of the data and the task is to select regions corresponding to buildings. Semi-automatic building extraction is suitable for small number of buildings, but is unfeasible on a large scale. Additionally, providing a suitable view of the data is challenging. In particular, for dense urban areas, where buildings are spaced closely together, or even adjoined, identification can be challenging.

Assuming that points acquired on buildings are somehow extracted, the next step is to create a surface model for each building. This topic is discussed in the following section.

4.3 Building Modeling

Modeling is the task of creating usable surface models from extracted building points. Most modeling approaches assume that points are separated into groups, where each group of points is sampled on a single building. As such, each building is modeled individually. The effectiveness of urban modeling depends largely on the geometric accuracy and detail of the building models. Commercial software tools for building modeling require, generally, a high degree of human interaction [DP08]. The techniques considered in this section are automatic, which as discussed previously, is required for modeling on a city-scale.

A substantial amount of research has gone into surface reconstruction from point data in the computer graphics community (e.g. [Hop94, KBH06]). However, there are two main issues with using such techniques in urban modeling [MN03]: (1) they assume dense regularly sampled points; and (2) they produce models consisting of a large number of triangles. ALS points are not regularly sampled and, furthermore, point sampling is not dense and may even be absent on vertical surfaces. Moreover, surface reconstruction techniques used in computer graphics typically attempt to connect all input points by defining suitable triangles. This results in large numbers of triangles, which added together for many buildings is prohibitive in terms of interactive rendering. Also, irregular point densities cause building appearances to vary significantly. Dedicated urban modeling surface reconstruction, therefore, attempts to create more thematic models using only the major planes of buildings [CC08]. The discussions below focus on dedicated building modeling for urban modeling purposes.

There are two main types of building modeling: (1) *model-driven*; and (2) *data-driven*. In model-driven approaches a set of user-provided geometric primitives are fitted to the point data. Data-driven approaches attempt to identify the presence of building structures based on the provided points. These two approaches are discussed in more detail below.

4.3.1 Model-driven Techniques

Model-driven techniques use a fixed set of geometric primitives, which are fit to the point data. Such techniques can be effective when point data is sparse, since the fitting of geometric primitives does not require complete overlap. Also, model-driven techniques are robust in the presence of noise and are often simple from a computational point of view. Further, model-driven techniques provide topologically correct models since primitives can be combined in valid patterns. However, model-driven approaches are limited by the number of user-provided primitives, which are often insufficient for describing all types of buildings.

You et al. [YHNF03] generate building models based on the fitting of superquadrics and demonstrate results for a university campus. However, these models lack in overall detail and contain no details for building walls. Maas and Vosselman [MV99] use a standard house type as the base primitive and thereafter add roof details, such as dorms, by intersecting planes. Four different house primitives are used by Haala et al. [HBA98]. These can be combined to form more complex buildings. However, these models lack details on vertical surfaces, since the method operates on elevation data. Further, Hu et al. [HYNP04] use a combination of simple primitives, such as cubes and planes, with curved surfaces to reconstruct a small number of buildings.

The techniques mentioned above have in common that details on building walls are ignored. Instead walls are often simply extruded from the ground-plane using provided or computed building outlines. Further, the number of primitives used has a large impact on the generated models in terms of diversity, since a small number of primitives can only be combined in fairly limited ways. The techniques discussed next do not use primitives, but rather try to define surfaces based solely on the provided data.

4.3.2 Data-driven Techniques

Data-driven techniques derive surface directly from point data and are capable of modeling arbitrary building shapes. In general, data-driven approaches are more flexible than model-driven approaches, but are often sensitive to noise in the input data. Many data-driven techniques operating on ALS data reconstruct roof shapes simply extrude walls based on the outlines of these shapes. As in the model-driven techniques, this produces building models that lack geometric details on building walls.

A common approach is to cluster points on building roofs based on their normal orientation. As mentioned, normal estimation is a noise-sensitive procedure, which may introduce errors. Points with similar normals are assumed to belong to the same roof plane. Intersection between roof planes then give the overall roof surface, and in some cases also provides the building outline. Building walls are simply extruded up to the roof height along the building outline to produce box-like buildings with detailed roofs. A common issue in data-driven approaches is enforcing the regularity often found in buildings for structural reasons. While it is possible to control regularity and symmetry on model-driven techniques this is harder with data-driven techniques, where surfaces are created separately before being intersected.

Zhou and Neumann [ZN10] created impressive buildings for a large urban area. However, these models are admittedly of an extruded nature and contain no details on building walls. Numerous authors (e.g. [DP08]) use an approach where the major planes are fitted to point data. Thereafter, these planes are intersected in order to derive a closed surface representation. As mentioned above, these methods often suffer from irregular sampling and cannot always guarantee building symmetry.

An interesting alternative used to create impressive looking imaginary cities is briefly discussed next. The reason for mentioning these approaches is that there has been some interest in adapting such methods for modeling real cities lately.

4.3.3 Procedural Modeling

In procedural modeling city models are created from grammars, describing rules for building appearances. Procedurally create urban models are visually realistic and are capable of producing large models very rapidly [PM01, WWSR03, LWW08]. However, although realistic, procedural city models do not portray existing cities. In some cases, such as movies or computer games, this is not a strict requirement, but for urban planning it clearly is. For generating long lost cities procedural modeling may be the only alternative, as in the cases of Pompeii [MWH*06] and Xkipché [MVW*06]. The main advantage of procedural modeling is the enforced regularity imposed by the grammars. Symmetry and proportions often found in real architecture for aesthetic, economic, and structural reasons can be enforced using appropriate grammars.

Interestingly, regularization approaches similar to grammars have been used to enforce regularity in models created from a combination of ALS and TLS data. Haala et al. [HBK06] use cell classification on building facades to achieve regularity and symmetry in window openings. Becker and Haala [BH07] use an approach similar to the previous one, adding the incorporation of facade images.

4.4 Summary

Urban planners and civil engineers require digital city models in a wide variety of applications, including both visualization (e.g. virtual cities) and simulation. Urban modeling is the field of creating digital city models and most often focuses on the creation of building models. There are three major steps involved in urban modeling: (1) *acquisition*; (2) *filtering*; and (3) *modeling.*

Acquisition involves acquiring spatial data for urban areas and laser scanning (Chapter 2) is used extensively for this task. With ALS it is possible to rapidly acquire point data for an entire city. However, currently ALS produces elevation data, which does not capture details on building walls.

Filtering is necessary to detect individual objects, most often buildings, in vast amounts of point data. This task requires a high degree of automation due to the enormous amounts of data acquired for cities. Digital images (Chapter 3) are commonly used in filtering of ALS point data. Points are mapped to the pixels of an image and pixel values are assigned according to some value, typically point elevation. Image processing techniques such as thresholding and region growing are then applied to the image in order to identify shapes corresponding to buildings. Currently, because ALS produces elevation data, building walls cannot be accurately detected, since overhanging roof parts occluded these features in top-down views. In some cases it is possible to use existing ground plans to identify buildings in ALS data, but such approaches are limited by the availability of such plans. The output of filtering is a separate set of points for each building, and modeling is based on taking each such set of points and creating a surface model from it.

Modeling is the task of creating digital surface models from point data. Most often the surface models created from point data are aimed at visualization and existing tools focus on creating such models. In several cases, the detection and modeling of roof shapes is used to create extruded models, which contain no details on vertical surfaces. Such models are not fully three-dimensional and are not suitable for the purpose of simulating building response, since openings, such as windows, on walls play a critical role in the load-bearing capacity of a building.

This chapter has presented existing approaches for urban modeling and the three major steps of such processing have been identified. The following chapter presents a type of simulations model commonly in building simulations.

Chapter 5

Solid Models

This chapter gives an introduction to solid models, which are used extensively to represent building geometries in engineering simulations. The purpose of this chapter is to provide the background necessary to describe a novel approach for creating solid models of building walls directly from point data, presented in Chapter 13. The main idea in solid modeling is to define the surface of an object such that the inside of the object is completely bound by the surface. Solid modeling is an interdisciplinary field, covering topics such as numerical analysis, approximation theory, applied mathematics, computational geometry and databases. A complete treatise of solid modeling is, therefore, beyond the scope of this thesis and the reader is referred to [Sab68, BFK84, RR84, Sed85, SP86, Req88, Far90, Req92, RR99] for in-depth discussions.

A solid model is a digital representation of the geometry of an object [RR99]. The goal of solid modeling is to represent, manipulate and reason about the three-dimensional shapes of solid physical objects using computers [HR96]. Solid model representations are unambiguous, complete and detailed digital approximations of the geometry of a single object or a collection of interacting objects, such as a car engine [RR99].

Solid modeling provides the fundamental tools for representing a large class of objects, and for performing the geometric computations required by various applications. This chapter focuses on the representation of buildings, with applications in engineering simulations and urban planning. Solid models are also widely used in the manufacturing industries, where precise models of parts are created using *Computer-Aided Design* (CAD) software. Designers manually specify points, curves, and surfaces, and connect them to define representations of objects.

Efforts to generate solid models automatically from external sources are of particular importance, since the cost of manually designing solid models of existing objects or scenes is far greater than other associated costs, such as hardware, software and personnel training [RR99].

Simpler shapes, such as blocks or cylinders, can be combined to represent complex objects. This approach is often used to model buildings, which tend to consist of regular, fairly sharp features [SMS04]. Such modeling schemes are referred to as *spatial decomposition* and are used extensively for creating solid models of buildings. The following section describes the type of solid models used in building simulations.

5.1 Solid Models in Building Simulations

Solids may be represented either exactly or approximately by a variety of space decomposition schemes. The entire three-dimensional space, or just the set that corresponds to the solid, is partitioned into non-overlapping three-dimensional regions referred to as *cells* [Sam90]. Spatial decomposition schemes may differ in the restrictions they impose on cells. These may be polyhedral or bounded by curved surfaces and are usually connected [LLM06]. Cells may be further restricted to be axis-aligned rectangular blocks, or regularly spaced cubes. The regular nature of spatial decomposition is attractive for modeling buildings.

Building are complex objects. The exteriors of buildings are largely varied, constrained only be the imaginations of architects. Moreover, interior parts of buildings may be equally complex. For these reasons buildings are rarely modeled in extreme detail. Instead, focus lies on trying to create models that behave realistically in simulations. As such, the most crucial parts to model are the load-bearing ones. For a vast majority of buildings walls are the most crucial load-bearing components [Hon11]. Therefore, simulations involving buildings focus on the modeling of building walls. Moreover, building walls are often modeled one at a time in order to reduce the complexity of the models.

Figure 5.1 shows an example of a solid model of a building wall. The cells of this model are defined and connected in such a way that they approximate the geometry of the building wall. Note that no cells are present for door and window openings. The reason for this is that openings are not load-bearing. Thus, the load-bearing parts of the wall are approximated by

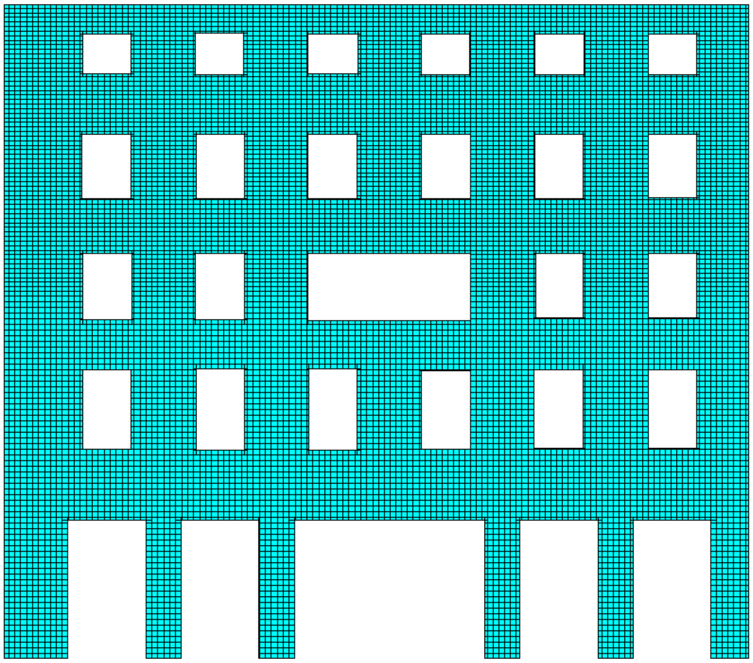


Figure 5.1: Solid model of a building wall.

connected cells. Individual cells are defined as rectangular blocks and must share a face in order to be connected.

In summary, solid models used in building simulations are conceptually similar. They consists of connected rectangular cells, which are defined in positions such that they together approximate the geometry of the building wall. In particular, it is of importance to model openings on building walls, since these are crucial factors in the load-bearing capabilities of a wall.

Chapter 6

Problem Statement

As was shown in Chapters 2–5 computational modeling of urban areas involves successfully combining acquisition, building extraction and solid modeling. In order to satisfy the goals of this thesis improvements are required across all of these steps. First, Chapters 2–5 are summarized, providing a concise description of topics covered thus far. Thereafter, motivations for carrying out the research in this thesis are given. In the subsequent problem statement the goal of this thesis is defined and the short-comings of existing tools with respect to this goal are discussed. Finally, a set of tools required to overcome the short-comings of existing tools is proposed, concluding Part I of this thesis. Detailed descriptions of these tools follow in Part II.

6.1 Background Summary

Chapters 2–5 introduced topics relevant to the problems addressed in this thesis. Key concepts from the aforementioned background chapters are summarized in this section. This is done to provide a concise and comprehensive understanding of the fields related to the work in this thesis.

Laser scanning (Chapter 2) is a remote sensing technology based on the principles of *Light Detection and Ranging* (LIDAR). In *Aerial Laser Scanning* (ALS) point data is acquired from an aircraft moving over the scene. When using *Terrestrial Laser Scanning* (TLS) point data is acquired from multiple ground-level positions. While TLS provides detailed point data for a single object, ALS is used to acquire point data for large areas. Point data acquired with ALS is typically less dense than that acquired with TLS, thereby trading coverage for data quality.

Both ALS and TLS are capable of acquiring vast amounts of point data. The regular nature and compactness of digital images (Chapter 3) is useful for visualizing large quantities of ALS data. Points can be mapped to pixels and pixel values computed from features in the mapped point data. Such images are used as visualization tools, but also as the starting point for filtering operations, such as building extraction. The regular sampling of digital images and the large body of existing image processing techniques make digital images useful in many applications related to ALS.

Laser scanning provides discrete point samples on surfaces and ideally these point samples are spaced closely enough to allow faithful reconstruction of the scanned surfaces. Urban modeling (Chapter 4) is the field of study which focuses on creating surface models of buildings over large areas, ideally for entire cities. Typically, urban modeling involves three steps: (1) *acquisition*; (2) *building extraction*; and (3) *surface modeling*. ALS is often used to acquire point data for large urban regions. Because of the enormous volumes of data acquired for large areas, it is desirable to perform building extraction with a high degree of automation. However, architectural diversity makes it difficult to define robustly what a building is. The output of urban modeling is typically a triangulated model suitable for visualization. For the purposes of building extraction and surface modeling, current urban modeling tools assume that ALS provides elevation data. Although this simplifies the surface modeling step, it means that models are of an extruded nature, rarely containing details for building walls.

Solid models (Chapter 5) are used by civil engineers as representations of physical objects in various types of simulations. The main difference between solid models and other types of surface representations is that solid models support simulations. The type of solid models used in building simulations consist of connected volumetric primitives, where it is common to use regular cubic primitives.

6.2 Motivation

Increasingly, large-scale computing efforts are being applied to urban planning and disaster management. Urban planners and civil engineers require large-scale three-dimensional geometric models of urban areas for a wide range of applications. In particular, solid models (Chapter 5) are commonly used by civil engineers to simulate building degradation in a number of different scenarios, including earthquakes, tunnel blasting, microclimate modeling, and vehicle-induced vibration. However, simulations are often limited to a small set of buildings due to the resources required to generate solid models and their subsequent computational expense. This leads to critical gaps in the analysis of large-scale effects, such as earthquakes.

At present, solid models are created manually from as-built surveying measurements or base maps. Acquisition of as-built measurements through surveying is costly and time-consuming, and substantial manual effort is required to create solid models from such data. In some cases governmental agencies maintain base maps of their jurisdictions, but the majority of existing base maps are limited to horizontal representations, requiring additional measurements to be made for vertical features. As such, the time and expense involved in generating solid models severely limits the availability of such models, thereby limiting the scale of simulations.

A higher degree of automation in the creation of solid models would increase the availability of such models, thereby supporting large-scale simulations. Additionally, this would enable simulations in scenarios that are time-critical or limited by economical factors. However, existing tools are not satisfactory for the task of large-scale creation of solid models of buildings. Therefore, it is necessary to develop a new set of tools to support this task. The following section clearly states the goal of this thesis and explains why existing tools fail to achieve this goal.

6.3 Problem Statement

As argued in the previous section, higher degrees of automation in the creation of solid models would be of significant benefit to the urban planning and civil engineering communities. The goal of this thesis is to create three-dimensional solid models of buildings on a city-wide scale directly from point data acquired by laser scanning. Existing urban modeling tools were designed for other purposes and it will be described how they are not satisfactory for achieving the goal of this thesis. Critical short-comings of existing tools are identified in this section and in the following section a new set of tools for achieving the goal of this thesis is proposed.

As was shown in Chapter 4 existing urban modeling tools involve three major steps. For the purposes of demonstrating why existing tools are not satisfactory for the goal of this thesis a fourth step, which involves converting triangulated models into solid models is added. The four steps are listed below:

1. Acquisition.
2. Building extraction.
3. Surface modeling.
4. Solid modeling.

Based on these four steps the following discussions will identify the short-comings of existing tools in relation to the goal of this thesis. Each step is discussed separately below.

6.3.1 Acquisition

With ALS it is possible to rapidly acquire elevation data for an entire city. However, such data does not include details on building walls. TLS can be used to acquire points on building walls, but does not scale well to an entire city. As such, existing acquisition techniques are not capable of rapidly acquiring fully three-dimensional urban point data, which is essential for allowing the creation of three-dimensional solid models of buildings on a large scale.

6.3.2 Building Extraction

Current methods for automatically extracting ALS points captured on buildings have not been tested for urban areas with diverse architecture and closely spaced buildings. In particular, existing methods operate on elevation data and are not able to take advantage of points acquired on building walls. Also, most existing methods do not address issues of scale, where large ALS data sets must be subdivided into smaller parts in order to fit into the memories of standard workstations. Further, since existing automatic methods cannot detect all buildings, semi-automatic building extraction methods are required in order to handle cases where automated approaches fail.

Additionally, an imaging technique capable of visualizing three-dimensional ALS point data is required to support identification of important features, such as building walls, in large ALS data sets. However, existing imaging techniques, most notably elevation images, are not capable of visualizing three-dimensional point data. The reason for this is that pixel values are derived from single points. While this is effective for elevation data, it fails to take advantage of the additional information present in three-dimensional data. This weakness of existing imaging techniques manifests itself both in semi-automatic and automatic building extraction methods. Semi-automatic building extraction requires high-quality visualizations in order for users to be able to identify regions for selection. Similarly, visualizations based on the provided ALS data are used to evaluate and analyze automatic extraction results in the absence of ground-plans.

6.3.3 Surface Modeling

Surface reconstruction from sparse three-dimensional point data is a complex and error prone procedure because of ambiguities in point connectivity. Conversion of three-dimensional point sets into triangle meshes is a well-studied topic in computer graphics. However, most such methods assume regularly sampled and closely spaced points, neither of which is true for points acquired with ALS. Large-scale urban modeling tools triangulate roof-shapes and extrude walls from building outlines. Such models are not fully three-dimensional and rarely information for building walls. Further, while standard triangle meshes are suitable for visualization purposes, they are not suitable for performing the type of simulation used for buildings. Such simulations require solid models of the type described in Chapter 5. Conversion from triangle meshes into solid models may be possible, but errors introduced in the surface reconstruction would be propagated to the solid models. As such, triangulated surfaces are not strictly required for engineering purposes.

6.3.4 Solid Modeling

Efforts to create simulation models from ALS point data have largely focused on flood plain mapping applications [CMHB03, RP04, HR05, MHH*08]. However, such models are created from sparse elevation data, and these methods are not easily extensible to fully three-dimensional data sets. Further, the created models are derived from triangulations and are not volumetric, as is required for solid models. Conversion from triangulated surface representations to solid models is error prone, since triangulation methods are likely to introduce errors when applied to sparse point data. To date, no methods for generating solid models of building walls directly from laser scanning point data have been published.

6.3.5 Summary

This section has stated the goal of this thesis and discussed the short-comings of existing tools with respect to this goal. The goal of this thesis is to create three-dimensional solid models of buildings on a city-wide scale directly from point data acquired by laser scanning. For this task three-dimensional point data for entire cities needs to be acquired in an efficient way. Existing laser scanning techniques have not been capable of rapidly providing three-dimensional data for entire cities. More specifically, TLS is not scalable to an entire city, while ALS traditionally has not consistently provided point data for building walls. Further, most existing building extraction methods have not been tested for densely built-up urban regions. Finally, existing modeling approaches have focused on generating visual models. Engineering models have been created for flood plain mapping simulations, but these methods do not extend naturally to three-dimensional solid modeling of buildings. In the following section a new set of tools is proposed in order to overcome the short-comings of existing tools with respect to the goal of this thesis.

6.4 Contributions

Existing tools are not satisfactory for achieving the goal of this thesis, as shown in the previous section. In order to address the short-comings of existing tools a new set of tools is proposed. Note that the order in which tools are listed below is a logical sequence for describing these tools, and does not necessarily correspond to the standard workflow used in urban modeling. This thesis contributes to existing knowledge through the description and evaluation of the following tools:

1. An ALS acquisition strategy that provides three-dimensional point data for urban areas.
2. A technique for processing large ALS data sets free from memory constraints.
3. An imaging technique capable of accurately visualizing three-dimensional ALS point data of urban scenes.
4. An semi-automatic building extraction tool particularly suited for extracting building walls.
5. An automatic building extraction technique for three-dimensional ALS point data of urban areas.

6. A method for directly converting point data into solid models.

Detailed descriptions of the tools listed above are given in Part II of this thesis. In conclusion, Part I has provided the background necessary to clearly state the problems addressed in this thesis. Existing tools have been discussed in relation to the goal of this thesis and the short-comings of existing tools have been discussed. In response, a new set of tools necessary to remedy these short-comings has been proposed. Remaining parts of this thesis provide the details of these new tools and evaluate their effectiveness.

Part II

Methodology

Chapter 7

Overview

Part I presented the background necessary to formulate the goals of this thesis and identified the short-comings of existing tools with respect to those goals. A set of new tools, required to achieve the goals of this thesis, was proposed in Section 6.4. Part II of this thesis describes the proposed tools in detail. Thereafter, results are presented in Part III.

Chapters 8–13 in this part provide detailed descriptions of the proposed tools. First, however, this chapter provides an overview by giving concise descriptions of each tool in the following sections. Note that tools are described in an order that is logical for explanation, which does not necessarily correspond to the order in which these tools might be used.

7.1 Urban ALS Flight Paths

Chapter 8 describes an approach for acquiring three-dimensional point data of urban areas using *Aerial Laser Scanning* (ALS). Through sampling analysis it is shown that horizontal surfaces are best scanned directly beneath the aircraft, whereas vertical surfaces are best scanned at the edges of flight strips. Additionally, low aircraft altitudes and large scan angles are beneficial to the acquisition of building walls in urban areas. It is possible to design flight paths to exploit these features. Guidelines for designing flight paths for urban areas in terms of flight strip overlap and flight track orientation with respect to the major street grid are presented.

The approach described in Chapter 8 was used to acquire a large ALS data set for a study area located in central Dublin, Ireland in 2007. This data set is presented in Part III, where it is shown that substantial parts of the building walls were acquired. Also, the Dublin data set is used in the evaluation of the other tools described in this part. Next, a summary of the framework used in this thesis for processing large ALS data sets is given.

7.2 ALS Scan Line Analysis

Chapter 9 describes a framework for processing large ALS data sets. An important feature of this framework is the fact that ALS points are grouped by scan line. The framework differs from existing approaches by organizing ALS points according to time, as opposed to horizontal location. Both ALS points and flight path points (Chapter 2) have accurate time-stamps, allowing aircraft position and orientation to be considered in relation to the scanned points. A tree-structure is used to represent aircraft motion over time, enabling efficient queries using ALS point time-stamps. Further, knowledge of the aircraft position and orientation allows offset angles for ALS points to be computed. Thus, ALS points can be grouped by offset angle, allowing them to be treated as pulse echoes. Thus, it becomes possible to identify pulses that have no echoes, and to approximate *missing echoes* for such pulses. The framework for scan line analysis is used in both visualization and building extraction, as described in the following sections.

7.3 ALS Occlusion Images

Chapter 10 presents an imaging technique suitable for visualizing three-dimensional ALS point data. Images created using this technique are referred to as *occlusion images*. Occlusion images are based on the idea that each pixel value is proportional to the number of positions in the sky from which surfaces in the pixel are visible. ALS points are acquired from an aircraft and the line-of-sight constraint allows points to be treated as visibility samples. Further, each pulse is treated as one unit of visibility, possibly spread across multiple echoes. Points are mapped to pixels and a set of unique aircraft positions is maintained for each pixel. The final pixel values are then computed based on the number of unique aircraft positions and the visibility contributions of mapped echoes.

Visualization of ALS data has applications in both semi-automated (Chapter 11) and automated building extraction (Chapter 12). In semi-automated approaches, it is necessary for a user to select features of interest from a visualization of ALS data. Similarly, detailed visualization is required to identify cases where automated approaches fail and to analyze why this happens.

In Part III comparisons between occlusion images and conventional images are presented. It is shown that occlusion image are capable of clearly visualizing details that are difficult to identify in conventional images.

7.4 Semi-automated Building Extraction

Chapter 11 present a tool for semi-automatically extracting user-specified regions of a large ALS data set. Region selection is based on visualizations of the available ALS data in the form of occlusion images. Two different selection mechanisms are provided: (1) *polygon selection*; and (2) *line selection*. Polygon selection is suitable for selecting entire buildings by specifying polygons corresponding to building footprints. Line selection, on the other hand, is suitable for selecting individual building walls. In particular, points are extracted in a local coordinate system based on the specified line

Results of semi-automatic extraction are presented in Part III, where both polygon and line selection are demonstrated. Additionally, line selection is used to extract building walls for solid modeling, where the transformation of points into local coordinates is beneficial for reasons discussed further on.

7.5 Automated Building Extraction

Chapter 12 presents an automatic building extraction approach based on scan line analysis. Classification of ALS points is done in individual scan lines, where building wall segments are identified as sequential points being stacked vertically. Results of scan line classification are mapped to a digital image in order to identify building outlines. A set of morphological operations (Section 3.4) are used to analyze building outlines in order to determine which buildings were successfully detected. As a result, building footprints can be generated, making it possible to extract a separate set of points for each detected building.

Automated building extraction results for the Dublin ALS data set are presented in Part III, and are evaluated using high-resolution occlusion images (Chapter 10).

7.6 Voxelizing Laser Scan Data for Solid Modeling

Chapter 13 describes a tool for creating solid models directly from point data. Rapid conversion of a point data set into a solid model is achieved by creating a volumetric structure based on the provided points. The cells of this structure are then treated as geometric primitives of the solid model. Further, cells are considered to be active or inactive depending on the presence of certain features of the points within them. Inactive cells are subsequently discarded, providing a reasonable approximation of the three-dimensional building geometry.

In Part III, the tool for creating solid models is applied to both ALS and TLS data sets. Basic engineering simulations are run on the created solid models, demonstrating the validity of the approach. The remaining chapters in this part describe the tools outlined in the sections of this overview.

Chapter 8

Urban ALS Flight Paths

Current *Aerial Laser Scanning* (ALS) system configurations and the dense geometry of urban environments are two major constraints in the acquisition of urban ALS data. This chapter outlines the difficulties related to effective point data acquisition, with an emphasis on vertical surfaces, in an urban environment for the purpose of three-dimensional modeling. A flight planning strategy to overcome these difficulties is presented.

ALS missions for urban areas are typically planned by specifying a series of straight lines, dictating the movement of the aircraft. A single line is referred to as a *flight track*. As the scanner moves along a flight track, point data is collected in an area beneath the scanner, known as a *flight strip*. The strip width depends on aircraft altitude and scan angle (Chapter 2). Most often, the area of interest is wider than a single strip, requiring several passes to be flown over the area of interest, to ensure that the entirety of the area is covered by at least one strip.

The path taken by the aircraft during the scan is referred to as the *flight path*. The flight path also includes such parameters as aircraft altitude and speed, which in turn have a large impact on the resulting data quality. While there are potentially many different flight paths that can ensure complete coverage, focus tends to be on achieving coverage with minimal ground data redundancy between the strips. Generally, some minimal strip overlap is recommended to enable strip adjustment. Beyond that, strip overlap has been considered redundant and unnecessary. This chapter will present a rethinking of this position and show mathematically how a fundamentally improved data set can be collected, one that opens new vistas for three-dimensional computational modeling, which is highly dependent on vertical data capture.

Traditionally, ALS has provided high quality data only on horizontal surfaces. Vertical surfaces are harder to capture accurately, principally because the laser pulses strike them obliquely, but also because of shadowing effects. In ALS both obliquity and shadows generate *dead zones* - regions in which data capture is either poor or non-existent. In particular, dead zones are generated by scan obliquity, facade shadows (building self-shadows) and canyon shadows (inter-building shadows). Thus, collecting densely sampled data for vertical surfaces involves planning a flight path to minimize these effects. The main body of this chapter, therefore, presents an analysis of these geometric constraints and provides recommendations on how to plan a suitable flight path to minimize these negative effects for a particular urban environment.

Baltsavias [Bal99b] and others note that the spacing between captured point samples is not uniform. However, nowhere is sampling density mentioned in relation to flight planning, where it can be used to obtain a more uniform sampling quality and strike a balance between flight strip overlap and data quality, as well as provide previously missing building wall data. The sampling analysis presented in this chapter shows how sampling patterns behave and how this can be used to design more optimal flight plans for urban scans.

8.1 Sampling Analysis

Resolution is a measurement of sampling quality. In the spatial context, resolution can be measured as the number of acquired scan points in a given area or a given unit of length, or reciprocally as the area or unit length per scan point. In ALS resolution is not uniform, with respect to width (across the flight track) and length (along the flight track), a discussion of resolution as a linear measure with respect to these two directions is presented in this section.

Contractors carrying out ALS missions typically quote the quality of the acquired data as scan points per square meter. For purposes outlined next, this must first be converted to *linear resolution*, R_L, the resolution in the along-track direction, which is assumed to be constant and depends on the velocity of the aircraft and the scan rate, i.e. the number of scan lines acquired per second. Secondly, conversion into *horizontal resolution*, R_H, the scan point spacing at ground level in the across-track direction is necessary. Finally conversion into *vertical resolution*, the vertical point spacing on building walls must occur. Of these, the first two will be discussed next. The vertical resolution will be discussed in Section 8.2.3.

8.1.1 Linear Resolution

The figure quoted for scan quality by contractors is notionally the number of scan points per square meter at *nadir* (R_N) - the point at ground level directly beneath the scanner. For the sake of argument it is assumed that $R_L = R_N$. This assumption is based on the fact that it is generally desirable to have uniform resolution throughout the entire set of acquired scan points. As will be shown further on, this is impossible to achieve in practice because of variations in horizontal and vertical resolution in the across-track direction.

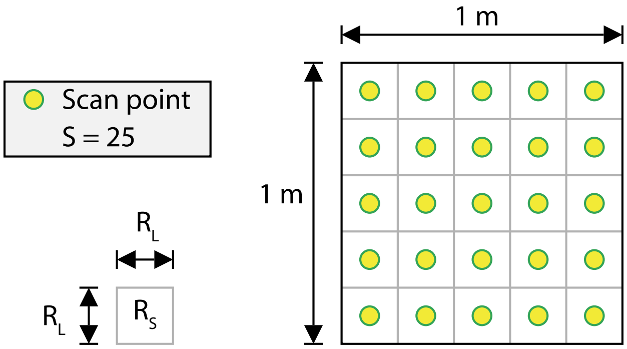


Figure 8.1: Linear resolution and scan point density.

If S is the number of scan points in $1m^2$, it follows that the relationship between S and R_L is $R_L = \sqrt{1/S}$, as shown in Figure 8.1. Consequently the *surface resolution* (R_S) as a measure of the area that each scan point represents in the sampled scene is related to R_L, through the expression $R_S = {R_L}^2$. Features smaller than R_S will not be recognizable in the acquired data. For example, if R_S is in the order of m^2 per scan point, objects such as fire hydrants and waste bins will not be recognizable.

The ability to estimate linear resolution preflight in the across-track direction is crucial to the methods presented in this chapter. Unlike the along-track resolution, which is constant and is determined by the way the aircraft moves during the scanning process, the scan points acquired across-track are at uniformly spaced instantaneous scan angles, not necessarily uniformly spaced on surfaces. The following section derives an approximation for horizontal resolution based on aircraft altitude and scan angle.

8.1.2 Notional Horizontal Resolution

In ALS, an aircraft moves above a city and laser pulses are emitted at small, regular angular intervals in the across-track direction (Chapter 2). The difference in angle to the scanner for two consecutive pulses, θ_L, is known as the angular resolution of the scanner. Together with the altitude of the aircraft, h, the angular resolution determines the horizontal spacing between two consecutive scan points at nadir (directly beneath the aircraft). A large altitude means that pulses will have traveled a long distance before striking the ground. Since pulses are emitted in different directions, this increase in traveled distance leads to scan points being spread further apart on the ground. Similarly, a large angular resolution leads to scan points being spread further apart since the differences in direction between consecutive pulses become larger. More precisely, the horizontal point spacing at nadir, R_N, can be expressed as: $R_N = h \tan \theta_L$. Thus, a typical ALS system with an angular resolution of 0.06 degrees flying at 500 meters gives: $R_N = 0.5$ meters. The relationship between h, θ_L, and R_N is illustrated on the left-hand side of Figure 8.2. Note that θ_L is exagerrated in the illustration to improve clarity.

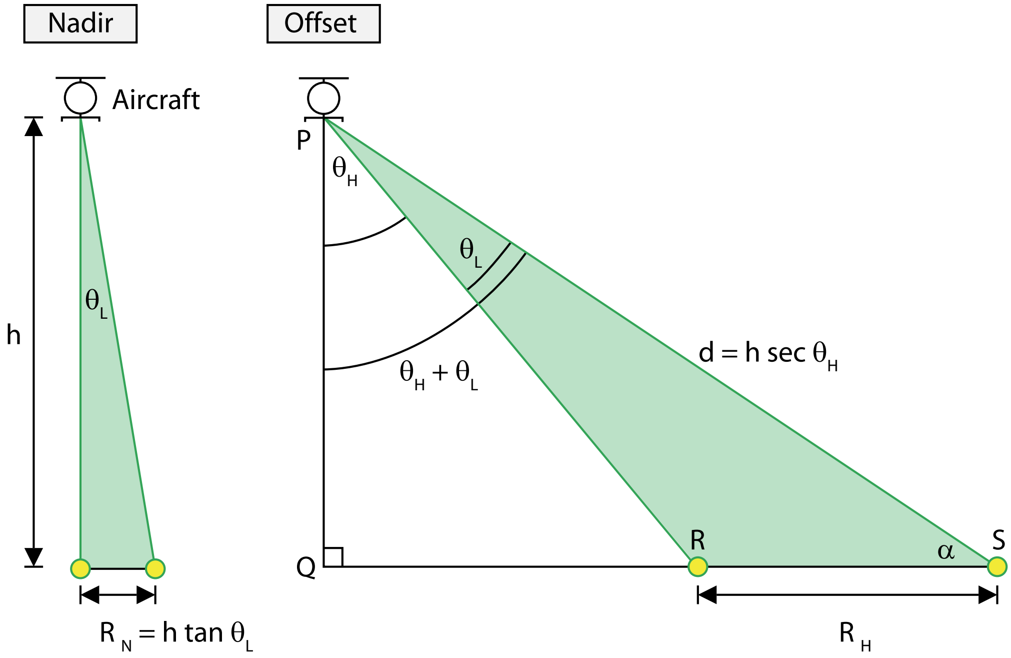


Figure 8.2: Horizontal resolution at nadir and at instantaneous scan angle θ_H, where $|\theta_H| > 0$.

The expression given previously for R_N is based on the assumption that pulses were emitted in directions very close to the vertical, striking the ground almost directly beneath the scanner. However, as pulses are emitted at larger angles away from the vertical this expression is no longer valid. The angle away from the vertical at which a pulse is emitted is referred to as the instantaneous scan angle and is denoted by θ_H (Figure 8.2). Thus, moving in the across-track direction away from nadir at ground level corresponds to increasing θ_H. Moreover, as θ_H increases the horizontal distance between consecutive scan points increases as well, as illustrated on the right-hand side of Figure 8.2. Horizontal point spacing at offset angle θ_H is denoted as R_H. By deriving an approximate expression for R_H it becomes possible to analyze how horizontal resolution varies with offset angle. Moreover, it enables the horizontal point spacing at maximum offset angles (i.e. at the edges of flight strips) to be compared with that at nadir.

In order to derive an expression for R_H the sine law is applied to triangle PRS (Figure 8.2). Note that the right triangle PQS allows the substitution: $\sin\alpha = \cos(90-\alpha) = cos(\theta_H + \theta L)$.

$$\begin{aligned}
\frac{\sin\alpha}{d} &= \frac{\sin\theta_L}{R_H} \Leftrightarrow R_H \sin\alpha = d\sin\theta_L && (8.1)\\
R_H\cos(\theta_H+\theta_L) &= h\sec\theta_H\sin_L && (8.2)\\
R_H &= \frac{h\sec\theta_H\sin\theta_L}{cos(\theta_H+\theta_L)} = \frac{h\sec\theta_H\sin\theta_L}{\cos\theta_H\cos\theta_L - \sin\theta_H\sin\theta_L} && (8.3)
\end{aligned}$$

For small θ_L the denominator of the right-hand side of Equation 8.3 is approximately equal to $\cos\theta_H$. Using this approximation Equation 8.3 can be rewritten as Equation 8.4:

$$R_H = h\sin\theta_L\sec^2\theta_H \qquad (8.4)$$

Since θ_L is small $\sin\theta_L$ is approximately equal to $\tan\theta_L$ resulting in Equation 8.5:

$$R_H = h\tan\theta_L\sec^2\theta_H = R_N\sec^2\theta_H \qquad (8.5)$$

From this, the term $\sec^2\theta_H$ can be interpreted as a scaling factor applied to the scan point spacing directly beneath the scanner, R_N. Let θ_W be the scan angle for a given scanner; hence the largest occurring offset angle is $\theta_H = \theta_W/2$. For a given height, angular resolution, and scan angle, the *effective* horizontal resolution (i.e. the worst case resolution, which occurs when $\theta_H = \theta_W/2$) can be computed. The expression for effective horizontal linear resolution, R_W, is obtained by inserting the height, angular resolution, and $\theta_W/2$ into Equation 8.5:

$$R_W = R_N\sec^2\left(\frac{\theta_W}{2}\right) \qquad (8.6)$$

The ALS system used in the case study presented in Part III is capable of angular resolutions in the order of 0.06 degrees and has a scan angle of 60 degrees. Inserting $\theta_W = 60$ degrees into Equation 8.6 gives that the effective horizontal resolution is approximately 33% larger than horizontal resolution at nadir, R_N. In effect, this means that if $R_N = 0.5$ meters, then the effective horizontal resolution is approximately 0.67 meters. Thus, the effective horizontal resolution is fairly uniform in the across-track direction. This means it does not matter hugely where in a flight strip a horizontal surface is scanned. However, as will be shown further on, the same is not true for vertical surfaces. The following section discusses further geometric issues related to acquisition of three-dimensional point data for urban areas.

8.2 Geometric Constraints on Urban Flight Paths

There are five principal constraints on designing a suitable flight path for an urban region: (1) *urban geometry*; (2) *flight geometry*; (3) *vertical scan obliquity*; (4) *self shadows*; and (5) *street shadows.* Each of these topics is treated in a separate subsection below, followed by a description of an ideal flight path resulting from these constraints.

8.2.1 Urban Geometry

Since the constraints on flight planning are principally geometric, characteristic geometry of urban environments must be acknowledged. There are three major factors that need to be considered: (1) *building geometry*; (2) *street geometry*; and (3) *street layout.*

Building geometry describes the shape of individual buildings. For structural and economical reasons, most buildings have vertical walls arranged in rectangular or near-rectangular shapes. While this is not true for buildings such as cathedrals or the Guggenheim Museum, a rectilinear pattern is broadly representative of most large urban aggregations and, as such, can be exploited when scanning. However, urban buildings are often closely spaced, abutting, or employing party walls, making it difficult to distinguish individual buildings - a key feature to individual building extraction.

Street geometry describes the shape of small groups of buildings aligned along a common communication and transportation area. Typically, a street consists of two rows of parallel buildings on opposite sides of an open space. Moreover, building plots along a given street are

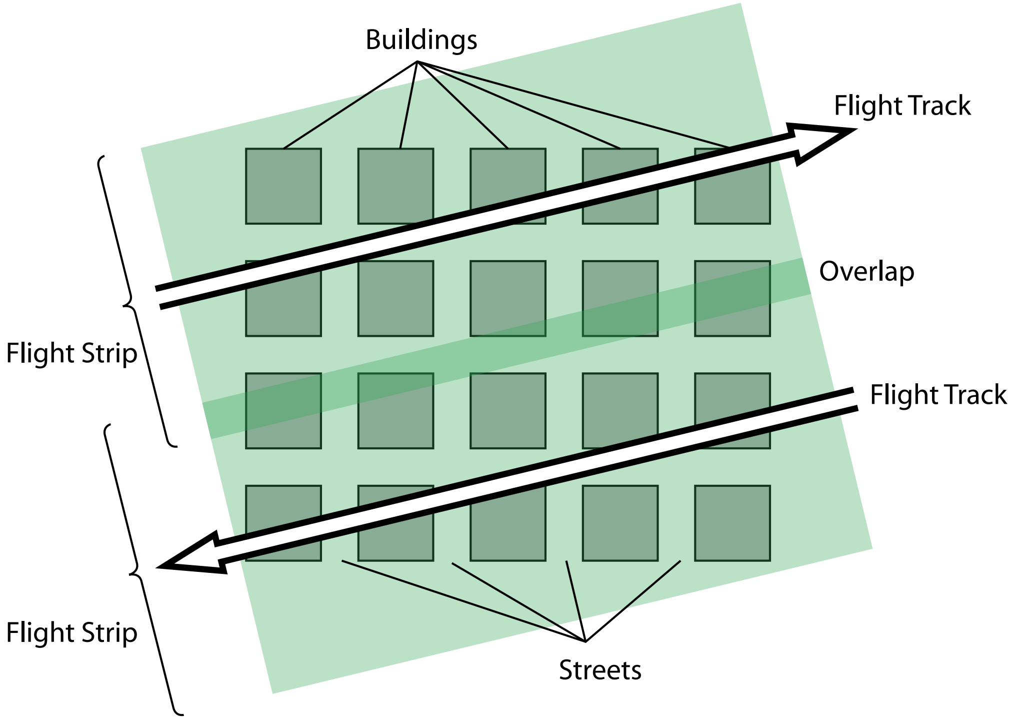


Figure 8.3: Ideal urban grid pattern with standard flight pattern superimposed.

fairly uniform in size and shape, with the result that barring topographic constraints, large parts of a city tend to have multiple streets parallel to each other. This, combined with the preferred rectilinear shape of buildings, tends to impose a strong geometric structure on the city as a whole, which can also be exploited when scanning (Figure 8.3).

Street layout describes the overall geometric structure of the city. While older portions of cities can be very complex, from casual observations one can observe that most cities fall into two basic patterns: (1) *regular rectangular grids*; or (2) *radial layouts*. However, within radial cities, the infill between the major radial streets tends to be rectangular in nature. As such, a reasonable approximation of street layout is that it tends to be locally regular but may be irregular at a larger scale. This localized structure can also be exploited when scanning. For simplicity, the balance of this chapter will assume that the city to be scanned can be represented locally as a rectangular grid (Figure 8.3).

8.2.2 Flight Geometry

While the flight path is notionally controllable, in practice it is easiest to fly in a straight line. Thus, most survey flight paths tend to be a set of straight lines (as opposed to a zigzag or radial pattern). Moreover, for a given altitude of flight, the LIDAR unit scans points on the ground, within a relatively fixed lateral offset (although this is dependent on the altitude of the ground). Since each straight line flown (i.e. flight track) corresponds to a rectangular strip of ground scan, flight planning is principally a question of choosing the rectangular strips to be scanned in such a way that the overlap covers the desired area. Given straight-line flight paths, this means that the most straight-forward method is to fly a series of parallel flight tracks, whose strips between them cover the entire area with only minimal overlap, as shown in Figure 8.3. For vertical surfaces, however, further adjustments are necessary to surmount geometric impediments.

8.2.3 Vertical Scan Obliquity

The principal geometric difficulty with ALS acquisition of vertical surfaces is that as the aircraft flies above a city, the laser beam scans a fairly narrow range of angles beneath the scanner - typically not more than 30 degrees away from the vertical on each side. Horizontal linear resolution has previously been shown to be rather uniform in the across-track direction. This is illustrated in the left-hand side of Figure 8.4, where horizontal resolution at maximum across-track offset differs from nadir, R_N, only by a factor of $\sec^2\theta$.

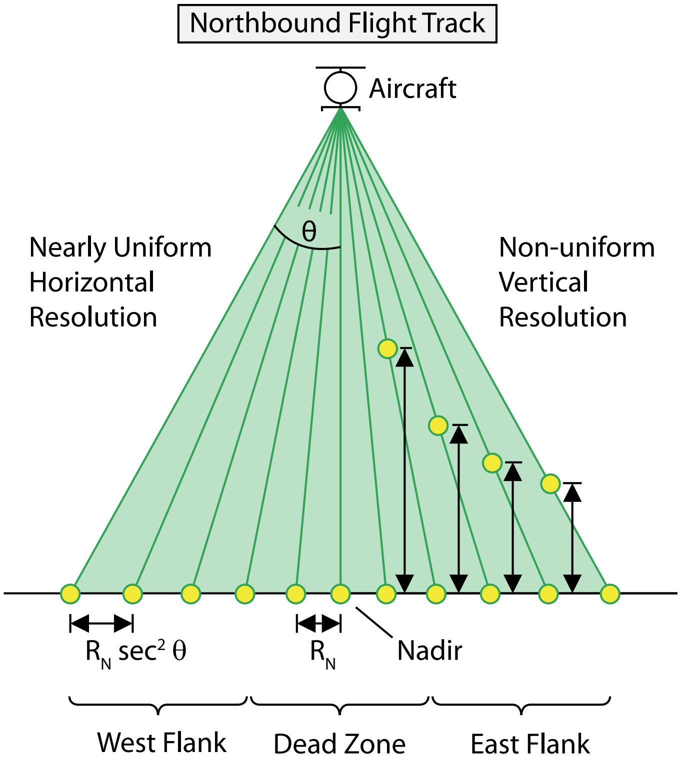


Figure 8.4: Horizontal and vertical scan resolutions.

For vertical surfaces, however, as visible by the arrows on the right side of Figure 8.4, the spacing of scan points on vertical surfaces increases as the vertical surfaces approach nadir. Intuitively, this is reasonable, since a vertical surface directly underneath the scanner will be parallel with the laser pulse, which will, therefore, strike the entire surface. In other words, the closer a vertical surface is to nadir, the worse the vertical scan resolution on that surface will be.

Since vertical resolution is worst directly beneath the scanner, guaranteeing scan quality of a particular level requires disregarding the data acquired directly beneath the aircraft, and for some distance off to the side: paradoxically, this *dead zone* of low quality vertical data is the zone of the *highest* quality horizontal scanning. The *flanks* of the scan, in contrast, generate the highest quality vertical scan and lowest quality horizontal data, as shown in Figure 8.4.

From this observation, it follows that the dead zone from one flight track will have to be scanned in the flank of another flight track. Obtaining this flanking data is most simply achieved if the flank is the same width as the dead zone (or possibly a rational fraction of it). The impact on scan quality of the width of the dead zone is addressed later in this chapter, but for now, consider the dead zone and flanks to be of equal width, and that each is 1/3 of the total swath width. Note that the width of the dead zone is not necessarily directly related to horizontal resolution at nadir, but rather to what is deemed to be acceptable vertical scan quality. As such, the rationale for defining the width of the dead zone varies with different types of applications and data requirements.

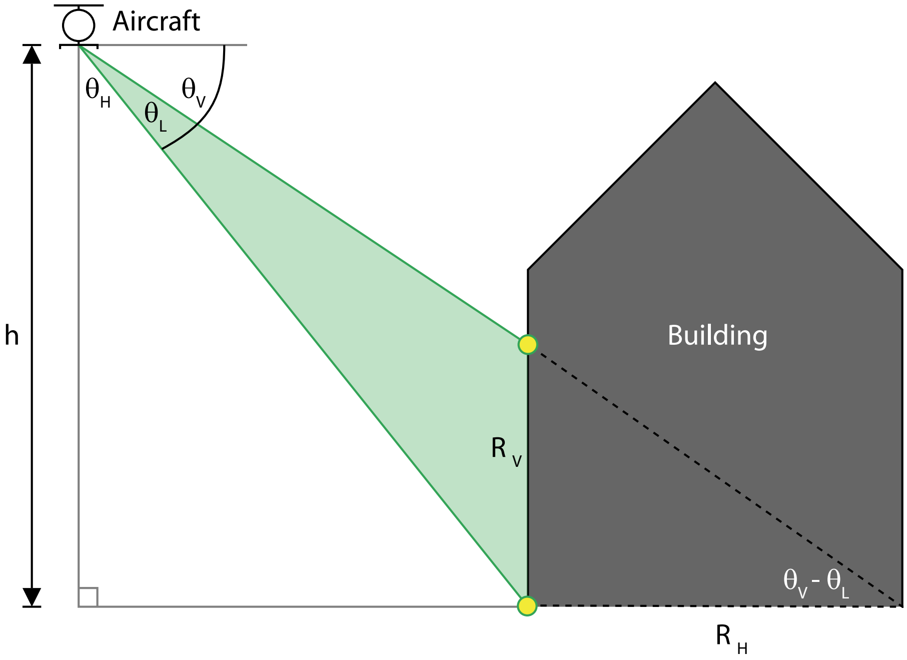


Figure 8.5: Vertical linear resolution.

As in the case of linear *horizontal* resolution, R_H, approximations for linear *vertical* resolution, R_V, can be derived. Defining an instantaneous vertical scan angle, θ_V, related to the horizontal offset angle by: $\theta_V = 90° - \theta_H$, as shown in Figure 8.5, is convenient for notation. The horizontal resolution at an instantaneous scan angle θ_H (Equation 8.4) can then be related to the effective vertical spacing through the expression:

$$R_V = R_H \tan(\theta_V - \theta_L) = R_H \tan(90^\circ - \theta_H - \theta_L) \tag{8.7}$$

Just as for effective horizontal linear resolution, it would be useful to define effective vertical linear resolution for a worst case scenario. However, R_V diverges to infinity at nadir, and, therefore, it is not possible to give a quantitative result for the worst case vertical linear resolution. However, the more interesting case is when a vertical surface laterally offset from nadir is being scanned. Since the instantaneous vertical scan angle θ_V will be largest at ground level of the vertical surface (where θ_H is smallest), the first interval on the vertical surface is used as the worst case, with the assumption that the laser samples the vertical surface exactly at the intersection between the surface and the horizontal plane (Figure 8.5).

Note that Equation 8.7 is a valid approximation up to $\theta_V = \theta_H = 45^\circ$. At this point horizontal resolution (R_H , Equation 8.4) starts diverging to infinity faster than $\tan(90^\circ - \theta_H - \theta_L)$ converges to zero. However, most ALS systems are only capable of θ_H up to about 30 degrees, and the proposed model is valid within this range.

Figure 8.6 shows how R_H (Equation 8.6) and R_V (Equation 8.7) vary with horizontal offset angle (θ_H). The angular resolution θ_L is assumed to be 0.01 degrees and the height used is 300 m, based on minimum allowed flyover height in urban regions. Note that an instantaneous scan angle $\theta_H = 45^\circ$ gives $R_H = R_V$.

8.2.4 Self Shadows

In addition to dead zones caused by scan line sampling patterns, not all vertical surfaces in the scan area will appear on the scan, as a result of *shadows* - since the laser is a form of light, it is blocked by solid objects. Furthermore, as any solid object has one side facing the scanner and the other side facing away, the side facing away will be *self shadowed* by the side facing toward the scanner, as shown in Figure 8.7. The consequence of this, shown in Figure 8.7, is that to acquire *both* sides of a given object, there must be (as a minimum criterion) two scans: one from the left and one from the right.

Even though each flank provides good quality vertical data, it only does so for half of the vertical surfaces. Specifically, the flank to the east of the flight path will only provide data on west-facing surfaces, and the flank to the west of the flight path will only provide data on

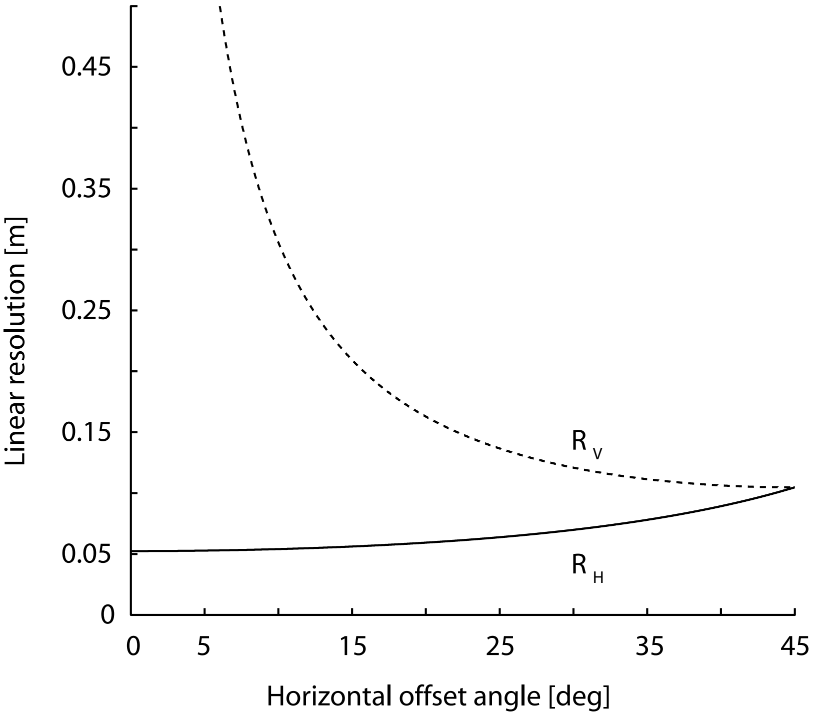


Figure 8.6: Horizontal and vertical resolutions as functions of instantaneous scan angle θ_H.

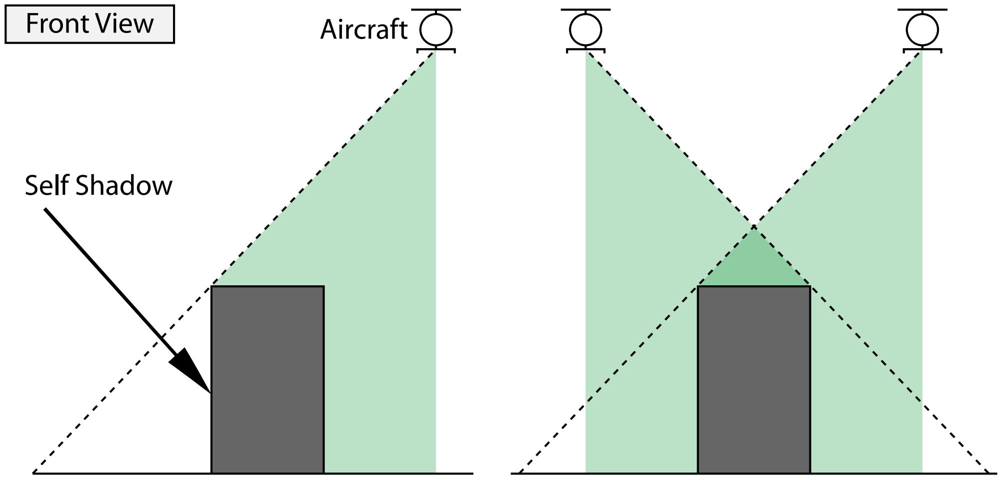


Figure 8.7: *Left*: Self shadows occur on building walls facing away from the aircraft. *Right*: At least two aircraft positions are required to avoid self shadowing on a simple building.

east-facing surfaces. Complete scan coverage thus requires that every building be covered from both flanks - one from each of two different flight segments (Figure 8.8).

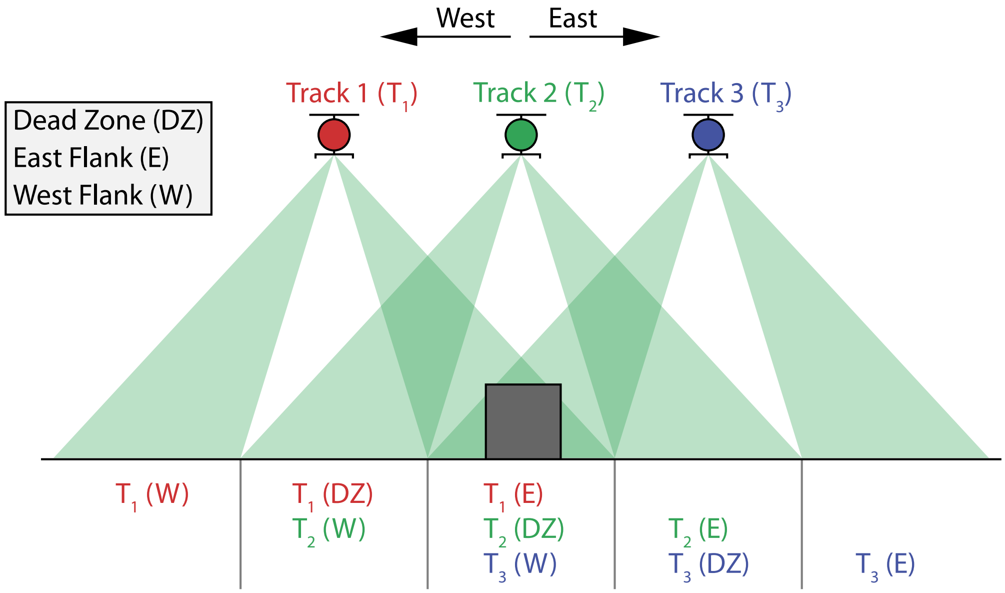


Figure 8.8: Overlapping flanks for full coverage.

It has been established that a uniform data quality is desirable and that the highest quality vertical data is obtained in the flanks of flight strips. Along-track resolution, R_L, is not subject to geometric constraints and is under the control of contractors. In Section 8.1 it was assumed that R_L is chosen to be the same as the horizontal resolution at nadir, R_N, in an attempt at achieving uniform resolution. However, it would be more efficient to chose R_L to match the vertical resolution, R_V, at the edges of the dead zone (i.e. the worst-case). The rationale behind this is that overall resolution is never better than the worst-case scenario, and achieving better resolution in the along-track direction will not improve overall resolution; presumably, lowering the along-track resolution would lead to faster and cheaper aerial scans.

8.2.5 Street Shadows

While self shadows are the result of a building shadowing itself, scans may also be blocked by the effect of street shadows, in which a building is shadowed by another building on the other side of a street. Geometrically, the effect of these shadows depends on the height of the shadowing building and the distance between the two buildings, as can be seen to the left in Figure 8.9. If the distance, w, between the buildings is less than $h \tan\theta$, then the bottom of one building will be shadowed by the adjacent building. Although the building height, h, is immutable, the distance between buildings is measured *perpendicular* to the flight path. If the flight path is parallel to the street, these distances are minimized, as shown by width, w_1, to the right in Figure 8.9. However, flight tracks at an angle φ to the direction of the street will effectively increase the distance between buildings, as shown by diagonal, w_2, in Figure 8.9.

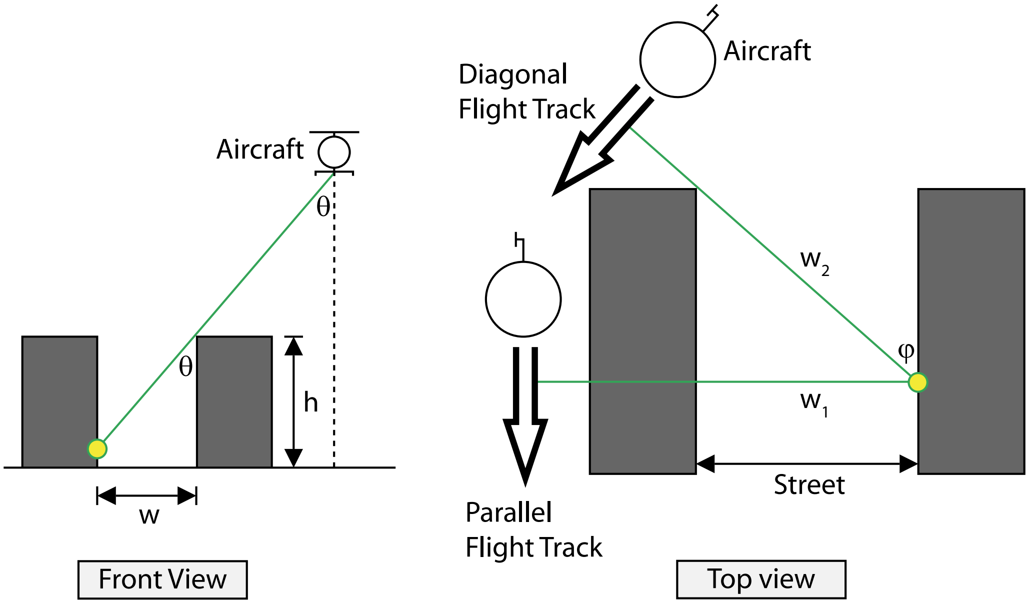


Figure 8.9: *Left*: Street shadows depend on the height of the shadowing building, h, and the distance between the two buildings, w. *Right*: Flight tracks at an angle φ to the direction of the street will effectively increase the distance between buildings, as shown by effective street widths $w_2 > w_1$.

Using the assumed perpendicular street grid, the optimal flight angle for avoiding street shadows is 45 degrees to the street grid. This will maximize w_2 for streets in both directions on a street grid where streets run in two principal, perpendicular directions. Unfortunately, flying diagonally to the street pattern compromises the resolution on buildings in the sense that the distance between samples in the street direction, (i.e. the *lateral resolution*, R_{LAT}, increases). Figure 8.10 illustrates the relationship between linear (along-track) resolution, R_L, and lateral resolution: $R_{LAT} = R_L \sec\varphi$. This dictates that the lateral resolution suffers as the angle φ approaches 90 degrees. However, as long as the lateral resolution does not exceed the vertical resolution this does not matter, since the area sampling is dependent on both these distances, and will always be limited by the one that is greater. Hence, it is reasonable to sacrifice lateral resolution in exchange for minimizing street shadows.

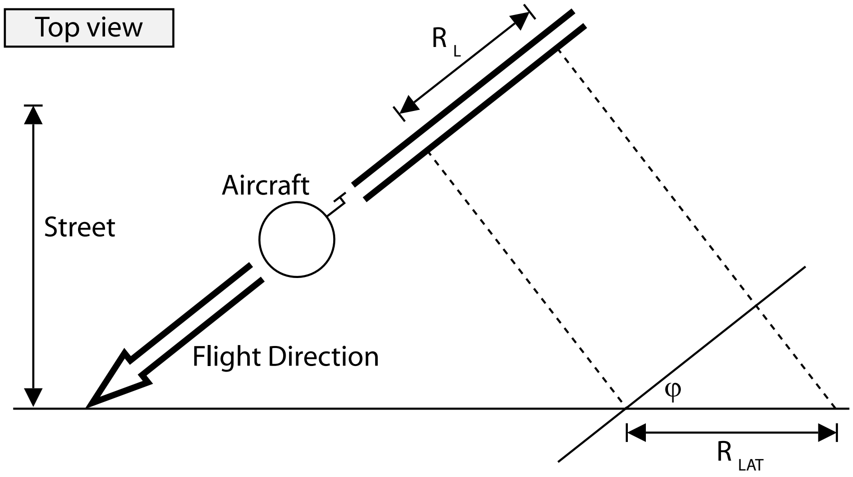


Figure 8.10: Lateral resolution.

8.3 Flight Planning

As noted above, the simplest approach is to set the dead zone equal in width to the flanks; for example, for a total scan width of 300 m, the dead zone and flanks should each be 100 m wide. From Figure 8.8, setting the distance between the flight paths equal to this width is shown, although a slight reduction may be desirable to achieve some additional overlap and avoid lacunae in the data. As such, the ideal flight path will consist of a series of parallel lines diagonal to the local street grid, spaced at a distance of 1/3 the total scan width (or possibly a rational fraction of it) from each other, in order to guarantee that each building is correctly scanned on all principal faces with a maximum of scan resolution and a minimum of shadowing. The specific amount of overlap chosen should reflect the along-track resolution and the importance of high data quality on vertical surfaces, and as such, is somewhat project specific.

Street patterns are never perfect grids in practice, and, therefore, trade-offs have to be made. One strategy would be to fly diagonally to as many streets as possible, regardless of individual street widths. This makes sense because the effective gain in street width by flying diagonally is the same, regardless of street width. A slightly more sophisticated approach would be to assign importance weights to the streets, and use these weights as decision factors for optimizing the flight path. One could also imagine using a numerical optimization method that takes a street pattern as input and computes a flight plan that flies at as close to 45 degrees as possible to as many streets as possible. Finally, if a three-dimensional model of the study area exists already, this may be used as input and would give an even better flight path since building geometries and shadowing effects could be taken into account. However, no algorithms for this type of flight planning exist at present.

Using the realistic assumption that buildings are bound by vertical discontinuities (e.g. vertical walls), reliable three-dimensional point data acquisition can be achieved for urban scenes by following three general guidelines: (1) large scan angle together with low aircraft altitude; (2) multiple flight strip overlap; and (3) flight tracks oriented as closely as possible at 45 degrees to the underlying street grid. Although multiple flight strip overlap is required to provide points on building walls, it also generates a fair amount of redundancy on horizontal surfaces. As a consequence, ALS data sets acquired using the guidelines presented above contain huge amounts of points. The following chapter provides a framework for working with large ALS data sets.

Chapter 9

ALS Scan Line Analysis

This chapter presents techniques for processing *Aerial Laser Scanning* (ALS) data. The relevant background for these discussions was presented in Chapter 2. The techniques described in this chapter aim to exploit angular sampling regularity within scan lines found in ALS data. Such patterns are exposed by associating information about the aircraft with acquired point data. Analysis of scan line points together with aircraft information allows reconstruction of individual pulses. Moreover, such analysis allows the identification of *empty pulses*, pulses for which no echoes were detected. Additionally, *missing echoes* can be approximated for empty pulses, enhancing the sampling regularity within scan lines. These techniques extend to arbitrarily sized ALS data sets, a necessary feature since ALS data sets are constantly growing in size. Coming chapters on visualization (Chapter 10) and building extraction (Chapter 12) make extensive use of the techniques presented in this chapter.

The goals of this chapter can be summarized as follows: (1) associating aircraft information with scan points; (2) robust reconstruction of individual pulses; (3) inserting approximated missing echoes in empty pulses; and (4) processing arbitrarily large ALS data sets in such a way that the first three operations are possible. Each of these items is treated below, starting with the description of a hierarchical structure describing the motion of an aircraft over time. This structure allows efficient querying of aircraft position and orientation using scan point time-stamps.

9.1 Flight Path Model

A crucial component of the scan line analysis presented in this chapter is the association of aircraft position and orientation with scan points. Aircraft position and orientation are recorded over time in the form of flight path points (Chapter 2). Flight path points provide instantaneous aircraft position and orientation at discrete, regularly spaced times. In the following discussions a flight path point is denoted as $\vec{p} = (t, x, y, z, \psi, \theta, \phi)$, where t is a time-stamp, (x, y, z) are positional coordinates and (ψ, θ, ϕ) are Tait-Bryan rotation angles describing aircraft orientation (Chapter 2).

Both flight path points and scan points have time-stamps. Thus, the time-domain can be used to relate aircraft motion to scanned points. In particular, points acquired in the same scan line are assumed to have identical time-stamps. Furthermore, the aircraft is notionally stationary during the acquisition of a single scan line. As will be shown further on in this chapter, associating aircraft information with the points of a scan line has several useful applications, as will be discussed in Section 9.2.

The time spacing between flight path points is ~ 1 second. However, modern ALS systems are capable of acquiring hundreds of scan lines per second, which means that scan point time-stamps have much finer resolution than flight path point time-stamps. Thus, interpolation is required in order to approximate the aircraft position and orientation from which a scan line was acquired. An approach for linearly interpolating aircraft position and orientation between two flight path points is provided in Section 9.1.2.

Linear interpolation of aircraft position and orientation for any given time requires identification of the two closest flight path points in the time-domain. In order to efficiently identify the two closest flight path points for any time, a hierarchical structure describing aircraft motion is presented in the following subsection. Thereafter, linear interpolation of flight path points is discussed.

9.1.1 Flight Path Hierarchy

A single flight path point describes instantaneous information about the aircraft. In order to describe aircraft motion over time it is necessary to explicitly derive some form of connectivity between flight path points. A hierarchical structure is used to describe connectivity for varying lengths of time. Describing aircraft motion over time in a hierarchical structure has two main purposes: (1) to enable fast querying of the two closest (in time) flight path points for any given scan point time-stamp; and (2) to enable aircraft motion to be decomposed into periods of continuous scanning. The hierarchical nature of the proposed structure is shown in Figure 9.1. The different levels of the structure shown have slightly different properties, as will be discussed below.

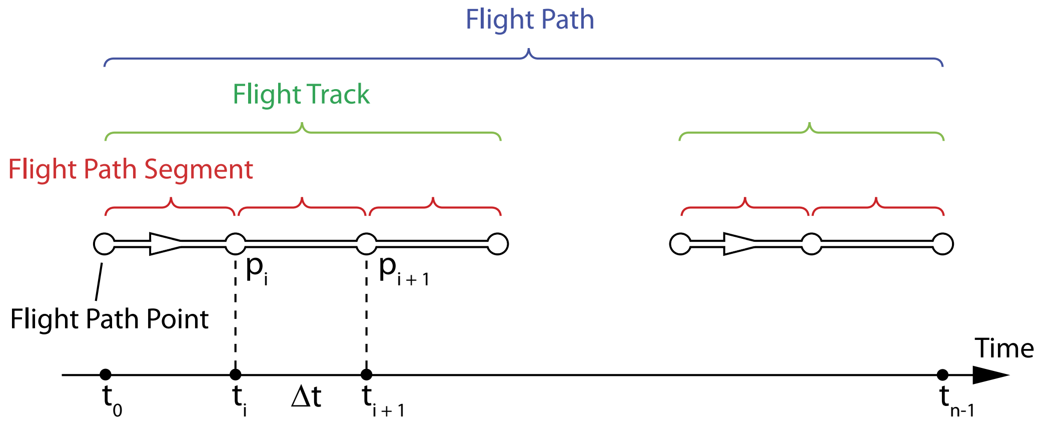


Figure 9.1: A hierarchical model describing the motion of an aircraft over time. Flight path segments (red) consist of two connected flight path points. Flight tracks (green) consist of one or more connected segments and the set of all tracks defines the flight path (blue).

Flight path points are at the lowest level of the hierarchy shown in Figure 9.1. As mentioned, these points provide instantaneous information about the position and orientation of the aircraft. Higher levels of the hierarchy are constructed based on connectivity between flight path points in the time-domain.

Flight path segments are defined as ordered pairs of connected flight path points, $(\vec{p}_i, \vec{p}_j)$, where $t_i < t_j$. As such, each segment has a start and an end point. Two distinct flight path points, $\vec{p}_i$ and $\vec{p}_j$, are said to be connected if $t_j - t_i = \Delta t$, where Δt is the time between flight path point recordings (typically ~ 1 second). Thus, each segment spans a time period of Δt.

Flight tracks are defined as a set of connected segments and must contain at least one segment. Two segments, s_i and s_j, are said to be connected if the end point of the first is equal to the start point of the second, i.e. if s_i consists of points $(\vec{p}_i, \vec{p}_j)$ and s_j consists of points $(\vec{p}_j, \vec{p}_k)$. Similar to a segment, a track has a start and end point, where the former is the start point of the first segment and the latter is the end point of the last segment. A track spans a time period of $M\Delta t$, where M is the number of segments in the track.

The *flight path* is defined as the set of all flight tracks. Thus, the flight path level serves as the entry point to lower levels of the hierarchy. A flight path spans the entire time period of the contained tracks. Having described different levels of the hierarchical flight path model, construction of the flight path model is discussed next.

Flight path hierarchy construction starts by connecting a provided set of distinct flight path points into segments. Flight path points are first sorted in ascending order with respect to time-stamps. Thereafter, pair-wise connectivity testing between consecutive points is used to create a set of flight path segments. Points that cannot be used to form a segment with another point are discarded. A similar approach is used to connect the created set of segments into flight tracks. By definition, the flight path level is then the set of all constructed flight tracks.

Having defined the a hierarchical structure describing the motion of an aircraft over time, the following subsection describes how such a structure can be used to efficiently interpolate aircraft position and orientation.

9.1.2 Flight Path Point Interpolation

Interpolation of flight path points is a crucial part of the techniques that will be presented in Section 9.2, where the association of aircraft position and orientation with the points of a single scan line is used extensively. Because scan point time-stamps have finer resolution than flight path point time-stamps, interpolation is required to approximate aircraft position and rotation for times in-between provided flight path points.

Linear interpolation, which has been found to be sufficient for the purposes in this thesis, of aircraft position and orientation requires identification of the flight path segment spanning the queried time. The start and end points of that segment will be used to provide interpolated information about the aircraft. For the purpose of efficiently finding the relevant segment, the hierarchical structure described in the previous subsection is used. In order to provide fast queries, the hierarchical flight path model is considered to be a *segment tree*, as described in [dBvKOS00]. In this case, flight tracks are considered to be segments. The segment tree makes it possible to identify which flight track, if any, a given time belongs to in $O(\log N)$, where N is the number of tracks in the flight path. Furthermore, finding a segment within a flight track can be done in constant time, assuming that segments have equal length. This is possible because segments within the same flight track must, by definition, be connected, i.e. there are no gaps in time within a flight track.

Having identified a suitable flight path segment, linear interpolation proceeds as follows. Consider the start and end points of a flight path segment, $\vec{p}_i$ and $\vec{p}_j$. Given a time $t_n \in (t_i, t_j)$ the goal is to create a new flight path point $\vec{p}_n$ that approximates the position and orientation of the aircraft at time t_n. The time-stamp of the new flight path point is simply t_n and the new position (x_n, y_n, z_n) is computed by linearly interpolating the positional coordinates of $\vec{p}_i$ and $\vec{p}_j$. Spherical linear interpolation, as described in [Sho85], is used to ensure constant angular velocity for the interpolated angles $(\psi_n, \theta_n, \phi_n)$.

This section has shown how flight path information can be efficiently associated with scan points. A hierarchical structure describing aircraft motion over time is used to enable efficient querying of the flight path information for scan points. Such flight path information needs to be interpolated due to the fact that scan point time-stamps have finer resolution than flight path point time-stamps. Linear interpolation methods were presented for this purpose. The following section presents applications of associating aircraft information with points acquired in a single scan line.

9.2 Pulse Reconstruction

As discussed in Chapter 2, an ALS point has two pieces of information about the pulse in which it was acquired. The first is the total number of echoes for the pulse and the second is the echo index, describing the which of the echoes the point represents. If ALS points are provided in an order where echoes of the same pulse are sequential, echoes can easily be grouped into pulses using the information provided about the total number of echoes for each pulse. However, as discussed in more detail in the following section, large ALS data sets are often split into smaller parts. Such splitting may change the order in which ALS points are provided during processing. Thus, ALS points are not guaranteed to be provided in an order corresponding to being echoes of the same pulse.

This section presents a technique for reconstructing pulses that is not based on the order in which ALS points are provided. Using the assumptions that ALS points are acquired in scan lines and that points acquired in the same scan line have identical time-stamps, points are first grouped by scan line. Thus, given a set of points known to have been acquired in the same scan line, the goal is to group these points further as into individual pulses. The purpose of this is twofold: (1) it allows the echoes of a pulse to be considered as a unit; and (2) it allows the insertion of approximated *missing echoes* for *empty pulses*, i.e. pulses for which no echoes were detected during scanning. The following subsections discuss these two topics in more detail.

9.2.1 Pulse Echoes

In order to reconstruct individual pulses, ALS points are first grouped by scan line. An interpolated flight path point is associated with the scan line points, using the methods described in Section 9.1. Pulses in a scan line are emitted at regular angular intervals. Moreover, the echoes of a pulse have approximately the same instantaneous scan angle. Thus, if instantaneous scan angles for ALS points can be computed, it is possible to associate points with pulses and to treat the points as echoes.

In order to compute instantaneous scan angles it is necessary to take into account the orientation of the aircraft. This is done by transforming points into a local *Aircraft Body Coordinate* (ABC) coordinate system. In this coordinate system the aircraft is at the origin and axes are defined such that $\hat{x}_{ABC}$ points in the forward direction of the aircraft, $\hat{y}_{ABC}$ points to the right, and $\hat{z}_{ABC}$ points downward [SL03]. The axes of this coordinate system are computed from interpolated aircraft rotation angles and the origin is defined using the interpolated aircraft position. A standard rigid body transformation [Str88] can be used to transform ALS points from global coordinates, (x_w, y_w, z_w), into aircraft body coordinates, $(x_{ABC}, y_{ABC}, z_{ABC})$. The instantaneous scan angle, θ_i, is then computed as:

$$\theta_i = \tan^{-1}\left(\frac{y_{ABC}}{z_{ABC}}\right)$$

for each ALS point. The instantaneous scan angle is used to assign each point to the pulse corresponding closest to that angle. The internal order of points within each pulse is given by the echo index. Interestingly, by reconstructing of a scan line, it becomes possible to identify pulses which have no echoes. The reason why some pulses do not have any echoes is that reflected pulse energy was not detected. In the following discussions, methods for approximating missing echoes are given.

9.2.2 Missing Echoes

As discussed previously, the pulses of a scan line are emitted at regular angular intervals. Also, when all pulses of a scan line are simultaneously reconstructed, it becomes possible to identify empty pulses, i.e. pulses that have no echoes. Moreover, it is possible to approximate missing echoes for empty pulses, thus increasing sampling uniformity within scan lines.

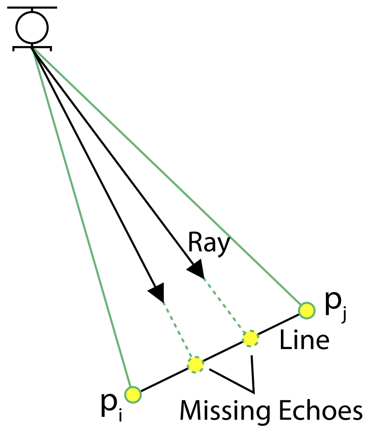


Figure 9.2: Missing echoes are approximated by computing the intersections between rays and a line segment defined between echoes of non-empty pulses.

The procedure for approximating missing echoes starts by identifying sequences of empty pulses. First, pulses are sorted by instantaneous scan angle. Thereafter, sequences of empty pulses delimited by non-empty pulses are identified. The missing echoes of a sequence of empty pulses lie on the same line (Figure 9.2). This line passes through the last echoes of the two delimiting non-empty pulses (p_i and p_j). Note that because two non-empty pulses are required to define the line, sequences starting or ending with the first or last pulse of the scan line are excluded. Since scan line pulse are emitted at regular angular intervals, the instantaneous scan angles for empty pulses are known. Using these angles it is possible to define rays that intersect the line. Linear component intersection is then used to compute the intersection points between empty pulse rays and the line [SE03]. These intersection points become the positions of the corresponding missing echoes.

A similar technique for inserting missing echoes was proposed in Höfle's thesis [HP07]. In that case missing echoes were approximated by linear interpolation between existing echoes.

However, it was shown in Chapter 8 that the neither horizontal or vertical spacing between consecutive echoes varies linearly with the instantaneous scan angle. As such, the technique presented above more closely mimics the way in which echoes are acquired. Although it assumes that the geometry between non-empty pulses is a straight line, which is of course not always the case.

9.3 Streaming

At present, large ALS data sets are of often split into several non-overlapping, horizontal (often rectangular) regions known as *tiles*. The reason for this is that entire ALS data sets are, in general, too large to fit in the main memory of a workstation. Each tile contains the points whose horizontal coordinates are within the tile boundaries. Tiles can be made small enough to reside in the main memory of a workstation and are processed separately, thereby avoiding memory limitation issues. However, tile splitting does not take into account the scan lines in which points were acquired. Thus, tile splitting may cause points acquired in a scan line to be divided across multiple tiles, making it difficult to process scan line points in unison. As such, dividing ALS data sets into tiles based on horizontal coordinates overcomes issues with memory limitations, but breaks up patterns inherent to the acquisition process.

As was shown in the previous section, when all points of a scan line are available simultaneously it becomes possible to reconstruct the echoes of pulses and to insert missing echoes in empty pulses. This section describes an approach for processing large ALS data sets while guaranteeing that points acquired in the same scan line are available simultaneously. The main idea of this approach is to create files containing points sorted with respect to time-stamps, as opposed to horizontal coordinates. Contents of such files are loaded into memory in small parts, a process often referred to as *streaming* in computer science literature. As with tiles, parts can be made small enough to fit in memory, which allows processing of arbitrarily large data sets. The approach described in this section is based on two important steps: (1) creating files on disk where scan line points are stored in sequence; and (2) reading points from these files such that scan line points are always available simultaneously. These steps are described separately in the following subsections, starting with a description of how points are stored on disk.

9.3.1 Data Storage

Large ALS data sets are stored in one or more files on a disk. Points within each file are sorted with respect to time-stamp. Recall that points acquired in the same scan line have identical time-stamps. Thus, points acquired in the same scan line are stored in sequential order in files. It is possible to store points in multiple files, but care must be taken that points with the same time-stamp are not split across multiple files. Organizing files so that scan line points appear in sequential order is beneficial when loading parts of such files into memory for processing. The following section describes how this is done while guaranteeing that at least one entire scan line is loaded into memory at any given time.

9.3.2 Read Operations

The goal of the streaming approach presented in this section is to process arbitrarily large ALS files, while at the same time guaranteeing that the points of any given scan line are available simultaneously. This could be easily achieved by loading the entire contents of a file into main memory. However, ALS point files may be too large to fit in main memory. Therefore, points are read into main memory in small *buffers*. The number of points in each buffer is user-specified and should be small enough to fit in main memory. Once the points in the current buffer have been processed, that buffer is discarded, freeing up memory for the next buffer to be read. This procedure is repeated until all the points in the file have been processed.

Recall from Chapter 2 that a fixed number of pulses is emitted for each scan line. The *scan rate* of an ALS system is the number of scan lines acquired per second. Similarly, the *pulse rate* is the number of pulses emitted per second. Thus, the number of pulses emitted in each scan line, M, is equal to the pulse rate divided by the scan rate. Multiple echoes may be detected for some pulses and each echo correspond to a point in the file. The maximum number of echoes that can be detected for a pulse, N, depends on the ALS system and is constant. Thus, a scan line is a stored as a sequence of no more than $K = MN$ points. As a consequence, the minimum allowed buffer size is also K, since buffers must be able to accommodate the largest possible scan line.

Pulses are not guaranteed to have the same number of echoes, which means that the number of points in scan lines varies. However, the buffer size is constant. Thus, points stored in the buffer do not necessarily belong to the same scan line. It is, therefore, necessary to examine the time-stamps of buffer points in order to establish sequences of points with identical time-stamps. In general, a sequence is a series of consecutive points with identical time-stamps, delimited by points with different time-stamps. This is not true for the first and last scan lines in a file, but in these cases the start and end markers of the file serve as delimiters.

Moreover, a potentially incomplete sequence may occur at the end of a buffer, where it is not possible to verify that a sequence is terminated properly. A *dual-buffer* approach is used to ensure that incomplete sequences are correctly processed. The two buffers are referred to as the *auxiliary buffer* and the *read buffer*. In a read operation, points are loaded from file into the read buffer. The auxiliary buffer is used to store incomplete sequences in-between read operations. The time-stamps of points in the auxiliary buffer are then tested against the first sequence in the read buffer. If there is a match, these two sequences are merged, otherwise the potentially incomplete sequence has been verified to be complete and is processed and subsequently discarded from the auxiliary buffer.

The number of read operations depends on the size of the file and the size of the read buffer. In practice, it is more efficient to use large buffers since there is a certain overhead involved in each read operation. Thus, the total overhead is shortened by minimizing the number of read operations. The buffer size should be as large as possible based on the memory constraints of the machine used. The only constraint on buffer size is that it should be at least K points. However, given that current ALS systems emit on the order of $1,000$ pulses per scan line and that up to 5 echoes are detectable for each pulse, most current machine are well within this limit.

In summary, this chapter has presented a framework for performing analysis on ALS points based on the scan lines in which they were acquired. An important aspect of this analysis is the association of aircraft position and orientation with scanned points. Such information enables the echoes of pulses to be fully reconstructed regardless of the order in which these echoes are stored on disk. Moreover, pulse reconstruction makes it possible to identify empty pulses and to insert approximated missing echoes in these cases. Additionally, the techniques presented in this chapter scale to arbitrarily large ALS data sets, since data can be streamed from disk.

The framework presented in this chapter is used in both automatic building extraction (Chapter 12) from large ALS data sets and visualization (Chapter 10) of ALS points. In terms of visualization, flight path information is used to create a novel type of image from ALS data, as described in the following chapter.

Chapter 10

Occlusion Images

The topic of this chapter is visualization of ALS data. The goal of *Aerial Laser Scanning* (ALS) data visualization is two-fold: (1) to visualize the objects that were scanned; and (2) to visualize points acquired on scanned objects. Visualizing scanned objects is of importance in cases where it is necessary for a human to identify objects. For instance, in semi-automatic building extraction the user must be presented with a clear visualization of buildings in order to perform the task of selecting relevant buildings. Further, in automated building extraction it is necessary to evaluate results using some ground truth. For this task it is important that visualizations not only show buildings, but also characteristics of the acquired data so that results can be properly analyzed.

Existing visualization tools are not suitable for three-dimensional ALS data sets of urban areas, such as those resulting from using the acquisition strategy presented in Chapter 8. Existing imaging techniques assume elevation data, and do not extend naturally to three-dimensional ALS data sets. Additionally, direct visualization of points does not accurately display the scanned objects due to the lack of surface information.

The novel imaging technique described in this chapter is based on visualizing patterns within ALS point data, as opposed to direct visualization of spatial measurements. The key idea of this technique is that ALS points can be treated as visibility samples. More specifically, occlusion patterns caused by the scanned geometry are visualized and the resulting images are referred to as *occlusion images*.

The first section of this chapter evaluates current techniques for ALS data visualization. This is followed by a description of occlusion patterns in the context of a method used in computer graphics to approximate realistic shadows. Thereafter, it is shown that the principles of this method can be applied in a reverse sense to ALS point data for creating accurate visualizations in the form of occlusion images.

10.1 ALS Data Visualization

At present there are two main approaches for visualizing ALS point data: (1) *direct point visualization*; and (2) *digital imaging*. These approaches are discussed separately below and will be clear from these discussion that neither alternative, in its present form, is suitable for visualizing large amounts of three-dimensional point data of an urban scene.

10.1.1 Direct Point Visualization

Small ALS data sets can be visualized by directly drawing the points on the screen. However, direct point visualization suffers from the fact that points are singular and only represent surfaces at discrete locations. Although three-dimensional points can easily be drawn on the screen, the lack of surface information between point samples causes objects to be visible through each other, making it difficult for users to clearly identify individual objects, let alone identify small features on objects. Existing techniques for visualizing points as continuous surface patches, commonly referred to as *point splatting* techniques, assume densely sampled points on a single object. Neither assumption is true for ALS data (e.g. [RL00, RPZ02]).

Letting users control a virtual camera interactively makes it possible to find clear views of individual objects in some cases. However, as the number of visualized points increases it becomes increasingly difficult to find clear views of individual objects. Also, consistently interactive frame rates become hard to achieve when the number of points is large. In fact, data sets may be too large to fit in main memory, requiring elaborate caching techniques to achieve interactive camera navigation.

In conclusion, direct point visualization is capable of visualizing three-dimensional point data, but does not scale well to urban areas. The other main approach for visualizing ALS data is to create a digital image from the points, as discussed next.

10.1.2 Digital Imaging

In contrast to direct visualization of unorganized ALS points, digital images present data in a well-structured, regularly sampled fashion. Digital images created from ALS data typically present top-down views and are easy to navigate since most users are already familiar with regular maps. Additionally, digital images do not require users to learn specialized software, as digital images are already part of many routine tasks. Finally, by choosing appropriate pixel dimensions it is possible to control memory usage, avoiding the need for elaborate caching techniques. For these reasons digital images are used extensively in ALS data visualization.

The most common type of digital image used in ALS data visualization is an elevation image. While elevation images are suitable for visualizing terrain, they are not ideally suited to visualizing urban regions. For instance, it is not possible to visualize the exact locations of walls in the case of overhanging roof parts, as eaves and cornices extend beyond the plan view of the structural walls. Also, vegetation close to buildings may in some cases occlude building edges. Additionally, elevation images are not suitable for visualizing small features, such as roof details. There are two main reasons for this: (1) small features are only represented by a few pixels, since pixel dimensions must accommodate large scenes; (2) small features do not vary significantly in elevation when compared to variations across a large scene. Finally, as discussed next, elevation images are not capable of accurately visualizing three-dimensional point data, making them unsuitable for analyzing tools that operate on such data.

Recall from Chapter 3 that elevation images are created by mapping points to pixels and assigning pixel values proportional to point elevations. For regularly sampled elevation data, pixel dimensions can be set so that the mapping between points and pixels is one-to-one, i.e. there is only a single point mapping to each pixel. In this case pixel values are derived directly from point data. Moreover, every point contributes to a pixel value.

However, for irregularly sampled elevation data it may not be possible to choose pixel dimensions that provide a one-to-one mapping between points and pixels. Cases where several points map to the same pixel are typically resolved by assigning a pixel value proportional to some extreme value (minimum or maximum) of point elevations. As such, pixel values are still derived directly from point data, but some points do not contribute to a pixel value.

Further, in the case of three-dimensional point data it is impossible to choose pixel dimensions such that there is a one-to-one mapping between points and pixels. Thus, even if pixel values correspond to some extreme value, significant portions of the point data have no effect on pixel values. At present, no satisfactory methods exist for computing pixel values from multiple points in such a way that both scanned objects and data characteristics are clearly visualized.

While images are suitable for visualizing large-scale elevation data acquired with ALS, they are not suitable for visualizing three-dimensional point data. Rather than visualizing point data measurements (e.g. elevation) directly, the new imaging technique presented in this chapter visualizes patterns within the data. More specifically, these patterns are based on occlusions caused by the three-dimensional geometries of the objects being scanned. The following section explains occlusion patterns, and how these are used to increase the realism of computer generated images. Thereafter, it is shown how occlusion patterns can be computed from ALS data in order to generate occlusion images.

10.2 Occlusion Patterns

Occlusion patterns are used in computer graphics to achieve higher degrees of realism in computer generated images. As shown in Section 10.2.2, an increased sense of depth can be added to a scene by including shadows derived from occlusion patterns. In order to discuss a technique related to the construction of occlusion images, known as *ambient occlusion*, it is first necessary to give a brief overview of the different types of illumination models used in computer graphics. Thereafter, the details of ambient occlusion are explained, showing how shadows are computed by testing visibility toward the sky for points on three-dimensional models. This reasoning can be applied in a reverse sense to ALS data in order to produce occlusion patterns with limited knowledge of the underlying geometry.

10.2.1 Illumination Models

In order to produce realistic looking images of computer generated scenes it is necessary to simulate how light interacts with objects. Light is a complex phenomenon and interactions with objects depend on a large number of factors, including object shape and surface material properties. For real-time applications it is common to use *local* illumination models (e.g. [Pho75, Bli77]), limited to interactions between light sources and points on surfaces. Interactions are evaluated locally at surface points, ignoring other surfaces in the scene, which results in fast simulations. However, local illumination models are not capable of accurately simulating global lighting effects, such as shadows, which require simultaneous consideration of multiple surfaces in the scene.

To overcome the limitations of local illumination models several *global* illumination models have been introduced. By enabling global lighting effects, such as accurate shadows and reflections, global illumination models add significant amounts of realism to computer generated scenes. In particular, shadows are important for visualizing complex geometry, providing an added sense of depth in the scene [LB00]. Sophisticated global illumination models that involve tracing individual photons [Jen09] produce realistic images at the price of being computationally expensive. Several other types of global illumination models exist, but a complete treatise of illumination models is beyond the scope of this thesis. The most common local and global illumination models are discussed in more detail in [Wat00].

10.2.2 Ambient Occlusion

The remainder of this section focuses on a global illumination technique known as ambient occlusion, which was first introduced by Zhukov et al. [ZIK98]. Ambient occlusion is a technique used to approximate shadow effects under simplistic lighting conditions. Typically, light is assumed to be emitted evenly from the sky, which is represented as a plane positioned above the scene. The fact that the entire sky acts as a light source produces soft shadows. Models visualized with ambient occlusion appear similar to how objects appear on an overcast day, where clouds provide evenly distributed lighting across the sky.

The main idea in ambient occlusion is that surface points that are exposed to the sky receive more light than points that are occluded. Exposure, or *visibility*, to the sky is computed for surface points on a model by casting rays in a fixed number of evenly distributed directions. Each ray that reaches the sky without intersecting a surface contributes some illumination to the surface point. As such, an occlusion factor, based on the number of directions with a clear view of the sky, can be computed. Increasing the number of rays cast for each point provides more accurate occlusion factors. However, there is an associated computational cost for testing additional rays and the number of rays must be chosen to meet the required accuracy. The details of ray casting will not be discussed here since they are not central to the main idea of ambient occlusion. More detailed discussions on this topic can be found in [Bun05, KL05, HPAD06].

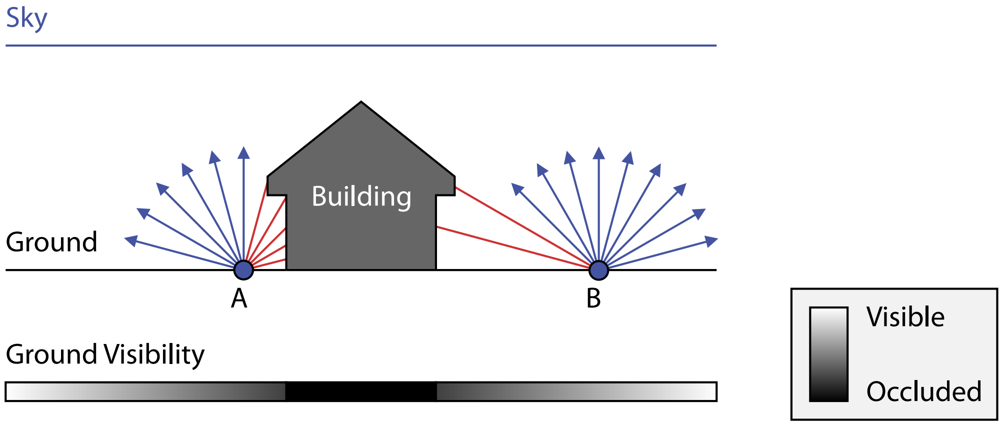


Figure 10.1: In ambient occlusion, visibility to the sky is tested in a fixed number of directions around each point. Blue arrows show directions with visibility to the sky, while red lines show directions that are occluded. An occlusion factor is computed based on the number of directions with a clear view of the sky. As such, point A has a higher occlusion factor than point B. Approximate occlusion along the ground plane is shown beneath the scene. Note how occlusion increases (or, equivalently, visibility decreases) near the building.

Figure 10.1 shows a simple scene consisting of a single building positioned on a ground plane. Consider the two points A and B located on the ground plane. Blue arrows show directions from these points in which the sky is visible, while red lines show directions in which visibility to the sky is occluded. Point B is more visible than point A because the sky is visible in a larger number of directions. Equivalently, point A is more occluded than point B. Approximate occlusion is shown beneath the ground plane. Interestingly, points on the ground closer to the building are more occluded than points further away from the building. Note that points on the ground plane beneath the building are completely occluded.

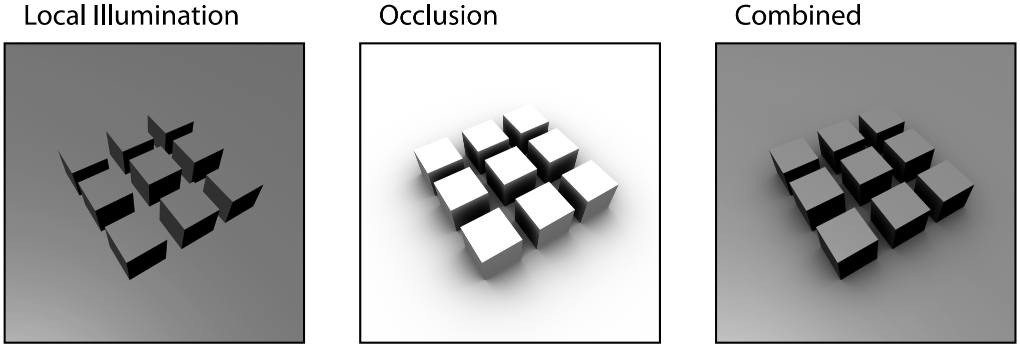


Figure 10.2: *Left*: A simple scene consisting of boxes on a plane rendered with local illumination. *Middle*: Occlusion pattern computed using ambient occlusion. *Right*: Local illumination combined with ambient occlusion.

The computed occlusion factors are often used in conjunction with local illumination models, thereby adding shadow effects that would otherwise be absent. Figure 10.2 illustrates the effects of ambient occlusion on a simple scene consisting of cubes resting on a plane. With a local illumination model it is difficult to achieve a sense of depth in the scene (left). An ambient occlusion pattern, using 256 visibility rays per point, is shown separately (middle). Darker regions in the occlusion pattern correspond to more occluded surface points. From this example it is clear that occlusion patterns can be fairly complex even for simple scenes. A combination of local illumination and ambient occlusion is shown (right) and it is clear that occlusion patterns provide clear visual cues to the nature of the underlying geometry. In fact, from the occlusion pattern alone (middle), the geometry of the cubes is visually clear. Thus, geometry can be visualized, to some extent, in terms of occlusion patterns. The following section describes an approach for generating occlusion patterns directly from ALS point data, where the key idea is to treat ALS points as visibility samples.

10.3 Visibility Sampling using ALS Data

In ambient occlusion, visibility to the sky is computed for points on the surfaces of objects. This section describes a reverse approach, where ALS points are treated as visibility samples on scanned surfaces. Instead of computing visibility from points on surfaces toward the sky, visibility is sampled from the sky toward points on surfaces. Thereby, it is possible to generate occlusion patterns without complete knowledge of the scanned geometry.

ALS points are acquired from an aircraft that moves above the scene. Since ALS is limited by line-of-sight, points are acquired on surfaces that are visible from the aircraft. Therefore, ALS points can be treated as a visibility samples, confirming that points on surfaces are visible from certain locations in the sky. The framework described in Chapter 9 is used to associate aircraft positions with ALS points, thus providing the locations in the sky from which points were acquired.

ALS points are acquired on surfaces, but the complete surfaces are unknown until some type of surface reconstruction has been carried out. In order to accumulate visibility for distinct regions points are mapped to image pixels. Pixels with points acquired from many different aircraft locations are considered to be highly visible and vice versa. Further, since ALS points are acquired on irregular geometries, visibility varies across an image. Using a horizontal image plane allows creation of an approximated occlusion pattern as perceived from a top-down view.

Next, details regarding visibility mapping are given, starting with the consideration of multiple echoes. Details on visibility accumulation from multiple aircraft locations are then provided. Thereafter, having described the basic mechanisms for creating occlusion patterns, an approach for producing smooth occlusion patterns is presented.

10.3.1 Pulse Visibility

As described in Chapter 2 it is possible for pulse energy to interact with multiple surfaces due to beam divergence, causing multiple echoes to be detected. Each echo corresponds to an ALS point and these points can be considered in the context of the pulse they were acquired in using the framework described in Chapter 9.

For the purpose of creating occlusion images, echoes are treated as visibility samples and each pulse defines a single unit of visibility. In the case of a pulse having multiple echoes, visibility is divided evenly across those echoes. The reason for this is that multiple echoes indicate partial visibility of multiple surfaces. Therefore, it is necessary to define the visibility contribution of each pulse echo, v_e, as follows:

$$v_e = \frac{1}{N_p} \tag{10.1}$$

where N_p is the total number of echoes for the pulse.

Visibility for pixels is determined by accumulating visibility contributed by mapped echoes. More specifically, pixel visibilities are equal to the sum of the visibility contributed by mapped echoes. However, a pixel cannot receive more than one unit of visibility from a single aircraft position. Thus, pixel visibility, v_p, is computed as:

$$v_p = \min(\sum v_e, 1) \tag{10.2}$$

where $\sum v_e$ is the total amount of visibility contributed by mapped echoes. As such, pixel visibilities measure the number of locations in the sky from which any surface point mapping to the pixel is visible. This corresponds to how visibility is tested in ambient occlusion, where ray casting is used to test visibility from selected surface points to sky locations.

Figure 10.3 illustrates the visibility distribution of two pulses. First, consider the case of small pixels. The left pulse has three echoes, where the second and third echoes map to the same pixel. These two echoes each contribute 1/3 visibility to that pixel, making a total of 2/3 visibility. The first echo maps to a different pixel, which receives a contribution of 1/3 visibility. The right pulse only has one echo, which contributes exactly one unit of visibility to the pixel it maps to.

Visibility distribution depends in no small part on pixel dimensions. Consider the large pixels in Figure 10.3. In this case, all echoes of the left pulse map to the same pixel and the fact that visibility is contributed from multiple echoes is unnoticeable. Thus, using smaller pixel dimensions will, in general, provide more fine-grained occlusion patterns. However, pixel dimensions must take into account the sampling density of ALS points. Using small pixel dimensions may results in many pixels without mapped echoes. Having demonstrated how

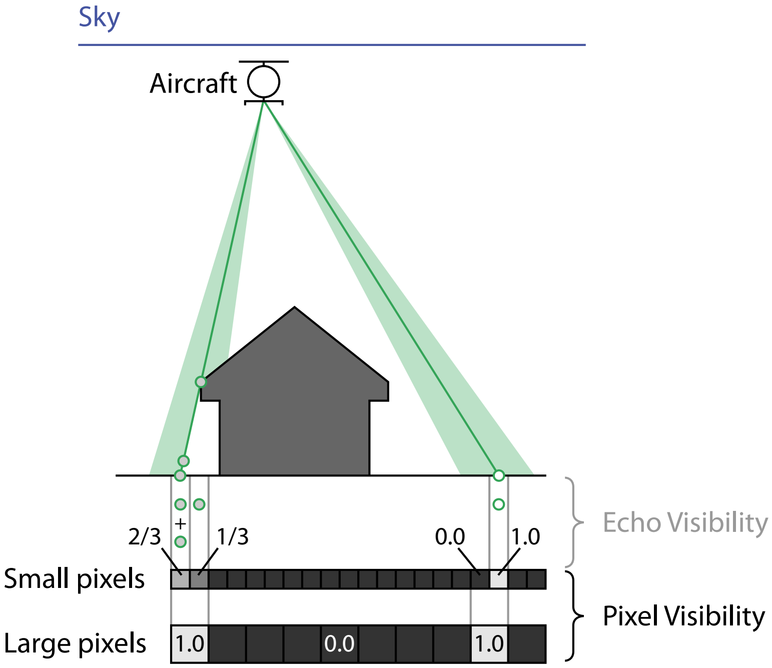


Figure 10.3: Each pulse is considered to be one unit of visibility, evenly distributed across echoes. Echoes are mapped to pixels and visibility is accumulated according to the contributions of the mapped echoes.

visibility is transferred from pulses to pixels, the following section addresses how to combine visibility information from multiple aircraft positions.

10.3.2 Visibility Accumulation

The previous section explains how visibility is mapped from pulse echoes to pixels in the case of a single aircraft position. In order to generate useful occlusion patterns it is necessary to combine visibility information from multiple aircraft positions.

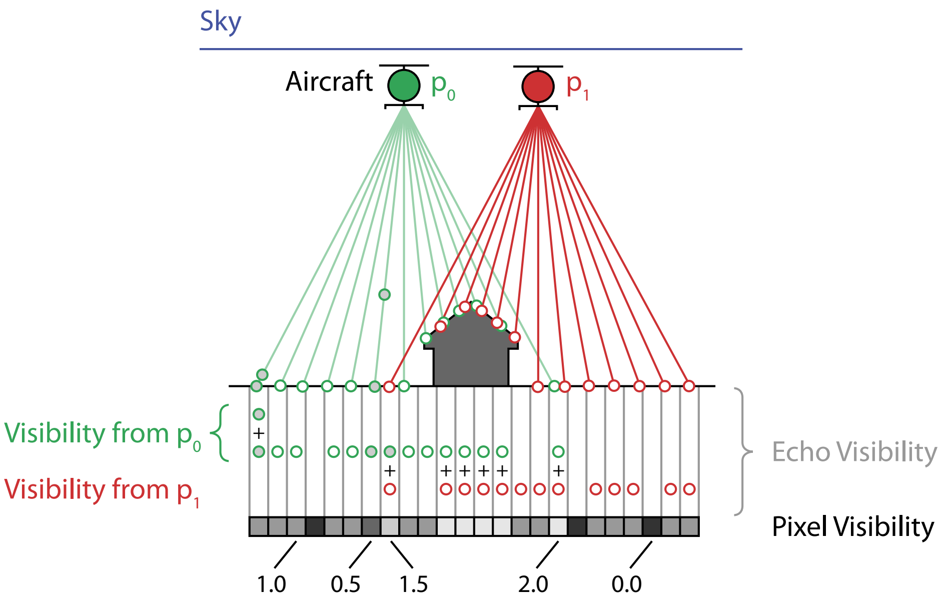


Figure 10.4: Echoes from multiple aircraft positions are mapped to pixels and visibility contributions are added and stored in pixels.

Visibility from multiple aircraft positions are added to produce accumulated pixel visibilities, as illustrated in Figure 10.4. Note that even though pixels can receive only a single unit of visibility from one aircraft position, it is possible for pixels to receive more than one unit of visibility when multiple aircraft positions are considered. In general, the maximum pixel visibility is equal to the number of aircraft positions considered. However, this is unlikely to occur in practice, since ALS data is acquired over vast areas with a limited field-of-view from any single aircraft position. When two aircraft positions are considered interesting occlusion patterns start to emerge. Note how the building roof is visible from both aircraft positions, whereas regions on the ground near the building are mostly visible from one of the positions. Such occlusion patterns exist on both large and small scales, and can be visualized using appropriate pixel dimensions. In order to visualize small scale occlusions it is necessary for pixel dimensions to be small in global coordinates.

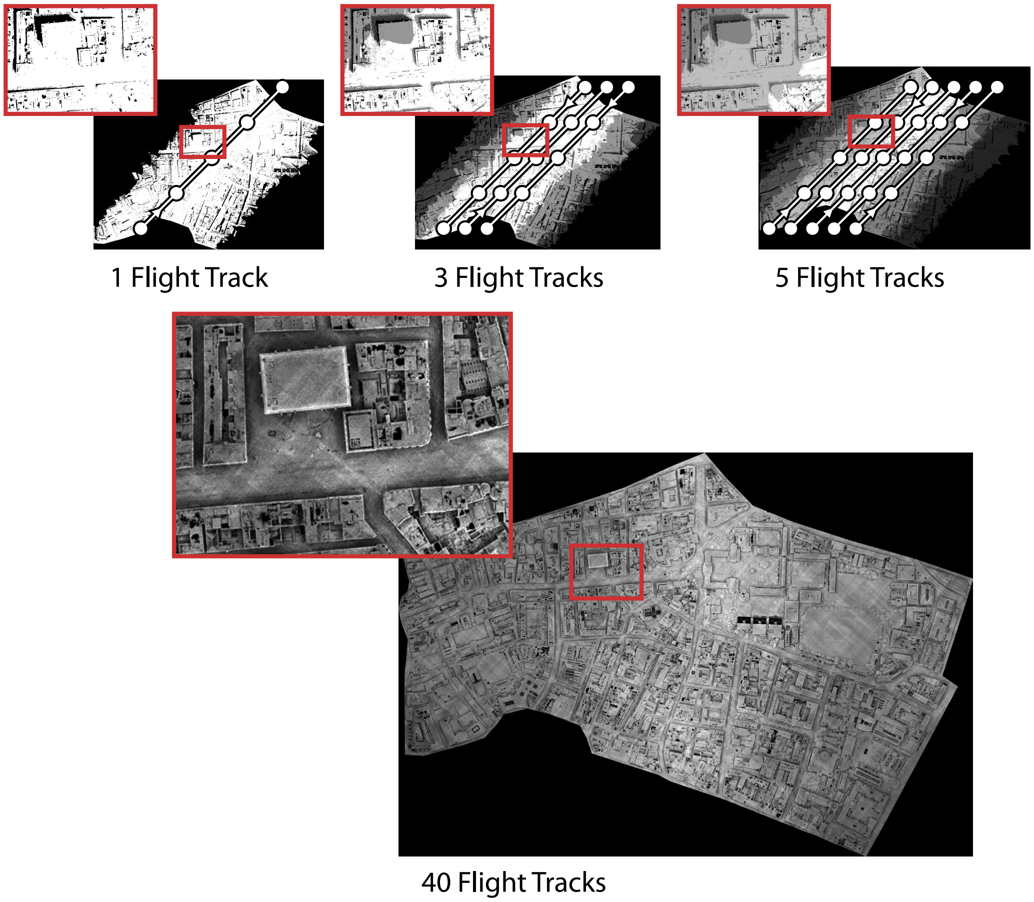
In the examples above each scan line has been considered to be acquired from a unique aircraft position. Even though this is true, in the sense that the aircraft is considered to be stationary during the acquisition of a single scan line, it is possible to extend the concept of unique aircraft positions to be more flexible, as discussed in the following section.

10.4 Flight Path Sampling

In this chapter ALS points are treated as visibility samples from aircraft positions in the sky. The number of ALS points from unique aircraft positions mapping to pixels are used to compute occlusion patterns. If the uniqueness of aircraft positions is taken literally, points in each scan line are acquired from distinct aircraft positions. This section extends the concept of uniqueness for aircraft positions. More specifically, the hierarchical flight path model introduced in Chapter 9 is used to generalize the concept of unique aircraft positions.

The different levels of the hierarchical flight path model presented in Chapter 9 correspond to aircraft motion over different periods of time. At the lowest level instantaneous aircraft positions are represented as flight path points. Higher levels represent aircraft motion over increasing periods of time. The idea presented in this section is that the concept of unique aircraft positions can be applied to any level of the flight path hierarchy. The example below is used to illustrate this idea.

Figure 10.5 shows an example where each flight track is considered as a unique aircraft position. In order to illustrate how an occlusion pattern emerges, points acquired in different flight tracks are added progressively. The leftmost image in the top row shows the occlusion pattern after considering points acquired in a single flight track. The other two images in the top row illustrate the effects of progressively adding flight tracks. The amount of detail visible in the insets increases with the number of aircraft positions (i.e. flight tracks) considered. This is not surprising, since visibility is being sampled from additional aircraft positions. Thus, visibility is considered in an increasing number of directions to the sky. The enlarged bottom image shows



1 Flight Track

3 Flight Tracks

5 Flight Tracks

40 Flight Tracks

Figure 10.5: Progressive flight track visibility accumulation. Pixel dimensions in global coordinates are $\Delta x_w = \Delta y_w = 0.5\ m$.

the occlusion pattern after adding 40 flight tracks. Clearly, the amount of detail has increased even further.

Besides illustrating how occlusion patterns emerge, Figure 10.5 emphasizes the requirement for flight strips to overlap. When ALS point data is acquired in overlapping flight strips, it becomes possible for points mapping to the same pixel to have been acquired from different aircraft positions. As such, the techniques presented in this chapter are suitable for visualizing ALS data acquired using the strategies presented in Chapter 8.

In summary, the imaging technique presented in this chapter differs from existing techniques in that pixel values are derived from patterns found in ALS data, as opposed to being derived from physical measurements, such as elevation. In Part III examples of occlusion images are presented and discussed in detail. Comparisons with existing techniques are made and the impact of parameters such as pixel dimension and flight path sampling are explored. Finally, in this thesis occlusion images are used in both (Chapter 11) and automatic (Chapter 12) building extraction. In the semi-automatic building extraction tool presented in the following chapter, occlusion images are used to provide users with clear visualizations of the available ALS data.

Chapter 11

Semi-automated Building Extraction

This chapter presents a simple tool for semi-automatically extracting regions of interest from large *Aerial Laser Scanning* (ALS) data sets. By interactively specifying regions in images created from ALS data (Chapter 10), users are able to select features of interest, such as buildings, for extraction. For the purpose of identifying regions of interest, occlusion images are used for visualization, since they clearly show buildings and building features. Digital images created directly from ALS data have the advantage that any blank spots in the data are identifiable. In contrast, maps and other resources do not visualize the available data and may be out of date.

The interactive tool presented in this chapter is complementary to the automatic building extraction tool presented in the following chapter (Chapter 12). While automatic building extraction tools are ideally capable of extracting all buildings of an ALS data set, it is sometimes necessary to extract buildings that could not be automatically detected, or even regions other than buildings. Although the tool presented in this chapter is not practical for extraction of all buildings from a large ALS data set, it provides flexibility in terms of allowing users to manually extract any region of an ALS data set. As such, the tool presented in this chapter is suitable for data exploration and allow parts of a larger ALS data to be examined in detail.

As mentioned, regions to be extracted are selected in occlusion images. At present, two types of selection are supported: (1) *polygon selection*; and (2) *line selection*. These two selection mechanisms have in common that a region is selected in an occlusion image. Points are tested against this region and are extracted if they are found to belong to the selected region. Regions are selected in the image plane, which typically lies in the horizontal plane of a global coordinate system. Thus, points are extracted based on horizontal coordinates.

Examples of semi-automatic building extraction are demonstrated in the form of images showing selections and the correspondingly extracted points. The two different types of selection are presented in separate sections below. In both cases a high-resolution occlusion image, with pixel dimensions in global coordinates being $\Delta x_w = \Delta y_w = 0.2\ m$, was used to visualize the provided ALS points. Polygon selection is discussed first, followed by line selection.

11.1 Polygon Selection

Polygon selection allows users to select regions by specifying the vertices of a polygon. ALS points with horizontal coordinates inside the region bounded by the polygon are extracted. This type of selection is useful for extracting entire buildings, since polygons corresponding to building footprints can be specified. Additionally, polygon selection is suitable for extracting large regions, which can be selected with a polygon of appropriate size and shape.

Figure 11.1 shows an example of polygon selection in an occlusion image. The selection covers the Rubrics building on the university campus of Trinity College Dublin, Ireland. Vertices of the polygon were specified as the four corners of the building, whereby the polygon area corresponds closely to the building footprint. A perspective view of the extracted points is shown in the inset, demonstrating that the correct points were extracted. The ALS data set used in this example was acquired using the methods described in Chapter 8. Note the resulting presence of points on the building walls. Further examples from this data set are shown in Part III.

Polygon extraction is suitable for extracting an entire building from a large ALS data set. However, for the purpose of creating solid models for building simulations it is necessary to extract individual walls of buildings. The line selection tool presented in the following section was designed particularly for this purpose.

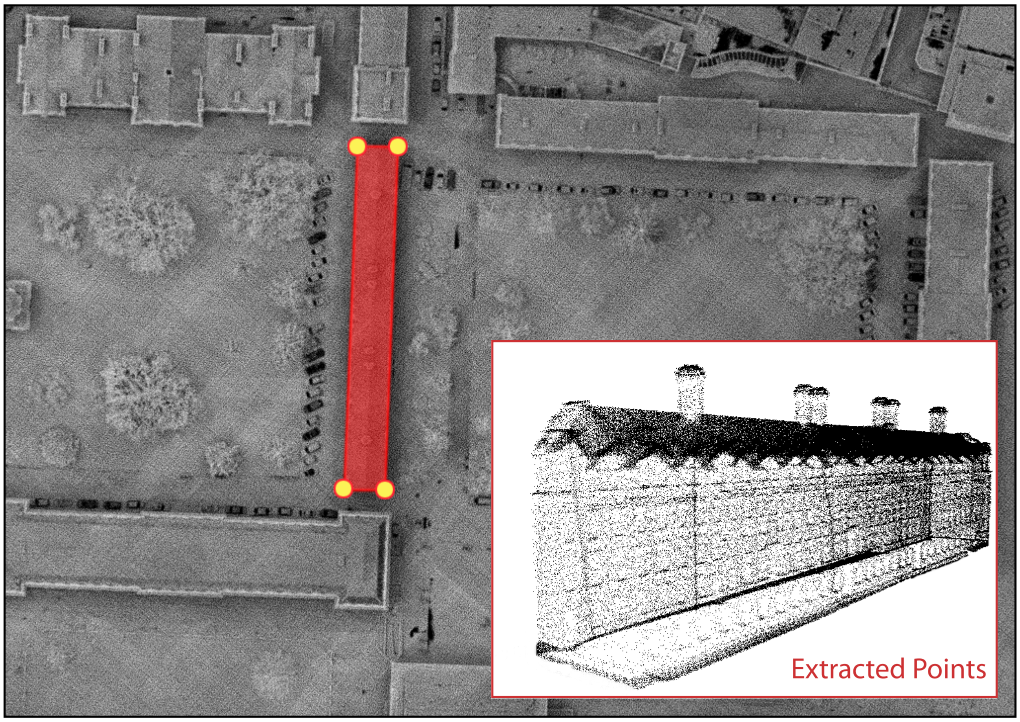


Figure 11.1: By specifying a polygon corresponding to a building footprint points acquired on that building were extracted. A perspective view of the extracted points is shown in the inset.

11.2 Line Selection

Line selection allows users to select regions by specifying the start and end points of a line segment together with a thickness. ALS points are extracted based on perpendicular distance to the line segment in the horizontal plane. More specifically, points within half the line thickness are extracted. The line segment thickness is typically quite small, since only points close to the wall are of interest. In addition to being extracted for further processing, ALS points are transformed into an orthogonal coordinate system that is local to the specified line segment. This coordinate system has the same vertical axis as the global coordinate system in which ALS points are expressed. However, the x-axis of this coordinate system is parallel to the line segment and the y-axis is perpendicular to the line segment. Therefore, since the thickness is small, y-coordinates of transformed points are close to zero. This means that transformed point coordinates vary mostly in two directions, which is useful for reasons discussed further in Chapter 13.

Figure 11.2 shows an example of line selection in an occlusion image. The selection covered a single building wall and a small line thickness was used to extract only points close to the wall. As in the previous example, the extracted points clearly illustrate the presence of points on the building wall.

In summary, semi-automatic building extraction is useful in cases where only a single building or building wall is of interest. The two different selection mechanisms offer flexibility and provide the functionality required to manually extract relevant parts of a large ALS data set. Line selection is particularly well-suited for extracting points on individual building walls, while polygon selection is better suited for extracting entire buildings. However, semi-automatic building extraction in general is not practical for urban areas containing large numbers of buildings. In such cases higher degrees of automation is required. The following chapter presents a tool for automatically extracting points on buildings in large ALS data sets.

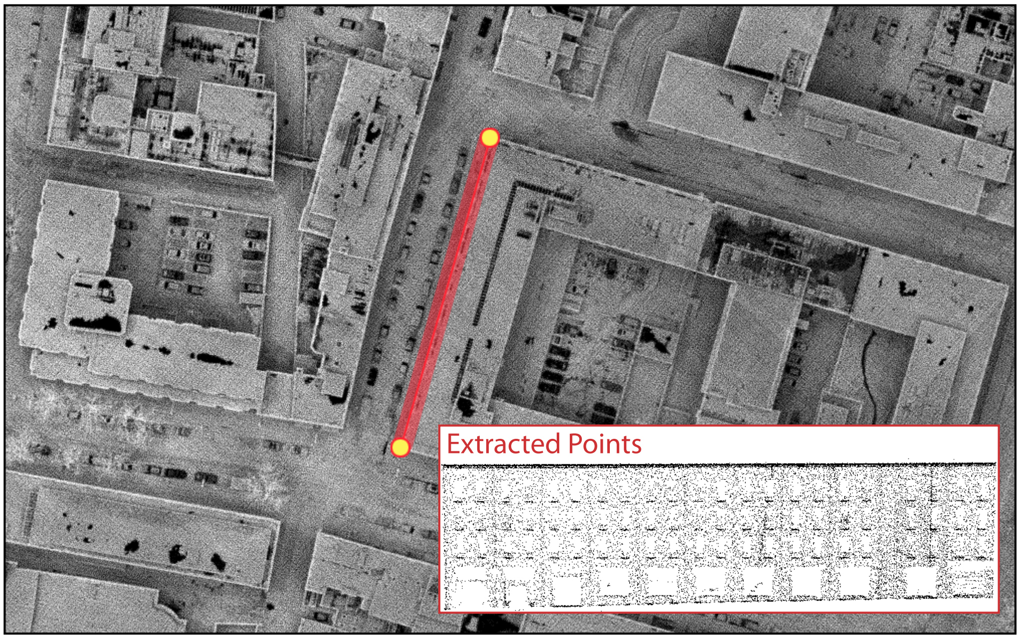


Figure 11.2: A line corresponding the horizontal projection of a building wall was specified by clicking two building corners in the occlusion image. Thereafter a line thickness was specified and points closer to the line than this thickness were extracted. An orthogonal view of the extracted points is shown in the inset.

Chapter 12

Automated Building Extraction

Given a large *Aerial Laser Scanning* (ALS) data set of an urban area, the goal in automatic building extraction is to divide the input data into several smaller parts, where each part contains the points acquired on a single building. Such division allows subsequent modeling steps to operate on individual buildings, greatly reducing the complexity of such tasks. Automatic detection and extraction of buildings in ALS data sets is a fundamental task in urban modeling, as discussed in Chapter 4. This chapter describes an automated building extraction technique based on the framework for ALS data processing presented in Chapter 9. Additionally, many of the digital image processing techniques introduced in Chapter 3 are relevant to the material presented in this chapter.

Cities are geometrically complex, containing objects of varying sizes and shapes. Previous successful approaches for automatic building detection have often been tested on suburban or rural areas, where buildings are clearly separated and roofs consist of well-defined, intersecting planes. In dense urban areas the situation is different. There buildings share walls, forming conglomerations of buildings and roofs are used as storage locations and are cluttered with smaller objects. Few automatic building extraction tools have been tested under these conditions and this area remains open for improvement.

The building extraction technique presented in this chapter operates under the common assumption that buildings are bound by a set of vertical surfaces, i.e. walls. This assumption is true for a vast majority of buildings, although there are exceptions in certain artistic architecture, examples being various cathedrals, museums, and opera houses. A shape in the horizontal plane representing the plan view of building walls is referred to as a *building outline* (Figure 12.1). Building outlines typically do not include overhanging roof parts. A building outline along with the interior region covered by a building is referred to as a *building footprint* (Figure 12.1). Note that building footprints require building outlines to be water-tight in the sense that there must exist a clearly defined interior region bounded by the building outlines. A more elaborate definition of a valid building outline is presented further on.

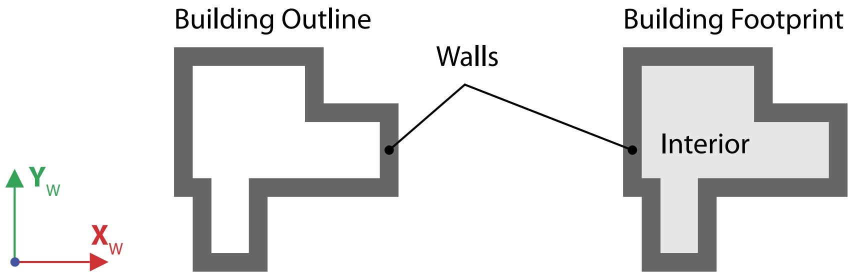


Figure 12.1: *Left*: A building outline is a shape representing the locations of the walls of a building in the horizontal plane. *Right*: A building footprint consists of building outlines along with the interior region covered by a building.

Given the building footprints of an urban area, points can be grouped by building using the mapping techniques described in Section 3.2. Assuming building footprints exist in the form of a digital image, ALS points are mapped to pixels, which contain information about building coverage. Existing building plans may provide building footprints and have been used with ALS data for building extraction [VD01], but such resources are not always available and up-to-date. Furthermore, for such an approach to work at all, building plans are required to be in digital form, whereby conversion is sometimes necessary and may be time-consuming.

The building technique presented in this chapter computes the building footprints for an urban area directly from ALS data, thus entirely providing the resources required for building extraction. Note that the technique presented in this chapter is not aimed at generating building footprints, but rather uses such information as an intermediate step. The generation of proper building footprints requires significant attention to the outlines of buildings, such as regularization of straight edges and enforcement of symmetry, which are beyond the scope of this thesis.

The main idea behind the building extraction technique presented in this chapter is that wall segments can be identified in individual scan lines found in ALS data. In particular, urban ALS data acquired using the flight path algorithms presented in Chapter 8 is suitable for this purpose, since this type of ALS data contains significant amounts of points on building walls. As such, the technique utilizes the presence of ALS points on building walls and is not limited to elevation data. ALS points are examined within the context of individual scan lines and points on wall segments are classified based on vertical stacking. Classified points are mapped to an image, transferring point classifications to image pixels. Hence, wall segments identified in scan lines are collected and, ideally, form water-tight building outlines in the image. Robust detection of valid building outlines is then performed on the image using a series of morphological operations. Further image analysis provides building footprints for correctly detected buildings. Finally, the resulting image contains building footprints, allowing points to be grouped by building.

Next, point classification in scan lines is described, followed by detailed descriptions of the image processing techniques used to produce the final building footprint image.

12.1 Scan Line Classification

Classification is the process of dividing entities into different categories [DHS01]. This section describes the process of classifying ALS points into one of three *categories*, or *states*. Note that the same categories are used in Section 12.2 to classify image pixels. Before presenting the methods used to classify points, an explanation of the categories is provided.

The three categories into which ALS points are classified are $\{\omega_v, \omega_{in}, \omega_{out}\}$ and these categories correspond to point locations in relation to vertical surfaces. The category ω_v is assigned to points believed to be sampled on vertical surfaces. For the purposes of ALS point classification any detected vertical surface is hypothesized to be a building wall segment. These hypotheses are tested in Section 12.3. As will be shown further on, it is useful to identify not only points sampled on vertical surfaces, but also the spatial relations of other points to these surfaces. Whereas ω_v points are believed to be sampled exactly on building outlines, the categories ω_{in} and ω_{out} are assigned to points believed to be horizontally inside or outside building footprints, respectively. As an example, points on (non-overhanging) roofs belong to ω_{in}, whereas points on streets belong to ω_{out}. Additionally, a null category is used to denote points that do not belong to one of the three categories above. Points classified in the null category do not contribute to the extraction of building footprints.

ALS point classification is performed on a per scan line basis. Individual scan lines are extracted from large ALS data sets using the framework presented in Chapter 9. Classification of points into categories is based on patterns within scan lines. In particular, the traversal order described next allows vertical surfaces to be detected as vertical stacking of sequential points in scan lines. Thereafter, methods for point classification are presented.

12.1.1 Scan Line Traversal

Scan line points are acquired by emission of pulses at regular angular offsets in the across-track direction. Due to the fact that the aircraft is notionally stationary during acquisition of a single scan line, the horizontal coordinates of points in the same scan line lie on a roughly straight line in the horizontal plane. As is typical for urban ALS applications, only the last echo of each pulse is used. Additionally, missing echoes in empty pulses may be inserted into each scan line, using the techniques described in Chapter 9. The benefits of inserting missing echoes for classification purposes are described in more detail in coming sections.

As discussed in Chapter 8, only vertical surfaces facing nadir can be scanned from a given aircraft position. Assuming fairly small rotation angles (a few degrees), the preceding statement is true regardless of aircraft orientation. Furthermore, if ALS points in a scan line are traversed from nadir out to the edges of a scan line, sequential points sampled on a vertical surface appear in an order where elevation increases while horizontal coordinates remain roughly equal.

Therefore, scan line points are divided into two groups, based on which side of nadir they are on. Division is based on horizontal perpendicular distance to the flight direction (Figure 12.2). A set of distances, $D = \{d_0, ..., d_{n-1}\}$, is computed, where d_i is the distance for point p_i and n is the number of points in the scan line. Based on the signs of distances in D, scan line points are divided into two groups D^+ and D^-. Points in D^+ and D^- are sorted in ascending order with respect to absolute distances, which allows sequential traversal of the points in each group from nadir out the respective edge of the scan line. Thus, points in D^+ and D^- exhibit similar patterns with respect to vertical surfaces, providing useful symmetry in the subsequent classification step, which is described next.

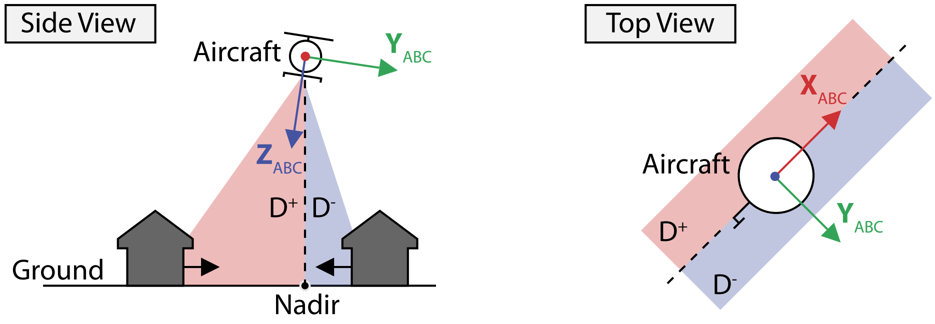


Figure 12.2: *Left*: Only vertical surfaces facing nadir can be scanned from a certain aircraft position. Scan line points are divided into two groups, D^+ and D^-, based on which side of nadir they are on. *Right*: Signed distances of points to the horizontal flight direction, $\vec{x}_{ABC}$, line is used to divide the points in two groups.

12.1.2 Scan Line Binning

As mentioned previously, scan line points are divided into two groups and each group is processed separately. Assuming that scan line points lie in a plane that is roughly vertical and that points are traversed from nadir to scan line edges, consecutive points sampled on the same vertical surface exhibit two interesting patterns: (1) points have roughly identical horizontal coordinates; and (2) points appear in increasing elevation order. These two patterns are used to classify scan line points, as explained below. The following discussions are valid for both D^+ and D^-, provided that these groups are traversed from nadir to the respective scan line edges.

The traversal order means that surfaces facing nadir are detected. Given that vertical surfaces are difficult to sample directly beneath the scanner, scan line points preceding building wall segments are assumed to be outside building footprints. Similarly, scan line points just after building wall segments are assumed to be inside building footprints.

In order to compare the horizontal coordinates of scan line points they are mapped to equally sized bins. Points mapping to the same bin are considered to have equal horizontal coordinates. The line extending from nadir to the horizontally most distant point in each group is divided into a set of bins $B = \{b_0, ..., b_{m-1}\}$, where m is the number of bins and is computed as:

$$m = \left\lceil \frac{\max d_j}{\Delta x_b} \right\rceil$$

where $\max d_j$ is the maximum horizontal distance to nadir in the group and $\Delta x_b > 0$ is a user-specified width of each bin. As such, every point is guaranteed to have a valid bin mapping. Points are traversed sequentially and are mapped to bins using the following expression:

$$b_i = \left[\frac{d_j}{\max d_j} \right]$$

where d_n is the horizontal distance to nadir of the mapped point and the operator $[x]$ rounds x to the nearest integer toward zero (Equation 3.3).

Mapped points are potentially stored in bins. There are two rules determining if a point is stored in the bin it maps to: (1) the previous point maps to the same bin; and (2) the point has a larger elevation than the previous point. An exception is the first point in each group, which is always added to the bin it maps to. After all points have been mapped to, and potentially stored in, bins, examination of bins proceeds based on the number of points stored in each bin.

Points in bins containing more than a user-specified threshold number, T_b, of points are classified as ω_v, indicating that these points are sampled on a vertical surface. Moreover, points in the preceding bin are classified as ω_{out}. Similarly, points in the next bin are classified as ω_{in}. Points that are not stored in bins classified according to the rules just mentioned are classified as null. Such points are not further considered for the purposes of building extraction.

The addition of missing echoes in empty pulses (Chapter 9) has advantages in the binning process described above. There are three main reasons for this: (1) pulse interaction with semi-transparent materials, such as glass, rarely provides detectable echoes, and the insertion of missing echoes in these cases provides a higher degree of sampling on vertical surfaces; (2) pulses often interact with vertical surfaces at oblique angles, owing to the fact that the aircraft is positioned high above the scene. Such interactions do not reliably provide detectable echoes; and (3) many roofs consist of metal plating, which reflect incoming pulse energy in a specular fashion. By inserting missing echoes, ω_{in} points are more likely to be close to detected vertical surfaces.

The point classification described in this section is local to each scan line. In order to extract building outlines, which are not present in individual scan lines, it is necessary to collect scan line classifications in a larger context. This is achieved by mapping classified points to the pixels of an underlying image, as described next.

12.2 Statistical Images

This section describes the type of image used to collect information from mapped ALS points classified in individual scan lines. Interactions between classified ALS points and image pixels allow wall segments detected in scan lines to be considered in the context of building outlines in a horizontal image plane. These images are referred to as *statistical images* and are distinguished by the fact that pixels store classification information from mapped ALS points. More specifically, each pixel stores the number of mapped points from each of the categories $\{\omega_v, \omega_{in}, \omega_{out}\}$, introduced at the beginning of Section 12.1. Thus, each pixel stores a triplet (n_v, n_{in}, n_{out}), where each value is the number of mapped points from the corresponding category. Pixels are initialized to $(0, 0, 0)$, which denotes the fact that no points have yet been mapped. Extraction of building outlines in statistical images is treated in Section 12.3. Before that, the details of statistical images are presented.

In order to map ALS points to pixels, statistical images must have bounds in global coordinates, as explained in Section 3.2. Statistical image global coordinate bounds are determined by the horizontal bounds of the provided ALS point data, guaranteeing that every ALS point maps to a valid pixel coordinate. The length of a pixel side in global coordinates, Δx_s, is a user-specified parameter that determines the pixel dimensions of the statistical image. Since each pixel stores information about the features within an area $A = \Delta x_s^2$ it is suitable to choose Δx_s such that there is a single feature within A. The main features of interest are building outlines and Δx_s should not be larger than the width of a building wall, including protrusions. If Δx_s is larger than the width of a building wall, building outlines may erroneously be connected. However, if Δx_s is too small, wall segments identified in scan lines will not form closed shapes in the image plane, thus preventing valid building outlines to be extracted.

In order to extract building outlines from statistical images it is necessary to classify pixels into one of the abovementioned categories. However, as mentioned, pixels do not store explicit classifications. Instead, each pixel stores enough information for classification to be determined by the evaluation of some function. A suitable function for this task is presented next.

12.2.1 Pixel Classification

Pixel classification is determined by evaluating a function, $f_p(n_v, n_{in}, n_{out}) \in \{\omega_v, \omega_{in}, \omega_{out}\}$, which takes as input a pixel triplet and evaluates to a classification category. The following function is used to determine pixel state:

$$f_p(n_v, n_{in}, n_{out}) = \begin{cases} \omega_v, & \text{if } n_v > 0 \\ \omega_{in}, & \text{if } n_{in} > n_{out} \\ \omega_{out}, & \text{if } n_{out} \geq n_{in} \end{cases} \tag{12.1}$$

Cases are tested in order, which means that the mapping of a single ω_v point causes pixel state to be ω_v. The rationale behind this is that ω_v pixels represent wall segments, which are more likely to form closed building outlines with a larger presence of ω_v pixels. Similarly, preference is given to ω_{out} in the case where $n_{out} = n_{in}$, motivated by the fact that a pixel is more likely to be outside a building outline than inside. In the special case of $n_v = n_{in} = n_{out} = 0$ pixels are classified as null, denoting the fact that no ALS points related to building extraction map to the pixel.

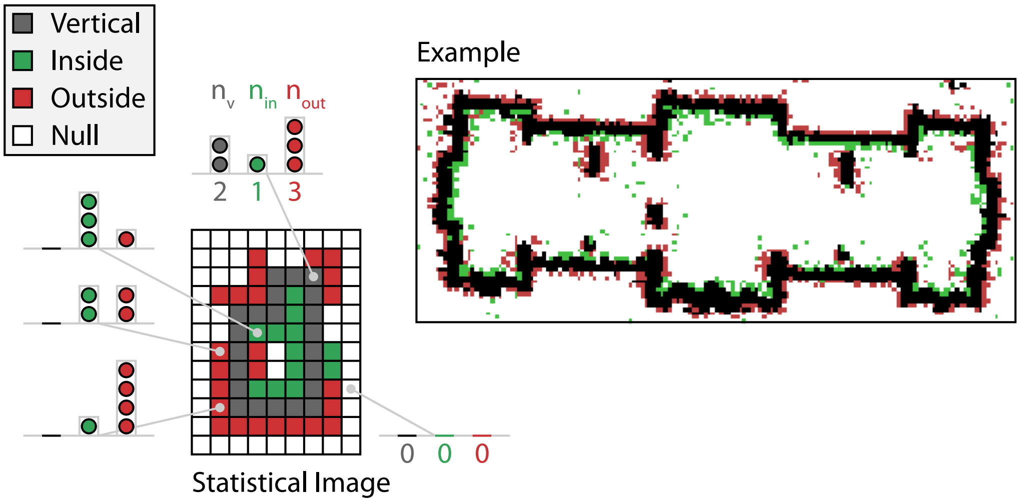


Figure 12.3: *Left*: Pixel classification is based on Equation 12.1, which takes as input the number of mapped ALS points in each category: (n_v, n_{in}, n_{out}). *Right*: An example demonstrating pixel classifications around a building. Note the inconsistencies where pixels outside the building outline are classified as $\omega_{in}$and vice versa for ω_{out} pixels.

Figure 12.3 illustrates pixel classification and shows an example of a statistical image generated from ALS data. To the left, the classification mechanism of pixels is shown, where pixels are classified based on the categories of mapped points using Equation 12.1. On the right an example is shown for a real building. Note that there are inconsistencies in that ω_{in} pixels appear outside the building outline and vice versa for ω_{out} pixels. These discrepancies are caused by the binning procedure used in the classification of scan line points, where ω_{in} and ω_{out} points do not always map to neighboring pixels of ω_v points.

In summary, statistical image are used to collect information from ALS points classified in individual scan lines. Statistical image pixels store the number of points in each classification category mapping to them and the classifications of pixels is based on this information. Ideally, building wall segments detected in scan lines appear as building outlines the subsequent in statistical images. However, in order to verify that building outlines are closed and to determine which regions within closed outlines belong to building footprints, further analysis is required and is discussed next.

12.3 Building Outline Extraction

Statistical images contain aggregate information from ALS points classified in scan lines. While building wall segments are identified as ω_v points within scan lines, statistical images are used to collect information about building wall segments, allowing these to be considered in the context of building outlines. This section describes automatic extraction of building outlines from statistical images. Building outlines are extracted for the purpose of further generating building footprints, which, as mentioned, are used to group ALS points acquired on the same building.

In order to identify building outlines in statistical images, analysis of ω_v pixel connectivity is required. More specifically, building outline candidates are defined as *8-Connected Components* (8-CC) of ω_v pixels (Section 3.3). Candidates are extracted using *8-Connected Component Labeling* (8-CCL), as described in Section 3.6. In the labeling step ω_v pixels are considered to be *active* and all other pixels are considered to be *inactive*.

Each 8-CC of ω_v pixels is examined separately using morphological dilation (Section 3.4) to determine the presence of well-defined interior regions. As mentioned, interior regions are required to generate building footprints. However, certain interior regions, such as courtyards, should be excluded from building footprints. Therefore, prior to building footprint generation, further analysis is carried out to determine if interior regions are part of building footprints using the information from ω_{in} and ω_{out} pixels. The separate treatment of building outline candidates is discussed next, followed by verification of these candidates.

12.3.1 Building Outline Sub-images

The statistical image created by mapping ALS points classified in scan lines is referred to as the *main image*. In order to isolate each 8-CC of ω_v pixels prior to further analysis, a rectangular region in the main image, containing the 8-CC itself and a one pixel neighborhood, is copied to a separate *sub-image* (Figure 12.4). The one pixel neighborhood is not possible at the edges of the main image and is not included in those cases. Pixel values from the main image are copied into the sub-image, with the exception that only ω_v pixels from one 8-CC of ω_v pixels are copied per sub-image. Thus, although sub-images from multiple 8-CC of ω_v pixels may overlap, each sub-image contains exactly one 8-CC of ω_v pixels.

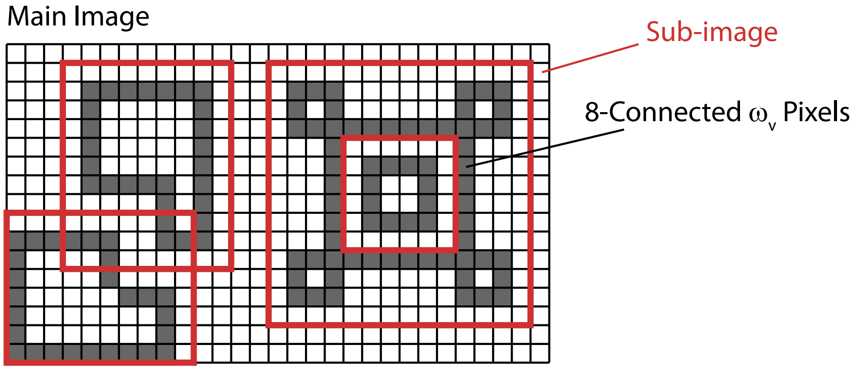


Figure 12.4: Sub-image separation for 8-CC of ω_v pixels.

The following discussions on building outline extraction assume that analysis is carried out in sub-images, where only a single 8-CC of ω_v pixels is present. Further, when sub-images are created, the location of the corresponding rectangular region in the main image is stored, such that results from sub-image analysis can be transferred back to the main image. Next, 8-CC of ω_v pixels are verified to find out if they form building outlines or not.

12.3.2 Building Outline Verification

In the context of building outline verification morphological dilation (Section 3.4) is used to define the neighborhood of an 8-CC of ω_v pixels. Considering ω_v pixels to be *active*, and all other pixels to be *inactive*, 8-dilation generates a set of dilated pixels in the sub-image (Figure 12.5). Dilation is performed separately in each sub-image containing exactly one 8-CC of ω_v pixels. Thus all dilated pixels have at least one 8-neighbor from the same 8-CC of ω_v pixels.

Following 8-dilation of ω_v pixels, *4-Connected Components* (4-CC) of dilated pixels are identified using *4-Connected Component Labeling* (4-CCL). A 4-CC of dilated pixels is referred to as a *dilation band* and at least one dilation band is guaranteed to exist. Further, a *closed* 8-CC is defined as an 8-CC with two or more dilation bands (Figure 12.5). In contrast, an *open* 8-CC has exactly one dilation band, which means that all dilated pixels are 4-connected, signifying that there is no clear separation between exterior and interior.

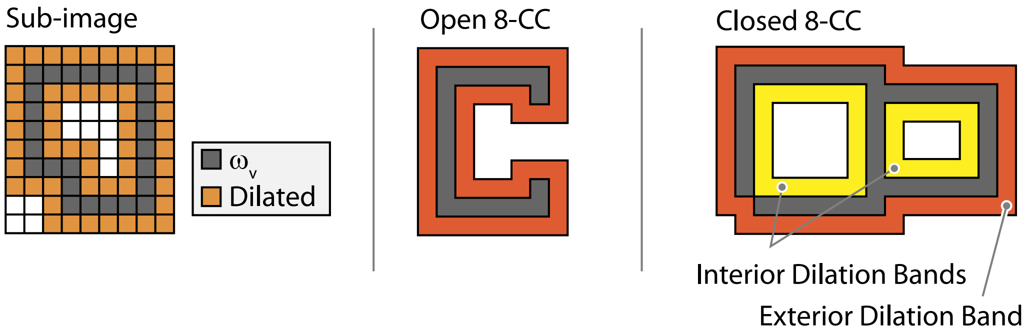


Figure 12.5: *Left*: Morphological 8-dilation is performed on an 8-CC of ω_v pixels. *Middle*: Open 8-CC have exactly one dilation band, signifying that there is no clear separation between exterior and interior. *Right*: Closed 8-CC have more than one dilation band.

Open 8-CC of ω_v pixels are not further considered as building outlines, since proper building outlines require the presence of clearly defined interiors and exteriors. As mentioned, closed 8-CC of ω_v pixels are distinguished by having more than one dilation band. In the presence of multiple dilation bands, at least one of them is guaranteed to be completely bounded by the closed 8-CC of ω_v pixels, verifying it as a building outline. Dilation bands are sorted by axis-aligned bounding box area. The dilation band with the largest bounding box area is referred to as the *exterior* dilation band. All other dilation bands are referred to as *interior* dilation bands.

The exterior dilation band is not bounded by the closed 8-CC of ω_v pixels. However, interior dilation bands are bounded by the closed 8-CC of ω_v pixels and are possibly located in interior regions of the building footprint. In order to verify that interior dilation bands are part of the building footprint they are classified as either ω_{in} or ω_{out}, as discussed next.

12.3.3 Dilation Band Classification

Following the verification of a closed 8-CC of ω_v pixels as a building outline, the associated dilation bands are classified as being either ω_{in} or ω_{out}. This is done to further investigate the nature of interior regions of building outlines, in order to determine if these regions are part of the building footprint or not. An example of an interior region that is not part of a building footprint is a courtyard, which is bounded by building outlines, yet does not belong to the building footprint.

Similar to pixel classification, a function is used to determine the state of each dilation band. Different functions are used to classify exterior and interior dilation bands. Both functions are of the form $f(m_{in}, m_{out}) \in \{\omega_{in}, \omega_{out}\}$, where m_{in} and m_{out} represent the number of classified ALS points in each category mapping to pixels overlapping those of the dilation band. The terms m_{in} and m_{out} are computed by adding the n_{in} and n_{out} channels of pixels within the dilation band. Note that this procedure does not count the categories of dilation band pixels, but rather sums the actual values stored at pixels. Additionally, all pixels within dilation bands are guaranteed to have $n_v = 0$, as they would otherwise be active in the dilation process.

Exterior dilation bands are classified using the function $f_{ext}(m_{in}, m_{out}) \in \{\omega_{in}, \omega_{out}\}$, defined as follows:

$$f_{ext}(m_{in}, m_{out}) = \begin{cases} \omega_{in}, & \text{if } m_{in} > m_{out} \\ \omega_{out}, & \text{if } m_{out} \geq m_{in} \end{cases} \tag{12.2}$$

In general, a larger presence of ω_{in} or ω_{out} points in the dilation band leads to a corresponding classification. However, in the case $m_{in} = m_{out}$, exterior dilation bands are classified as ω_{out}. The rationale behind this is that most exterior dilation bands do in fact denote the outside of a building. Therefore, preference is given to ω_{out} for exterior dilation bands.

Interior dilation bands are classified using the function $f_{int}(m_{in}, m_{out}) \in \{\omega_{in}, \omega_{out}\}$, defined as follows:

$$f_{int}(m_{in}, m_{out}) = \begin{cases} \omega_{in}, & \text{if } n_{in} \geq n_{out} \\ \omega_{out}, & \text{if } n_{out} > n_{in} \end{cases} \tag{12.3}$$

Again, a larger presence of ω_{in} or ω_{out} points in the dilation band leads to a corresponding classification. However, in the case $m_{in} = m_{out}$, interior dilation bands are classified as ω_{in}. With similar reasoning as above, this is due to the fact that most interior dilation bands do in fact denote the inside of a building, and for this reason preference is given to ω_{in}.

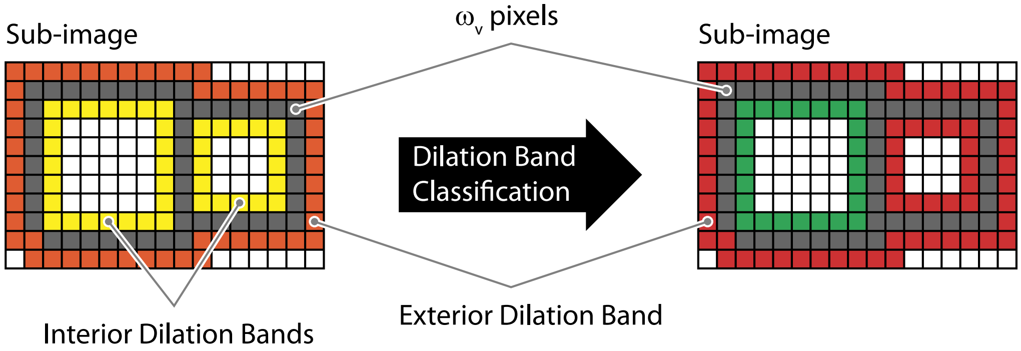


Figure 12.6: Dilation band classification.

Following classification of dilation bands, all pixels within dilation bands are considered to have the dilation band classification (Figure 12.6). This provides uniformity within dilation bands and resolves ambiguities, since dilation band pixels may originally belong to different categories. Together with a closed 8-CC of ω_v pixels, classified dilation bands provide information in a one pixel neighborhood around building outlines. In the following section this information is used to generate building footprints.

12.4 Building Footprint Extraction

Building outline extraction in separate sub-images, together with the analysis described in the previous section, provides verified building outlines along with classified dilation bands. Such information is used to generate building footprints, which not only describe the locations of building walls, but also explicitly define the interior regions covered by buildings. This section describes the processes of generating building footprints and merging sub-images back into full context. The final output of these processes is an image containing all detected building footprints, where a unique identifier has been assigned to each building footprint.

In order to generate building footprints from building outlines, interior dilation band classifications are propagated inward, thus providing building coverage in the horizontal image plane. While individual building footprints are generated in sub-images, results are subsequently transferred back into the full context of the study area. Separate building footprints are given unique building identifiers and ALS points mapping to the same building footprint can be grouped according to this identifier. Next, the determination of interior regions for building footprints is discussed.

12.4.1 Building Interiors

Sub-images containing a closed 8-CC of ω_v pixels together with classified dilation bands provide complete classification of pixels in a one pixel neighborhood around building outlines. For the purpose of generating building footprints, it is necessary to classify building outline interior regions so that the horizontal coverage of a building is explicitly defined. This is done by propagating information inward from interior dilation bands using 4-flood filling (Section 3.5). Interior dilation band pixels, which share the same classification, act as seed pixels and ω_v pixels are boundaries (Figure 12.7).

Interior dilation band pixels are located in regions that are bounded by ω_v pixels and, thus, propagation does not affect any pixels outside of the building outlines. Regardless of classification, exterior dilation band classifications are never propagated, since there are no clear boundaries for such propagation. Also, exterior dilation bands are technically not part of the building outlines or footprints.

Propagation is constrained to be within the building outlines defined by ω_v pixels and interior regions are filled with the classifications of the corresponding dilation bands. Interior regions filled with ω_{in} are part of building footprints, whereas interior regions filled with ω_{out} are not. Together ω_v and ω_{in} pixels provide the horizontal coverage of a building, i.e. the building footprint. Up to this point building footprint generation has been limited to sub-images. In order to group ALS points sampled on the same building it is necessary to transfer building footprints generated in the sub-image back into the full context of the study area, as described next.

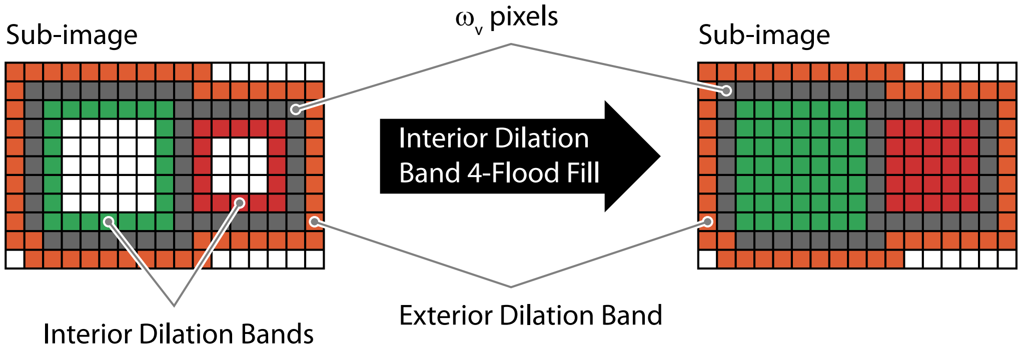


Figure 12.7: Pixels in interior dilation bands are used as seed pixels to 4-flood fill interior regions where ω_v pixels act as boundaries.

12.4.2 Building Masks

Each sub-image contains a separate building footprint and these footprints must be merged back into the full context of the study area in order to map ALS points to these footprints. This section describes the creation of a *building mask*, which is a binary image where *active* pixels denote building coverage. Recall that main image coordinates are stored during sub-image creation, which enables the transferal of pixel information from sub-images back to the main image. A building mask with the same pixel dimensions as the main statistical image (Section 12.2) is created from information in sub-images and these procedures are described below.

A building mask is created in two steps. In the first step, two intermediate binary images are created: an ω_{in}-mask and an ω_{out}-mask. Active pixels in the ω_{in}-mask correspond to transferred ω_v and ω_{in} pixels from all sub-images. Active pixels in the ω_{out}-mask correspond to ω_{out} pixels from all sub-images. Exterior band pixels are ignored in the creation of intermediate masks, the reason being that such pixels are by definition not part of building footprints since they are outside the building outlines. In the second step, the building mask is finalized by subtracting the ω_{out}-mask from the ω_{in}-mask on a per pixel basis.

Thin protrusions in the building mask can be removed using morphological opening (Section 3.4.4), if deemed necessary. Such protrusions often correspond to vertical structures, e.g. walls, attached to buildings and are typically not desirable in subsequent modeling steps. The final building mask contains building footprints that are guaranteed to be separated by at least one pixel. Before ALS points can be grouped by building, each building footprint in the building mask is given a unique identifier, as described next.

12.4.3 Building Identifiers

A building mask contains information on pixels that are covered by building footprints. In order to generate a unique identifier for each building footprint, 8-CCL is used to identify 8-CC of active pixels in a building mask, as shown in Figure 12.8. Each 8-CC corresponds to a building footprint and the pixels of each 8-CC are assigned a unique identifier. An image storing unique building identifiers is referred to as a *label image*. Pixels that are not part of a building footprint are given a null value.

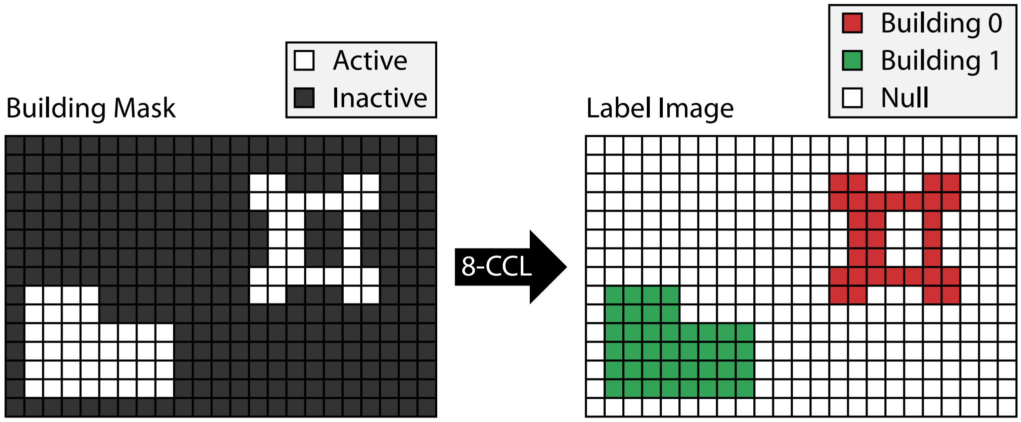


Figure 12.8: A label image is created by applying 8-CCL to a building mask in order to identify 8-CC of active pixels. The pixels of each building footprint are assigned a unique identifier, which makes it possible to group mapped ALS points by building footprint.

Label images are given the same global coordinate bounds as the main statistical image, enabling ALS points to be mapped to pixels. Hence, by mapping ALS points to the pixels of a label image it is possible to obtain a building identifier for each point, allowing points to be grouped accordingly.

In Part III the techniques described in this chapter are applied to an area of central Dublin, Ireland and the majority of the present buildings were detected.

Chapter 13

Voxelizing Laser Scan Data for Solid Modeling

A simple, yet effective, method for creating engineering models from point data is presented in this chapter. In particular, solid models used in building simulations are in focus (Chapter 5). Engineers use these models to simulate building response in various scenarios, such as earthquakes and tunnel excavations. The goal in this chapter is to create models of the type described in Chapter 5 from point data, most likely acquired with *Aerial Laser Scanning* (ALS) or *Terrestrial Laser Scanning* (TLS) data.

Current engineering practice is to study building response for individual walls. The main reason for doing so is to save time, both in modeling and simulation. Also, building walls are the main load-bearing components of masonry buildings, which are the most vulnerable to ground movements, thus it makes sense to concentrate simulation resources in this area. Further, solid models used by engineers in building wall simulations consist of connected cuboid primitives. Cuboids are arranged so that positions inside a cuboid correspond to being part of the building wall, while positions outside cuboids correspond to non-bearing parts of the building wall, such as windows and other openings. Small ornaments are typically ignored, since they do not add to the structural properties of a building.

Further, internal structures are often ignored in building wall simulations, which means that data acquired by laser scanning on the outside of buildings contains the necessary geometric information for further modeling. The challenge lies in finding a suitable conversion from raw point data into solid models. This chapter presents an automatic approach for this type of conversion.

Although automatically created models will most likely be inferior to hand-crafted models, they can be generated in a fraction of the time. Tools for automatically converting acquired data into engineering models enable simulations on a large scale, allowing simulations to be used in scenarios where this was previously unfeasible. Even though such models may not be accurate enough for detailed simulations, they offer new opportunities in large-scale simulations, where a significant number of buildings are present.

As was shown in Chapter 8, it is possible to acquire substantial portions of building walls with ALS in urban scenes. Thus, geometric data can be made available for a large number of buildings. Additionally, TLS can be used to acquire highly detailed point data for individual buildings and is complementary to ALS in the sense that detailed data can be collected for critical buildings within an ALS data set. Point data can be used to create models portraying structurally important elements, such as windows and other openings on walls.

Before presenting the technique for converting point sets into solid models it is necessary to introduce a volumetric data structure, which is done next. Thereafter, existing techniques using this data structure are reviewed and a point-based version of these techniques is described in detail. Finally, conversion from points into solid models is presented.

13.1 Voxel Grids

This section describes *voxel grids*, which are used extensively in coming sections. A voxel grid divides a bounded three-dimensional region into cells, which are referred to as *voxels*. Conceptually, this is very similar to the two-dimensional pixel grids used in digital images (Chapter 3). The main difference is that voxels are volumetric, whereas pixels are defined as areas in some image plane. Complementary to the description of voxel grids in this section are those in [Kau87, CK90].

Figure 13.1 shows a voxel grid. Since voxel grids are three-dimensional, six values are required

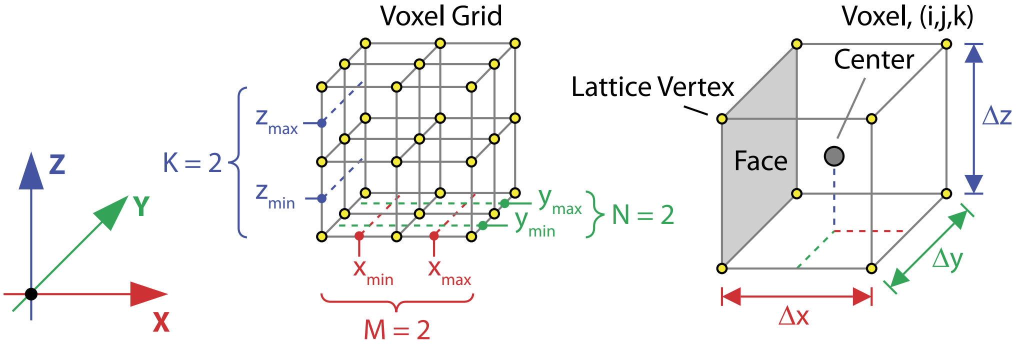


Figure 13.1: A voxel grid spans a volume in a three-dimensional space bounded by (x_{min}, x_{max}), (y_{min}, y_{max}), and (z_{min}, z_{max}). This volume is divided into non-overlapping cells, referred to as voxels, where (M, N, K) are the number of voxels in each direction. The dimensions of each voxel are $(\Delta x, \Delta y, \Delta z)$ and values are stored at voxel centers.

to define the bounds in some coordinate system:

$$(x_{min}, x_{max}),$$
$$(y_{min}, y_{max}),$$
$$(z_{min}, z_{max})$$

Grid dimensions are represented by an integer triplet, (M, N, K), where M is the number of voxels in the x-direction, N is the number of voxels in the y-direction, and K is the number of voxels in the z-direction. Voxel grid coordinates are given as (i, j, k), where $i \in [0, N-1]$, $j \in [0, M-1]$, and $k \in [0, K-1]$. The dimensions of individual voxels, $(\Delta x, \Delta y, \Delta z)$, are related to grid dimensions and bounds as follows:

$$\begin{aligned}\Delta x &= \frac{(x_{max} - x_{min})}{M-1}\\ \Delta y &= \frac{(y_{max} - y_{min})}{N-1}\\ \Delta z &= \frac{(z_{max} - z_{min})}{K-1}\end{aligned}$$

Each voxel has eight associated *lattice vertices*, which define the corners of the voxel (Figure 13.1). As such, lattice vertices define the geometry of each voxel. Each voxel has a rectangular shape with six faces. Note that lattice vertices are in some cases shared between voxels. In some cases up to four lattice vertices are shared, which corresponds to two voxels sharing a face. Voxel faces are used to define connectivity between voxels, as discussed below. Additionally, the volumes spanned by voxels' lattice vertices are used in visualization and modeling, as will be discussed in more detail in coming sections.

Similar to pixels, values are stored at voxel centers. In general, any type of value can be stored in voxels and this topic is discussed in more detail further on. In fact, many of the digital image processing techniques presented in Chapter 3 can be generalized to voxels, but only a relevant subset of such operations are presented in this section. Two important concepts relating to voxel grids, which are also relevant to solid modeling, are: (1) mapping of points to voxels; and (2) connectivity between voxels. Next, a mapping technique, similar to that in Section 3.2, for associating points with voxels is presented.

13.1.1 Mapping

Given the similarities between voxel grids and pixel grids, mapping a point (x, y, z) to a voxel is simply a matter of extending the mapping equations for digital images (Equation 3.2) to include a third dimension. A mapping from point (x, y, z) to grid coordinates (i, j, k) is defined in the following way:

$$i = \left[(M-1)\frac{(x - x_{min})}{(x_{max} - x_{min})} + 0.5\right] \tag{13.1}$$

$$j = \left[(N-1)\frac{(y - y_{min})}{(y_{max} - y_{min})} + 0.5\right] \tag{13.2}$$

$$k = \left[(K-1)\frac{(z - z_{min})}{(z_{max} - z_{min})} + 0.5\right] \tag{13.3}$$

where the operator $[x]$ rounds x to the nearest integer toward zero (Equation 3.3). Equations 13.1–13.3 yield valid grid coordinates (i, j, k) for points (x, y, z) in the ranges:

$$x \in \left[x_{min} - \frac{\Delta x}{2}, x_{max} + \frac{\Delta x}{2}\right] \quad \mapsto \quad i \in [0, M-1] \tag{13.4}$$

$$y \in \left[y_{min} - \frac{\Delta y}{2}, y_{max} + \frac{\Delta y}{2}\right] \quad \mapsto \quad j \in [0, N-1] \tag{13.5}$$

$$z \in \left[z_{min} - \frac{\Delta z}{2}, z_{max} + \frac{\Delta z}{2}\right] \quad \mapsto \quad k \in [0, K-1] \tag{13.6}$$

The mapping of points (x, y, z) outside the ranges in Equations 13.4–13.6 yield invalid grid coordinates. Applications of mapping points to voxels are discussed further on in this chapter. Next, connectivity between voxels is defined.

13.1.2 Voxel Connectivity

As for pixels, connectivity between voxels is a fundamental concept used in numerous processing techniques. The connectivity concepts used for pixels, presented in Section 3.3, including *adjacency*, *paths*, and *connected components* can easily be extended to voxels. However, at the root of these concepts is the the concept of *neighbors*, which is necessarily slightly different for voxels since they are volumetric and, thus, have neighbors in three directions, as opposed to two for pixels.

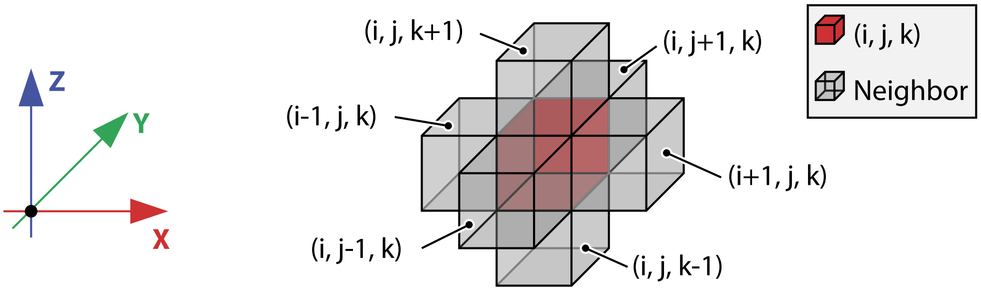


Figure 13.2: A voxel with grid coordinates (i, j, k) has up to six neighbors that share a face.

A voxel v at grid coordinates (i, j, k) has up to six neighbors that share a face (Figure 13.2) whose grid coordinates are given by:

$$\begin{array}{ll} (i-1, j, k), & (i+1, j, k), \\ (i, j-1, k), & (i, j+1, k), \\ (i, j, k-1), & (i, j, k+1) \end{array}$$

This set of voxel coordinates is referred to as the *6-neighborhood* of v and is denoted by $N_6(v)$. Other types of neighborhoods for voxels exist, but are not further discussed since they are not used in the solid modeling approach presented below. It is possible for coordinates in $N_6(v)$ to lie outside the voxel grid if (i, j, k) is on the edge of the voxel grid. The way outside neighbors are handled is application-specific, but failure to detect such cases leads to undefined results. The work in this chapter detects and ignores cases where voxel grid coordinates lie outside the voxel grid. Thus, not all voxels have six neighbors.

The 6-neighborhood defined above can be used to define adjacency, paths, and connected components of voxels as described in Section 3.3. In particular, *6-connected components* are useful in the solid modeling technique presented below. Having introduced voxel grids in some detail, the following section describes how to create and initialize a voxel grid from a provided set of three-dimensional points.

13.2 Voxelization

The term *voxelization* describes the conversion of an object, in any geometric representation, into a volumetric representation stored in a voxel grid [KPT99]. Voxelized representations of objects are used in numerous applications [ACY93], including, but not limited to, medical imaging [AMM92, SFF91] and terrain modeling [CO97]. The voxelization technique presented in this section is suitable for point data acquired by ALS or TLS. This voxelization technique is further used in the solid modeling approach presented in the following section for creating geometry from point data.

The basic idea in most voxelization algorithms is to test if voxels belong to the object or not and to classify them as *active* or *inactive*, respectively. This is accomplished by either examining if voxel centers lie inside the object, or by classifying all voxels that intersect the object surface as active [KS86]. More sophisticated algorithms generate smooth voxel representations and involve filtering of the volume [WK93, WK94], subdivision of the original object [DKW94], or calculation of the distance to the object surface from voxel centers [Jon96]. Notably, the majority of voxelization techniques are oriented toward a single type of object representation, e.g. polygon meshes or parametric surfaces.

Moreover, most existing voxelization techniques operate on surface representations of objects, where a significant part of the problem is to identify which voxels the surfaces pass through. Such methods are referred to as *surface-based voxelization* (Figure 13.3). In contrast, the voxelization approach presented in this section operates directly on point data and is referred to as *point-based voxelization*. Point-based voxelization avoids surface reconstruction, which is error-prone [REFS!], especially for sparse point sets. Although surface-based voxelization may guarantee some degree of connectivity between active voxels, this connectivity is not guaranteed to be accurate if errors are introduced in the surface reconstruction step. Further, when point sampling is dense compared to object details and voxel dimensions, points approximate object surfaces well.

As explained previously (Section 13.1), points can be mapped directly to voxels, allowing

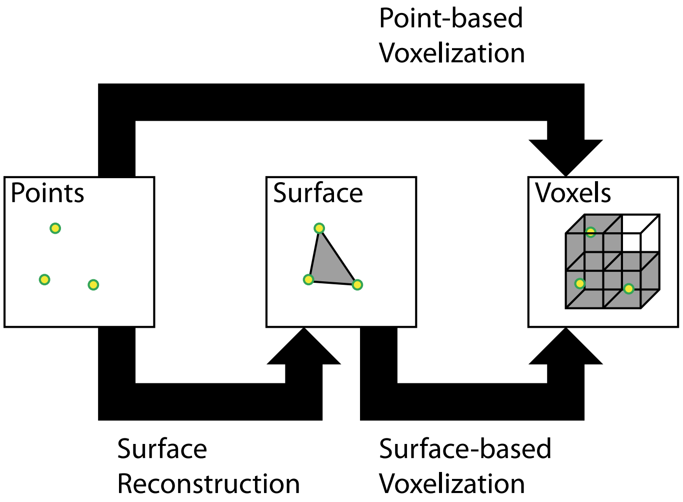


Figure 13.3: Point-based voxelization avoids surface reconstruction and operates directly on point data. Surface-based voxelization guarantees some degree of connectivity between active voxels, but surface reconstruction from points may introduce errors, which are then propagated to the voxelized model.

classification of voxels based on the presence of mapped points. As such, point-based voxelization is conceptually much simpler than existing surface-based voxelization algorithms. The mechanisms behind point-based voxelization are well-known, but they have not been applied to solid modeling of buildings from ALS or TLS data. Point-based voxelization is discussed in more detail below.

13.2.1 Point-based Voxelization

The input to the point-based voxelization algorithm presented in this section takes as input a set of points $P = \{\vec{p}_0, ..., \vec{p}_{n-1}\}$, where n is the number of points and each point, $\vec{p}_i$, is of the form $\vec{p}_i = (x_i, y_i, z_i)$. Points may be expressed in any coordinate system, but it will be shown further on that choosing a suitable coordinate system may improve results. No additional information, such as intensity or colour, is required for points. Since the only required input is a set of three-dimensional points, point-based voxelization is oblivious to the method used in point acquisition. Notably, ALS and TLS are suitable candidates for generating point data, but stereo photogrammetry can also be used. Point-based voxelization presented in this section consists of three steps: (1) creation of a voxel grid based on the provided set of points; (2) mapping of provided points to the created voxel grid; and (3) classification of voxels based on the previous point mapping. The first two steps are described below and the third step is described separately thereafter.

Creation of the voxel grid starts by examining the provided point set P. The bounds of the voxel grid are set according to the axis-aligned extents of P as:

$$\begin{aligned}
(x_{min}, x_{max}) &= (\min x_i, \max x_i) \\
(y_{min}, y_{max}) &= (\min y_i, \max y_i) \\
(z_{min}, z_{max}) &= (\min z_i, \max z_i)
\end{aligned}$$

These bounds guarantee that every point $\vec{p_i} \in P$ has a valid voxel mapping, as dictated by Equations 13.4–13.6. Voxel dimensions, $(\Delta x, \Delta y, \Delta z)$, are provided by the user and determine the voxel grid dimensions (M, N, K) as follows:

$$\begin{aligned}
M &= \left\lceil \frac{(x_{max} - x_{min})}{\Delta x} \right\rceil \\
N &= \left\lceil \frac{(y_{max} - y_{min})}{\Delta y} \right\rceil \\
K &= \left\lceil \frac{(z_{max} - z_{min})}{\Delta z} \right\rceil
\end{aligned}$$

Typically, $\Delta x = \Delta y = \Delta z$, which means that voxels are regularly sized cubes. It is also possible to let users specify the voxel grid dimensions (M, N, K) directly, but it has been found that providing voxel dimensions offers more intuitive control when it comes to choosing an appropriate level of detail for the voxel grid.

After creating a suitable voxel grid, points are mapped (Equations 13.1–13.3) to voxels and each voxel stores the number of points mapping to it. Voxels are classified based on this statistic, as described next.

13.2.2 Voxel Classification

Given a voxel grid where voxel values correspond to the number of mapping points, some classification mechanism is used to determine if voxels are active or inactive. A function of the form $f_v(n) \in \{\text{active}, \text{inactive}\}$ is used to classify voxels, where the argument n is the number of mapping points to a voxel. This function is applied to each voxel, classifying it into one of the two categories mentioned above. The function used in this chapter is simplistic and is defined as:

$$f_v(n) = \begin{cases} \text{active}, & \text{if } n \geq T_n \\ \text{inactive}, & \text{if } n = 0 \end{cases} \tag{13.7}$$

where T_n is a user-specified threshold value. Typically $T_n = 1$, which mean that voxels with at least one mapping point are classified as active. In this case only voxels to which no points map are classified as inactive. The classification function presented above is rather simple and it is possible to design more sophisticated classification functions. For instance, the distribution of points within voxels could be examined.

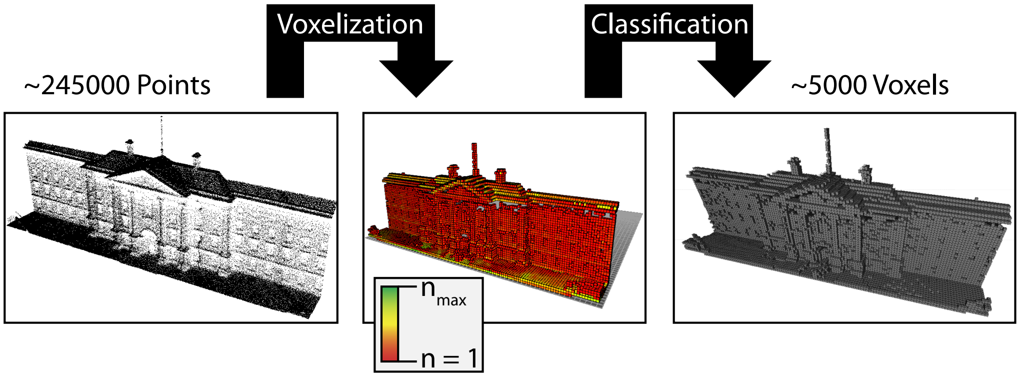


Figure 13.4: *Left*: $\sim$ 245000 ALS points acquired on the front building of Trinity College Dublin, Ireland, are provided as input to the voxelization process. *Middle*: $\Delta x = \Delta y = \Delta z = 0.25\ m$ was used to create the voxel grid, where voxels with $n \geq 1$ are visualized; n_{max} is the largest number of points mapping to a single voxel. *Right*: Using $T_n = 1$ the $\sim$ 5000 active voxels are shown in grey.

Figure 13.4 illustrates voxelization of an ALS point set acquired on the front building of Trinity College Dublin, Ireland. The point set contains $\sim$ 245000 points and was extracted manually from a larger data set using the tools presented in Chapter 11. Using $\Delta x = \Delta y = \Delta z = 0.25\ m$, a voxel grid was created and points were mapped to this grid. Voxels with $n > 0$ are shown and are colour-coded based on the number of points mapping to each voxel. Further, voxels are classified, resulting in $\sim$ 5000 active voxels. Since $T_n = 1$ was used in the classification, all voxels with $n \geq 1$ are classified as active.

This section describes the process of voxelization using a point-based approach. A voxel grid is created and points are mapped to this grid. Statistics are collected for each voxel, which is subsequently classified based on these statistics. The result is a voxel grid where voxels are classified as active or inactive, where active voxels are considered to be part of the object. The following section presents techniques for converting voxelized point sets into solid models, which can be further used in building simulations.

13.3 Solid Modeling

This section describes the process of transforming voxelized point sets into solid models suitable for engineering simulations. In particular, such models are used in engineering studies of building walls. Further, the solid modeling technique presented in this section is suitable for laser scanning data acquired with ALS or TLS (Chapter 2). For ALS data, the tools described in Chapter 11 can be used to extract points acquired on a single building wall. Terrestrial scans typically focus on a single building and the separation into individual walls is done manually. The only required input data to the solid modeling technique described in this section is a set of points acquired on a building wall.

Point-based voxelization is applied directly to the provided point data and the created voxel grid stores voxels classified as active or inactive, as discussed previously (Section 13.2). The presence of mapped points determines voxel classification. Thus, for point data acquired with laser scanning, active voxels correspond to the presence of surfaces within voxel volumes. Importantly, laser scanning does not acquire points on transparent materials, such as glass. Because of this windows and other openings are classified as being inactive.

Given that building models used by engineers consist of connected cubes, the geometry of active voxels, provided by lattice vertices (Section 13.1), is used to create a solid model of a building wall, as shown in Figure 13.5. Thus, load-bearing parts of the building are modeled, allowing simulations that test building stability to be performed. The exact format of such solid models is application specific and is, therefore, not discussed in this thesis.

More specifically, the solid models used in building wall simulations consist of *6-connected components* (6-CC) of cubes. However, point-based voxelization does not guarantee that all active voxels are 6-connected. Therefore, 6-CC of active cubes must be identified in the voxel grid. This is done using connected component labeling techniques similar to those used for

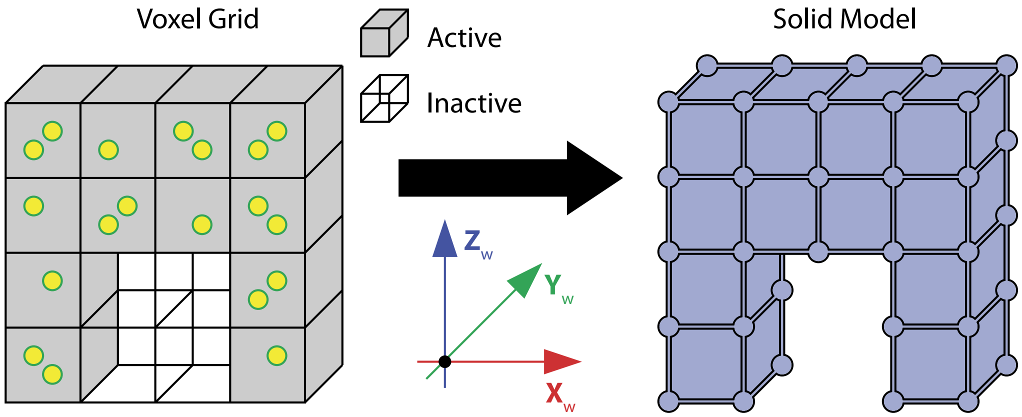


Figure 13.5: Active voxels provide the geometry necessary to create a solid model.

digital images (Section 3.6). The 6-CC containing the largest number of voxels is chosen to represent the building wall.

Two important issues relating to voxel connectivity are discussed below: (1) voxel dimensions; and (2) voxel aliasing. The first topic is discussed below, where general guidelines for choosing voxel dimensions are addressed. Thereafter, the second topic is addressed by proposing a simple technique to increase 6-connectivity between active voxels. Experiments and results are presented in Part III.

13.3.1 Voxel Dimensions

Voxel dimensions, $(\Delta x, \Delta y, \Delta z)$, are user-provided parameters to the point-based voxelization process. Typically, $\Delta x = \Delta y = \Delta z$, which means that voxel sides are equal. The choice of voxel dimensions is constrained by two factors: (1) the sizes of features that are being modeled; and (2) the sampling density of the provided point set. Large voxels are more likely to contain points, and, thus be active. However, if voxels are too large, small features cannot be accurately modeled. Moreover, if voxels are too small, connectivity between voxels suffers, especially if the point sampling density is low. Additionally, small voxels may lead to highly detailed models, with a negative impact on simulation times.

In general, voxel dimensions should be chosen to match the sizes of the modeled features. This leads to models that are well-suited to simulations. Also, connectivity issues can be resolved by manually activating voxels that were not automatically classified as active. Since the voxelization process is fast, especially when compared to manually creating solid models, it is possible to run the voxelization step multiple times in order to optimize the voxel dimensions. This does make the approach less automated, but offers the possibility to achieve good results. Next, connectivity issues relating to the axis-aligned nature of voxels is discussed.

13.3.2 Voxel Aliasing

Jagged edges, often referred to in computer graphics as *aliasing*, is a sampling problem that occurs when geometry is approximated on a regular grid (e.g. voxel grid or digital image). In general, lines that are diagonal to the grid axes cannot be represented without introducing jaggedness. This is illustrated in Figure 13.6. Axis-aligned lines, however, can be represented without aliasing.

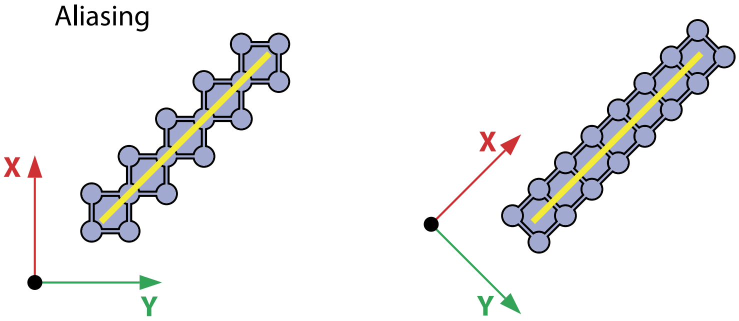


Figure 13.6: *Left*: A straight wall (yellow line) is to be voxelized. Here the wall is diagonal with respect to the coordinate system, which leads to voxels whose faces are not connected. *Right*: By aligning the coordinate system to the wall orientation voxel faces are connected.

Aliasing on voxel grids results in fewer 6-connected active voxels, which in turn leads to quality degradation in the created solid models. Since the basic geometry of a building wall is a vertical plane, aliasing can be remedied by expressing points sampled on the wall in a coordinate system where one of the horizontal axes is parallel to the wall (Figure 13.6). As such, many of the features on the wall become axis-aligned, as explained in Chapter 11, which reduces artefacts caused by aliasing.

There are different ways of identifying the wall plane. From a computational point of view, a line fitting based on the horizontal coordinates of the provided points would find a good approximation to the wall plane. The fitted line together with the vertical axis can be used to define the third axis needed to express a coordinate system suitable for further modeling. The tools presented in Chapter 11 allow users to manually specify a line and a thickness for the purpose of extracting building wall points. The line drawn by users can be used instead of a fitted line, following the same reasoning as above.

In summary, this chapter has described a point-based voxelization technique for creating solid models of building walls. Solid models are created directly from point data. In particular, the semi-automatic extraction tool presented in Chapter 11 is useful for extracting point data on building walls. Examples of solid models created with the technique presented in this chapter are shown in Part III. Both ALS and TLS point data were used to create solid models. Standard simulations were tested on the created models to confirm the feasibility of the approach.

Part III

Results and Discussion

Chapter 14

Overview

In Part I the goal of this thesis was formally stated, after first providing the necessary background. The goal is to create solid models of buildings on a city-scale. In order to achieve this goal a workflow involving a set of tools was proposed in the problem statement (Chapter 6). Part II presented methods and techniques for creating the required tools and in this part (Part III) these tools are evaluated by demonstrating practical results. This introductory chapter provides an overview of this part and gives a brief introduction to the results presented therein.

Figure 14.1 shows the workflow proposed for achieving the goal of this thesis. The tools involved in the workflow are shown as colour-coded boxes, where color is representative of the four categories: (1) *acquisition*; (2) *visualization*; (3) *building extraction*; and (4) *solid modeling.* The chapters relevant to each category are given at the bottom of the figure. Not shown in the workflow are the ALS data processing techniques presented in Chapter 9, which enable coupling of aircraft position to point data and allow very large data sets to be efficiently processed. These techniques are used throughout the workflow to provide scalability and efficiency and are important components of several tools. Further, solid lines show completed connections between tools, while dashed lines show connections between tools that are incomplete, i.e. have not been implemented. Notably, the connection between automatic building extraction and solid modeling is incomplete, for reasons discussed in coming chapters. Results in each of the five categories are presented in separate chapters in this part. First, however, results for each category are briefly outlined below.

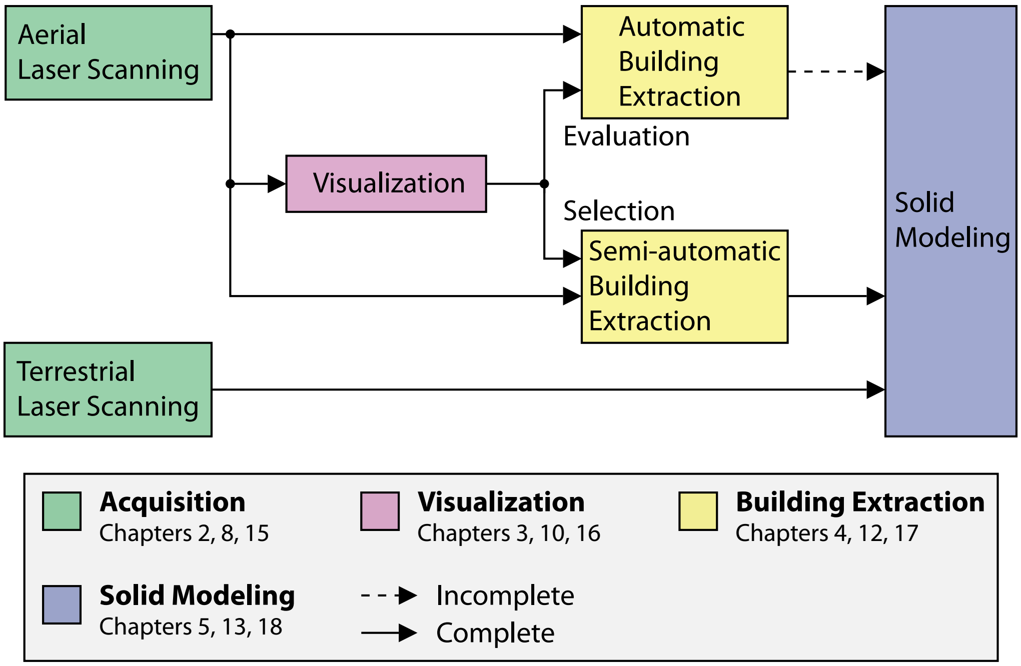


Figure 14.1: The tools involved in the workflow proposed in this thesis are represented as boxes. Box colour is based on the categories shown at the bottom, where the chapters relevant to each category are also given. Solid lines represent completed connections for which results will be demonstrated, whereas dashed lines represent incomplete connections.

Acquisition is the task of gathering geometric data for buildings and this thesis focuses on the use of laser scanning, as described in Chapter 2, for this purpose. Both *Aerial Laser Scanning* (ALS) and *Terrestrial Laser Scanning* (TLS) are used. Two data sets were acquired specifically for evaluation of the proposed workflow. The first data set was acquired using ALS over a study area located in central Dublin, Ireland in 2007. The study area is $\sim 1\ km^2$ and contains ~ 200 buildings. Approximately 225 million points were acquired within the study area, giving a point density of $\sim 225\ points/m^2$. The flight path was designed using the algorithms presented in Chapter 8 and point data was acquired for a majority of building walls. Thus, the Dublin ALS data set differs from traditional ALS data sets in that it is fully three-dimensional, as opposed to being focused on elevation information. In terms of acquisition, ALS and TLS are complementary, since TLS can be used to acquire detailed point data for a single building, while ALS point data is less detailed but can be acquired for large urban areas. Details regarding the acquired ALS data set are presented in Chapter 15. The second data set was acquired using TLS in a standard way and covers a single building within the study area. Solid models were created directly from this data in order to compare results from using ALS and TLS data sets.

To support building extraction from ALS data, both automatic and semi-automatic, digital images (Chapter 3) were used. The techniques presented in Chapter 10 were used to create occlusion images, in which the scanned structures are clearly visible. Use of the ALS data processing techniques presented in Chapter 9 enabled consideration of the aircraft position with acquired ALS points. Occlusion images were used in both automatic and semi-automatic building extraction, as discussed below. Visualization results of the ALS study area are presented and discussed in Chapter 16.

As discussed in Chapter 4, extracting points on buildings is a fundamental task in ALS data processing. The proposed workflow uses two different tools for this purpose: (1) *semi-automatic building extraction*; and (2) *automatic building extraction.* A semi-automatic building extraction tool was presented in Chapter 11. Occlusion images were used to visualize the ALS data set and regions were interactively selected for extraction. Semi-automatic building extraction was used to extract building walls in the ALS data set, allowing solid modeling to be applied to ALS points.

An automatic building extraction method was presented in Chapter 12. This method was tested on the ALS study area and a clear majority of buildings were detected. High-resolution occlusion images were used both to analyze results and to define a ground truth used in the evaluation of successfully detected buildings. Ideally, automatically extracted buildings could be used in the subsequent solid modeling step. However, the connection between these two steps has not been finalized and solid modeling was tested on semi-automatically extracted building points, as discussed below.

Solid models were introduced in Chapter 5. In particular, the type of solid models used in building simulations was described. Thereafter, a solid modeling approach for converting point data directly into solid models was proposed in Chapter 13. This tool takes as input a point set and produces a solid model suitable for use in building simulations. In Chapter 18 solid modeling results are presented for semi-automatically extracted building walls in the ALS data set and for TLS points. It is shown that the created solid models are compatible with existing computational modeling software and produce feasible results.

Results outlined above are presented in more detail in the following chapters. After results have been presented separately for each workflow category, discussions are provided in the last chapter of this part.

Chapter 15

Acquisition

This chapter presents acquisition results using *Aerial Laser Scanning* (ALS). An ALS data set was acquired for a study area located in central Dublin, Ireland in 2007. The methods for acquiring three-dimensional ALS data of urban regions, presented in Chapter 8, were used to design the flight path for this mission. The outcome is presented in Section 15.1, where details of the acquired ALS data set are provided and hypotheses regarding linear resolution and three-dimensional point sampling proposed in Chapter 8 are tested. The acquired ALS data set is used extensively in subsequent chapters of this part as input to later stages of the workflow.

15.1 ALS Case Study: Dublin, Ireland

In 2007 an ALS mission was flown over Dublin, Ireland and an area of $\sim 1\ km^2$ was chosen as the study area. The flight path of this mission was planned using the guidelines described in Chapter 8. Two principal hypotheses are tested in this case study: (1) linear resolution can be predicted from scanner specifications on horizontal and vertical surfaces; and (2) adherence to the flight planning discussed in Chapter 8 leads to scans with sufficient vertical resolution that building wall details are clearly detectable. Additionally, towards the end of this case study, missing echo distribution is examined using the techniques presented in Chapter 9. Missing echoes are used in several other chapters, most notably in automatic building extraction (Chapter 17). First, however, some general information regarding the acquired data is presented.

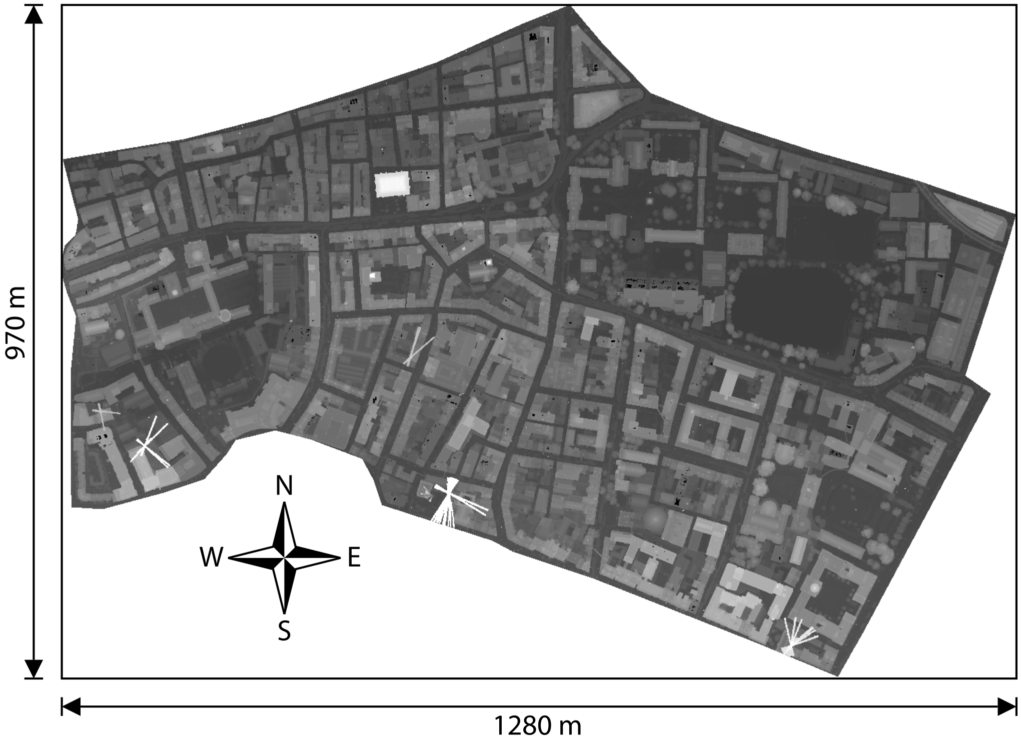


Figure 15.1: Elevation image of the $\sim 1\ km^2$ study area for ALS acquisition in central Dublin, Ireland. Pixel dimensions in world coordinates are: $\Delta x_w = \Delta y_w = 1.0\ m$.

Echo	**Count**	**Percentage**
1^{st}	217,497,975	96.33%
2^{nd}	7,902,595	3.5%
3^{rd}	383,840	0.16%
4^{th}	4,028	0.001784%
Total:	**225,788,438**	**100%**

Table 15.1: Distribution of echoes for acquired ALS points.

In Figure 15.1 an elevation image of the study area is shown. A total of $\sim$ 225 million points were acquired for the study area, which contains 205 buildings. Buildings in the study area are often adjoined, sharing an inner wall with another building. There are relatively few stand-alone buildings and most buildings are part of a larger building block. Building configurations will be discussed in further detail in Chapter 17 where building extraction is tested.

ALS was carried out by contractors using the FLI-MAP 2 system. This system has a scan rate of 150 scan lines per second and a pulse rate of $150,000$ pulses per second. Thus, $1,000$ pulses were emitted for each scan line and a total of $370,154$ scan lines were acquired. The system operated at a scan angle of 60 degrees, with an angular spacing of 60/1000 degrees between pulses, which is roughly equal to one milliradian. Acquired points were provided in a global coordinate system, relating to the use of a *Global Navigation Satellite System* (GNSS) to determine the aircraft position during scanning. The quoted accuracy of the FLI-MAP 2 system is 8 cm in the horizontal plane and 5 cm in the vertical direction, including both laser range and navigational errors.

Further, the FLI-MAP 2 system is capable of recording up to four echoes for each emitted pulse. The distribution of echoes is shown in Table 15.1. A vast majority of points are first echoes and although secondary echoes are present they constitute only a small portion of the points. This is expected for an urban study area, since multiple echoes are known to occur most frequently in vegetated areas. The study area does contain some vegetation, but an overwhelming majority of surfaces in the study area are solid, in the form of streets and buildings.

In addition to point data, the FLI-MAP 2 system also provides spectral data in two different forms: (1) *intensity*; and (2) *colour*. An intensity value is provided for each point and is proportional to the amount of reflected pulse energy. Colour information is provided by cameras acquiring images during the flyover and is transferred to scan points. Examples of images created from spectral data, both intensity and colour, are shown in Chapter 16, where they are compared with other types of images. Outside of visualization, spectral information was not used in this thesis, which focuses on geometric data in the form of scanned points. Before testing the hypotheses regarding linear resolution prediction and flight strip overlap, the flight path used to acquire data for the study area is presented.

15.1.1 Flight Path

As mentioned, the guidelines presented in Chapter 8 were applied in the design of the flight path used to acquire ALS data for the study area. Three major parameters were considered: (1) low aircraft altitude and wide scan angle; (2) flight track orientation diagonal to the major street grid; and (3) overlapping flight strips. The minimum allowed flying altitude over urban areas is strictly regulated, but was chosen to be as low as possible and varied between $\sim 380 - 480\ m$, with an average value of $\sim 400\ m$. Further, the widest possible scan angle for the ALS system (60 degrees) was used. Flight track orientation relative to the major street grid is discussed below and the effects of flight strip overlap are shown in the following section.

In Figure 15.2 an illustration of the flight path overlaid on the major street grid is shown. Since Dublin does not have a perfectly regular street grid, the strictly ideal case of diagonal flight paths was not possible. However, like many other urban centers, there is a tendency towards east-west, north-south street orientation, and the dominant directions of the flight tracks were chosen to be north-east, north-west, south-east and south-west.

In total 44 flight strips were acquired and $2,823$ flight path points were recorded, providing instantaneous aircraft positions over time. The flight path points provided were used to create a hierarchical flight path structure presented in Chapter 9. The total length of the flight path was $\sim 43\ km$, counting only those parts of the flight path for which scan data was being acquired. Flight strips were acquired with substantial overlap and the effects of this with respect to building wall acquisition are discussed next.

15.1.2 Flight Strip Overlap

The results of flight strip overlap for the acquisition of point data on building walls is most clearly shown by comparison against the scanner output from a single flight strip (akin to traditional

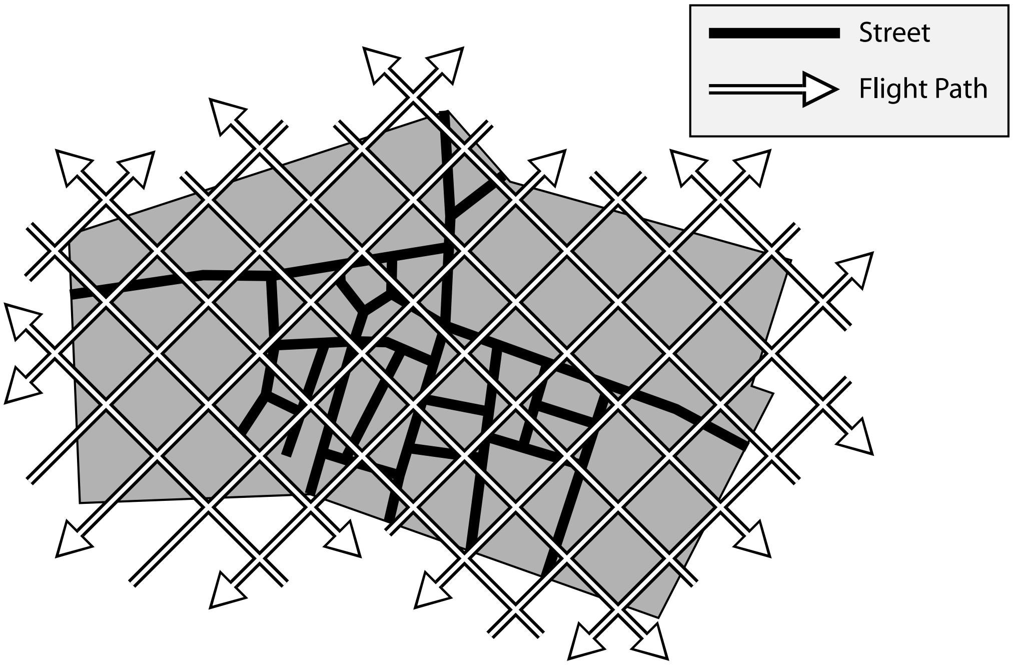
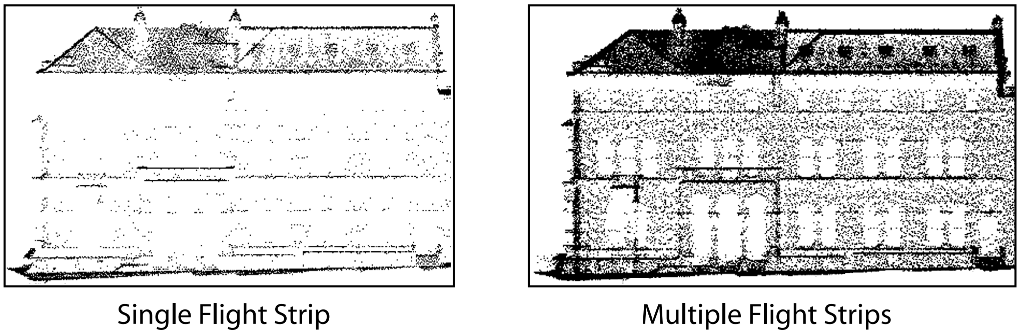


Figure 15.2: Designed ALS flight path overlaid on the major street grid.

minimally overlapping flight strip acquisition) for one isolated building wall. In Figure 15.3 point sampling on a building wall is shown both from a single flight strip (left) and for multiple flight strips (right). Although the overall configuration of the building wall are visible on the left, structural details are not. In comparison, with multiple flight strip overlap, individual windows and door openings are clearly visible, although shadowing effects from the eaves still occur at the top of the wall just below the roof.



Single Flight Strip

Multiple Flight Strips

Figure 15.3: *Left*: ALS points from a single flight strip on a building wall. *Right*: ALS points from multiple flight strips on a building wall.

The presence of ALS points on building walls is a feature that several other tools in the proposed workflow exploit and other examples of ALS acquisition on building walls will be shown in coming chapters in this part. Next, hypotheses regarding linear resolution predictions made in Chapter 8 are tested.

15.1.3 Linear Resolution

To confirm the predictions of linear resolution made in Chapter 8, points from a representative scan line were extracted using the techniques presented in Chapter 9. For simplicity only points on one side of nadir were chosen and only last echoes were used. Given that extracted points were acquired in the same scan line, they were also acquired from the same aircraft position, denoted by $\vec{p}_s = (x_s, y_s, z_s)$. Extracted point coordinates are denoted by $\vec{p}_i = (x_i, y_i, z_i)$ and a horizontal offset angle, θ_i, was computed for each point using the following formula:

$$\begin{aligned}(\Delta x, \Delta y, \Delta z) &= (x_s - x_i, y_s - y_i, z_s - z_i) \\ \theta_i &= \tan^{-1}\left(\frac{\sqrt{\Delta x^2 + \Delta y^2}}{\Delta z}\right)\end{aligned}$$

Extracted points were sorted with respect to increasing θ_i, which ordered them in a way that corresponds to being consecutive samples in a single scan line.

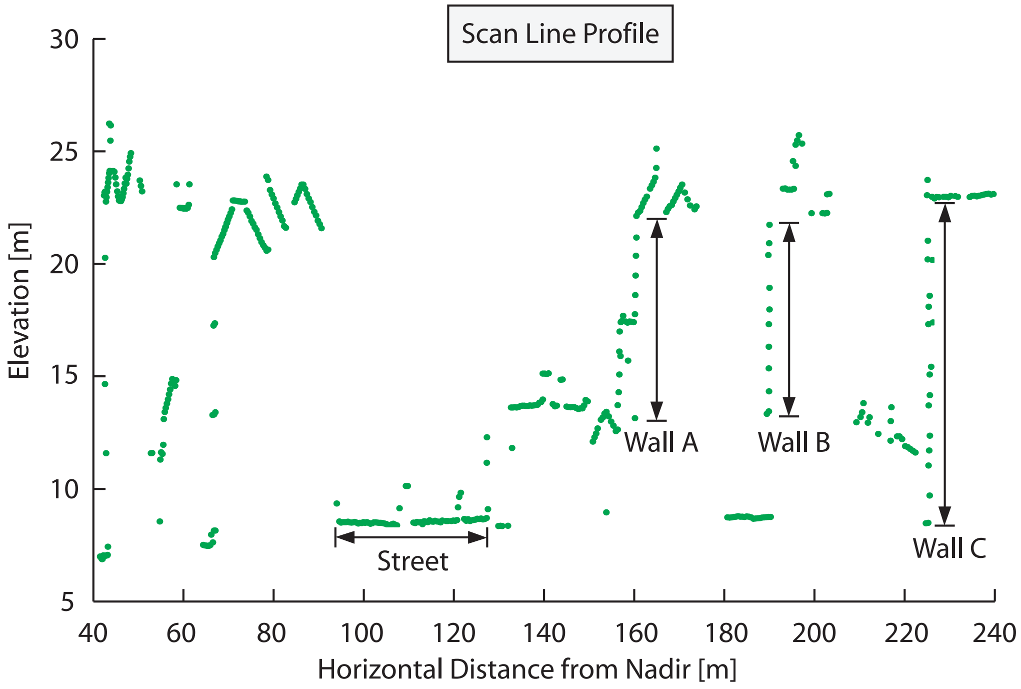


Figure 15.4: Point elevation plotted against horizontal distance to nadir. Points sampled on a street and three walls are marked in the diagram.

By computing the horizontal distance d_i to nadir (i.e. $(x_s, y_s, 0)$):

$$\begin{aligned}(\Delta x, \Delta y) &= (x_s - x_i, y_s - y_i) \\ d_i &= \sqrt{\Delta x^2 + \Delta y^2}\end{aligned}$$

and plotting (d_i, z_i) a cross-sectional view (scan line profile), can be generated as shown in Figure 15.4. Note that the smallest horizontal distance in Figure 15.4 is 40 m. It was desirable when extracting points to avoid working with points close to nadir in order to avoid infinite vertical resolution prediction. However, those issues do not exist as the horizontal distances increase, and the extracted points, therefore, extend to the maximum horizontal offset. This plot clearly shows the high horizontal resolution on the street combined with the lower vertical resolution on walls. Note that no vertical points are present for the walls of the triangular roof shapes on the left (closer to nadir), whereas both roofs and walls are sampled on the right (further away from nadir). This illustrates the effect of dead zones, where points are sampled on building walls only in scan line flanks.

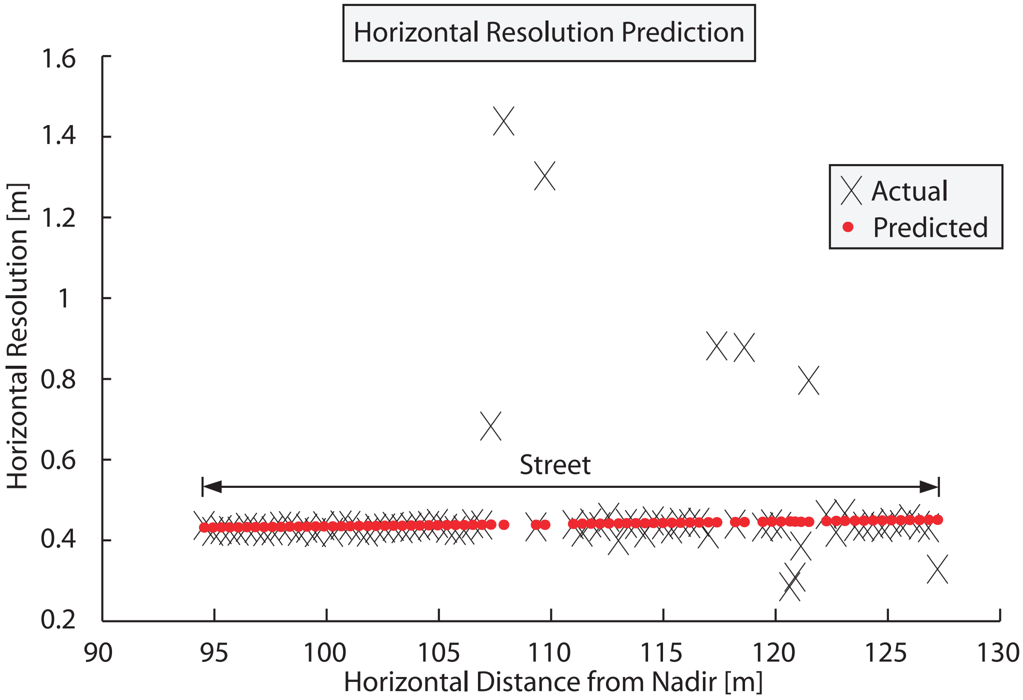


Figure 15.5: Horizontal resolution prediction.

Recall from Equation 8.4 and Equation 8.7 that expected horizontal and vertical linear resolution can be predicted. Horizontal resolution prediction (Equation 8.4) assumes that two consecutive point samples have the same vertical (z) coordinate, i.e. are sampled on the same horizontal surface. For points on the ground such as those marked as *Street* in Figure 15.4, this is a reasonable approximation. For points on walls, diagonal elements such as roofs, or on trees, cars or other objects, greater variance is expected, but there are enough points along ground surfaces to make a comparative plot meaningful. For a single scan line, the predicted horizontal resolution comes directly from Equation 8.4, while the measured horizontal resolution is taken to be the horizontal distance to the next consecutive point. Figure 15.5 shows the result of this plot for the points marked as *Street* in Figure 15.4. The data in Figure 15.5, with the exception of some outliers, show that the agreement for the prediction with horizontal surfaces is extremely good.

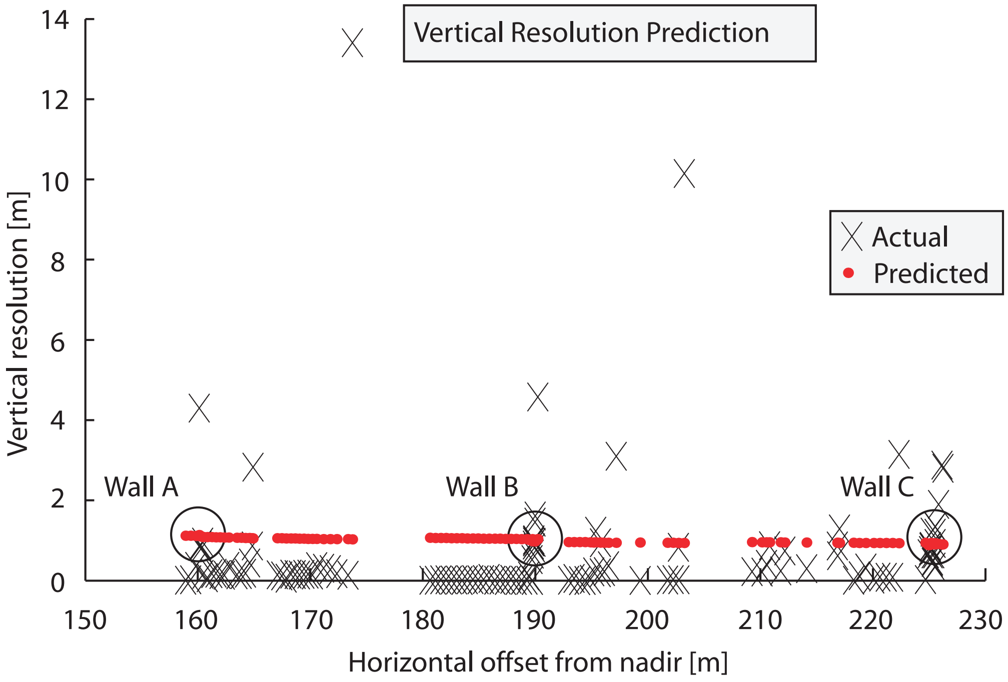


Figure 15.6: Vertical resolution prediction.

Similarly, Figure 15.6 compares the actual vertical resolution against that predicted by Equation 8.7. Here, actual vertical resolution was computed as the vertical distance to the next consecutive point. Figure 15.6 shows the plotted vertical resolution for the sequence of apparent wall/roof elements at the right-hand end of the scan line profile shown in Figure 15.4. This plot is harder to interpret, as wall elements share essentially the same horizontal distance to nadir. However, horizontal surfaces are flat and, thus, the points sampled on them have nearly zero difference in the vertical direction. More importantly, for the vertical surfaces (i.e. the building walls) the actual resolutions cluster close to the predicted values, especially near 160 m, 190 m

and 225 m from nadir (marked with circles in Figure 15.6), which agrees with the presence of walls noted in the scan line height profile (Figure 15.4). An important observation is that even for a set of points all with nearly identical horizontal offsets it is unlikely that the distances between these will match perfectly the predicted resolution. Most likely the highest point in such a set is actually sampled on an adjoining non-vertical surface (i.e. roof); thereby appearing to belong to the wall, when in fact this is not the case. Similarly, it is unlikely for the lowest point on a wall to be at the intersection of the wall and the horizontal plane. This may lead to situations where a vertical distance is computed between a point on a horizontal surface and a point a short distance up a wall, resulting in smaller measured vertical distance than predicted. In practice, this means that some outliers are to be expected, even when samples appear to be from the same vertical surface.

Moreover, in this study the vertical distance between consecutive points is described as an absolute value. Negative vertical distances would arise when consecutive points are sampled on separate surfaces with different elevations. This occurs frequently because of building shadowing, where building walls not facing nadir are not sampled. These situations arise where the first of two consecutive points is sampled on the building roof, and the next one is sampled on a different building or street.

15.1.4 Missing Echoes

In Chapter 9 methods for inserting missing echoes in empty pulses were presented. These methods were applied to the Dublin data set and results are demonstrated in this section. The results presented in this section focus on visualizing the distribution of missing echoes and verifying that missing echoes are in fact inserted at plausible positions.

In total, $\sim$ 13.8 million missing echoes were inserted in the scan lines of the Dublin ALS data set, which corresponds to 6.1% of the scanned points. Figure 15.7 shows an image where pixel intensity is proportional to the number of mapped missing echoes. Missing echoes are fairly evenly distributed throughout the study area, with the exception of grass-covered areas, appearing as large dark regions. Also, a large number of missing echoes are inserted on building roofs, which are often constructed from metallic materials. As explained in Chapter 2, metallic materials are difficult to acquire with laser scanning because of their highly reflective nature. Also, in some cases pools of water form on top of roofs and the refractive properties of water

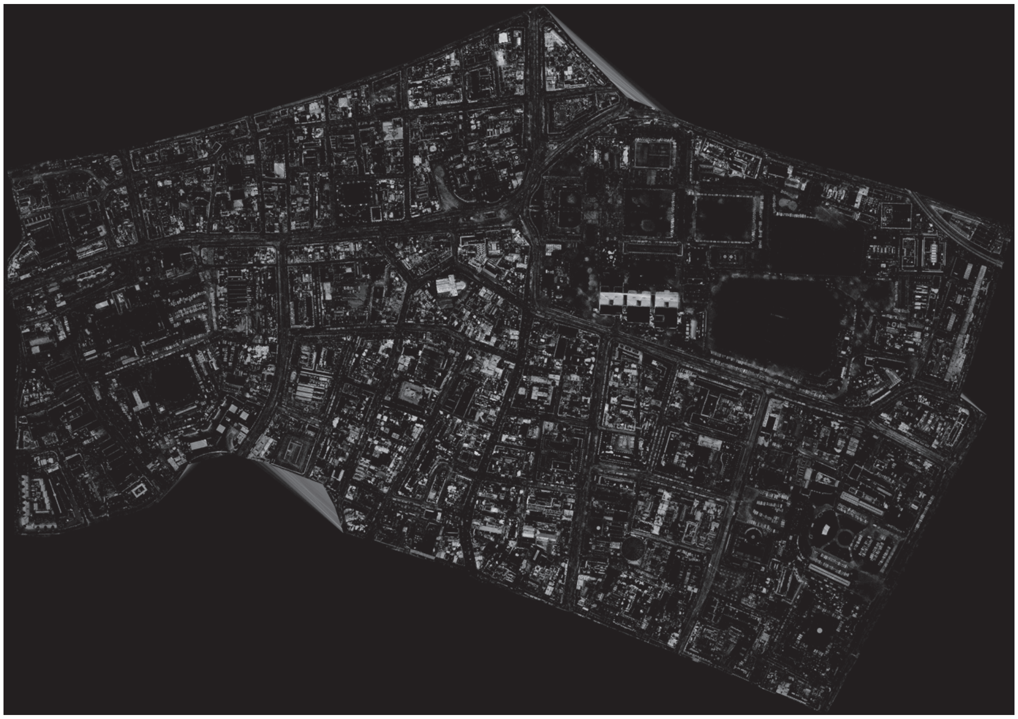
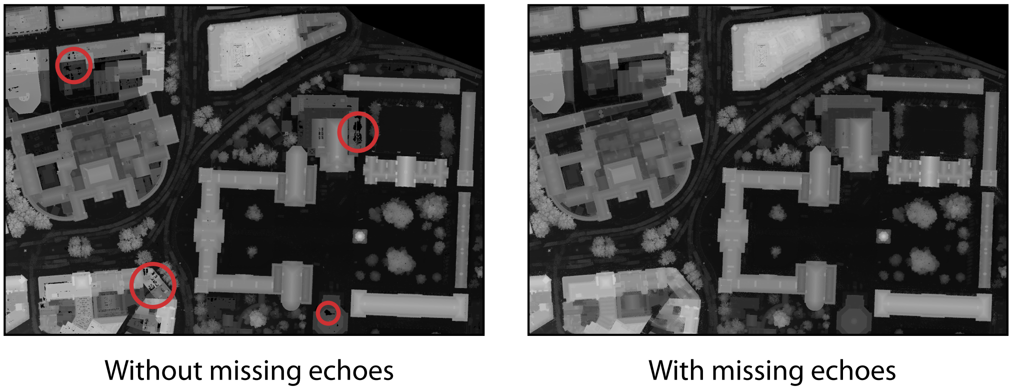


Figure 15.7: Pixel intensity is proportional to the number of mapped missing echoes. Pixel dimensions in global coordinates are: $\Delta x_w = \Delta y_w = 1.0\ m$

are known to interfere with laser pulse reflection. Further, the wide gray bands at the bottom and top of the image are caused by the fact that the study area was extracted from a larger data set. For some scan lines large portions of points were removed in the extraction process. The removed points were then re-inserted as missing echoes.


Without missing echoes

With missing echoes

Figure 15.8: *Left*: Pixels in an elevation image to which no points map are black and dark regions correspond to areas where no points are available (red circles). *Right*: The insertion of missing echoes increases the availability of point data, thus reducing dark regions caused by an absence of information. In both elevation images pixel dimensions in global coordinates are: $\Delta x_w = \Delta y_w = 0.2\ m$.

Figure 15.8 shows two elevation images. In both elevation images pixel dimensions in global coordinates are: $\Delta x_w = \Delta y_w = 0.2\ m$. The elevation image on the left was created without inserting missing echoes. Dark regions on roofs (red circles) are caused by an absence of point sampling on these surfaces. For the reasons stated above, roofs are in some cases difficult to acquire with laser scanning. On the right, missing echoes have been inserted and it is clear that gaps are filled in. Additionally, inserted echoes conform well to existing echoes, which is indicated by the fact the image created using missing echoes is very smooth and without visible discontinuities caused by misplaced echoes.

In summary, this chapter has described an ALS data set acquired for a fairly large study area located in central Dublin. This data set was used extensively in the evaluation of the proposed workflow. In the following chapter results of applying the imaging technique presented in Chapter 10 are demonstrated and compared to other imaging techniques.

Chapter 16

Visualization

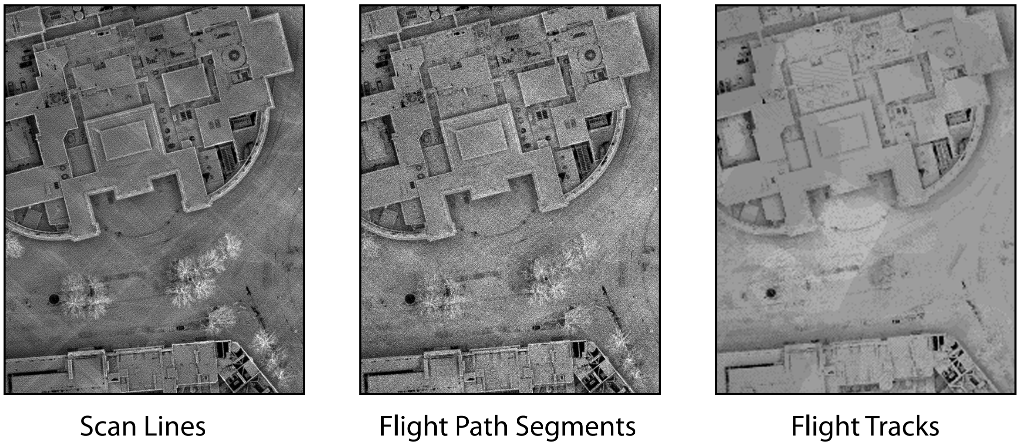
In the previous chapter the acquisition of an *Aerial Laser Scanning* (ALS) data set was described and the data set was presented. This chapter presents results of visualizing that data set using occlusion images, presented in Chapter 10. The goal of this visualization is to show the scanned structures, in particular buildings and building features. Additionally, visualizations should provide information about the point sampling on the scanned structures. The visualization results presented in this chapter, in the form of digital images, are further used in the extraction of buildings from ALS data, both automatic and semi-automatic, as will be demonstrated in Chapter 17.

Recall that occlusion images are based on treating ALS points as visibility samples and visibility is accumulated at pixels by mapping points. Visibility is accumulated under the constraint that a unique aircraft position may only contribute a single unit of visibility to a pixel. The large amount of flight strip overlap in the Dublin data set ensures that points mapping to the same pixels are acquired from several different aircraft positions. This is a crucial requirement for useful occlusion patterns to occur.

The aircraft positions from which ALS points were acquired are interpolated using the flight path hierarchy described in Chapter 9. Different levels of this hierarchy can be treated as unique aircraft positions and the effects of this are tested in Section 16.1. Thereafter, the effects of varying pixel dimensions are tested in Section 16.2, followed by an evaluation of the effects of inserting missing echoes in Section 16.3. Finally, occlusion images are qualitatively compared against existing imaging techniques in Section 16.4.

16.1 Flight Path Sampling

As explained in Chapter 10, different levels of the flight path hierarchy can be treated as unique aircraft positions during the creation of occlusion images. At the finest level, interpolated aircraft positions for every scan line are treated as unique aircraft positions. Moving up the hierarchy, the next level is to treat points acquired in each flight path segment, representing one second of scanning, as being sampled from unique aircraft positions. Similarly, entire flight tracks can be treated as unique aircraft positions, where the scanning time represented by each flight track may vary.



Scan Lines

Flight Path Segments

Flight Tracks

Figure 16.1: Three different occlusion images of the same area created with different flight path sampling. Pixel dimensions in global coordinates are: $\Delta x_w = \Delta y_w = 0.2\ m$. *Left*: Scan line sampling. *Middle*: Flight path segment sampling. *Right*: Flight track sampling.

Figure 16.1 shows three different occlusion images of the same area with pixel dimensions in global coordinates being $\Delta x_w = \Delta y_w = 0.2\ m$. In the left image, every scan line was treated as being acquired from a unique aircraft position. Small features, such as parked cars (top left) and obstacles on roofs, are clearly visible. Additionally, vegetation in the form of trees appear as bright shapes in cloudy patterns. Bright diagonal streaks, perpendicular to the flight path (not shown), are caused by small variations in aircraft orientation over time. More specifically, small variations in pitch between consecutive scan lines may cause the same area to be scanned in multiple scan lines, thereby drastically increasing the visibility for that area, resulting in the previously mentioned streaks.

The middle image shows an occlusion image where every flight path segment was treated as a unique aircraft position. The amount of detail visible in this image is similar to the one on the left. However, since unique aircraft positions are considered to be longer time periods, small variations in aircraft pitch have no visible effect.

In the right image, every flight track was treated as a unique aircraft position. As higher levels of the flight path hierarchy are treated as unique aircraft positions, the number of unique positions decreases. The effects of this are visible in the images in terms of progressively decreasing contrast. Small features on roofs are not clearly visible in this image. Although the decrease in contrast effectively removes vegetation from the image, for the purpose of examining building features, less detail is provided.

In summary, the choice of flight path sampling affects the balance between contrast, detail and smoothness. Sampling lower levels of the flight path hierarchy leads to increased contrast and visibility of small features. However, at the scan line level artefacts in the form of bright streaks perpendicular to flight tracks are introduced if there are variations in aircraft pitch over time. On the other hand, sampling higher levels reduces contrast and visibility of small details. However, small variations in aircraft orientation do not affect the final images. The following section examines the effects of varying pixel dimensions.

16.2 Pixel Dimensions

Besides flight path sampling, pixel dimensions are an important factor for occlusion images. In order to evaluate the effects of pixel dimensions three images of the same area were created. The flight path was sampled at flight path segment level for all images, while the pixel dimensions were varied.

Figure 16.2 shows three occlusion images of the same area with varying pixel dimensions, increasing from left to right. On the left is a high-resolution occlusion image with pixel dimensions $\Delta x_w = \Delta y_w = 0.2\ m$ in global coordinates. Note that some graininess can be seen in this image. This artefact is caused by non-uniform visibility in neighboring pixels and suggests that at this fine level point sampling is not uniform.

The middle image has pixel dimensions $\Delta x_w = \Delta y_w = 0.5\ m$ in global coordinates. Using

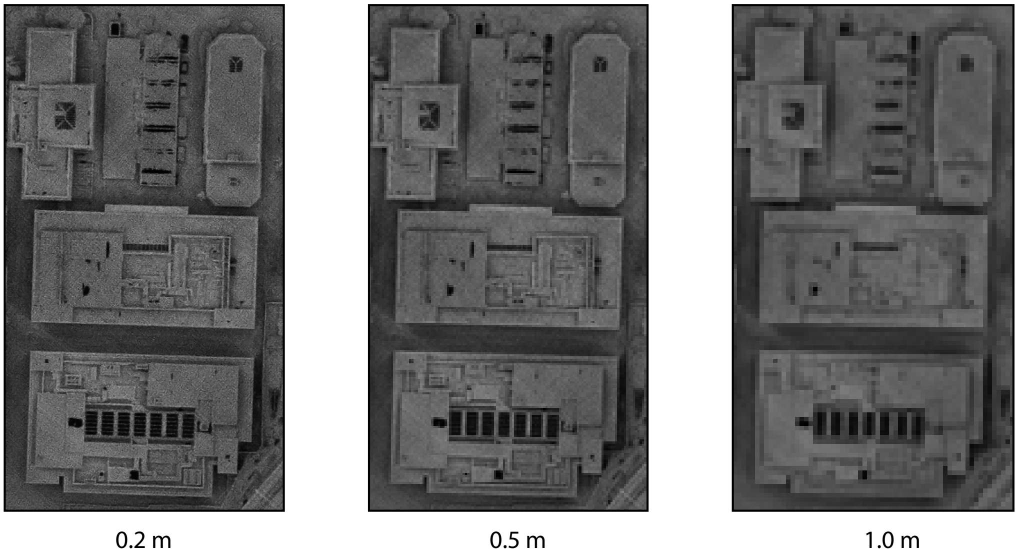


Figure 16.2: Three occlusion images of the same area sampled at flight path segment level with varying pixel dimensions. *Left*: $\Delta x_w = \Delta y_w = 0.2\ m$. *Middle*: $\Delta x_w = \Delta y_w = 0.5\ m$. *Right*: $\Delta x_w = \Delta y_w = 1.0\ m$.

slightly larger pixels eliminates much of the graininess, but the increased pixel size makes it slightly more difficult to identify small features such as details on roofs.

The image on the right has pixel dimensions $\Delta x_w = \Delta y_w = 1.0\ m$ in global coordinates. Again, increasing the pixel dimensions further produces a smoother image, at the price of further reducing the amount of detail in the image.

In summary, pixel dimensions control the sizes of building features that can be visualized. Thus, pixel dimensions should be chosen to match the sizes of interesting features. However, if pixel dimensions are too small, point sampling will not be uniform, causing artefacts in the form of graininess. Next, the effects of inserting missing echoes are tested.

16.3 Missing Echoes

In this section the effects of inserting missing echoes during the creation of occlusion images are examined. This was done by creating two occlusion images with identical parameters, except that for one of the images missing echoes were inserted. Comparison of these two images reveal effects caused by missing echoes.

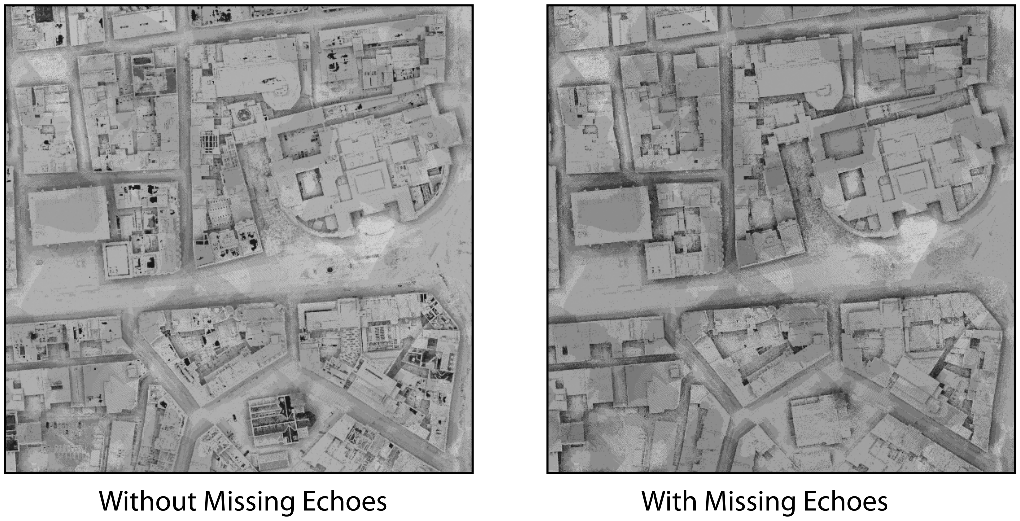


Figure 16.3: Two occlusion images of the same area sampled at flight track level are shown. Pixel dimensions in global coordinates are: $\Delta x_w = \Delta y_w = 0.5\ m$. *Left*: Occlusion image created without inserting missing echoes. *Right*: Occlusion image created with inserted missing echoes.

Figure 16.3 shows two images which were created with identical parameters, except that missing echoes were inserted during creation of the image on the right. The flight path was sampled at flight track level and pixel dimensions in global coordinates were $\Delta x_w = \Delta y_w = 0.5\ m$. Recall from the previous chapter that missing echoes are commonly found on building roofs. When comparing the two images it becomes evident that roof details are largely suppressed by the additional visibility inserted in the form of missing echoes. Interestingly, this suggests that the absence of echoes is in some cases beneficial to visualization, especially for small features. It could, therefore, be argued that the insertion of missing echoes provides overly strong sampling uniformity, whereby only large geometric irregularities, such as buildings, remain detectable. Depending on the application, the absence of roof details may or may not be desired, making it difficult to determine objectively the value of inserting missing echoes.

Having presented results demonstrating the effects of various parameters for creating occlusion images, the following section compares occlusion images to existing imaging techniques for the purpose of visualizing ALS data.

16.4 Comparison

This section compares existing techniques for visualizing ALS point data with occlusion image created using the methods presented in Chapter 10.

In terms of visual quality, occlusion images are compared to existing imaging tools for visualizing large ALS point data sets. The comparison focuses on imaging techniques, since direct visualization of large point data sets is inappropriate, as discussed in Chapter 10. More specifically, occlusion images are compared against three other types of images: (1) elevation images; (2) intensity images; and (3) colour images. The creation of elevation images was explained in Chapter 3 and intensity and colour images are created in a very similar way, where pixel intensity or colour is assigned from the most elevated mapping point. Note that intensity values were taken directly from ALS points, and were not normalized as suggested in [HP07].

The same pixel dimensions ($\Delta x_w = \Delta y_w = 0.5m$) were used in all images, and the same area is shown enlarged in each of the figures below. None of the images below were generated using missing echoes. Also, two versions of elevation images are shown, one where pixel intensity is proportional to elevation and another were elevation has been mapped to colours using the *Hue-Saturation-Value* (HSV) colour space [Smi78], treating elevation as the hue-component.

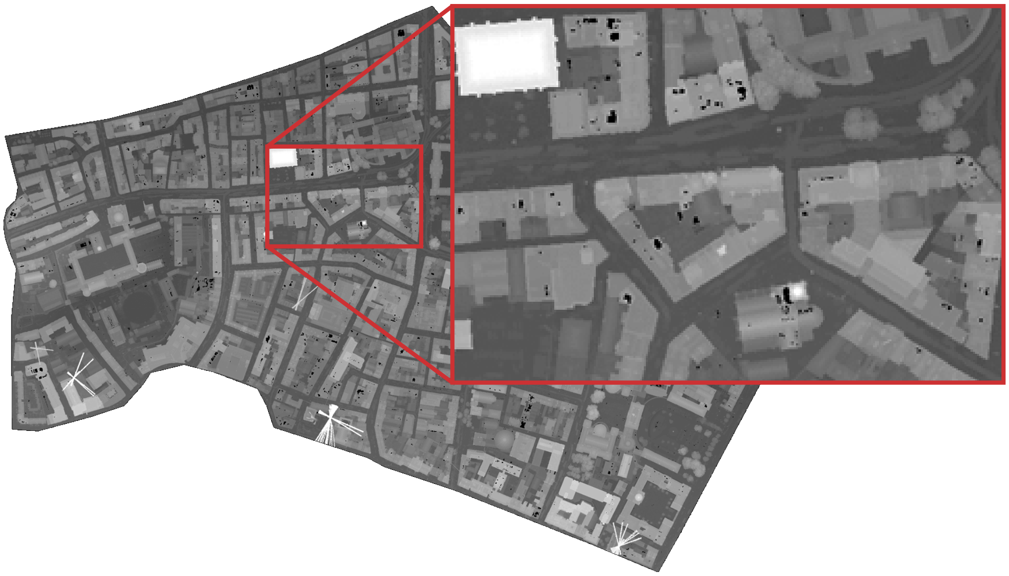


Figure 16.4: Elevation image of study area.

Figure 16.4 shows an elevation image of the entire Dublin data set, where a smaller area has been enlarged. Large structural shapes are visible but roof details are difficult to identify. Additionally, for reasons relating to the creation process, elevation images cannot visualize three-dimensional properties of point data. Similarly, Figure 16.5 shows an elevation image where intensity has been colour-mapped using the HSV colour space. Elevation is treated as hue, which provides different colour tones for different elevations. Although the colours are in some cases better adapted to visualizing small elevation differences, the fact that three-dimensional properties of the point data cannot be visualized remain.

As described in Chapter 2, ALS points often have an associated intensity value that is proportional to the amount of reflected pulse energy. Figure 16.6 shows an image where the intensity of the most elevated point mapping to each pixel has been used as the pixel value. However, the intensity values provided with ALS points are known to be unreliable [HP07]. Also, intensity images visualize spectral properties, as opposed to the geometry visualized in elevation images. As shown in the image it is nearly impossible to identify building details from the provided intensity values.

In cases where colour information is available for points, points can be mapped to pixels to

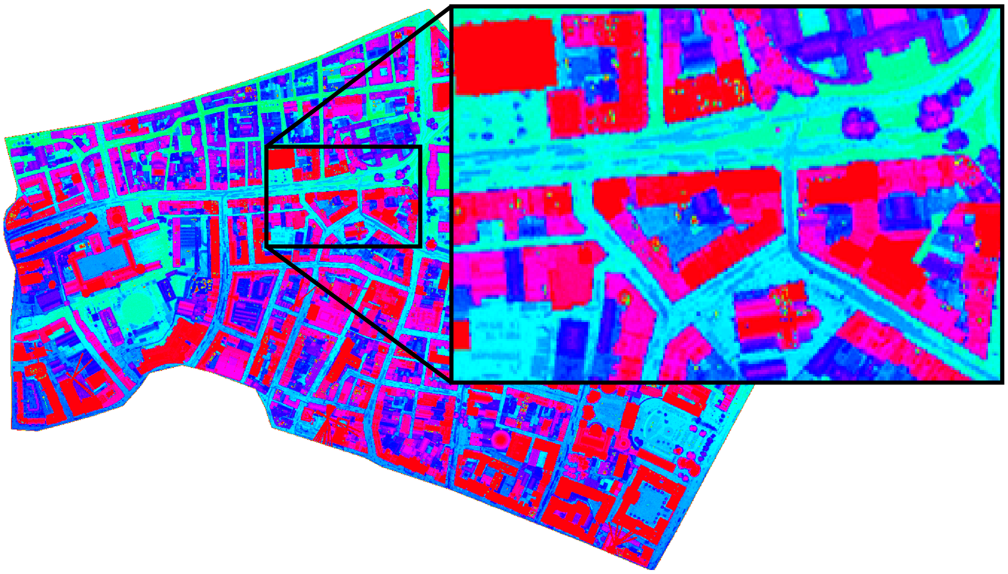


Figure 16.5: HSV image of study area were elevation is hue.

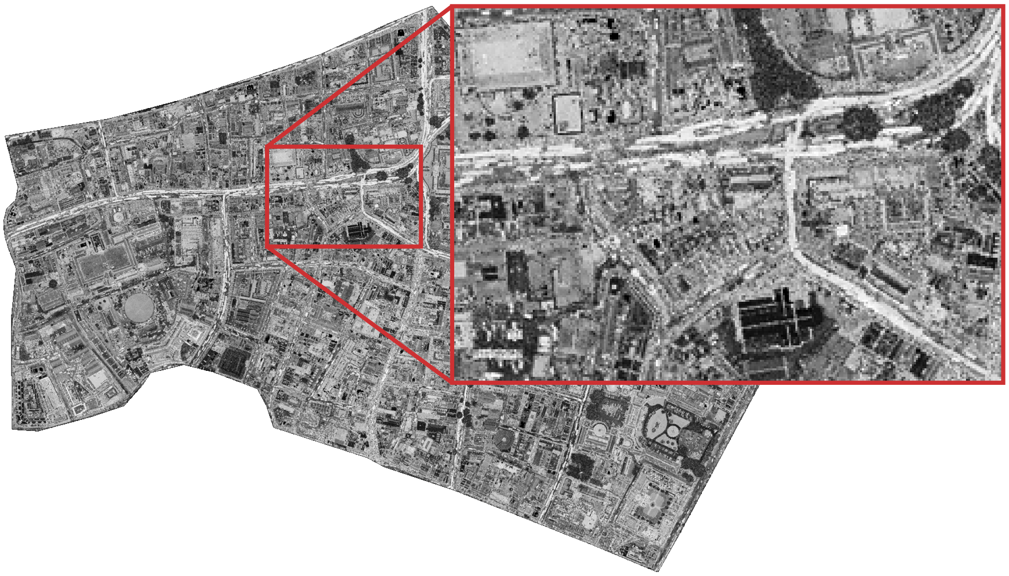


Figure 16.6: Intensity image of study area.

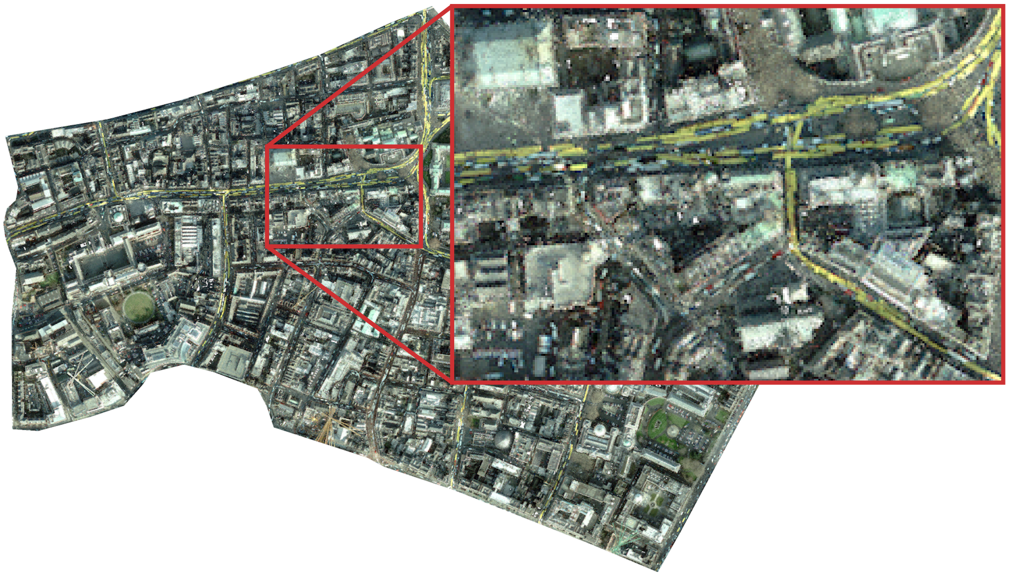


Figure 16.7: Colour image of study area.

create a colour image that is conceptually similar to an orthophoto. Figure 16.7 shows a colour image where the most elevated point mapping to each pixel transfers its colour to that pixel. As for intensity images, colour images do not directly visualize geometry, but rather spectral information. However, such information can be of interest. For instance, the yellow streaks in Figure 16.7 correspond to the roofs of buses and it may be possible to identify bus routes from such images. However, buildings appear in many different spectral forms, making them difficult to identify.

Figure 16.8 shows an occlusion image, where visibility was sampled at flight path segment level, of the entire Dublin data set. Buildings are clearly visible in this image. Also, small features on roofs are visible.

In Figure 16.9 the zoomed in regions from the previous figures are presented side by side. For the purpose of visually identifying buildings the elevation image (top left) and occlusion image are clearly preferable to the spectral images.

Elevation images suffer from quantization, where the difference in pixel values for small elevation changes is small, making such differences difficult to identify. The range of elevation values directly affects this quantization. Occlusion images do not suffer in the same way from

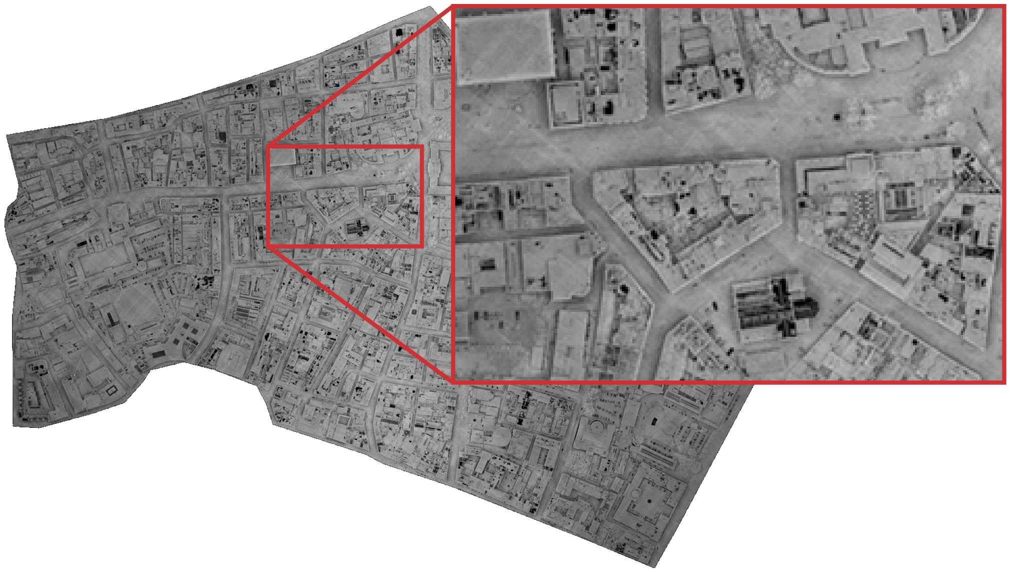
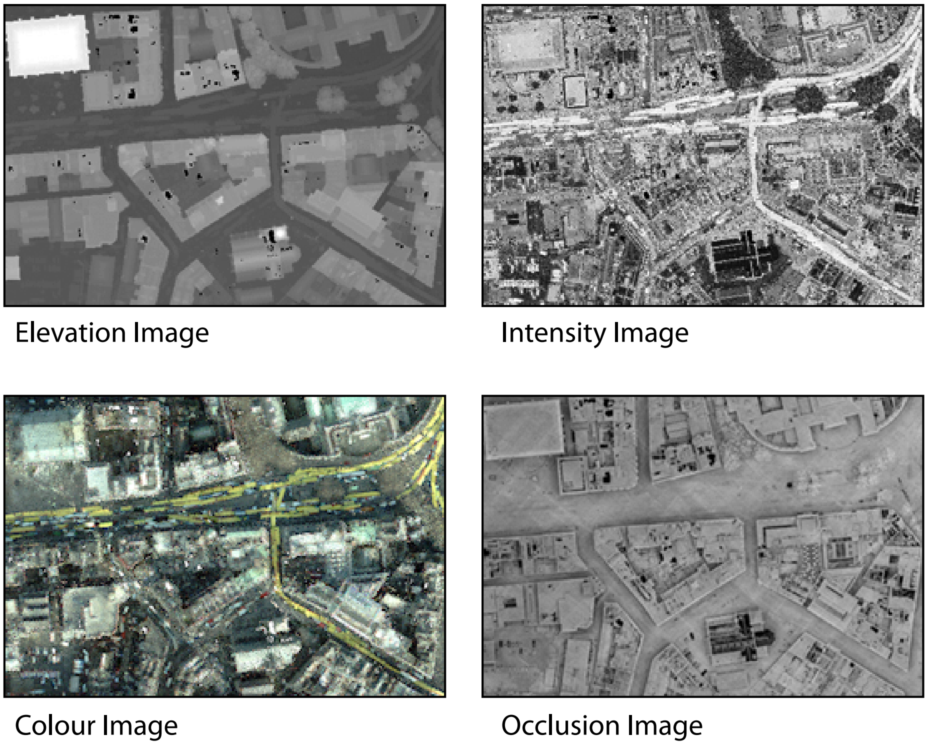


Figure 16.8: Occlusion image of the study area.

quantization issues, since visibility distribution is not directly related to elevation differences. For small areas where elevations do not vary hugely, quantization effects are less of an issue. However, for large areas, where elevations do vary hugely, quantization may assign the same value to pixels within a fairly large range.

In Figure 16.10 two images are compared with respect to visual quality for building feature identification. On the left side of Figure 16.10 an elevation image is shown. Major elevation discontinuities, e.g. building walls are visible and building footprints are clearly separated visually from the background as bright pixels. Details on roofs, however, are difficult to identify since pixel values are very similar for features with small variations in elevation. Also, some building parts are blocked from view by overhanging vegetation. In such cases, pixel values are assigned based on vegetation elevation, occluding the underlying building structures. On the right side of Figure 16.10 an occlusion image is shown. In this image it is possible to identify wall placement despite overhanging vegetation. Note how small features such as roof details and parked cars are clearly visible in the occlusion image.

Figure 16.11 shows another comparison between an elevation image and an occlusion image. The purpose of this comparison is to illustrate how objects that move over time are invisible in



Elevation Image

Intensity Image

Colour Image

Occlusion Image

Figure 16.9: Comparison between occlusion image and other common types of images used for ALS data visualization.

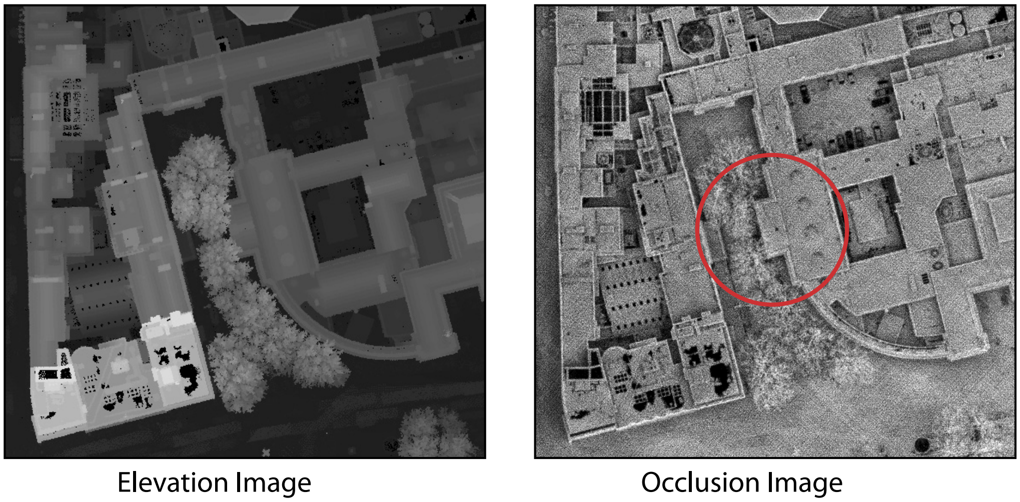


Figure 16.10: Comparison between elevation image (left) and occlusion image (right). Note that the elevation image contrast was manually optimized for viewing in this case. Pixel dimensions in global coordinates are: $\Delta x_w = \Delta y_w = 0.2\ m$.

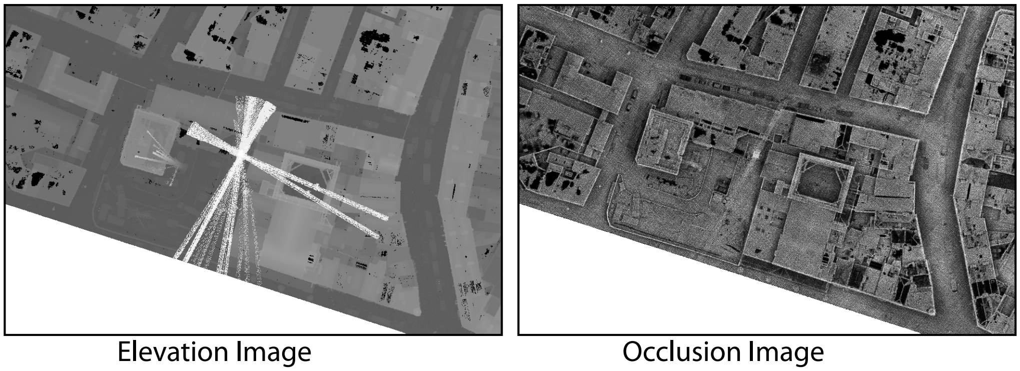


Figure 16.11: Comparison between elevation image (left) and occlusion image (right). Note that the elevation image contrast was manually optimized for viewing in this case. Pixel dimensions in global coordinates are: $\Delta x_w = \Delta y_w = 0.2\ m$.

occlusion images. The elevation image on the left contains a moving crane captured in several poses in different flight strips. The reason why the crane is captured in several poses is that ALS data is acquired in overlapping flight strips, where the same area is scanned multiple times. Because the crane is a tall structure it has a dominating effect in the elevation image. In occlusion images, however, the fact that pixel values are based on visibility means that the crane is nearly invisible. In particular, the crane does not hugely occlude points beneath it, since it is so much higher than the underlying buildings.

Chapter 17

Building Extraction

Extraction of individual buildings or building walls in large *Aerial Laser Scanning* (ALS) data sets is an important part of the workflow proposed in this thesis. Two different approaches for doing this were proposed earlier: (1) *semi-automatic building extraction*; and (2) *automatic building extraction*. Semi-automatic building extraction requires users to directly specify regions to be extracted. A semi-automatic extraction tool was presented in Chapter 11, providing mechanisms for extracting both buildings and individual building walls. In cases where a particular building is being studied, semi-automatic tools provide flexibility and robustness, since the user specifies directly the regions to be extracted. Semi-automatic building extraction was demonstrated in Chapter 11.

However, large ALS data sets of an urban area may contain hundreds of buildings and semi-automatically extracting every building would be tedious and time-consuming. Therefore, automatic approaches to building extraction are required and results of using the techniques presented in Chapter 12 are given in Section 17.1.

17.1 Automatic Building Extraction

While semi-automatic approaches to building extraction are extremely useful for extracting individual buildings, they are not well-suited for extracting all the buildings of an entire city. An automatic building extraction technique was presented in Chapter 12 and the results of applying this technique to the Dublin ALS data set are presented and evaluated in this section.

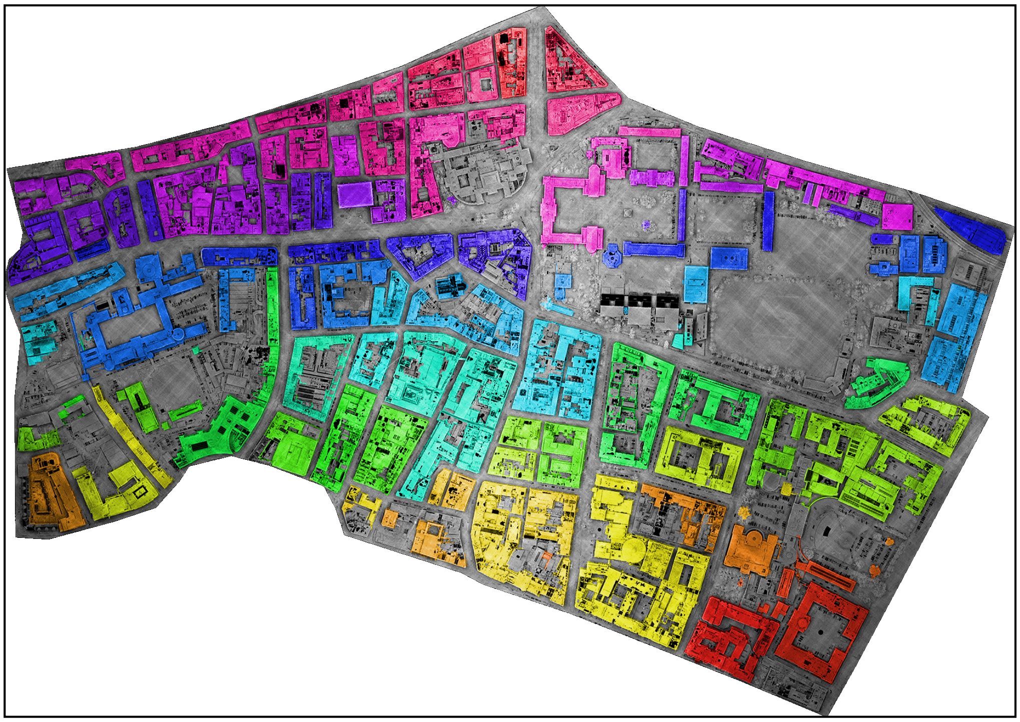


Figure 17.1: Detected buildings shown as uniquely colored labels overlaid on an occlusion image.

The automatic building extraction technique presented in Chapter 12 was applied to the Dublin ALS data set with pixel dimensions $\Delta x_w = \Delta y_w = 0.5\ m$ in global coordinates. Scan line bin dimensions were $\Delta x_b = 0.5\ m$ and the threshold for bins to be classified as belonging to building outlines was $T_b = 6$. The building extraction process took $\sim$ 1 hour on a standard laptop with a 1.8 GHz processor. The majority of the time was spent in the scan line classification step, where reading data from disk was one of the major bottle-necks. The image processing steps were performed within seconds. Furthermore, the methods were implemented in C++, but the code was not optimized for speed.

Figure 17.1 shows the resulting label image overlaid on an occlusion image. Each label corresponds to an *8-connected component* (8-CC) of pixels classified as belonging to a building footprint and labels have been assigned unique colors for clarity. A large number of buildings were detected.

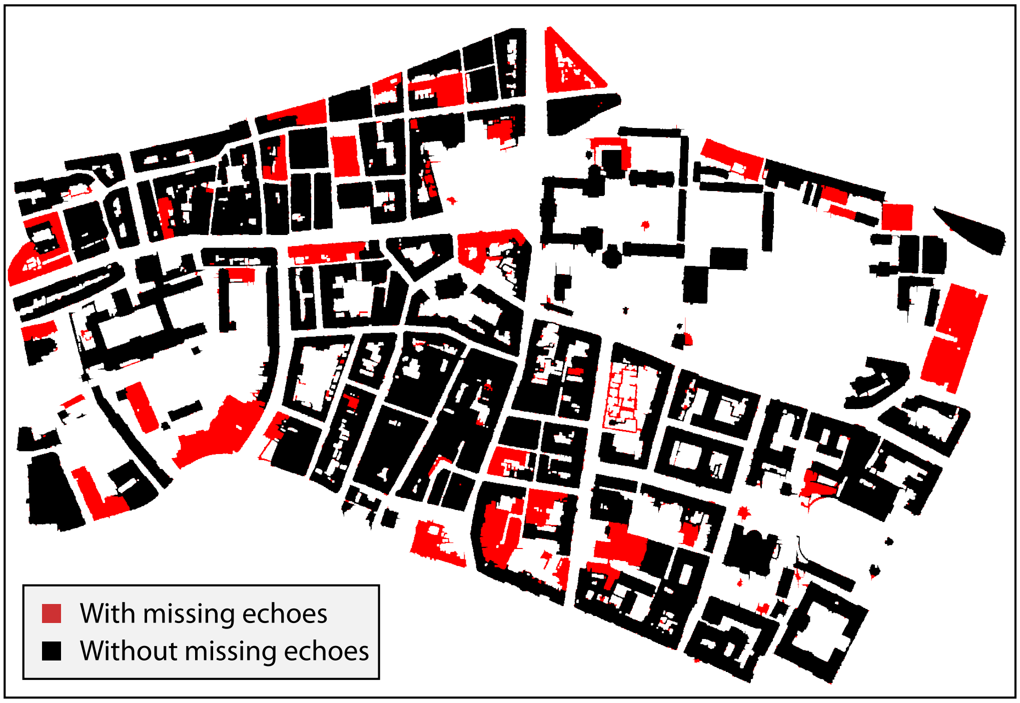


Figure 17.2: Detected buildings shown in black and red. Red buildings result from using missing echoes in the scan line classification step.

In addition to the building extraction parameters given above, missing echoes were found to improve building extraction results. Figure 17.2 shows the improved results when inserting missing echoes. As seen in the image, a number of additional buildings were detected as a result of inserting missing echoes. Moreover, given that missing echoes increase the availability of points, buildings detected without them must by definition be detected also in the case where they are inserted. From what is seen in the image it is clear that the use of missing echoes enables the detection of additional buildings.

In the following discussions extraction results are quantitatively evaluated and compared to existing methods, to the extent that this is possible.

17.1.1 Quantitative Evaluation

Quantitative evaluation of building extraction results is done using the same methods as Rottensteiner et al. in [RTCK05]. Use of these methods provides some means of comparing the results presented in this section to existing methods, although there are certain inherent difficulties involved in such comparisons, as will be discussed further on.

In order to quantitatively evaluate the extracted building footprints it was necessary to establish a ground-truth for comparison. Since no building footprint information for Dublin existed in any suitable format, a reference image containing building footprints was manually created. This was done by tracing building outlines in a vector format using a high-resolution occlusion image as guidance. This manual tracing took $\sim$ 12 hours. The resulting vector representation was then rasterized at the same resolution used in the building extraction process, enabling per-pixel comparison with the reference image. Figure 17.3 shows the reference image created alongside the extracted building footprints.

Two quantities often used in the evaluation of building extraction techniques are *completeness*, $\eta_{complete} \in (0, 1]$, and *correctness*, $\eta_{correct} \in (0, 1]$. Completeness is the percentage of reference building footprint pixels that were detected and correctness is the percentage of correctly detected building footprint pixels [HMWJ97]. For both of these quantities the optimum value is 1, corresponding to perfectly extracted building footprints with respect to the reference image.

Three important concepts are used in the computations of completeness and correctness: (1) *true positives*; (2) *false positives*; and (3) *false negatives* [HMWJ97]. These concepts are illustrated in Figures 17.4–17.6. Given that the reference image and the extracted building footprint image have the same pixel dimensions, per-pixel comparison was possible.

True positives (Figure 17.4) are defined as pixels that belong to a building footprint in both images. As is shown, there is a large degree of overlap between the detected and reference building footprints. In some cases, large portions of building outlines are detected. Additionally, interior regions may be bounded by these outlines. However, if none of the bounded interior regions corresponds to the main building footprint, buildings are not fully detected.

False positives (Figure 17.5) are pixels that belong to a building footprint in the extracted image but not in the reference image, thus signifying that a pixel was erroneously classified as belonging to a building footprint. As evident from Figure 17.5, the majority of false positive

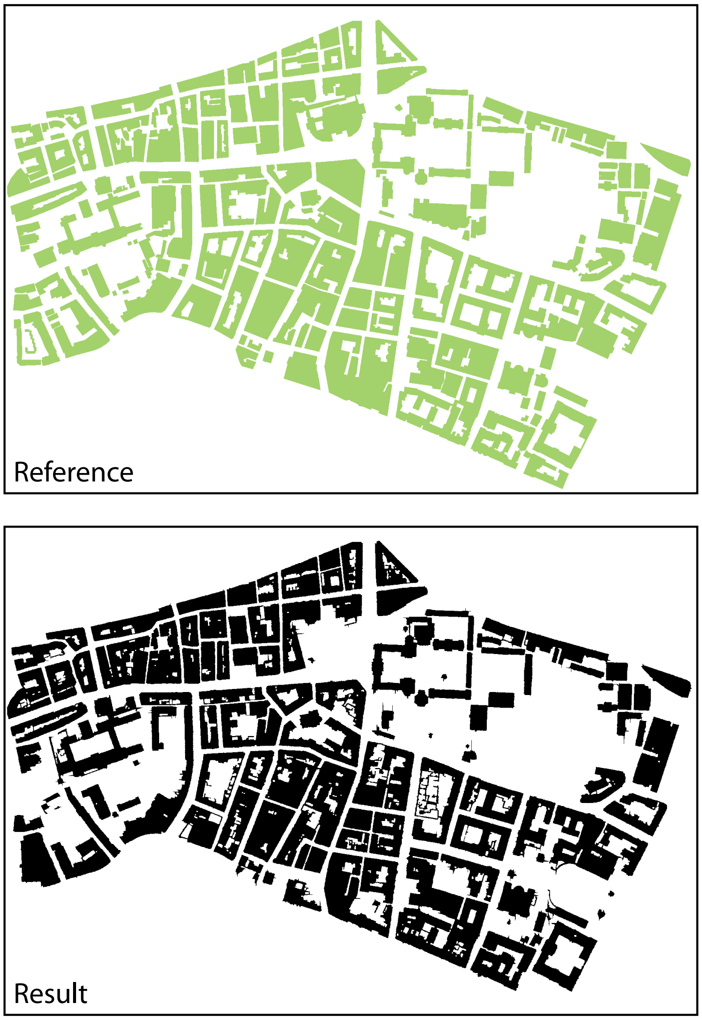


Figure 17.3: *Top*: Reference image of building footprints in the study area. *Bottom*: Automatically extracted building footprints in the study area.

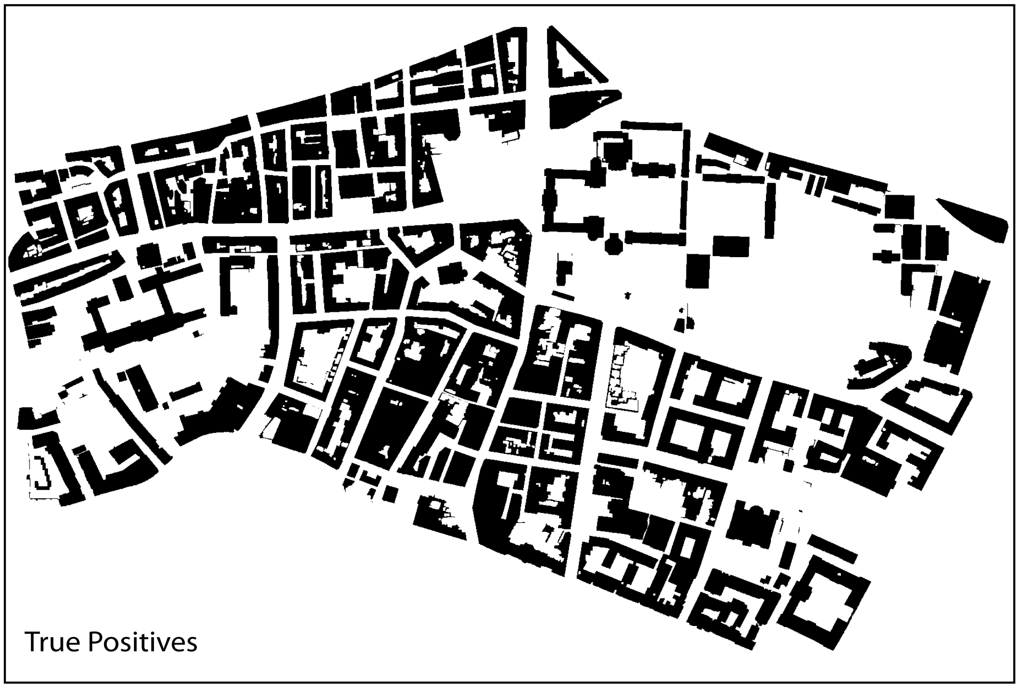


Figure 17.4: True positives are building footprint pixels existing in both reference and extracted image. This corresponds to correctly detected building footprint pixels.

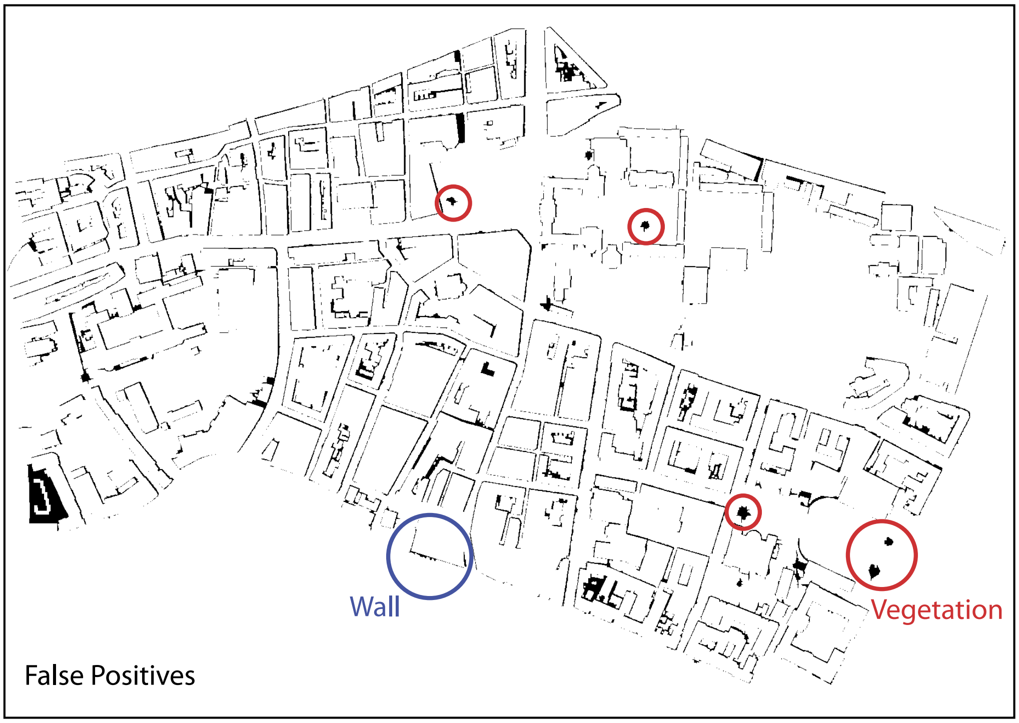


Figure 17.5: False positives are building footprint pixels in the extracted image that have no counterpart in the reference image. This corresponds to erroneously detected building footprints.

pixels appear in narrow bands around buildings. This suggests that most false positive pixels could be removed by applying some sort of post-processing step to the detected building labels. Alternatively, since false positives seem to occur almost exclusively around building outlines, some sort of scale bias may have been introduced in the reference image. In particular, it seems that false positives occur to a large extent in north and east directions, further suggesting a systematic bias. The blobs in the bottom-right corner correspond to vegetation erroneously classified as building footprints. Other examples of misclassified vegetation can be found at the top of the image. Dense vegetation may exhibit similar statistics to building walls, where point samples are to a large extent vertically stacked, explaining why the algorithm detects them as walls. Similarly, walls and other building extensions are sometimes erroneously detected. Fortunately, however, such extensions rarely form closed paths and are, thus, not confused with building footprints. An example of a wall partly surrounding a building site is shown at the bottom of the image.

Finally, false negatives (Figure 17.6) are building footprint pixels in the reference image that were not classified as belonging to a building footprint in the extracted image, denoting cases were reference buildings, or parts of buildings, were not correctly detected. Large black regions in Figure 17.6 correspond to entire buildings that could not be detected. There are several scenarios in which buildings cannot be detected with the method presented. In particular, buildings that do not fit well into the description of being bound by vertical planes cannot be detected. Examples of this are monumental buildings, where the outer roof parts are held up by pillars. An practical example of this can be seen at the top of Figure 17.6, where the Bank of Ireland building was not properly detected. Moreover, some modern buildings have walls that are largely constructed from glass or metal. The north-south oriented building block on the left is a good example of this. In this case, the walls are constructed from a combination of glass and metal, leading to poor point sampling quality on those vertical surfaces. Another example is the library building on Trinity College Dublin (TCD) campus, located in the middle of the image. In this case, the north walls consist of a slanted glass structure that cannot be detected as wall by the proposed algorithm. Because these materials cannot be sampled well with ALS, these walls cannot be detected. Failure to acquire points on building walls may also occur in other scenarios. For instance, vegetation close to buildings may occlude substantial parts of building walls. Occlusion also happens between closely spaced buildings, as discussed

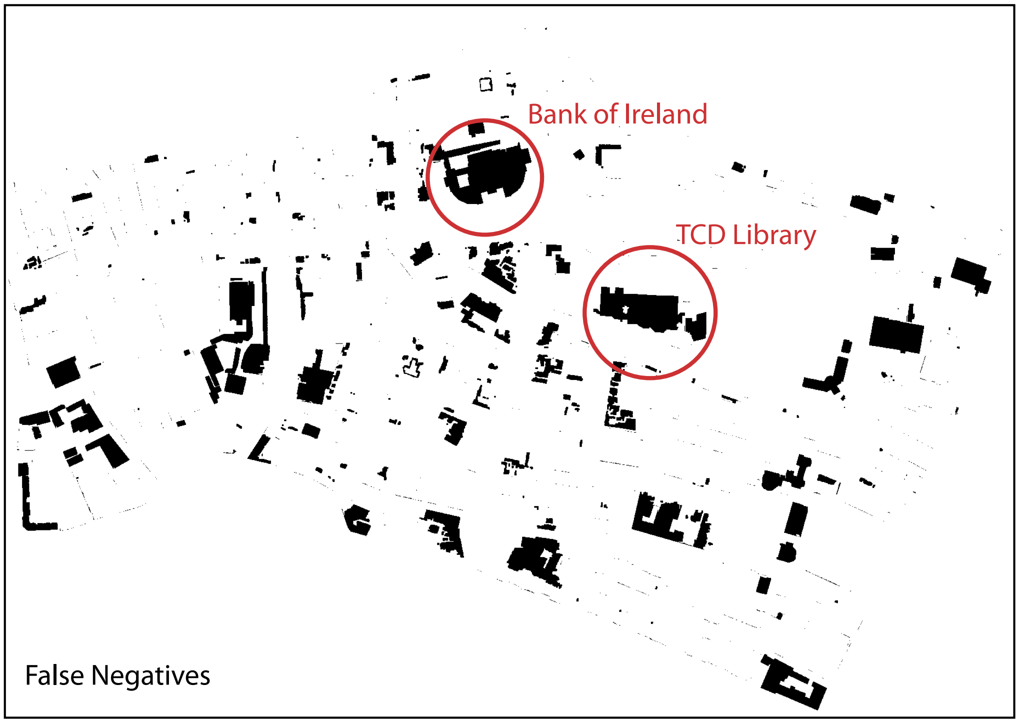


Figure 17.6: False negatives are building footprint pixels in the reference image that have no counterpart in the extracted image. This corresponds to building footprints that were not detected.

in Chapter 8. Examples of this can be found on the right hand side of Figure 17.6, where several building blocks where not detected. Finally, although it is not clear from the image, the method presented is sensitive to small gaps in detected building outlines. A single misclassified pixel may lead to an entire building not being correctly classified. Introducing a post-processing step on statistical images prior to building footprint extraction could most likely result in significant improvements in the number of false negatives.

Having introduced the basic concepts used to determine completeness and correctness, these two quantities are computed as follows:

$$\eta_{complete} = \frac{T^+}{T^+ - F^-} \tag{17.1}$$

where T^+ is the number of *true positive* pixels and F^- is the number of *false negative* pixels. Similarly correctness is computed as:

$$\eta_{correct} = \frac{T^+}{T^+ - F^+} \tag{17.2}$$

where F^+ is the number of *false positive* pixels.

Completeness (Equation 17.1) and correctness (Equation 17.2) for the extraction results presented in this section are 0.82 and 0.87, respectively. These values were computed on a per-pixel basis, where the quantities involved were represented by the number of pixels in each category. In the following discussions these values are compared with similar quantities reported for existing methods.

17.1.2 Comparison

The completeness and correctness values for extraction results provided in the previous subsection enable comparison with existing methods that provide similar quantitative analysis. It should, however, be noted that such comparisons inherently have a large degree of uncertainty since existing methods were tested on different ALS data sets. Given that the automatic building extraction technique presented in this thesis relies heavily on the presence of points on building walls, it would most likely perform worse than existing methods on elevation data. Although comparison between building extraction methods are somewhat uncertain, success rates can be compared and this is done below.

Table 17.1 shows statistics for some existing approaches. Importantly, some of the existing methods use multi-sensor approaches, typically a combination of ALS points and orthophotos.

Method	**Data**	**Density** [points/m^2]	**Area** [km^2]	**Type**	$\eta_{complete}$	$\eta_{correct}$
[MHH03]	ALS Intensity	~ 2.5	2	Suburban	0.91	0.84
[VS03]	ALS	1	1	Rural	0.93	0.96
[RTCK05]	ALS RGB	1	4	Suburban	0.94	0.85
[RTCK07]	ALS RGB	1/1.8	4/4	Suburban/ Industrial	0.87/0.91	0.91/0.92
[MKH07]	ALS RGB	17	1.4	Suburban	0.93/0.93	0.91/0.84
[DP08]	ALS	6	N/A	Suburban	0.75	N/A
[MHA*10]	ALS RGB	~ 3	5	Suburban	0.89-0.97	0.78-0.91
Chapter 12	**ALS**	$\sim$ **225**	**1**	**Urban**	**0.82**	**0.87**

Table 17.1: This table gives information about the data used in existing approaches along with reported success rates. Some of the existing techniques use multi-sensor approaches and, thus, have two entries in the data-column. The term *RGB* refers to color information, typically in the form of orthophotos, being used.

Moreover, in all existing methods points acquired with ALS are treated as elevation data. Additionally, the type of each data set refers to the characteristics of buildings within each study area. Importantly, types of study areas shown in the table are those given by the authors.

Matikainen et al. [MHH03] use a combination of elevation differences and textural characteristics in an intensity image to define building regions. Most buildings, especially those covering more than a certain area, were detected using their approach. Voegtle and Steinle [VS03] presented an approach for detecting buildings from ALS data. Their approach uses fuzzy logic to detect buildings using the gradients of an elevation image. A high degree of success is reported for a rural area. Rottensteiner et al. [RTCK05] presented a method that uses both ALS points and orthophotos. A probabilistic approach based on the fusion of color information is used to detect the buildings of a suburban region. In a later publication, Rottensteiner et al. [RTCK07] presented an updated version of the 2005 method, which was tested on the two data sets. Results were better for an industrial region, mostly because buildings in such regions tend to be larger. A machine learning approach was presented by Matikainen et al. [MKH07], where part of a data set was used to "train" a classification tree. Two slightly different variations of the classification tree were then applied to the rest of the data. Dorninger and Pfeifer [DP08] presented a comprehensive collection of methods for creating building models. The building extraction part of their workflow is based on identifying planar segments in point data. However, while their results were convincing in terms of modeling, their success rate in terms of building extraction was fairly low. In 2010, Matikainen et al. [MHA*10] presented an updated version of their classification tree-based methods, this time incorporating color information. However, while robustness (i.e. correctness) was improved, the completeness was slightly lower.

The completeness of the automatic building extraction method presented in this thesis is fairly low when compared against the existing method mentioned above. However, when compared against only those methods relying solely on ALS data ([VS03, DP08]), it turns out that these results are more competitive. Moreover, the fact that these results were achieved for a densely built-up urban area should be considered in such comparisons. Also, the point density used in this thesis is an order of magnitude greater than in any of the other methods. Because methods have not been compared on the same data sets it is difficult to draw any conclusions from this. In general, existing methods have been tested mostly on suburban regions. An interesting extensions to the work in this thesis would be to test existing methods on the Dublin data set.

In summary, while there is some degree of uncertainty involved in comparing building extraction on different data sets, it has been shown in this chapter that the methods presented in this thesis perform on the same level as existing methods. Additionally, existing methods have mostly been tested on suburban regions, in some cases using multi-sensor approaches. The following chapter presents results of computational modeling, where ALS points acquired on building walls were used to created solid models for building simulations.

Chapter 18

Solid Modeling

This chapter demonstrates simulation results on solid models created using the technique presented in Chapter 13. Both *Aerial Laser Scanning* (ALS) and *Terrestrial Laser Scanning* (TLS) data sets were used to create solid models. Simulations of stress under self-weight were performed on the created models with ANSYS, which is a widely used simulation software package. It is important to note that these simulations were carried out to demonstrate the feasibility of the solid modeling technique presented. As such, stress distributions are qualitatively analyzed with respect to expected behavior. Quantitative analysis and accuracy measurements of simulated stress levels are beyond the scope of this thesis and require further support from the civil engineering community.

As mentioned, both ALS and TLS data sets were used in the evaluation of the solid modeling tool. ALS points on building walls were semi-automatically extracted, as the automatic approach does not support extraction of individual walls at present. For terrestrial scans, building walls were manually separated. The models used in the simulations below were created within seconds, as opposed to the many hours required to create such models using traditional approaches. However, the created models are of a simplistic nature and no effort was spent on assigning material properties to the models. As such, the tests below aim first and foremost to demonstrate that usable solid models can be created directly from point data. Bench-marking such models with respect to accuracy requires large amounts of testing. Such testing is currently being performed and will be presented in a separate thesis written by a civil engineering student [Hon11]. In this chapter qualitative analysis is given and it is demonstrated that the created models behave as expected. Solid modeling and simulation results are presented separately for ALS and TLS points in the following two sections.

18.1 ALS Case Study

Points acquired on a building wall were extracted from the Dublin ALS data set using the tools described in Chapter 11 and the solid modeling tool described in Chapter 13 was applied to the extracted points. In the extraction process points within a small distance of the wall plane were extracted. Further, points were transformed to a coordinate system where one of the horizontal axes was parallel to the wall plane, thus avoiding aliasing artefacts in the voxelization step.

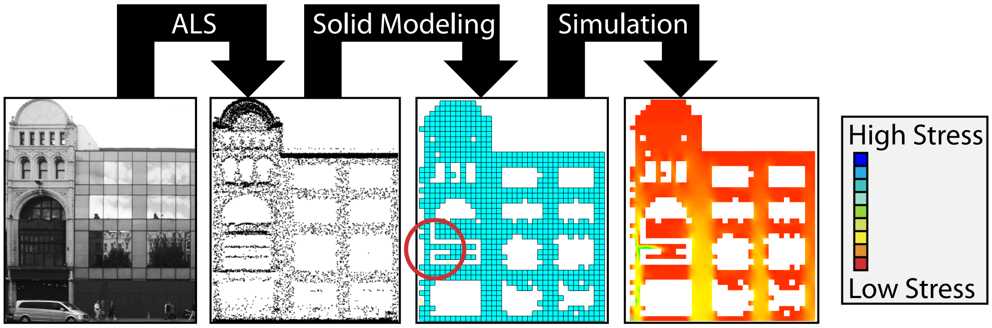


Figure 18.1: ALS points acquired on a building wall were extracted and a solid model was created from those points. The solid model was used to simulate stress under self-weight and stress distributions appear to be realistic since high amounts of stress occur at load-bearing regions. The red circle shows a structural feature that was incompletely modeled, resulting in attachment to the wall on only one side. Simulation shows significant stress in this area because of this.

In the voxelization step the parameters for voxel dimensions and classification were $\Delta x = \Delta y = \Delta z = 0.25\ m$ and $T_n = 1$. Thus, every voxel containing a mapped point was considered in the solid model. Note that the solid model is planar in the sense that all voxel centers lie in the same plane. Simulation results are shown in Figure 18.1. Red corresponds to low amounts of stress and blue corresponds to high amounts of stress. The building selected for testing has several window openings. Although these window openings were not perfectly modeled, as can be seen by the jagged window edges, simulation results show that stress is focused around thin, load-bearing structural parts. To the left, on the second floor of the building it can be seen from the point data that there are two bars spanning a window opening. These bars were not successfully modeled and are only attached to the wall at a single point. Interestingly, this causes large amounts of stress on the left side, where the weight of the bars is only supported on one side. Clearly, the fairly sparse point sampling introduces some errors in the voxelization step. However, these errors are reflected in the simulation, demonstrating the validity of the created model. Next, stress simulations using solid models created from TLS points are presented.

18.2 TLS Case Study

The previous section presented solid modeling results using ALS points. This section presents solid modeling results using point data acquired with TLS. TLS points acquired on a single building wall were used to create a solid model used in a simulation similar to the previous one.

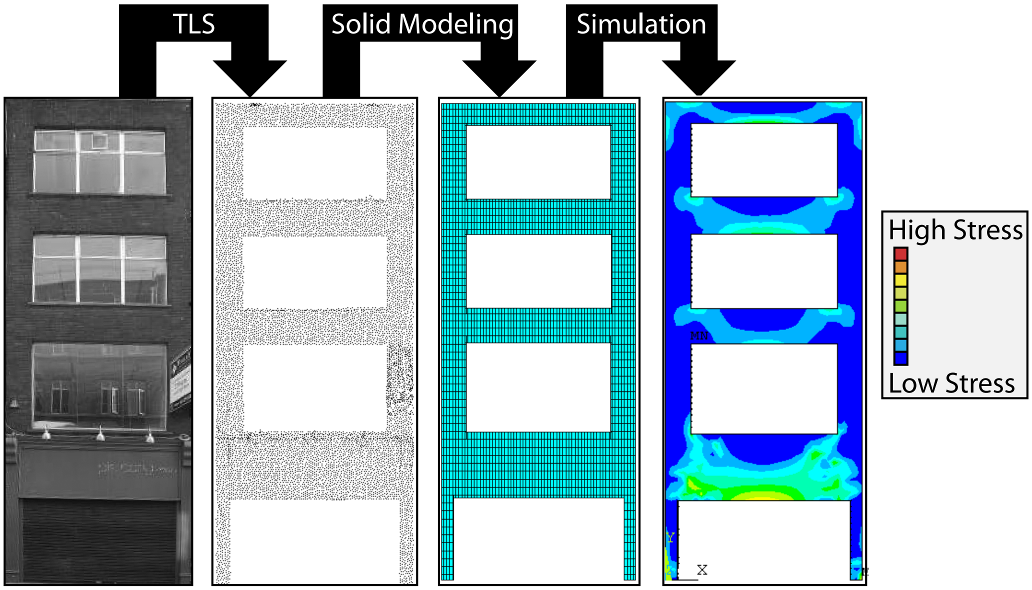


Figure 18.2: Results of running a stress simulation on solid model created from TLS data.

Simulation results for a solid model created from TLS points are shown in Figure 18.2. In the voxelization step the parameters for voxel dimensions and classification were $\Delta x = \Delta y =$

$\Delta z = 0.1\ m$ and $T_n = 1$. Lower amounts of stress are shown in blue and higher amounts of stress are shown in red. Compared to the previous study using ALS points, a smaller voxel size was used in this study, owing to the fact that TLS points were more densely sampled on the building wall. The modeled building wall is supported at ground level only by two thin strips of wall on either side of a large opening. Although some stress is carried by these thin strips, the majority of stress occurs just above the large opening. Similar stress patterns occur at the smaller openings higher up on the wall, where the level of stress appear to be proportional to the opening size. Qualitatively speaking, these stress patterns are realistic and occur in expected regions.

In summary, solid models created from point data acquired with laser scanning have been shown to work with a widely used simulation software. Solid models created from TLS points appear to have fewer artefacts than those created from ALS points. The main reason for this is that TLS points are more densely sampled and, thus, are less likely to be absent on important features. However, as ALS technology improves over time, it is expected that increases in point density will follow. Further, the simulation test carried out demonstrate plausible behavior in terms of stress distribution. Quantitative verification of stress levels remains to be done, but is beyond the scope of this thesis. Thus, all tools in the proposed workflow have been evaluated and the final chapter of this part discusses the results presented in this and previous chapters.

Chapter 19

Discussion

Previous chapters in this part presented results using the tools in the workflow shown back in Figure 14.1. Tools in this workflow were divided into four categories: (1) *acquisition*; (2) *visualization*; (3) *building extraction*; and (4) *solid modeling.* In this chapter results are discussed in terms of the goals of this thesis, which were stated in Part I (Chapter 6). In short, the goal of this thesis is to create solid models on a city-scale for use in building simulations. Results in each category are discussed in the following sections.

19.1 Acquisition

Acquisition of geometric data for buildings was done using both *Aerial Laser Scanning* (ALS) and *Terrestrial Laser Scanning* (TLS), which were introduced in Chapter 2. TLS was used to acquire points on a single building using a standard approach and will not be further discussed in this section as this data was mainly acquired to evaluate other tools in the workflow. ALS points were acquired for an urban study area using a flight path specifically designed to acquire points on building walls using the guidelines presented in Chapter 8. These two data sets were presented in detail in Chapter 15. The following discussions, however, will focus on the results of ALS acquisition.

As mentioned, the flight path used in the scanning of a $\sim 1\ km^2$ area located in central Dublin, Ireland was designed using the guidelines presented in Chapter 8. The three guidelines were: (1) low aircraft altitude and wide scan angle; (2) multiple overlap between flight strips;

and (3) flight tracks aligned at 45 degrees to the major street grid. It was hypothesized that by following these guidelines points could be acquired for a substantial portion of building walls in the study area.

Verification of the effects of low aircraft altitude and wide scan angle were not verified explicitly since these two parameters were constant during the scan, making it difficult to study their impact on the acquired data. However, the guidelines for these parameters were motivated by predictions of linear sampling resolution. These predictions were shown to be accurate through study of acquired points in a representative scan line. Thus, guidelines regarding aircraft altitude and scan angle were implicitly verified. Similar reasoning can be applied to flight track orientation with respect to the major street grid, which was also based on linear resolution predictions. The effects of multiple flight strip overlap were demonstrated visually in Chapter 15 by comparing point sampling on a representative building wall for a single flight strip versus multiple flight strips. It was clear from this comparison that multiple flight strips provided a significant increase in point sampling and that details, such as windows, became clearly detectable. Additionally, point sampling on building walls was confirmed by automatic building extraction results, where a clear majority of the buildings in the study are were detected using a technique that relies heavily on the presence of points on building walls. Building extraction results are discussed further in Section 19.3.

Comparisons between ALS and other remote sensing techniques, such as aerial imagery, for acquisition of geometric data over urban areas have not been made since such data was not available. Interesting factors to compare would be cost, time spent on acquisition and accuracy. The first two factors are related, since cost is proportional to acquisition time under the valid assumption that the actual scanning is done by contractors owning the necessary equipment. Further, as for traditional ALS, aerial imagery provides elevation data. As such, extensions are required to existing aerial imaging techniques in order to conduct a fair comparison in terms of cost, time and accuracy for the same end product.

The ALS data set acquired for the study area differs from traditional ALS data sets in two main areas: (1) points are sampled on building walls as well as other surfaces; and (2) point density is extremely high [$\sim 225\ points/m^2$]. Geometric data, in the form of point samples, on building walls is exploited in several further stages of the workflow and enables solid models of building walls to be created from ALS data. As mentioned above, substantial flight strip overlap was used to acquire points on building walls. As a consequence, there was a significant amount of redundancy in point sampling on other surfaces, which explains the unusually high point density. Sampling redundancy caused by flight strip overlap was exploited for visualization purposes, as discussed next.

19.2 Visualization

Digital images, introduced in Chapter 3, are commonly used in visualizations of ALS data. In Chapter 10 a new imaging technique was presented and images created with this technique are referred to as *occlusion images*. Occlusion images were created for ALS data of the Dublin study area and were presented in Chapter 16.

Occlusion images are based on the concept of treating ALS points as visibility samples. Using the framework for processing ALS data presented in Chapter 9 it becomes possible to associate an aircraft positions with every scanned point. Additionally, this framework allows extremely large ALS data sets to be processed without memory constraints. The aircraft position from which ALS points were acquired are known and ALS points confirm visibility from positions in the sky to surfaces on which points were sampled. Pixel values of occlusion images are derived from the number of unique aircraft positions from which surfaces in pixels are visible. As such, occlusion is measured as visibility toward the sky, where the idea is that some surfaces are visible from a larger number of positions in the sky than others and that such occlusion patterns are well-suited for visualizing three-dimensional ALS data.

Existing imaging techniques, elevation imaging in particular, are not ideally suited for visualizing small features present in extremely densely sampled ALS data sets. Moreover, existing imaging techniques are not suitable for visualizing three-dimensional point data, owing to the fact that pixel values are derived from singular points. It has been shown that while small building features do not vary significantly in elevation, occlusion patterns around such features make it possible to detect them. Finally, substantial flight strip overlap is required for useful occlusion patterns to emerge. However, such a strategy can also be used to acquire point data for building walls, thus serving a dual purpose.

19.3 Building Extraction

Building extraction is the process of identifying and extracting building from an ALS data set, as explained in Chapter 4. Two methods for building extraction were presented in this thesis: (1) *semi-automatic* (Chapter 11); and (2) *automatic* (Chapter 12). Results for automatic building extraction were presented in Chapter 17.

Semi-automatic building extraction was implemented in terms of an interactive tool, letting users manually extract regions specified by drawing shapes on top of a visualization of the ALS data. For visualization purposes occlusion images were used and were found to be satisfactory for the task of identifying buildings and building walls. The results of semi-automatic building extraction were easily verifiable by establishing that extracted points corresponded to the specified region.

The quantitative analysis in Chapter 17 shows that the presented building extraction method performs at roughly the same level as existing methods. Most existing automatic building extraction methods have been tested on a fairly small number of data sets. Further, data sets used in testing tend to have similar characteristics in terms of point density and building structures. A notable exception is the comparative study conducted by Sithole and Vosselman in [SV04], where several methods were tested on several different data sets. As for most existing methods, the building extraction method presented in this thesis has only been applied to the Dublin ALS data set. An important reason for this is that the Dublin ALS data set differs substantially from existing data sets in that points are, to a large extent, available for building walls. Given that the presented method relies heavily on such sampling, it would, in all likelihood, perform poorly on traditional elevation data sets. Therefore, a better comparison would be to test existing building extraction methods on the Dublin data set. At present this has not been done and doing so would provide an interesting extension to the work presented in this thesis.

Additionally, the presented building extraction method is fairly simplistic in that a single threshold value is used for bins, regardless of bin distance to nadir. For equally tall buildings and assuming no occlusion, the sampling predictions indicate that larger amounts of points should be available on walls in flight strip flanks. Thus, it might be possible to scale the threshold by distance from nadir. However, not all buildings are equally tall, and without using any a priori knowledge of buildings it is difficult to compute a proper scale factor.

Although a large number of buildings were automatically extracted, computational modeling was conducted instead on semi-automatically extracted building walls. The reason for this was that individual walls were not automatically identified and points extracted for buildings were, therefore, not suitable for creating the type of solid models used in building simulations. Solid modeling results using semi-automatically extracted buildings walls and TLS points are discussed next.

19.4 Solid Modeling

The type of solid model commonly used in building simulations was described in Chapter 5. A method for converting point data into this type of solid model, consisting of connected cubes, was presented in Chapter 13. Although this method is fairly simplistic, where cubes are defined based solely on the presence of points, solid models created with this method were demonstrated to behave plausibly in the stress simulations shown in Chapter 18.

Simulation tests showed that stress distributions were realistic for solid models created directly from both ALS and TLS points. However, evaluation of stress level accuracy was not carried out in this thesis. Such evaluation needs to be carried out in the engineering community, where simulation results can be compared with manually created solid models in order to quantify errors for further analysis.

Although quantitative error analysis is beyond the scope of this thesis, it is possible to discuss the factors that influence the uncertainty in simulations. The two main sources of uncertainty in the simulation tests are believed to be: (1) geometrical artefacts; and (2) uncertainty in material properties. For structural and esthetical reasons, openings on building walls, such as windows and doors, tend to be regular, not only in shape, but also in placement. Geometrical artefacts appear in the form of irregularly modeled openings, where straight edges suffer from aliasing caused by the use of regular voxel grids. Further, irregular modeling of individual openings is detrimental to overall regularity in terms of opening placement. Material properties were empirically selected to test the solid models created, which at present are created from geometric data only. It is unclear to what extent geometrical irregularities affect simulated stress levels and investigations need to be made in these directions.

In terms of creating usable solid models, voxel dimensions should be chosen such that load-bearing elements are modeled at a suitable level of detail, allowing simulations to run quickly without overly compromising accuracy. Simulation speed is an important factor when modeling an entire city. This is especially true, if the intent is to use the models in disaster management response. TLS provides dense point samplings at a resolution that is often more than sufficient for modeling at the desired scale. However, ALS data may be fairly sparse and is not regularly sampled. Sampling density is lower at the base of a wall, for reasons discussed in Chapter 8. Also, parts of building walls just beneath overhanging roof parts may be occluded when scanned from above, causing sampling gaps between the wall and the roof. As such, it is sometimes difficult for ALS points to support the desired voxel dimensions.

The quality of the created solid models is largely dependent on the input point data. For this reason the more dense point samplings provided by TLS allow creation of solid models with a higher level of detail and fewer artefacts. However, although ALS data on building walls is often fairly sparse at present, advances in ALS technology are likely to change this. It is, therefore, of importance to have demonstrated the feasibility of being able to create solid models directly from ALS data. Future generations of ALS systems are likely to provide denser point sampling of buildings walls, and it is expected that this will further improve the quality of solid models created from such data.

Part IV

Conclusions and Future Work

Chapter 20

Conclusions

Part I of this thesis provided the background necessary to clearly state the problems being addressed. Part II presented methods for solving these problems and in Part III results using these methods were presented and discussed. This final part (Part IV) summarizes the previous parts and evaluates the overall results in terms of achieving the goals of this thesis. The first chapter of this part presents conclusions drawn from the material presented in previous parts and the second chapter gives possible directions for future work.

This chapter begins with a review of the methodology used to achieve the results presented in this thesis. Thereafter, conclusions are presented for each part of the methodology, based on the results demonstrated in Part III. Toward the end of this chapter conclusions are summarized and the results presented in Part III are discussed in a wider context, extending beyond the scope of this thesis.

20.1 Methodology Overview

This section presents an overview of the methodology used in this thesis. The goal of this thesis was to create solid models to support building simulations on a city-scale. A set of tools required to achieve this goal was proposed and the workflow associated with these tools was shown in Figure 14.1. The use of these tools is described below and provides the necessary background for interpreting the results that were presented in Part III.

In 2007, *Aerial Laser Scanning* (ALS) [Chapter 2] was used to acquire geometric data, in the form of point samples, for a $\sim 1\ km^2$ study area in central Dublin, Ireland. For this mission, a flight path specifically designed to include point sampling on building walls, involving substantial flight strip overlap, was used (Chapter 8).

The acquired ALS data was visualized using a novel type of digital images (Chapter 3), referred to as *occlusion images* (Chapter 10). Occlusion images were created by associating aircraft positions with scan line points (Chapter 9) and treating points as visibility samples. In the subsequent extraction of points sampled on buildings such images were used both to enable users to manually select buildings and to verify the results of automatic building extraction, as discussed below.

As explained in Chapter 4, a fundamental step in the processing of urban ALS data is the extraction of points acquired on buildings. In this thesis two different approaches were used to extract buildings. In the first approach, points on buildings were semi-automatically extracted using an interactive tool (Chapter 11). Regions to be extracted were selected on top of an occlusion image. This tool was used to extract both entire buildings and individual building walls, were the latter were used as input to the solid modeling tool. In the second approach, building footprints were automatically detected (Chapter 12) and points were extracted separately for each detected building footprint. Identification of building footprints was based on the fact that consecutive scan line points sampled on building wall segments were vertically stacked. Interpolated aircraft positions associated with scan lines made it possible to insert *missing echoes*, providing approximated points in cases where reflected pulse energy could not be detected during scanning (Chapter 9). Automatic building extraction was evaluated using ground plans created manually with the assistance of a high-resolution occlusion image. Although the majority of buildings in the study area were successfully extracted automatically, the further separation of extracted building points into individual walls was not implemented. Therefore, semi-automatically extracted ALS points were used as input to the solid modeling tool.

Solid models (Chapter 5) were created from points sampled on building walls. Such models were created both from ALS and TLS points. Models were created directly from point data using a voxelization approach (Chapter 13). Importantly, simulation results for these models demonstrated realistic stress distributions, as will be discussed further in the following section. This section has described how the tools in the proposed workflow were used together in order to achieve the goal of this thesis. In the following section results using the methodology described above are summarized and conclusions are drawn from these results.

20.2 Results

The previous section gave a brief review of how the tools presented in this thesis were used. This section gives conclusions drawn from the achieved results, presented in Part III, with respect to the goal of this thesis. Results are summarized in separate categories and conclusions are presented separately. The following section summarizes the conclusions drawn in this section and discusses the impact of the achieved results in a context extending beyond this thesis.

20.2.1 Acquisition

ALS was used to acquire point data for a study area in central Dublin and the results were presented in Chapter 15. It was shown that multiple flight strip overlap generated significantly improved point sampling on building walls. Although only a few examples of point sampling on building walls were presented, the results of automatic building extraction (Chapter 17) further confirm point sampling on building walls, since a majority of buildings in the study area could be detected based on such sampling. In addition to increased flight strip overlap, flight tracks were oriented at 45 degrees to the major street grid and a low aircraft altitude combined with a wide scan angle were used. However, no direct comparisons were made to demonstrate the effects of these parameters. Instead, validation of linear resolution predictions, on which the choices for these parameters were made, were used to indirectly support these choices.

Given that point data was rapidly acquired for a large number of building walls in a densely built-up urban environment, acquisition using ALS was deemed successful in terms of providing sufficient input to subsequent tools in the proposed workflow. Further, future generations of ALS systems are likely to provide even higher point densities. In particular, increased scan angles and angular resolutions will result in denser sampling in flight strip flanks, providing additional detail on building walls.

The major conclusion from the results presented in Chapter 15 is that current ALS systems have the potential to return significant amounts of data related to vertical details of buildings in urban environments. The successful acquisition of such data is, however, highly dependent on understanding where high quality data is acquired with respect to the flight path, namely in flight strip flanks. Therefore, the flight path should be designed so that flight strip overlap accounts for zones of lower quality vertical data directly beneath the aircraft. Further, flying diagonally to the major street grid reduces the issue of blind spots in the data caused by building shadows. Finally, the methods presented for acquisition of point data for building walls in an urban environment can be readily applied using existing ALS systems and are, thus, simple to incorporate into future projects. As mentioned, the acquired ALS data was used as input to several subsequent tools in the proposed workflow, including visualization, which is discussed next.

20.2.2 Visualization

Visualizations of the ALS data acquired for the study area, in the form of occlusion images, were presented in Chapter 16. Occlusion images were compared to images created with existing imaging techniques. It was demonstrated that building features were visible to a larger extent in occlusion images, compared to other imaging techniques.

The visibility sampling required to create occlusion images relies on the fact that ALS data was acquired with substantial flight strip overlap, such that visibility for pixels is sampled from multiple positions in the sky. However, given that substantial strip overlap is required to acquire three-dimensional data, occlusion images are a powerful visualization tool. Large structures, such as buildings, as well as smaller details, such as roof inventories, are clearly visible. Because buildings are clearly visible in occlusion images, they were used in both semi-automatic and automatic building extraction, as discussed next.

20.2.3 Building Extraction

Two different building extraction approaches were presented in this thesis: (1) *semi-automatic*; and (2) *automatic*. Building extraction results, both semi-automatic and automatic, were presented in Chapter 17.

For semi-automatic building extraction two types of selection were demonstrated: (1) *polygon selection*; and (2) *line selection*. Polygon selection was used to select a horizontal region corresponding to a building footprint. Line selection was used to extract a building wall and points where transformed into an orthogonal coordinate system using the specified line as one of the horizontal axes. This transformation was done in order to avoid aliasing when creating solid models, which will be discussed further below.

While the semi-automatic approach does not scale well to selecting all the buildings of a city, it was sufficient for selecting a few building. In terms of data exploration a semi-automatic tool is more flexible than a fully automatic tool, since regions can be selected based on users' interests, rather than having to be detected by an algorithm. The option to extract any region is highly desirable from a data exploration point of view. Further, it was found that occlusion images provided suitable visualizations for selecting regions of interest.

Automatic building extraction results showed that a majority of building footprints in the study area were detected. The percentage of detected buildings was slightly lower than in other studies reported in literature, but those numbers were in many cases reported for suburban areas, containing regularly shaped, stand-alone buildings.

Interestingly, the presented method for automatic building extraction is fairly simple and uses a straight-forward approach to detect building wall segments in scan lines. Even with such simple means it was possible to detect the majority of buildings in the study area. Moreover, the method proved to be robust in the sense that few false positives were reported as buildings. A key concept in the extraction method is the association of aircraft positions with point data, which is used to process scan line points sequentially. Given that the flight path has significant impact on the acquisition phase, it is not surprising that such information is useful in later processing stages as well. Also, given the relative simplicity of the presented method, there is plenty of room for improvement.

20.2.4 Solid Modeling

Solid modeling results were presented in Chapter 18. Solid models were created from points acquired with both ALS and *Terrestrial Laser Scanning* (TLS). As a consequence of the denser point sampling provided by TLS, solid models created from such data had fewer artefacts in the forms of jagged openings and disjoint structural elements. In both cases test simulations showed that stress distributions behaved as expected, although exact stress levels were not quantitatively analyzed.

Even though solid models were not created for all buildings in the study area, due to shortcomings in automatic building extraction, it was possible to demonstrate the feasibility of the presented solid modeling approach through case studies of individual buildings. It has been demonstrated in this thesis that ALS can be used to acquire the data required to create usable solid models for building simulations. However, improvements are needed in automatic building extraction to fully automate this process. The quality of the created solid models is directly related to the quality of the input points. Since solid models created from densely sampled TLS data were of high quality, it is expected that similar quality can be achieved using future generations of ALS.

The presented solid modeling approach allows solid models to be created rapidly and with a high degree of automation, once points on building walls have been extracted. However, significant amounts of testing remain before the presented approach can be used in real-world scenarios. Many different building types must be tested and errors in stress levels resulting from geometric artefacts and lack of material information must be analyzed. This thesis contributes to future research in this direction by presenting a simple method for converting point data directly into solid models. From these results it is possible to further refine the geometry of the solid models, as well as incorporate material properties. The final section of this chapter gives a summary of the conclusions presented in this chapter and discusses the overall impact of the results of this thesis in a greater context.

20.3 Summary

The most important results in this thesis are the acquisition of point data for building walls with ALS and the conversion of such point data into solid models. In addition, associating aircraft positions with scanned points allowed improved visualization of ALS data as well as a robust method for automatically extracting buildings. Moreover, by inserting missing echoes additional buildings were detected. Solid models were created from semi-automatically extracted ALS points. Hence, solid modeling results have not been demonstrated on a city-scale, but it has been shown that solid models can be created from point data that was acquired on a city-scale.

In a context extending beyond this thesis the creation of solid models on a city-scale has several important applications. Many cities in the world are located in earthquake sensitive areas and simulations are a useful tool for predicting the impact of such natural disasters. Additionally, large engineering projects, such as tunnel excavation, may cause significant stress on buildings. It has been demonstrated in this thesis that it is possible to rapidly and efficiently create solid models on a city-scale. The availability of such models has traditionally been limited by the time and cost spent on manual creation. As a consequence, the use of such models has been limited to individual buildings and has not been applied on a city-scale. The work in this thesis may change the way simulations are used in the sense that larger areas may be studied and a more wide-spread availability of solid models may allow simulations to be applied in situations where solid models were previously unavailable.

Directions for future research based on the material presented in this thesis include finding better automatic building extraction tools. Besides more accurately detecting buildings, such tools should also be able to separate extracted building points into distinct wall planes for solid modeling. Furthermore, the solid modeling approach presented in this thesis could most likely be improved by employing a more sophisticated voxelization technique. Additionally, quantitative comparison of simulation results with manually created solid models is required in order to establish the numerical accuracy of the presented method. Directions for future work are discussed in more detail in the final chapter of this thesis.

Chapter 21

Future Work

The final chapter of this thesis discusses possible directions for future work based on the results and conclusions presented in previous chapters. Clearly, many issues related to city-scale computational modeling remain open for further research and extensions to the methods presented in this thesis are required to remedy these issues.

In terms of acquisition, it has been demonstrated that *Aerial Laser Scanning* (ALS) is a technology capable of rapidly and accurately acquiring vast amount of geometric data for large areas. Future generations of ALS systems are likely to provide even more detailed geometric data. It has been shown that the flight path has a significant impact on the amount of detail acquired for building walls in urban areas. However, while the flight path used in this thesis was based on flight strip overlap and flight track orientation relative to the major street grid of the study area, more sophisticated approaches are possible. For instance, with the availability of an existing city model it becomes possible to take into account the geometry of the study area. Visualization tools could be designed to interactively show approximate point densities over the study area, enabling trade-offs in terms of time and cost versus data quality to be made pre-flight.

A crucial factor in city-scale solid modeling is automation. This thesis has demonstrated that it is possible to create usable solid models from ALS points, but solid models have not been created from automatically extracted building points. While semi-automatic building extraction provides flexibility by relying on users to identify regions of interest, it does not scale well when the number of buildings to be extracted becomes large. Extraction of individual walls from automatically extracted buildings remains an issue and is a crucial part of an effective, city-scale solid modeling approach. Additionally, the automatic building extraction method presented operates on individual scan lines and is, therefore, not able to fully take advantage of information provided from flight strip overlap. A possible extension to the method presented would be to perform automatic building extraction in multiple passes, where an initial pass would gather relevant statistics in an auxiliary digital image. This information would then be available in subsequent passes by mapping points to the auxiliary image.

The solid models created in this thesis have been shown to behave plausibly in terms of simulated stress distributions, although errors in stress levels have not been quantified. Continued research is required to produce quantitative error estimates, which are required in order to guarantee safety in real-world scenarios. Further, the regular nature of voxel grids makes it difficult to accurately model curved shapes, such as arched windows. An interesting possibility for remedying this issue would be to initially create solid models using standard voxelization and, thereafter, adjust lattice vertices to fit closer to the point data. This would take advantage of the regular topology of voxel grids, while providing additional flexibility in terms of allowing irregularly shaped voxels. Additionally, in order to realistically simulate stress, incorporation of material properties is most likely required. A possible solution is to derive material properties from color images. However, this is a challenging task given the many different materials that buildings are constructed from and the fact that color images may be acquired under varying lighting conditions. Another interesting possibility would be to derive material properties from full-waveform ALS data, where differences in peak characteristics may provide useful information. Moreover, assuming that solid models can be automatically created for an entire city, challenges arise in the area of communicating simulation results on a city-scale. While individual building simulations can be visualized using standard techniques, communicating simulation results for an entire city is more challenging.

Finally, the results presented in this thesis are encouraging and clearly demonstrate the high potential of solid modeling from ALS data. However, as discussed above, many of the tools presented in this thesis could be improved through continued research. A few examples of future research directions have been provided in this concluding chapter and hopefully the work presented in this thesis will be an inspiration for further research.

Bibliography

[AB02] Alharty A., Bethel J.: Heuristic filtering and 3d feature extraction from Lidar data. In *International Archives of Photogrammetry and Remote Sensing, XXXIV, 3A* (2002), pp. 23–28.

[Ack99] Ackermann F.: Airborne laser scanning – present status and future expectations. *ISPRS Journal of Photogrammetry & Remote Sensing 54* (1999), 64–67.

[ACY93] A.Kaufman, Cohen D., Yagel R.: Volume Graphics. *IEEE Computer 26* (1993), 51–64.

[AMM92] A.Pommert, M.Bomans, M.Riemer: Volume Visualization in Medicine: Techniques and Applications. *Focus on Scientific Visualization* (1992), 41–71.

[AMR05] Andersen H. E., McGaughey R. J., Reutebuch S. E.: Estimating forest canopy fuel parameters using LIDAR data. *Remote Sensing of Environment 94* (2005), 441–449.

[AP05] Andrews L. C., Phillips R. L.: *Laser beam propagation through random media.* MIT Press, 2005.

[ARDA06] Arnold N. S., Rees W. G., Devereux B. J., Amable G. S.: Evaluating the potential of high-resolution airborne LiDAR data in glaciology. *International Journal of Remote Sensing 27*, 5-6 (2006), 1233–1251.

[ASP03] ASPRS: *LIDAR Data Exchange Format Standard.* Tech. rep., ASPRS, 2003.

[Axe99] Axelsson P.: Processing of laser scanner data - algorithms and applications. *ISPRS Journal of Photogrammetry & Remote Sensing 54*, 2-3 (July 1999), 138–147.

[Bal99a] Baltsavias E. P.: Airborne laser scanning - an introduction and overview. *ISPRS Journal of Photogrammetry & Remote Sensing 54*, 2-3 (July 1999), 68–82.

[Bal99b] Baltsavias E. P.: Airborne laser scanning: basic relations and formulas. *ISPRS Journal of Photogrammetry & Remote Sensing 54*, 2-3 (July 1999), 199–214.

[Bal99c] Baltsavias E. P.: Airborne laser scanning: existing systems and firms and other resources. *ISPRS Journal of Photogrammetry & Remote Sensing 54*, 2-3 (July 1999), 164–198.

[Bal99d] Baltsavias E. P.: A comparison between photogrammetry and laser scanning. *ISPRS Journal of Photogrammetry & Remote Sensing 54*, 2-3 (July 1999), 83–94.

[BFK84] Boehm W., Farin G., Kahmann J.: A survey of curve and surface methods in CAGD. *Computer Aided Geometric Design 1*, 1 (1984), 1–60.

[BH05] Böhm J., Haala N.: Efficient Integration of Aerial and Terrestrial Laser Data for Virtual City Modeling Using LASERMAPS. In *IAPRS Vol. 36 Part 3/W19 ISPRS Workshop Laser scanning* (2005), pp. 192–197.

[BH07] Becker S., Haala N.: Combined Feature Extraction for Facade Reconstruction. In *Proceedings of the ISPRS Workshop* (2007), vol. XXXVI, Part 3/W52, IAPRS, pp. 44–49.

[BHF01] Brenner C., Haala N., Fritsch D.: Towards fully automated 3D city model generation. In *Automatic Extraction of Man-Made Objects from Aerial and Space Images III* (2001), pp. 47–57.

[Bli77] Blinn J. F.: Models of light reflection for computer synthesized pictures. In *SIGGRAPH '77: Proceedings of the 4th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1977), ACM, pp. 192–198.

[BM92] Besl P. J., McKay N. D.: A Method for Registration of 3-D Shapes. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 2 (1992), 239–256.

[BM99] Baillard C., Maitre H.: 3-D Reconstruction of Urban Scenes from Aerial Stereo Imagery: A Focusing Strategy. *Computer Vision and Image Understanding 76*, 3 (December 1999), 244–258.

[Bre00] Brenner C.: Towards fully automatic generation of city models. *International Archives of Photogrammetry & Remote Sensing 33*, B3 (2000), 85–92.

[Bre10] Brenner C.: *Airborne and terrestrial laser scanning.* Whittles Publishing, 2010, ch. Chapter 5 - Building Extraction.

[Bun05] Bunnell M.: *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation.* Addison-Wesley Professional, 2005, ch. Dynamic ambient occlusion and indirect lighting, pp. 223–233.

[CC08] Chen J., Chen B.: Architectural Modeling from Sparsely Scanned Range Data. *Int. J. Comput. Vision 78*, 2-3 (2008), 223–236.

[CK90] Cohen D., Kaufman A.: Scan Conversion Algorithms for Linear and Quadratic Objects. *Volume Visualization* (1990), 280–301.

[CM91] Chen Y., Medioni G.: Object modeling by registration of multiple range images. In *IEEE International Conference on Robotics and Automation* (1991), pp. 2724–2729.

[CM92] Chen Y., Medioni G.: Object modelling by registration of multiple range images. *Image Vision Comput. 10*, 3 (1992), 145–155.

[CMHB03] Cobby D. M., Mason D. C., Horritt M. S., Bates P. D.: Two-dimensional hydraulic flood modelling using a finite-element mesh decomposed according to vegetation and topographic features derived from airborne scanning laser altimetry. *Hydrol. Process. 17* (2003), 1979–2000.

[CO97] Cohen-Or D.: Exact antialiasing of textured terrain models. *The Visual Computer 13* (1997), 184–199.

[CR07] Clode S., Rottensteiner F.: Detection and vectorization of roads from lidar data. *Photogramm. Eng. Remote Sens. 73*, 5 (2007), 517–536.

[dBvKOS00] de Berg M., van Kreveld M., Overmars M. H., Schwarzkopf O.: *Computational Geometry, Algorithms and Applications*, second ed. Springer-Verlag, 2000.

[DHS01] Duda R. O., Hart P. E., Stork D. G.: *Pattern Classification*, second ed. Wiley, New York, 2001.

[DKW94] D.Cohen, Kaufman A., Wang Y.: Generating a Smooth Voxel-Based Model from an Irregular Polygon Mesh. *The Visual Computer 10* (1994), 295–305.

[DP08] Dorninger P., Pfeifer N.: A Comprehensive Automated 3D Approach for Building Extraction, Reconstruction, and Regularization from Airborne Laser Scanning Point Clouds. *Sensors 8* (2008), 7323–7343.

[DST92] Dillencourt M. B., Samet H., Tamminen M.: A general approach to connected-component labeling for arbitrary image representations. *J. ACM 39*, 2 (1992), 253–280.

[Dug04] Duggal S. K.: *Surveying Volume 2*. Tata McGraw-Hill Publishing Company Limited, 2004.

[EB02] Elaksher A. F., Bethel J. S.: Building Extraction Using Lidar Data. In *ASPRS-ACSM Annual Conference and FIG XXII Congress* (2002), pp. 22–26.

[Elb10] Elberink S. O.: *Acquisition of 3D topography - Automated 3D road and building reconstruction using airborne laser scanner data*. PhD thesis, University of Twente, 2010.

[ELF97] Eggert D., Lorusso A., Fisher R.: Estimating 3D rigid body transformations: a comparison of four major algorithms. *Mach. Vision and Appl. 9*, 5–6 (1997), 272–290.

[Far90] Farin G.: *Curves and Surfaces for Computer-Aided Geometric Design*, second ed. Academic Press, 1990.

[Fil03] Filin S.: Analysis and implementation of a laser strip adjustment model. *International Archives of Photogrammetry & Remote Sensing 34*, 3/W13 (2003), 65–70.

[Flo99] Flood M.: Commercial development of airborne laser altimetry: a review of the commercial instrument market and its projected growth. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 32*, 3W/4 (November 1999), 13–20.

[Flo01a] Flood M.: Laser altimetry: From science to commercial lidar mapping. *PE & RS 67*, II (2001), 1209–1217.

[Flo01b] Flood M.: Lidar activities and research priorities in the commercial sector. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 34*, 3W/4 (2001), 678–684.

[FN07] Forlani G., Nardinocchi C.: Adaptive Filtering of Aerial Laser Scanning Data. In *Laser Scanning and SilviLaser* (September 2007), ISPRS, pp. 130–135.

[FNSZ06] Forlani G., Nardinocchi C., Scaioni M., Zingaretti P.: Complete Classification of Raw LIDAR Data and 3D Reconstruction of Buildings. *Pattern Analysis Applications 8*, 4 (2006), 357–374.

[För99] Förstner W.: 3D-city models: Automatic and semiautomatic acquisition methods. In *Photogrammetric Week 99* (1999), Fritsch D., Spiller R., (Eds.), Wichmann Verlag, pp. 291–303.

[FV04] Filin S., Vosselman G.: Adjustment of airborne laser altimetry strips. *International Archives of Photogrammetry & Remote Sensing 35*, B3 (2004), 5.

[FZ03] Früh C., Zakhor A.: Constructing 3D City Models by Merging Aerial and Ground Views. *IEEE Computer Graphics & Applications 23*, 6 (2003), 52–61.

[FZ04] Früh C., Zakhor A.: An Automated Method for Large-Scale, Ground-Based City Model Acquisition. *International Journal of Computer Vision 60*, 1 (2004), 5–24.

[GLS03] Geist T., Lutz E., Stötter J.: Airborne laser scanning technology and its potential for applications in glaciology. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 34*, 3/W13 (2003), 101–106.

[GMGP05] Gelfand N., Mitra N. J., Guibas L. J., Pottmann H.: Robust Global Registration. In *Symp. Geometry Processing* (2005), pp. 197–206.

[GS00] Geibel R., Stilla U.: Segmentation of laser altimeter data for building reconstruction: different procedures and comparison. *International Archives of Photogrammetry & Remote Sensing 33*, B3 (2000), 326–334.

[GW02] Gonzalez R. C., Woods R. E.: *Digital Image Processing*, second ed. Prentice-Hall, 2002.

[HA97] Haala N., Anders K.-H.: Acquisition of 3D urban models by analysis of aerial images, digital surface models and existing 2D building information. In *Proc. SPIE Conf. Integrating Photogrammetric Techniques with Scene Analysis and Machine Vision III* (1997), SPIE, pp. 212–222.

[Haa99] Haala N.: Combining Multiple Data Sources for Urban Data Acquisition. In *Photogrammetric Week '99* (1999), pp. 329–340.

[HB99] Haala N., Brenner C.: Extraction of Buildings and Trees in Urban Environments. *ISPRS Journal of Photogrammetry & Remote Sensing 54*, 2-3 (July 1999), 130–137.

[HBA98] Haala N., Brenner C., Anders K.-H.: 3D Urban GIS from Laser Altimetry and 2D Map Data. *International Archives of Photogrammetry & Remote Sensing 32*, 3 (1998), 339–346.

[HBK06] Haala N., Becker S., Kada M.: Cell Decomposition for the Generation of Building Models at Multiple Scales. *Symposium Photogrammetric Computer Vision 36*, Part III (2006), 19–24.

[HHL*04] Hyyppä J., Hyyppä H., Litkey P., Yu X., Haggrén H., Rönnholm P., Pyysalo U., Pitkänen J., Maltamo M.: Algorithms and methods of airborne laser-scanning for forest measurements. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 36*, 8/W2 (2004), 82–88.

[Hil09] Hill L. L.: *Georeferencing.* MIT Press, 2009.

[Hir08] Hirschmüller H.: Stereo processing by semi-global matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell. 30*, 2 (2008), 328–341.

[HMWJ97] Heipke C., Mayer H., Wiedemann C., Jamet O.: Evaluation of automatic road extraction. *International Archives of Photogrammetry & Remote Sensing 32*, 3-2W3 (1997), 47–56.

[HO03] Hamill J., O'Sullivan C.: Virtual Dublin - A Framework for Real-Time Urban Simulation. *Winter School of Computer Graphics 11*, 1 (2003), 221–225.

[Hon11] Hong L. T.: *Automatic Generation of Solid Models of Building Facades from LIDAR Data for Computational Modelling.* PhD thesis, University College Dublin, 2011.

[Hop94] Hoppe H.: *Surface Reconstruction from Unorganized Points.* PhD thesis, University of Washington, 1994.

[HP07] Höfle B., Pfeifer N.: Correction of laser scanning intensity data: Data and model-driven approaches. *ISPRS Journal of Photogrammetry & Remote Sensing 62*, 6 (December 2007), 415–433.

[HPAD06] Hegeman K., Premože S., Ashikhmin M., Drettakis G.: Approximate ambient occlusion for trees. In *I3D '06: Proceedings of the 2006 symposium on Interactive 3D graphics and games* (New York, NY, USA, 2006), ACM, pp. 87–92.

[HR96] Hoffmann C. M., Rossignac J. R.: A road map to solid modeling. *IEEE Transactions on Visualization and Computer Graphics 2*, 1 (1996), 3–10.

[HR05] Haile A. T., Rientjes T. H. M.: Effects of LIDAR DEM resolution in flood modelling: a model sensitivity study for the city of Tegucigalpa, Honduras. In *Laser Scanning 2005* (September 2005), pp. 168–173.

[HS08] Hebel M., Stilla U.: Pre-classification of points and segmentation of urban objects by scan line analysis of airborne LiDAR data. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 37*, B3a (2008), 105–110.

[HW97] Hug C., Wehr A.: Detecting and identifying topographic objects in laser altimeter data. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 32*, 3-4/W2 (1997), 19–26.

[HWEK06] Hollaus M., Wagner W., Eberhöfer C., Karel W.: Accuracy of large-scale canopy heights derived from LiDAR data under operational constraints in a complex alpine environment. *ISPRS Journal of Photogrammetry & Remote Sensing 6*, 5 (2006), 323–338.

[HWK05] Hollaus M., Wagner W., Kraus K.: Airborne laser scanning and usefulness for hydrological models. *Advances in Geosciences 5* (2005), 57–63.

[HYN03] Hu J., You S., Neumann U.: Approaches to Large-Scale Urban Modeling. *IEEE Computer Graphics & Applications 23*, 6 (2003), 62–69.

[HYNP04] Hu J., You S., Neumann U., Park K. K.: Building Modeling from LiDAR and Aerial Imagery. In *ASPRS'04* (May 2004), pp. 1–6.

[Jel92] Jelalian A. V.: *Laser Radar Systems.* Artech House, London, 1992.

[Jen09] Jensen H. W.: *Realistic Image Synthesis Using Photon Mapping.* A. K. Peters, Ltd., Natick, MA, USA, 2009.

[Jon96] Jones M.: The Production of Volume Data from Triangular Meshes Using Voxelization. *Computer Graphics Forum 15* (1996), 311–318.

[JS06] Jutzi B., Stilla U.: Range determination with waveform recording laser systems using a Wiener Filter. *ISPRS Journal of Photogrammetry & Remote Sensing 61*, 2 (2006), 95–107.

[Kag04] Kager H.: Discrepancies between overlapping laser scanning strips - simultaneous fitting of aerial laser scanner strips. In *International Archives of Photogrammetry and Remote Sensing, XXXV, B/1* (2004), pp. 555–560.

[KAHS05] Kaasalainen S., Ahokas E., Hyyppä J., Suomalainen J.: Study of surface brightness from backscattered intensity: calibration of laser data. *IEEE Geoscience and Remote Sensing Letters 2*, 3 (2005), 255–259.

[Kau87] Kaufman A.: Efficient Algorithms for 3D Scan-Conversion of Parametric Curves, Surfaces, and Volumes. *Computer Graphics 21*, 4 (1987), 171–179.

[KBH06] Kazhdan M., Bolitho M., Hoppe H.: Poisson surface reconstruction. In *SGP '06: Proceedings of the fourth Eurographics symposium on Geometry processing* (Aire-la-Ville, Switzerland, 2006), Eurographics Association, pp. 61–70.

[Kev03] Kevany M. J.: GIS in the world trade center attack - trial by fire. *Computers, Environment and Urban Systems 27*, 6 (2003), 571–583.

[KHE96] Kilian J., Haala N., Englich M.: Capture and Evaluation of Airborne Laser Scanner Data. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 31*, 3 (1996), 383–388.

[KL05] Kontkanen J., Laine S.: Ambient occlusion fields. In *I3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games* (New York, NY, USA, 2005), ACM, pp. 41–48.

[Klu11] Kluckner S.: *Semantic Interpretation of Digital Aerial Images Utilizing Redundancy, Appearance and 3D Information.* PhD thesis, Graz University of Technology, Institute for Computer Graphics and Vision, 2011.

[KP98] Kraus K., Pfeifer N.: Determination of terrain models in wooded areas with airborne laser scanner data. *ISPRS Journal of Photogrammetry & Remote Sensing 53*, 4 (1998), 193–203.

[KPT99] Karabassi E.-A., Papaioannou G., Theoharis T.: A fast depth-buffer-based voxelization algorithm. *J. Graph. Tools 4* (December 1999), 5–10.

[Kra07] Kraus K.: *Photogrammetry - Geometry from Images and Laser Scans*, second ed. de Gruyter, Vienna, Austria, 2007.

[KS86] Kaufman A., Shimony E.: 3D Scan Conversion Algorithms for Voxel-Based Graphics. In *Proc. ACM 1986 Workshop on Interactive 3D Graphics* (1986), pp. 45–76.

[Lam60] Lambert J.: *Photometria Sive de Mensura et Gradibus Luminus, Colorum et Umbrae.* Eberhard Klett, 1760.

[Lat02] Latypov D.: Estimating relative lidar accuracy information from overlapping flight lines. *ISPRS Journal of Photogrammetry & Remote Sensing 56* (2002), 236–245.

[Lat05] Latypov D.: Effects of laser beam alignment tolerance on lidar accuracy. *ISPRS Journal of Photogrammetry & Remote Sensing 59*, 6 (November 2005), 361–368.

[LB00] Langer M. S., Buelthoff H. H.: Depth discrimination from shading under diffuse lighting. *Perception 29*, 6 (2000), 649–660.

[Lin93] Lindenberger J.: *Laser-Profilmenssungen zur topographischen Gelandeaufnahme.* PhD thesis, Deutsche Geodätische Kommission bei der Bayerischen Akademie der Wissenschaften, Reihe C, Heft Nr. 400, 1993.

[LLM06] Li M., Langbein F. C., Martin R. R.: Constructing regularity feature trees for solid models. *Geom. Modeling and Processing 4077* (2006), 267–286.

[LP06] Laefer D., Pradhan A.: Evacuation Route Selection Based on Tree-Based Hazards Using LiDAR & GIS. *Journal of Transportation Engineering 132*, 4 (2006), 312–320.

[LR07] Laefer D. F., Ryan A.: *UREKA participation in GUILD Alison Ryan Summer 2007.* Tech. rep., Report for Science Foundation Ireland, 2007.

[LWW08] Lipp M., Wonka P., Wimmer M.: Interactive visual editing of grammars for procedural architecture. *ACM Trans. Graph. 27*, 3 (2008), 1–10.

[MB04] Melzer T., Briese C.: Extraction and modeling of power lines from als point clouds. In *Proceedings of* 28th *Workshop of the Austrian Association for Pattern Recognition (ÖAGM)* (2004), pp. 47–54.

[MCUP02] Matas J., Chum O., Urban M., Pajdla T.: Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. In *In British Machine Vision Conference* (2002), pp. 384–393.

[MGPG04] Mitra N. J., Gelfand N., Pottmann H., Guibas L.: Registration of point cloud data from a geometric optimization perspective. In *SGP '04* (2004), pp. 22–31.

[MHA*10] Matikainen L., Hyyppä J., Ahokas E., Markelin L., Kaartinen H.: Automatic Detection of Buildings and Changes in Buildings for Updating of Maps. *Remote Sens. 2*, 5 (2010), 1217–1248.

[MHC*08] Morvan Y., Hinks T., Carr H., Laefer D. F., O'Sullivan C., Morrish W. S.: Post Facto Registration Tools for Urban Modelling. In *Eurographics 2008* (April 2008), pp. 215–218.

[MHH03] Matikainen L., Hyyppä J., Hyyppä H.: Automatic detection of buildings from laser scanner data for map updating. *International Archives of Photogrammetry & Remote Sensing 34*, 3/W13 (2003), 218–224.

[MHH*08] Mandlburger G., Hauer C., Höfle B., Habersack H., Pfeifer N.: Optimisation of LiDAR derived terrain models for river flow modelling. *Hydrol. Earth Syst. Sci. Discuss. 5* (2008), 3605–3638.

[MKH07] Matikainen L., Kaartinen H., Hyyppä J.: Classification Tree Based Building Detection From Laser Scanner and Aerial Image Data. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 3*, 3/W52 (2007), 280–287.

[MMH05] Monteiro L. S., Moore T., Hill C.: What is the accuracy of DGPS? *The Journal of Navigation 58* (2005), 207–225.

[MN03] Mitra N. J., Nguyen A.: Estimating surface normals in noisy point cloud data. In *Proceedings of the nineteenth annual symposium on Computational geometry* (New York, NY, USA, 2003), SCG '03, ACM, pp. 322–328.

[MPBM03] Matusik W., Pfister H.-P., Brand M., McMillan L.: A data-driven reflectance model. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), ACM, pp. 759–769.

[MPD06] Makadia A., Patterson IV A., Daniilidis K.: Fully Automatic Registration of 3D Point Clouds. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2006), pp. 1297–1304.

[MRC*07] Munoz C., Rodriguez A., Comeron A., Batet O., Garcia D., Rocadenbosch F., Sicard M.: Speed measurements with a continuous wave Lidar prototype. In *International Geoscience and Remote Sensing Symposium (IGARSS-07), Conference on Atmospheric Lidar* (July 2007), IEEE International Geoscience and Remote Sensing Symposium (IGARSS-07), pp. 2775–2778.

[MT00] Morgan M., Tempfli K.: Automatic building extraction from airborne laser scanning data. In *International Archives of Photogrammetry and Remote Sensing, XXXIII, B3* (2000), pp. 616–623.

[MV99] Maas H.-G., Vosselman G.: Two Algorithms for Extracting Building Models from Raw Laser Altimetry Data. *ISPRS Journal of Photogrammetry & Remote Sensing 54*, 2-3 (July 1999), 153–163.

[MVW*06] Müller P., Vereenooghe T., Wonka P., Paap I., Van Gool L.: Procedural 3D Reconstruction of Puuc Buildings in Xkipché. In *Eurographics Symposium on Virtual Reality, Archeology and Cultural Heritage (VAST)* (2006), EG, pp. 139–146.

[MWH*06] Müller P., Wonka P., Haegler S., Ulmer A., Van Gool L.: Procedural modeling of buildings. *ACM Trans. Graph. 25*, 3 (2006), 614–623.

[MWK09] Möser S., Wahl R., Klein R.: Out-of-core topologically constrained simplification for city modeling from digital surface models. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 37*, 5/W1 (February 2009), 1–7.

[NDM05] Ngan A., Durand F., Matusik W.: Experimental Analysis of BRDF Models. In *Rendering Techniques* (2005), pp. 117–126.

[OTDS04] Oda K., Takano T., Doihara T., Shibasaki R.: Automatic building extraction and 3-D city modeling from lidar data based on Hough transformation. *International Archives of Photogrammetry & Remote Sensing 35*, B3 (2004), 1–4.

[PB07] Pfeifer N., Briese C.: Geometrical aspects of airborne laser scanning and terrestrial laser scanning. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 36*, part 3/W52 (2007), 311–310.

[PDHF07] Pfeifer N., Dorninger P., Haring A., Fan H.: Investigating terrestrial laser scanning intensity data: Quality and functional relations. In 8^{th} *Conf. on O3D* (July 2007), pp. 328–337.

[Pho75] Phong B. T.: Illumination for computer generated pictures. *Commun. ACM 18*, 6 (1975), 311–317.

[PHYH06] Pottmann H., Huang Q. X., Yang Y. L., Hu S. M.: Geometry and convergence analysis of algorithms for registration of 3D shapes. *Int. J. Computer Vision 67*, 3 (2006), 277–296.

[PM01] Parish Y. I. H., Müller P.: Procedural modeling of cities. In *SIGGRAPH '01* (New York, NY, USA, 2001), ACM Press, pp. 301–308.

[PSTA05] Persson A., Söderman U., Töpel J., Ahlberg S.: Visualization and analysis of full-waveform airborne laser scanner data. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 36*, 3/W19 (2005), 103–108.

[RB02] Rottensteiner F., Briese C.: A new method for building extraction in urban areas from high-resolution LIDAR data. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 34*, 3A (2002), 295–301.

[RB03] Rottensteiner F., Briese C.: Automatic Generation of Building Models from Lidar Data and the Integration of Aerial Images. *International Archives of Photogrammetry & Remote Sensing 34*, 3/W13 (2003), 174–180.

[Ree01] Rees W. G.: *Physical Principles of Remote Sensing*, second ed. Cambridge University Press, Cambridge, 2001.

[Req88] Requicha A.: *CAD Based Programming for Sensory Robots.* Springer Verlag, 1988, ch. Solid modelling: a 1988 update, pp. 3–22.

[Req92] Requicha A.: Solid modeling and beyond. *IEEE Computer Graphics & Applications (Special issue on CAGD) 12*, 5 (September 1992), 31–44.

[Res09] Reshetyuk Y.: *Self-calibration and direct georeferencing in terrestrial laser scanning.* PhD thesis, Royal Institute of Technology, Sweden, 2009.

[RHHP08] Rutzinger M., Höfle B., Hollaus M., Pfeifer N.: Object-based point cloud analysis of full waveform airborne laser scanning data for urban vegetation classification. *Sensors 8* (2008), 4505–4528.

[RJ99] Rangachar K., Jain R. C.: *Computer Vision Principles.* IEEE Computer Society Press, 1999.

[RKM08] Ressl C., Kager H., Mandlburger G.: Quality checking of als projects using statistics of strip differences. In *International Archives of Photogrammetry and Remote Sensing, XXXVII* (2008), pp. 253–260.

[RL00] Rusinkiewicz S., Levoy M.: QSplat: a multiresolution point rendering system for large meshes. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 343–352.

[RL01] Rusinkiewicz S., Levoy M.: Efficient Variants of the ICP Algorithm. In *Proc. of the* 3^{rd} *Intl. Conf. on 3D Digital Imaging and Modeling* (2001), pp. 145–152.

[RMAC03] Reutebuch S., McGaughey R., Andersen H., Carson W.: Accuracy of a high-resolution lidar terrain model under a conifer forest canopy. *Canadian Journal of Remote Sensing 29* (2003), 527–535.

[Rot03] Rottensteiner F.: Automatic Generation of High-Quality Building Models from Lidar Data. *IEEE Computer Graphics & Applications 23*, 6 (November 2003), 42–50.

[RP66] Rosenfeld A., Pfaltz J. L.: Sequential Operations in Digital Picture Processing. *Journal of the Association for Computing Machinery 13* (1966), 471–494.

[RP04] Rath S., Pasche E.: Hydrodynamic floodplain modeling based on high-resolution LIDAR measurements. In 6^{th} *International Conference on Hydroinformatics* (2004), pp. 1–8.

[RPZ02] Ren L., Pfister H., Zwicker M.: Object Space EWA Surface Splatting: A Hardware Accelerated Approach to High Quality Point Rendering. *Computer Graphics Forum 21*, 3 (September 2002), 461–470.

[RR84] Rossignac J., Requicha A.: Constant-Radius Blending in Solid Modeling. *ASME Computers in Mechanical Engineering (CIME) 3* (1984), 65–73.

[RR99] Rossignac J., Requicha A.: *Encyclopedia of Electrical and Electronics Engineering.* John Wiley and Sons, 1999, ch. Solid modeling.

[RTCK05] Rottensteiner F., Trinder J., Clode S., Kubik K.: Using the DempsterShafer method for the fusion of LIDAR data and multi-spectral images for building detection. *Information Fusion 6*, 4 (2005), 283–300.

[RTCK07] Rottensteiner F., Trinder J., Clode S., Kubik K.: Building detection by fusion of airborne laser scanner data and multi-spectral images: Performance evaluation and sensitivity analysis. *ISPRS Journal of Photogrammetry & Remote Sensing 62*, 2 (June 2007), 135–149.

[Sab68] Sabin M. A.: *The use of potential surfaces for numerical geometry.* B.A.C. Weybridge, 1968.

[Sam90] Samet H.: *Application of Spatial Data Structures.* Addison-Wesley, Reading, MA, 1990.

[SB05] Schuhmacher S., Böhm J.: Georeferencing of Terrestrial Laserscanner Data for Applications in Architectural Modelling. *International Archives of Photogrammetry & Remote Sensing 36*, 5/W17 (2005), 1–7.

[SD03] Sohn G., Dowman I.: Building extraction using lidar DEMs and Ikonos images. *International Archives of Photogrammetry & Remote Sensing 34*, 3/W13 (2003), 174–180.

[SD07] Sohn G., Dowman I.: Data fusion of high-resolution satellite imagery and LiDAR data for automatic building extraction. *ISPRS Journal of Photogrammetry & Remote Sensing 62*, 1 (May 2007), 43–63.

[SE03] Schneider P. J., Eberly D. H.: *Geometric tools for computer graphics.* Elsevier, 2003.

[Sed85] Sederberg T. W.: Piecewise Algebraic Surface Patches. *Computer Aided Geometric Design 2* (1985), 53–59.

[Ser82] SERRA J.: *Image Analysis and Mathematical Morphology.* Academic Press, 1982.

[SFF91] STYTZ M. R., FRIEDER G., FRIEDER O.: Three-dimensional Medical Imaging: Algorithms and Computer Systems. *ACM Computing Surveys 23* (1991), 421–499.

[SG00] STAUFFER C., GRIMSON W. E. L.: Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 8 (2000), 747–757.

[Shi01] SHIODE N.: 3D Urban Models: Recent Developments in the Digital Modeling of Urban Environments in Three-Dimensions. *GeoJournal 52*, 3 (2001), 263–269.

[Sho85] SHOEMAKE K.: Animating rotation with quaternion curves. *SIGGRAPH Comput. Graph. 19*, 3 (1985), 245–254.

[SHS03] SUZUKI K., HORIBA I., SUGIE N.: Linear-time connected-component labeling based on sequential local operations. *Computer Vision and Image Understanding 89*, 1 (January 2003), 1–23.

[SL03] STEVENS B. L., LEWIS F. L.: *Aircraft Control and Simulation*, second ed. Wiley-IEEE, 2003.

[Smi78] SMITH A. R.: Color Gamut Transform Pairs. *Computer Graphics 12*, 3 (1978), 376–383.

[SMS04] SCHWEIER C., MARKUS M., STEINLE E.: Simulation of earthquake caused building damages for the development of fast reconnaissance techniques. *Natural Hazards and Earth System Sciences 4* (2004), 285–293.

[SMS05] SCHWALBE E., MAAS H.-G., SEIDEL F.: 3D building model generation from airborne laser scanner data using 2D GIS data and orthogonal point cloud projections. *International Archives of Photogrammetry & Remote Sensing 36*, 3/W19 (2005), 209–214.

[SP86] SEDERBERG T. W., PARRY S. R.: Free-form deformation of solid geometric models. *SIGGRAPH Comput. Graph. 20*, 4 (1986), 151–160.

[SS01] Shapiro L. G., Stockman G. C.: *Computer Vision.* Prentice Hall, 2001.

[ST88] Samet H., Tamminen M.: Efficient component labeling of images of arbitrary dimension represented by linear bintrees. *IEEE Trans. Pattern Anal. Mach. Intell. 10*, 4 (1988), 579–586.

[Ste86] Sternberg S. R.: Grayscale Morphology. *Computer Vision, Graphics, and Image Processing 35* (1986), 333–355.

[Ste01] Steele C.: A critical review of some traffic noise prediction models. *Applied Acoustics 62*, 3 (2001), 271–287.

[Str88] Strang G.: *Linear Algebra and Its Applications.* Harcourth Brace Jovanovich, 1988.

[SV03] Sithole G., Vosselman G.: Automatic Structure Detection in a Point Cloud of an Urban Landscape. In *Proc.* 2^{nd} *GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas* (2003), URBAN2003, pp. 67–71.

[SV04] Sithole G., Vosselman G.: Experimental comparison of filter algorithms for bare-earth extraction from airborne laser scanning point clouds. *ISPRS Journal of Photogrammetry & Remote Sensing 59*, 1-2 (August 2004), 85–101.

[UMF82] Ulaby F. T., Moore R. K., Fung A. K.: *Microwave remote sensing: Active and passive. Volume 2 - Radar remote sensing and surface scattering and emission theory.* Addison-Wesley Publishing Co., 1982.

[VD01] Vosselman G., Dijkman S.: 3D building reconstruction from point clouds and ground plans. *International Archives of Photogrammetry & Remote Sensing 34*, 3/W4 (2001), 37–43.

[VGSR04] Vosselman G., Gorte B. G. H., Sithole G., Rabbani T.: Recognising structure in laser scanner point clouds. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 46*, 8/W2 (2004), 33–38.

[VKH06] Verma V., Kumar R., Hsu S.: 3D Building Detection and Modeling from Aerial LIDAR Data. In *CVPR '06: Proceedings of the 2006 IEEE Computer*

Society Conference on Computer Vision and Pattern Recognition (Washington, DC, USA, 2006), IEEE Computer Society, pp. 2213–2220.

[VM01] Vosselman G., Maas H.-G.: Adjustment and filtering of raw laser altimetry data. In *OEEPE Workshop on Airborne Laserscanning and Interferometric SAR for Digital Elevation Models* (2001), pp. 62–72.

[VM10] Vosselman G., Maas H.-G. (Eds.): *Airborne and terrestrial laser scanning.* Whittles Publishing, 2010.

[VMY04] Vu T. T., Matsuoka M., Yamazaki F.: Employment of LIDAR in Disaster Assessment. In *Proc.* 2^{nd} *International Workshop on Remote Sensing for Post-Disaster Response* (October 2004), MCEER, pp. 1–6.

[Vos00] Vosselman G.: Slope based filtering of laser altimetry data. *International Archives of Photogrammetry & Remote Sensing 33*, B4 (2000), 958–964.

[Vos08] Vosselman G.: Analysis of planimetric accuracy of airborne laser scanning surveys. *International Archives of Photogrammetry & Remote Sensing 34*, part 3A (2008), 99–104.

[VS03] Vögtle T., Steinle E.: On the quality of object classification and automated building modeling based on laserscanning data. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 34*, 3W13 (2003), 149–155.

[War92] Ward G. J.: Measuring and modeling anisotropic reflection. *SIGGRAPH Comput. Graph. 26*, 2 (1992), 265–272.

[Wat00] Watt A.: *3D Computer Graphics*, third ed. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000.

[WK93] Wang S., Kaufman A.: Volume Sampled Voxelization of Geometric Primitives. In *Proc. of the Visualization '93 Conference* (1993), IEEE Computer Society Press, pp. 78–85.

[WK94] Wang S., Kaufman A.: Volume-Sampled 3D Modeling. *IEEE Computer Graphics & Applications 14* (1994), 26–32.

[WL99] Wehr A., Lohr U.: Airborne laser scanning - an introduction and overview. *ISPRS Journal of Photogrammetry & Remote Sensing 54*, 2-3 (July 1999), 68–82.

[WMDS10] Wilkinson B. E., Mohamed A. H., Dewitt B. A., Seedahmed G. A.: A Novel Approach to Terrestrial LiDAR Georeferencing. *Photogrammetric Engineering & Remote Sensing 76*, 6 (2010), 1–8.

[WUD*06] Wagner W., Ullrich A., Ducic V., Melzer T., Studnicka N.: Gaussian decomposition and calibration of a novel small-footprint full-waveform digitising airborne laser scanner. *ISPRS Journal of Photogrammetry & Remote Sensing 60*, 2 (April 2006), 100–112.

[WUM*04] Wagner W., Ullrich A., Melzer T., Briese C., Kraus K.: From single-pulse to full-waveform airborne laser scanners: potential and practical challenges. *International Archives of Photogrammetry, Remote Sensing & Spatial Information Sciences 35*, B3 (2004), 201–206.

[WWSR03] Wonka P., Wimmer M., Sillion F., Ribarsky W.: Instant architecture. *ACM Trans. Graph. 22*, 3 (2003), 669–677.

[YHNF03] You S., Hu J., Neumann U., Fox P.: Urban Site Modeling From LiDAR. In *Proc. of* 2^{nd} *International Workshop on Computer Graphics and Geometric Modeling (CGGM'2003)* (May 2003), CGGM, pp. 579–588.

[ZIK98] Zhukov S., Inoes A., Kronin G.: An ambient light illumination model. In *Rendering Techniques '98* (1998), Drettakis G., Max N., (Eds.), Springer-Verlag, Wien, New York, pp. 45–56.

[ZN08] Zhou Q.-Y., Neumann U.: Fast and extensible building modeling from airborne LiDAR data. In *GIS '08: Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems* (New York, NY, USA, 2008), ACM, pp. 1–8.

[ZN10] Zhou Q.-Y., Neumann U.: 2.5D Dual Contouring: A Robust Approach to Creating Building Models from Aerial LiDAR Point Clouds. In 11^{th} *European Conference on Computer Vision (ECCV 2010)* (September 2010), pp. 1–14.