Classification of Points Acquired by Airborne Laser Systems

Examensarbete utfört i Reglerteknik vid Tekniska högskolan i Linköping av Jakob Ruhe och Johan Nordin

LITH-ISY-EX--07/3874--SE
Linköping 2007
Classification of Points Acquired by Airborne Laser Systems

Examensarbete utfört i Reglerteknik vid Tekniska högskolan i Linköping av

Jakob Ruhe och Johan Nordin

LITH-ISY-EX--07/3874--SE

Handledare: Christina Grönwall
isy, Linköpings universitet
Ulf Söderman
FOI

Examinator: Thomas Schön
isy, Linköpings universitet

Linköping, 13 March, 2007
Klassificering av punkter insamlade med flygburen laserradar

Classification of Points Acquired by Airborne Laser Systems

Jakob Ruhe och Johan Nordin

During several years research has been performed at the Department of Laser Systems, the Swedish Defense Research Agency (FOI), to develop methods to produce high resolution 3D environment models based on data acquired with airborne laser systems. The 3D models are used for several purposes, both military and civilian applications, for example mission planning, crisis management analysis and planning of infrastructure.

We have implemented a new format to store laser point data. Instead of storing rasterized images of the data this new format stores the original location of each point. We have also implemented a new method to detect outliers, methods to estimate the ground surface and also to divide the remaining data into two classes: buildings and vegetation.

It is also shown that it is possible to get more accurate results by analyzing the points directly instead of only using rasterized images and image processing algorithms. We show that these methods can be implemented without increasing the computational complexity.

Airborne laser systems, LiDAR, point clouds, outlier detection, ground estimation, classification
Abstract

During several years research has been performed at the Department of Laser Systems, the Swedish Defense Research Agency (FOI), to develop methods to produce high resolution 3D environment models based on data acquired with airborne laser systems. The 3D models are used for several purposes, both military and civilian applications, for example mission planning, crisis management analysis and planning of infrastructure.

We have implemented a new format to store laser point data. Instead of storing rasterized images of the data this new format stores the original location of each point. We have also implemented a new method to detect outliers, methods to estimate the ground surface and also to divide the remaining data into two classes: buildings and vegetation.

It is also shown that it is possible to get more accurate results by analyzing the points directly instead of only using rasterized images and image processing algorithms. We show that these methods can be implemented without increasing the computational complexity.

Sammanfattning

Under ett flertal år har det pågått forskning på avdelningen för lasersystem på FOI för att ta fram metoder som skapar högupplösta 3D-modeller baserat på data framtagen med flygbara laser. För 3D-modellerna finns det ett flertal olika användningsområden, såväl militära som civila, t.ex. uppdragsplanering, krishantering och infrastrukturplanering.

Vi har utvecklat och implementerat ett nytt format för att lagra information erhållen från flygburna lasersystem. Istället för att som tidigare lagra rastervade bilder så sparar vårt nya format även punkternas exakta positioner. Vi har också tagit fram och implementerat nya metoder för att hitta felaktiga punkter och för att ta fram en terrängmodell samt för att identifiera byggnader och vegetation i terrängen.

Vi visar att det är möjligt att få ett bättre resultat genom att analysera punkterna direkt och inte bara använda rastervade bilder och bildbehandlingsalgoritmer såsom tidigare skett. I denna rapport visas också att dessa metoder kan implementeras utan en markant ökning av beräkningstiden.
Acknowledgments

First of all, we would like to thank our supervisors Dr. Ulf Söderman and Dr. Christina Grönwall for all the support during our work and for pushing us to finish this thesis on time. Second, we would like to thank Dr. Gustav Tolt at FOI for all his help and advice and for many interesting conversations. We would also like to thank Simon Ahlberg and Åsa Persson at FOI for all valuable comments and suggestions and FOI for giving us the opportunity to do this thesis. Last but not least we would like to thank our examiner Dr. Thomas Schön and our two opponents, Nina Wollinger and Sofia Sjögren.
# Contents

1 Introduction ................................................. 1  
  1.1 Purpose .................................................. 1  
  1.2 Reader’s guide ........................................... 2  
  1.3 Notation .................................................. 3  
    1.3.1 Symbols .............................................. 3  
    1.3.2 Abbreviations and acronyms ......................... 3  

2 Airborne Laser Systems .................................... 5  
  2.1 Background .............................................. 5  
  2.2 Laser transmitter ....................................... 5  
  2.3 Diffuse and specular reflection .......................... 6  
  2.4 Multiple returns ....................................... 7  
  2.5 Coordinate system ..................................... 8  
  2.6 Scanning patterns ..................................... 8  
  2.7 Camera ................................................... 10  
  2.8 Accuracy and performance ............................... 10  
  2.9 Outliers ................................................ 10  
  2.10 File formats ........................................... 10  
    2.10.1 LAS Format ........................................ 11  
    2.10.2 TIFF Format ........................................ 11  

3 Methods and Algorithms .................................... 13  
  3.1 Convolution ............................................. 13  
  3.2 Gaussian smoothing .................................... 14  
  3.3 Rank-ordered filters ................................... 15  
  3.4 Morphological filters .................................. 15  
    3.4.1 Dilation ........................................... 15  
    3.4.2 Erosion ............................................ 15  
    3.4.3 Closing ............................................ 16  
    3.4.4 Distance transform ................................ 16  
  3.5 Watershed segmentation ................................ 17  
  3.6 Connected component labeling .......................... 17  
  3.7 Principal Components Analysis .......................... 18  
  3.8 Delaunay triangulation ................................ 19
4 Previous methods

4.1 Splitting up the data in squares
4.2 Rasterization
4.3 Removing outliers
4.4 Ground estimation
  4.4.1 Ground estimation using active contours
  4.4.2 Ground estimation using watershed segmentation
4.5 Classification
  4.5.1 Neural net classification
  4.5.2 Watershed classification
4.6 Buildings
4.7 Extraction of single trees
4.8 Tree species
4.9 3D landscape model

5 New methods

5.1 Laser Point Structure
  5.1.1 Limitations with the previous format
  5.1.2 The new format
  5.1.3 Point table
  5.1.4 Extraction of points in a neighborhood
5.2 Detect and Remove Outliers
5.3 Ground Estimation
  5.3.1 Improvements of the active contour algorithm
  5.3.2 The cone ground estimator
5.4 Classification
  5.4.1 Why classification is a hard problem to solve
  5.4.2 Classifying with thread segmentation and PCA
  5.4.3 Classifying by using spheres

6 Results

6.1 Outlier detection
  6.1.1 Test regions
  6.1.2 Test results
  6.1.3 Computation time
6.2 Ground estimation
  6.2.1 Test regions
  6.2.2 Test results
6.3 Classification
  6.3.1 Test regions
  6.3.2 Test results
  6.3.3 Computation time

7 Concluding Remarks

7.1 Conclusions
7.2 Future research
List of Figures

1.1 Example of a 3D landscape model generated from data acquired by an airborne laser system and aerial photos. 2

2.1 Light gray shows the flight path of the pulse and the dark gray area shows the position of the pulse at the moment. 6

2.2 The left image shows a specular reflection in water and the right image illustrates a diffuse reflection on the ground. 7

2.3 One transmitted pulse can generate multiple returns. 8

2.4 Illustration of RT90. 9

2.5 Different methods to cover an area of interest with the laser scanner. 9

2.6 Outliers can occur when the laser pulse hits for example birds, windows or areas with water. 11

3.1 Principle of convolution. 13

3.2 An example of a Gaussian $21 \times 21$ kernel with $\sigma = 3.0$. Axes are in pixel units. 14

3.3 Dilation of image $b$ with the kernel $a$ gives the output image $c$. 15

3.4 Erosion of image $b$ with the kernel $a$ gives the output image $c$. 16

3.5 Dilation followed by an erosion of image $b$ with the kernel $a$ gives the output image $c$. 16

3.6 Two different distance transforms applied to an image. 17

3.7 Example of the watershed algorithm. 18

3.8 Threads in $x$- and $y$-direction and the resulted segments. 19

3.9 Delaunay triangulation of a point set. 19

4.1 Simplified description of the data flow and the methods used. 21

4.2 A square typically includes an overlap with data from adjacent squares. 22

4.3 Rasterization with the cell grouping method. 23

4.4 Different steps of the watershed classification process. 27

4.5 Example of 3D models of buildings generated from laser point data. 27

4.6 Illustration of how the different layers are merged to generate a 3D landscape model. 28

5.1 Artificial points are added in areas that are shadowed by buildings or trees. 30

5.2 (a): Ground estimation without identified outliers. (b): After the outliers are removed. 32

5.3 (a): The square containing the points. (b): The points sorted by $z$-value. 33

5.4 A schematic view of how the cone is used to find ground point candidates. 35
5.5 (a): Interpolated $z_{max}$ surface. (b): Thread segmentation over the area. \\
5.6 (a): Erroneous ground points in a building segment. (b): Ground points in a ground segment. \\
5.7 Building segments with the inner border in gray. \\
5.8 The figure shows a ground segment that includes a bridge. The points in the segment are points classified as ground points. \\
5.9 A bridge surface separated from the ground surface. \\
5.10 Flowchart for the method Classification with thread segmentation and PCA. \\
5.11 Calculation of a point normal by using the normals of the surrounding triangles. \\
5.12 Normalized spherical coordinates defined in a Cartesian coordinate system. \\
5.13 A roof of a building is usually larger than the building itself. This means that there are points that may have been detected on the walls or on the ground directly below the roof. \\
5.14 The points inside the sphere of the current point. \\
5.15 An interpolated $z_{max}$-image is shown to the left and to the right the smallest eigenvalues are shown. White corresponds to high values. \\

6.1 The points from region 1 with $z_{h2}$ and $z_{l2}$ shown as horizontal lines. \\
6.2 The points in region 2. The diamonds are the points with the $z$-values $z_{l1}$ and $z_{h1}$, respectively. The horizontal lines represent $z_{l2}$ and $z_{h2}$. \\
6.3 Ground estimation of region 1. Axes are in meters. \\
6.4 Ground estimation of region 2. Axes are in meters. \\
6.5 Ground estimation of region 3. Axes are in meters. \\
6.6 Class annotation. \\
6.7 The classification images for region 1. \\
6.8 The classification images for region 2. \\
6.9 The classification images for region 3. \\
6.10 The classification images for region 3 (continued). \\

List of Tables \\
4.1 The rasterized images used to represent laser point data. \\
6.1 Computation times for three different classification algorithms evaluated in three regions (minutes:seconds).
Chapter 1

Introduction

During several years research has been performed at the Department of Laser Systems, the Swedish Defense Research Agency (FOI), to develop methods to produce high resolution 3D environment models based on data acquired with airborne laser systems. The 3D models are used for several purposes, both military and civilian applications, for example mission planning, crisis management analysis and planning of infrastructure.

Many algorithms have been implemented to automatically classify the laser data. It is possible to generate digital terrain models, 3D models of buildings and to identify trees and classify their species. In figure 1.1 an example 3D landscape model is shown.

1.1 Purpose

So far all algorithms developed at FOI have been based on image processing. A lot of work has been done to improve them but it seems that by only using image processing it is hard to improve the results. The purpose of this thesis is to investigate if better algorithms can be developed if they work directly on the points — or on a combination of both. It was a requirement that the methods should be implemented in IDL\textsuperscript{1} and ENVI\textsuperscript{2}. IDL is a software for data analysis and application development in general and ENVI is an extension which contains many powerful image processing algorithms and visualization tools. Both applications are developed by ITT\textsuperscript{3}.

\textsuperscript{1}IDL is a software for data analysis and application development, see http://www.ittvis.com/idl/
\textsuperscript{2}ENVI is an extension to IDL, see http://www.ittvis.com/envi/
\textsuperscript{3}ITT developer of IDL and ENVI, see http://www.itt.com
1. **Introduction**

Gives a brief introduction to the subject and the purpose of this report. The notation used in the report is also available in this chapter.

2. **Airborne Laser Systems.** The principles of Airborne Laser Systems are explained as well as related data formats.

3. **Methods and Algorithms.** Here the theory of some of the algorithms related to laser data processing are explained.

4. **Previous Methods.** The methods previously developed at FOI are described in this chapter.

---

**Figure 1.1.** Example of a 3D landscape model generated from data acquired by an airborne laser system and aerial photos.
5. **New Methods.** In this chapter the methods that we have developed and implemented are described.

6. **Results.** Here a comparison between our methods and the previous methods is made.

7. **Future Research.** Some weaknesses in our algorithms are explained together with possible future development to improve our methods.

### 1.3 Notation

#### 1.3.1 Symbols

- \(i\): reflected intensity of a point
- \(r\): return number of a point
- \(x, y, z\): location of a point
- \(z_{\text{max}}\): image where the pixels represent the maximum elevation at each location
- \(z_{\text{min}}\): image where the pixels represent the minimum elevation at each location

#### 1.3.2 Abbreviations and acronyms

- **2D**: Two dimensional
- **3D**: Three dimensional
- **DGPS**: Differential Global Positioning System
- **DSM**: Digital surface model
- **DTM**: Digital terrain model
- **FOI**: Swedish Defense Research Agency
- **INS**: Inertial Navigation System
Introduction
Chapter 2

Airborne Laser Systems

This chapter contains information about airborne laser systems and subjects related to this field. A more detailed description can be found in [Pyy06].

2.1 Background

The use of airborne laser systems (ALS) started in the 1970s and as many other technologies the development and usage of it began in the military industry. Today the technology is widely spread and commercially available. It can be used to generate detailed elevation models of the ground surface, digital city models with 3D buildings and for forest inventory.

The central part of an airborne laser system is the laser scanner. The laser scanner is often referred to as LIDAR - Light detection and ranging. It generates short laser pulses and measures the reflected light. The time between transmitting and when the light is received again is measured. By letting $\Delta t$ be that time and by knowing the speed of light $c$, the distance $d$ can be calculated as

$$d = \frac{\Delta t}{2} \cdot c.$$  (2.1)

The distance to the point where the laser pulse is reflected does not give enough information. Included in an ALS is also a device that can measure the position of the airborne vehicle and also the orientation of it. For the former a Differential Global Positioning System device (DGPS) is often used. A DGPS device measures position by analyzing received signals from GPS satellites and from a ground reference station with known position. For the latter different sensors for measuring the orientation exists.

2.2 Laser transmitter

Laser is an acronym for Light Amplification by Stimulated Emission of Radiation. The laser transmitter, which is used in ALS, is usually operating at wavelengths
in the near infrared region. It generates short pulses at a high repetition rate, e.g. 30-100 kHz. A high repetition rate gives the possibility to fly faster and at a higher altitude while keeping the point density on the ground at the same level. Note however, that if the system is built in such a way that a pulse is received before the next is transmitted, the flight altitude, $h$, is limited by the repetition rate $f$ of the transmitter, by:

$$h \leq \frac{c}{2f}. \quad (2.2)$$

To illustrate this limitation a repetition rate of 100 kHz limits the maximum flight altitude to 1500 m. The intensity and the divergence of the laser beam also limits how high above ground it is possible to fly without loss of accuracy. The divergence of the laser beam is small, usually lower than 1 mrad. In some systems it is possible to adjust the divergence, giving the possibility to keep the size of the laser footprint on ground while flying at different altitudes. The resolution depends on the cross-section of the laser beam and the length of the pulse, see Figure 2.1.

\begin{figure}[h]
\centering
\includegraphics[width=0.4\textwidth]{figure2.1.png}
\caption{Light gray shows the flight path of the pulse and the dark gray area shows the position of the pulse at the moment.}
\end{figure}

### 2.3 Diffuse and specular reflection

A certain amount of the transmitted pulse must be reflected back to the scanner to be detected. A common way to describe the reflection of surfaces is to refer to diffuse and specular reflection. Diffuse surfaces reflect light in every direction and the amount of reflected light is only dependent of the angle of incidence. A dull white paper is an example of a diffuse surface. A specular surface reflects different
amount of light in different directions. An example of a pure specular surface is a mirror.

Most of the surfaces found in normal terrain are more or less diffuse. That means that some of the transmitted light is reflected back to the sensor. Specular surfaces also exist and two examples are water and glass. Often these surfaces are flat and therefore the laser pulse is almost never returned back to the sensor. It may happen that the second surface that reflects the pulse is diffuse and therefore detected by the scanner. The measured point is in this case not a correct one and should be identified as an outlier. Outliers are described in Section 2.9. Glass surfaces that are shaped like domes can be detected because the pulse is reflected back in the direction where the sensor is. For the same reason, waves on the water, increase the probability for the laser pulse to be directly reflected back to the sensor. An illustration of the two different types of reflections is given in Figure 2.2. More information about diffuse and specular reflections can be found in [Has06].

![Figure 2.2](image)

Figure 2.2. The left image shows a specular reflection in water and the right image illustrates a diffuse reflection on the ground.

### 2.4 Multiple returns

The receiver measures the amount of received light and generates a sampled waveform. The waveform is analyzed and parts with high amplitude is interpreted as reflected pulses. The laser scanner is usually able to detect several returns from one transmitted pulse. The amplitude of the reflected pulse is also stored. Another term that is being used for multiple returns is multiple echoes. When the laser scanner is able to detect several returns it is usually possible to extract the return number, \( r \), for a point, and insert it into one of these categories:

- **single return.** \( r = 0 \), this point is the only return detected from the pulse.
- **first return.** \( r = 1 \), this point is the first of several detected pulses.
- **middle return.** \( r = 2 \), this point is not the first return nor the last return from the pulse.
last return. $r = 3$, this point is the last return measured from the pulse.

In Figure 2.3 the laser pulse is reflected on the edge of a building. A part of it continues and hits the ground and is reflected back to the scanner. Multiple returns are often generated at the edges of buildings and in areas with vegetation.

![Figure 2.3. One transmitted pulse can generate multiple returns.](image)

### 2.5 Coordinate system

The coordinate system used for the laser data at FOI is usually RT90 (Rikets koordinatsystem 1990). RT90 is the national coordinate system for Sweden. It is based on a Gaussian cylindric projection of the Bessel ellipsoid. The central meridian for the coordinate system is located $15^\circ 48' 29.8''$ east of Greenwich. The $x$-coordinates are counted positive in the north direction from the equator and the $y$-coordinates positive in the east direction from the central meridian. To avoid negative values for the $y$-coordinates 1500 km is added. For an illustration of RT90, see Figure 2.4. More information about RT90 can be found at Lantmäteriet\(^1\), Swedish National Land Survey.

The points measured by an airborne laser systems also have an elevation value, $z$. This is the height above sea level. All coordinates are given in meters.

### 2.6 Scanning patterns

In order to cover an area of interest the laser beam has to be swept. Different methods have been proposed. An ideal case would be if the points were spread out evenly in the $x,y$-grid. This would make the analysis and visualization easy since the laser data would be very well approximated by images.

Figure 2.5 shows three different scanning techniques. The arrows show the direction of flight. The Saw tooth pattern is used by for example ALTM 3100EA

---

\(^1\) Swedish National Land Survey [http://www.lantmateriet.se](http://www.lantmateriet.se)
from Optech\textsuperscript{2}. Rotating mirror pattern is used by TopEye\textsuperscript{3} MKII. The swing mode pattern is used by TopoSys\textsuperscript{4} Falcon III. We have used laser data captured with all three different scan patterns.

\textbf{Figure 2.4.} Illustration of RT90.

\textbf{Figure 2.5.} Different methods to cover an area of interest with the laser scanner.

\begin{itemize}
\item Saw tooth pattern
\item Rotating mirror pattern
\item Swing mode pattern
\end{itemize}

\textsuperscript{2}Optech: http://www.optech.ca
\textsuperscript{3}TopEye: http://www.topeye.com
\textsuperscript{4}TopoSys: http://www.toposys.com
2.7 Camera

A camera is often mounted on the airborne vehicle to acquire aerial photos at the same time. The camera usually has the ability to measure light on four different bands. The bands consist of: visible light of red, green and blue and often also an IR-channel.

2.8 Accuracy and performance

The inaccuracy of the position of the points depends on which ALS that is used and at which altitude the data is collected. The vertical inaccuracy is normally about 0.1 m and horizontally 0.15–0.25 m. The point density varies depending on the application of scanning, from 1 points/m² up to 50 points/m². Laser datasets with low density can be used to get a rough model of the terrain and to find large buildings and vegetation. To be able to determine the shape of buildings and find individual trees a point density of 10–20 points/m² is required.

2.9 Outliers

Usually some of the measured points can be identified as outliers. There are two types of outliers. Type 1 outliers occur when the laser beam hits for example a flying bird. Since our aim is to generate landscape models, not counting and classifying birds, we consider these points as unwanted outliers. Another type of outliers is referred to as type 2 outliers and consist of points with a too low z-value. They occur when the laser beam hits for example water and is reflected with the angle of incidence and not directly back to the sensor. If the laser pulse hits another surface some of it may eventually be reflected back and detected by the sensor. It will then be inappropriately measured as a lower point because of the extra travel time for the pulse.

Figure 2.6 shows how outliers can be generated. Point 1 is identified as a type 1 outlier because the laser beam hit some flying birds instead of any object on the ground. Point 2 occurs because the pulse hit a specular surface (i.e. a window) on a building and it is reflected and hits the ground. The ground is a diffuse surface so some of the transmitted pulse is reflected back to the sensor and stored as point 2 shows. This type of outlier is hard to detect since in this case it is neither too low or too high compared to the other points. Point 3 occurs when the pulse hits water and is later reflected back to the sensor after it has hit a building. This point can be identified as an outlier because it is lower and isolated compared to the other points.

2.10 File formats

There are many different file formats to store images and vectorized data. In the field of laser point processing and aerial photogrammetry, two file formats are of
2.10 File formats

Figure 2.6. Outliers can occur when the laser pulse hits for example birds, windows or areas with water.

common use today. Brief information about them is available in the following two sections.

2.10.1 LAS Format

In May 2003 a file format standard was proposed to be used for laser point data — The LAS Format\(^5\). Version 1.1 of the standard was proposed in March 2005. The LAS Format is today the format that is most often used. A LAS file consist of a header with information about when the data was acquired and by which system. The header may be followed by some optional fields. These fields can be used to add additional information about the data, for example which projection and coordinate system that is used. The optional fields are followed by a table with point entries for each measured point. A point entry consists of different fields and the most important are: location \(x, y, z\), intensity \(i\) and return number \(r\). The locations of the points are stored with a common offset, defined in the header, to save space.

2.10.2 TIFF Format

TIFF is an acronym for Tagged Image File Format and it is used to store image data. The standard is from the mid 1980’s. Information about location and pixel size in geographical units can be stored with the GeoTIFF Format\(^6\).

\(^5\)LAS Format Specification: http://www.lasformat.org
Chapter 3

Methods and Algorithms

In this chapter the theory behind the algorithms used throughout the thesis is described. It covers convolutional filters, rank-ordered filters, morphological filters and other methods such as Principal Components Analysis and Delaunay triangulation.

3.1 Convolution

Convolutional filters are fundamental in the domain of image processing. A kernel is used to slide over the pixels in the input image. For every position of the kernel an output pixel value is calculated by forming the sum of each kernel pixel multiplied by the input image pixels. It is often desirable to let the output image have the same size as the input image. This can be accomplished by adding a border of zeroes around the input image. The width of this border is half the kernel size. Figure 3.1 illustrates this. More theory about convolutional filters and kernels is available in [BJ00].

By convolving an $m \times n$ input image $I$, with a $k \times k$ kernel $H$, a resulting
output image $R$, is generated with the following value in pixel $(x, y)$:

$$R(x, y) = \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} I(x + i - \lfloor k/2 \rfloor, y + j - \lfloor k/2 \rfloor) H(i, j).$$  \hspace{1cm} (3.1)

### 3.2 Gaussian smoothing

Gaussian smoothing is well known in signal processing. The two dimensional version is used to smooth images. The kernel is defined as:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{\frac{x^2+y^2}{2\sigma^2}},$$ \hspace{1cm} (3.2)

where the parameter $\sigma$ determines how much the output image should be smoothed. In this case the same variance $\sigma$ is used in both $x$ and $y$ directions and this gives a symmetric kernel. A large $\sigma$ needs a larger kernel to approximate the Gaussian well. In Figure 3.2 a Gaussian kernel is visualized.

![Figure 3.2](image.png)

**Figure 3.2.** An example of a Gaussian $21 \times 21$ kernel with $\sigma = 3.0$. Axes are in pixel units.
3.3 Rank-ordered filters

Rank-ordered filters are very simple filters, but still useful. In this category all filters that sort the points in some way are included. They work by positioning a kernel in the input image and the points covered by the kernel are sorted. The output value at this position is one of the sorted pixel values. Example of common rank-ordered filters are minimum-, median- and maximum-filters. All three filters can be used to interpolate pixel positions where no measures are available. More information is available in [BJ00].

3.4 Morphological filters

The theory behind morphological filters is shortly explained in the following sections. A more detailed description is available in [BJ00].

3.4.1 Dilation

Dilation is a morphological operation used to grow regions of connected pixels or connect adjacent regions in a binary image. A kernel defines how the growing is performed. The kernel is placed with the center pixel over each pixel that is set in the image and all the pixels which are covered by the kernel are set in the output image. An example is given in Figure 3.3.

![Figure 3.3. Dilation of image b with the kernel a gives the output image c.](image)

3.4.2 Erosion

Erosion is a morphological operation used to shrink regions of connected pixels. A kernel defines how the shrinking is performed. The kernel is placed with the center pixel over each pixel in the image. If all pixels covered by the kernel are set in the input image the center pixel will be set in the output image. See example
in Figure 3.4 where an image $c$ is generated by eroding image $b$ with the kernel $a$. The center pixel of the kernel is marked with a dot.

\[ a \rightarrow b \rightarrow c \]

**Figure 3.4.** Erosion of image $b$ with the kernel $a$ gives the output image $c$.

### 3.4.3 Closing

The closing operation is a dilation followed by an erosion. See Figure 3.5 for an example.

\[ a \rightarrow b \rightarrow c \]

**Figure 3.5.** Dilation followed by an erosion of image $b$ with the kernel $a$ gives the output image $c$.

### 3.4.4 Distance transform

The distance transform calculates a distance map where each foreground pixel is set to the distance to the nearest background pixel. In Figure 3.6 an image is shown together with two different distance transforms applied to it — *City block distance* and *Chessboard distance*.

City block distance is also known as the Manhattan distance and this metric only allows horizontal and vertical moves. The city block distance $D_{city}$ between
point \((x_1, y_1)\) and point \((x_2, y_2)\) is:
\[
D_{\text{city}} = |x_2 - x_1| + |y_2 - y_1|.
\] (3.3)

In Figure 3.6 the distance transform using Chessboard distance metric is also shown. This metric allows horizontal, vertical and diagonal moves. The distance \(D_{\text{chessboard}}\) between two points \((x_1, y_1)\) and \((x_2, y_2)\) is:
\[
D_{\text{chessboard}} = \max(|x_2 - x_1|, |y_2 - y_1|).
\] (3.4)

\[\begin{array}{c}
\begin{array}{cccc}
\hline
& & & \\
& & & \\
& & & \\
& & & \\
& & & \\
\hline
\end{array}
\end{array}\]

Original image City block distance metric Chessboard distance metric

**Figure 3.6.** Two different distance transforms applied to an image.

### 3.5 Watershed segmentation

The watershed algorithm is used to segment an image. Consider the surface in Figure 3.7, where low and high pixel values are represented by valleys and peaks, respectively. The algorithm can be explained as if water falls on each pixel of the surface from above. The water flows from pixels with high values to pixels with lower values. The water that flows to the same local minimum creates a segment. The result of the watershed algorithm can be seen in Figure 3.7. The image contains three local minimas and this results in three different segments. In [BJ00], more information is available.

### 3.6 Connected component labeling

Connected component labeling is used to segment images. In this thesis this method is referred to as thread segmentation. One way to describe it is to let a thread exist between neighboring pixels if the difference between them is less then a threshold, \(\epsilon_h\). The thread segmentation is performed both horizontally and vertically on the input image. The two images are merged together to one image and the segments in the resulting image consists of areas with similar values. Figure 3.8 gives an example of how the two thread images are merged to form the resulting segment image. We have implemented this method and it is used by several of our algorithms.
Original image visualized as a surface

The resulting watershed segments

Figure 3.7. Example of the watershed algorithm.

3.7 Principal Components Analysis

Principal Components Analysis (PCA) is a useful statistical technique to reduce the dimension of a dataset. In this thesis the dataset consists of \( n \) points in three dimensions \( x, y \) and \( z \).

To be able to compute the covariance matrix, the algorithm begins by calculating the mean value for \( x \), \( y \) and \( z \) for the \( n \) points by:

\[
\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i. \tag{3.5}
\]

The covariance between two vectors \( x \) and \( y \) with \( n \) elements each is defined as:

\[
\text{cov}(x, y) = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{(n - 1)}. \tag{3.6}
\]

The covariance matrix consists of the covariances between the vectors \( x \), \( y \) and \( z \) and is defined as:

\[
C = \begin{pmatrix}
\text{cov}(x, x) & \text{cov}(x, y) & \text{cov}(x, z) \\
\text{cov}(y, x) & \text{cov}(y, y) & \text{cov}(y, z) \\
\text{cov}(z, x) & \text{cov}(z, y) & \text{cov}(z, z)
\end{pmatrix}. \tag{3.7}
\]

As the covariance matrix is symmetric we can calculate an orthogonal basis by finding its eigenvectors together with the corresponding eigenvalues. The eigenvalue tells how much the data varies in the direction of its eigenvector.

By using PCA on 3D points information about how they are organized are obtained. These three important cases exist:

**3 large eigenvalues.** The points are unorganized.

**2 large eigenvalues.** The points are located on a plane.

**1 large eigenvalue.** The points are located on a line.

A more detailed description of PCA in general is [Smi02]. For PCA on points in 3D a recommended source for information is [HG06].
Delaunay triangulation

Delanay triangulation is performed for a planar point set \( P \). The triangulation is performed so that no points in \( P \) are inside the circle of the triangles and it also maximizes the minimum angle for the triangle. In Figure 3.9 an example is given and in [MdB97] a more detailed description about Delaunay triangulation can be found. The triangles generated by the Delaunay triangulation can be used to generate a rasterized image with pixel values interpolated from the original points.

![Figure 3.8. Threads in \( x \)- and \( y \)-direction and the resulted segments.](image)

![Figure 3.9. Delaunay triangulation of a point set.](image)
Chapter 4

Previous methods

In this chapter the previous methods developed at FOI are explained. Figure 4.1 shows how the laser data and the aerial photos are used by the methods to finally generate a realistic 3D landscape model.

Figure 4.1. Simplified description of the data flow and the methods used.

4.1 Splitting up the data in squares

The company responsible for the laser scanning normally deliver the laser data in LAS files and the aerial photos in TIFF format. Both formats are briefly explained in Section 2.10. The LAS files are usually huge and divided on flight strip basis
instead of geographic basis. In order to simplify the analysis of the data, the process begins by splitting the LAS files on a geographical basis in different squares. A typical size for a square is $120 \times 120$ m, which includes a border overlap of 10 m with data from adjacent squares as shown in Figure 4.2. The overlap is needed in order to get a correct ground model and a better classification of buildings near the border. This also minimizes the border effects of convolutional filters. The TIFF files are split in a similar way.

![Figure 4.2](image)

**Figure 4.2.** A square typically includes an overlap with data from adjacent squares.

### 4.2 Rasterization

The method used for sampling the irregularly spaced data into uniformly-sized cells is called *cell grouping*. When using the cell grouping method, all the points are rasterized into different cells as shown in Figure 4.3. In one cell zero, one or many points may exist. The value of the pixels in the resulting image depends on what kind of image that is wanted. The parameter $\delta$ defines the size of one pixel in the grid. Typical value for $\delta$ is 0.25 m. One image is referred to as the $z_{\text{max}}$-image and the value of each pixel is the $z$-value of the highest point inside each cell. In a similar way a $z_{\text{min}}$-image is generated. Another image is called the *hitmask* where the pixel values represent the number of hits in each cell. Two other images give information about multiple echoes and another one about the reflected intensity, $i$, from the laser pulse. Table 4.1 summarizes the raster images generated during the rasterization process. See [BH03] for more information and examples of how these images look like and how they can be utilized.
4.3 Removing outliers

Often some of the points in a dataset should be identified as outliers. The reason for this and why they are part of the laser data is explained in Section 2.9. The algorithm that is in use today to detect outliers is divided in two parts: one that removes outliers of type 1 and one that removes outliers of type 2. To find type 1 outliers it starts with a $z_{\text{min}}$-image and creates a $z_{\text{min}}$-surface by filtering the image in different steps to remove all big slopes and to smooth the image. This surface is moved down 1 m and all pixels that are below this surface are identified as outliers. The same method is used to remove Type 2 outliers, with the difference that it starts with a $z_{\text{max}}$-image and the surface is moved up 30 m. Pixels above this surface are identified as outliers.

4.4 Ground estimation

The ground estimation is a very important step for the future analysis. Two different master theses previously written at FOI cover ground estimation algorithms. One is based on the theory of active contours and another one on the watershed algorithm.
4.4.1 Ground estimation using active contours

The ground estimation algorithm that uses active contours is presented in Elmqvist [Elm00]. It can be explained as a 2D elastic cloth being pushed upwards from an initial position below all points. The lower points attract the cloth when it moves upwards and the elasticity will prevent the cloth to grow up in buildings where there are no ground points. This algorithm has a high accuracy in forest areas and in less urban environments. In more urban environments it can happen that the cloth grows into the buildings, especially in the periphery of the image. Another negative aspect of this method is that it is rather time consuming.

4.4.2 Ground estimation using watershed segmentation

The ground estimation algorithm which uses the watershed segmentation is developed at FOI by Landgård and is presented in [Lan05]. The method begins by performing a watershed segmentation on an interpolated \( z_{\text{min}} \)-image. It uses seed points which are the lowest point in each segment. Then it starts with the lowest point in the image and decides that this point is a ground point. After that a region growing algorithm classifies the pixels in the surrounding to ground as long as the difference in \( z \)-value is less than a threshold (0.3m). When a region cannot grow anymore the process around the next seed point will begin. Finally an interpolation is performed in the areas where no points have been classified as ground. This algorithm is able to estimate the ground with a high accuracy in urban areas. In rough terrain it can have problems to estimate the ground surface well because it has problems to climb up on steep hills. This method is much less time consuming than the one that uses active contours.

4.5 Classification

The next step in the process is to classify the regions. Two classification algorithms have been developed and implemented at FOI. The first one is developed by Brandin and Hamrén and is presented in [BH03]. This one is referred to as Neural net classification. The second one has been proposed by Landgård as presented in [Lan05]. The method has been developed further by Gustav Tolt at FOI as described in [GTLS06]. The latter classification method is referred to as watershed classification. Both classification methods are based on image processing approaches.

4.5.1 Neural net classification

This classification method uses neural nets to perform classification. The most critical part in the algorithm is the segmentation. The \( z_{\text{max}} \)-image is used to find pixels that are more than 2 meters above ground. All these pixels are classified and the method first segments the pixels into groups using information from the echo-image. If this segmentation fails, which often happen when trees stand close
to buildings, the classification will also fail in these areas. For all segments three parameters are calculated:

**Hough value** - indicates if the segment have any straight lines or not.

**Maximum slope value** - the mean value of the first derivative of the object’s pixels.

**LaPlacian value** - the mean value of the second derivative of the object’s pixels.

These parameters are used by the neural net to classify the segments. The method can classify power lines, posts, buildings and vegetation.

### 4.5.2 Watershed classification

The watershed classification method uses various types of filtration methods to find areas above the ground level that contain flat surfaces. All points that are more than 2 meters above ground are used to create a mask and then a distance transform (Section 3.4.4) is used. On this image an upside-down watershed segmentation (Section 3.5) is performed. The watershed segments that include some planar objects are classified as buildings. This method fails if there are some parts of the trees that are flat and in areas where trees stand very close to buildings. A tree with the major part of the crown area outside a building can be separated from each other by using the fact that two maxima can be found in the distance image. An illustration of the method can be seen in Figure 4.4.

### 4.6 Buildings

By using the classification image and the $z_{max}$-image it is possible to create 3D models of the buildings. This is done by first cluster the normals extracted from the $z_{max}$-image for each building. This gives the different roof segments. The next step is to create the edges of the roof segments. To accomplish this step a 2D Hough transform is used on the border of the segments. The roof segments are adjusted to fit together to generate a complete roof. When the roof segments are estimated the walls are created by generating polygons from the edges of the estimated roof segments and the ground surface. Figure 4.5 shows two 3D models of buildings generated from laser data. More information about this process can be found in [US06].

### 4.7 Extraction of single trees

To identify trees the $z_{max}$-image is used together with the results from the classification process. In this section the method is described very briefly. For a detailed description we refer to the work done by Åsa Persson [Per01] in this area.

This method begins by interpolating the $z_{max}$-image. Then two different images are generated by filtering the $z_{max}$-image with Gaussian kernels with different $\sigma$-values. Gaussian smoothing is explained in Section 3.2. The first one will be
filtered using a small $\sigma$ so that small trees will have just one peak value in the $z_{max}$-image. The second one will be filtered with a larger $\sigma$ so that large trees get just one peak value in the $z_{max}$-image. The location of the trees are estimated at the positions of the local maxima in these images.

An upside-down watershed method is applied to the filtered images to segment the tree crowns. The operation is done on both images and the results are compared. A parabolic surface is used to fit the surface canopy of each tree detected in the two filtered images. The estimated trees that best fits the surfaces are used as the final tree estimation in that region.

### 4.8 Tree species

To identify the species of the trees, the laser points in combination with an aerial photo that contains a red, green, blue and an IR-channel, is used. By using the information from the IR-channel it is possible to separate the deciduous trees from the coniferous trees with a high accuracy. Separation of spruce from pine can also be done using the aerial photo but with less accuracy. By combining the information from the aerial photo with the laser points it is possible to improve the classification of the tree species. The previously calculated parabolic surfaces for the tree crowns are used again in this step. A parabolic surface is more pointed for spruces compared to pine and deciduous trees. Again this is a very simplified description of the method and for the curious readers we recommend [Per01] and [APS06].

### 4.9 3D landscape model

When all the presented algorithms are applied, it is possible to create a very realistic 3D landscape model of the area. See Figure 4.6 for an example. More information can be found in [US06].
Figure 4.4. Different steps of the watershed classification process.

Figure 4.5. Example of 3D models of buildings generated from laser point data.
Figure 4.6. Illustration of how the different layers are merged to generate a 3D landscape model.
Chapter 5

New methods

In this chapter the new methods that we have designed and implemented are described. The chapter begins with explaining the new structure used to store the laser data. The following sections are about our new method to detect outliers, estimate ground and to classify laser data.

5.1 Laser Point Structure

At FOI extensive research has been done on data measured by airborne laser systems for several years. All the operations, such as estimating ground and classifying vegetation and buildings, have been performed on raster images. The main idea of this master thesis is to develop a format that keep more information about the measured points. This new format is described in this section.

5.1.1 Limitations with the previous format

The reason why raster images have been used is that ordinary image processing algorithms can be used to process the data. Many efficient image processing algorithms have been developed during the last decades and therefore a lot of robust and fast solutions can be used to analyze the raster images. The previous format used to store laser data is explained in Section 4.2.

The limitations with raster images is that information about the exact point location is discarded. This can be compensated by choosing a small pixel size, $\delta$. But by decreasing the pixel size the amount of pixels, of which many are empty, increases. This gives a waste of both storage space and increased processing time for the algorithms. It should also be noted that the empty pixels need to be taken care of somehow by the algorithms that process the data. One way is to make the algorithms process empty pixels in a special way.

Another more common approach has been to first interpolate the images and then use standard image algorithms. The disadvantage with interpolation is that artificial height data can be inserted. This happens for example when a building shadows the laser beam so that there is an area behind the building without points.
An interpolation method may add many artificial z-values, that were not part of the original data. This is illustrated in Figure 5.1.

![Original datapoint and regularly sampled point](image)

**Figure 5.1.** Artificial points are added in areas that are shadowed by buildings or trees.

When several points are associated with the same pixel, information is lost because only the points with highest and lowest position are kept. Information about the return number, \( r \) (see Section 2.4), a point has, is also lost with the previous format. This is an important reason why it has been necessary to develop a new format to store the laser point data.

### 5.1.2 The new format

A requirement of the new format is that all the raster images that were used before should be able to be generated fast. The reason for this is that it should still be possible to use the current algorithms that are implemented to work on raster images and also to be able to easy visualize the data as images. Some or all of these algorithms may in the future be rewritten to take advantage of the new information available.

All formats and standards need a name and for our format we have chosen *Laser Point Structure* (LPS). We have implemented functions to convert from other laser point formats, for example LAS, and to load and save LPS files. We have also developed a graphical user interface to work with LPS files. The implementation language and environment we have used is IDL and ENVI from ITT as described in Section 1.1.

### 5.1.3 Point table

Every point is stored in a table with attributes such as position \((x, y, z)\), reflected intensity \((i)\) and return number \((r)\). Each point also has an index which is the location of the point in the table. The format also supports addition of extra
attributes to the points. An example of an extra attribute is the class that the point belongs to.

5.1.4 Extraction of points in a neighborhood

Extracting points in a certain region is a very common operation during analysis. A naive approach is to loop through the points and extract those points which are inside the region to extract. The time needed to extract a region is proportional to the number of points in the data set which is unacceptable when working on larger datasets. One solution is to always keep the datasets small but what if this is not a reasonable alternative? We solve this by grouping the points in subsquares. Inside these subsquares the points are sorted in ascending z order. When a region is extracted the operation that is needed is to find the subsquares that contain the points. By dividing the points in subsquares it is easy to find neighboring subsquares. A quad-tree approach is also a possible solution but as the points are evenly distributed, using fixed sized subsquares is good enough.
5.2 Detect and Remove Outliers

As explained in Section 2.9 an outlier can be either of type 1 or type 2. Outliers exist in most datasets and it is important to remove them in order to get the algorithms used in later steps to generate accurate results. A step that gives a less accurate result is the ground estimation that is not able to classify points above type 2 outliers as ground points. An example of this is shown in Figure 5.2. Type 2 outliers are mainly a problem for the algorithm that detect trees. The outliers may make the tree algorithm to generate too high trees.

![Figure 5.2](a): Ground estimation without identified outliers. (b): After the outliers are removed.

To remove the outliers we look at all points in a square, $s_o$, as is shown in Figure 5.3. The $z$-values of the points are then sorted in ascending order. As the outliers usually are very few or none we can directly consider all middle points, 99.5% of all points, as correct points. The highest and lowest height values of these correct points are $Z_{h1}$ and $Z_{l1}$, as shown in Figure 5.3. There are still points that we want to keep. Therefore we subtract a small value, $\epsilon_{low}$ from $Z_{l1}$ and we add a value, $\epsilon_{high}$, to $Z_{h1}$. These two new values are called $Z_{h2}$ and $Z_{l2}$. We add a larger value to $Z_{h1}$ because high objects usually have fewer points at the same level compared to low ground areas. To avoid problems in the periphery of the square we only identify outliers in an inner square, $s_i$. All points in $s_i$ above $Z_{h2}$ or below $Z_{l2}$ are identified as outliers. The square, $s_i$, is moved over the whole image so that all points get tested.
Figure 5.3. (a): The square containing the points. (b): The points sorted by $z$-value.
5.3 Ground Estimation

The classifier used today is dependent of a good ground surface to produce acceptable results. Although it is possible to make the classifier less dependent of the ground surface a good ground estimator is needed for other reasons, for example to estimate tree heights and to produce realistic 3D landscape models. For that reason we are still interested in improving the current methods. There are two different ground estimators in use at FOI today. They are both briefly explained in Section 4.4. In this section our work regarding ground estimation is described.

5.3.1 Improvements of the active contour algorithm

Speed improvement

As ground estimation using active contours is a rather compute intensive method, we noticed a need for a faster algorithm. An intuitive way was to use the same algorithm but to work on smaller datasets. We do this by dividing the points into squares. The size of the squares correspond to the resulted resolution of the estimated ground surface and can is chosen by the user. As input to the active contour algorithm we use the points with lowest elevation in each square. To get a more visual attractive surface a bilinear interpolation is performed on the output from the active contour algorithm. This is followed by a low pass filter to get a smooth surface.

By working at a coarser scale the active contour algorithm has fewer points to process and because this algorithm is quite computational intensive this gives an important speed improvement. The computational time for the algorithm is proportional to the number of points. If we choose a scale of 4, which means that we use the points with lowest elevation in every 4×4 region, the algorithm has to process only 1/16th of the points.

A ground surface estimated at a coarser scale is not per definition a less realistic model of the ground. By using the points with lowest elevation we remove some small objects that otherwise would have been part of the ground surface. In general the true ground surface vary slowly, which besides the important speed improvement, makes the idea of working on a coarser scale motivated.

Remove non-ground points using $r$

We have also been looking at a simple approach to exclude points that are likely not to have hit the ground. As we now have access to the return number, $r$, of the points, we are able to find points that are not good ground candidates. If a point is not the last point of a pulse (i.e $r = 1$ or $r = 2$), than this point is probably not a ground point. We have also tested to remove every point that is indeed the last point generated from the laser pulse but is part of a pulse that has generated more than one point (i.e $r = 3$). Those points might of course be ground points. By using only single echo points ($r = 0$), we have removed many of the points that belong to vegetation and also points that are part of the building edges. Note that
this method excludes information that may be needed to estimate a good ground surface in a very dense forest area. However we have not found any such area yet.

5.3.2 The cone ground estimator

Inspired by [Sit05], we decided to implement a new ground estimator. We have previously shown in Section 5.3.1 how we can use $r$ to remove some of the points that are likely to not belong to the ground surface. This new approach is even more simple than active contours and is based on one idea: points that belong to ground must have no points with a lower $z$-value in the immediate surrounding. This must of course be formulated in a more precise way to be implemented and used. We use a cone to check whether a point is a ground candidate or not. We locate the cone top at the current point and the cone is orientated so that the top is upwards. A point must have no points inside the cone to be classified as a ground point.

Figure 5.4 shows a schematic view of how the cone is used to find out if a point is a ground candidate or not. The left figure shows some points that are part of the ground surface and some that has hit a building. To illustrate how the cone ground estimator works the cone is positioned on two points. The point to the left with the cone has hit the building and as there is one point inside the cone this point will not be classified as a ground point. The other point is indeed part of the ground and the cone has no points inside so this will be correctly classified as a ground point. The right figure shows how the parameters $h$, $d$ and $\alpha$ defines the cone.

Figure 5.4. A schematic view of how the cone is used to find ground point candidates.

Parameters

The cone that is used by the ground estimation method is defined by an angle $\alpha$, a minimum distance $h$ and a maximum diameter $d$. Figure 5.4 shows what the cone looks like. If we choose a too large $\alpha$ we will reject ground points where the
ground is steep. If $\alpha$ is too small we will accept many points as ground points which are part of vegetation or buildings. The parameter $h$ is needed if we want points being near other ground points to also be classified as ground points. This is desirable when we take into account that there is a small uncertainty of the position of the points and also if we want to include steep but small changes of the ground surface. The last parameter defines the maximum diameter of the cone. By using the LPS format it is possible to make a quite efficient implementation of this algorithm. Extraction of an area that may be inside the cone is possible by extracting a region as described in Section 5.1.4. To see results from the algorithm the reader is referred to Chapter 6.

Using the cone ground estimator together with thread segmentation

The cone ground estimator removes many of the points that do not belong to the ground. It is very intuitive and it is easy to understand how the parameters affect the result. There is one problem yet not mentioned though. The algorithm classifies points that belongs to roofs of large buildings as ground points. What we mean with large buildings really depends on how tall they are and what parameters we choose for the cone. But for the algorithm to correctly classify the points on the roof of a large and low building we need to use a very wide cone (i.e. a large $\alpha$ value). Apart from being slower this will also make the cone selector classify a lot of points belonging to the ground surface as non ground points.

To classify points on large buildings as non ground points we use our thread segmentation method. In Figure 5.5 a small area including a building and some trees has been segmented. Here it shows how the building and the ground create large segments and the trees small segments. Figure 5.6 shows two segments, one ground segment and one building segment. The points shown in the segments are the points that the cone selection algorithm has classified as ground points. To identify the segment, the points inside the inner border of the segment are analyzed. The segment with the inner border is shown in Figure 5.7. If very few or none of these points have been classified as ground points the segment is identified as a building segment. Otherwise we identify this segment as a ground segment. For the former case, all points inside the segment are classified as non ground points.

Classifying bridges

Some small experiments have been made to classify bridges. To find them we compare the ground segments described in Section 5.3.2 with the points classified as ground by the cone selector. The bridge is part of a ground segment, but the cone ground estimator has not classified the points on the edges of the bridge as ground points. The reason for this is that there are points on the ground below at the edges of the bridge. An example of this is shown in Figure 5.8. The gray areas show where no points have been classified as ground. If the bridge is high or narrow this might include the whole bridge.

When an area is identified as a bridge, the cone ground estimation algorithm can be used on the points in this area. By excluding the points below the bridge,
the points of the bridge surface, are identified as ground points by the cone ground estimation algorithm. The points identified as ground in this step can then be classified as bridge points. An example of a bridge classification can be seen in Figure 5.9.
Figure 5.7. Building segments with the inner border in gray.

Figure 5.8. The figure shows a ground segment that includes a bridge. The points in the segment are points classified as ground points.

Figure 5.9. A bridge surface separated from the ground surface.
5.4 Classification

The classification algorithms developed during this master thesis uses the laser data and the estimated ground as input. The latter can consist of an estimated rasterized ground surface or a list of the points that are classified as ground points. The major difference with our methods compared to previously methods developed at FOI is that we classify points instead of pixels. That means that a pixel can contain points with different classes, for example when a building is partly hidden by a tree. Our new LPS format, which is described in Section 5.1, is used to obtain information about the points and also to store the classification results.

5.4.1 Why classification is a hard problem to solve

To implement a complete classifier a lot of things need to be considered, including the following:

1. Into which classes should the points be divided?
2. How to define a building and how to separate a hedge or vehicle from it?
3. How to treat bridges?

We have based our classifier on the fact that flat surfaces could normally be found on buildings. To detect other roof types like domes other suitable methods should be used.

When you start to use your newly implemented classifier on laser data, you will very soon begin to realize that it is not enough just to detect flat surfaces. On buildings you often find a lot of points belonging to objects like chimneys, antennas and balconies. To classify these points as part of buildings as well, a simple rule like "points near buildings are part of the building" might be used. Another rule that is useful is "a group of points high above the ground surface which all have similar z-values are very likely to belong to a building". Points that belong to very small and isolated flat surfaces should perhaps not be classified as building points because they are usually found by coincidence in vegetation or they are anyhow too small to be buildings of reasonable size.

We have implemented two methods. The first one uses image algorithms to preprocess and segment the image and then it works on the points. A lot of testing and fine tuning have been done. The other one analyzes the points directly, but is just at experimental state as some of the rules just mentioned must be added in order to get it perform well.

5.4.2 Classifying with thread segmentation and PCA

The classifier using thread segmentation and PCA is described in this section and in Figure 5.10 the different steps are shown.
Extract ground points

The ground must be estimated before this algorithm is used. If the output from the ground estimation algorithm is just a ground surface we need to see which points that belong to that surface. The ground estimator using cones described in Section 5.3.2 is an example of a ground estimation algorithm that classifies the points instead of generating a ground surface. To find points that belong to the ground surface, the height difference between the point and the ground surface at the location of the point is calculated. If this value is less than a threshold, this point is classified as a ground point.

Image Segmentation

To classify building points, we begin with interpolating the points by using a Delaunay triangulation and generate a regular sampled height grid from the triangles. This grid is segmented using thread segmentation as described in Section 3.6. If the height difference between neighboring cells is larger than a threshold, $\epsilon_h$, the thread will be cut. Cells that contain ground points also cut the thread. This is because we suppose that we have no points that belong to the ground surface inside buildings. The result of this will be that the roofs of the buildings form large segments. Many small segments are also generated and most of them contain points that belong to vegetation and edges of buildings.

Finding flat surfaces

The segmented image generated in the previous step needs further processing. This is done in several steps that depend on the size of each segment and is described in Figure 5.10. Large segments with a high mean elevation compared to the ground level will be classified directly as buildings. Segments that are a bit smaller are harder to classify because it can be a building, vegetation, or a combination of both. Therefore we make a new triangulation of the points for each such segment. Then the normal for each triangle is calculated. By using these normals we are able to calculate the normals of each point. The normal for a point is calculated by summing the normal of each triangle that are connected to the point. Figure 5.11 illustrates this. Each triangle normal is weighted by the area of the triangle. This is done in order to make larger triangles have more influence on the point normal. Normals of small triangles have less accuracy because the uncertainty of the points have more influence on them.

As shown in Figure 5.12, the normals are converted from $(x, y, z)$ to normalized spherical coordinates $(\varphi, \theta)$ by:

\[
\varphi = \arctan \frac{x}{y}, \quad (5.1)
\]

\[
\theta = \arctan \frac{\sqrt{x^2 + y^2}}{z}. \quad (5.2)
\]
The next step is to perform PCA on all points that have similar $\theta$ and $\varphi$ and are close to each other in the segment. If the PCA shows that the points mainly are spread in two dimensions, a plane, this will for sure be a part of a building. As the segments usually are smaller than the buildings, the size of the segment is increased. This occurs if trees partly covers the building or if there are points on the ground below the edges of the roof. The latter situation is shown in Figure 5.13. All points with a distance less then a threshold, $\varepsilon_{\text{limit}}$, from the plane in this segment, are classified as building points. The value of $\varepsilon_{\text{limit}}$ depends on the accuracy of the points. With this method it is possible to not just separate trees close to buildings, but also to classify buildings partly hidden by trees.

Very small segments, less than a couple of square meters, will not be considered as they usually consist of vegetation or edges of buildings.
Triangulation

Thread segmentation

Calculate size of segments and sort in descending order

(each segment)

Segment large enough?

Yes

Segment large and high?

No

Triangulate and calculate normals

Pick out points with similar normals

Do PCA on points

Third eigenvalue low?

Yes

Classify points as building

No

Points left to classify?

Yes

No

Done!
Figure 5.11. Calculation of a point normal by using the normals of the surrounding triangles.

Figure 5.12. Normalized spherical coordinates defined in a Cartesian coordinate system.
Figure 5.13. A roof of a building is usually larger than the building itself. This means that there are points that may have been detected on the walls or on the ground directly below the roof.
5.4.3 Classifying by using spheres

We have also implemented another algorithm to find points that are part of flat surfaces. Instead of working on interpolated images this algorithm works on the points directly. This is done by centering a sphere with radius $r$ on each point. The points inside this sphere is used to calculate how flat the local neighborhood of the corresponding center point is. The points inside the sphere can be found by first extracting a small region as described in Section 5.1.4. Figure 5.14 shows the sphere centered on a point in 3D space.

![Sphere centered on a point in 3D space](image)

Figure 5.14. The points inside the sphere of the current point.

Different methods can be utilized to analyze a point and its local neighborhood. One way would be to estimate how well the points fit to a plane by using a least mean square method. Another solution is to use PCA. By using the latter method we get more information about the local neighborhood. If we look at the eigenvalues from the PCA we have these cases:

1. If the largest eigenvalue is large compared to the other two then the points found in this region are approximately located on a line.

2. If the first two eigenvalues are large the points in the region are approximately located on a plane with a normal parallel to the eigenvector with the smallest eigenvalue.

3. If all three eigenvalues are large the points have no structure but are more or less spread out in 3D space.

The objects we want to detect in the first case are powerlines but if only a few points are found in the sphere they may also generate this case. The points
are also located naturally on lines because of the way the laser scanner works. In Section 2.6 different laser scanning patterns are shown.

The second case is common for flat ground and building objects. The eigenvector with the smallest eigenvalue is pointing in the direction of the normal of the plane the point belongs to. By segmenting neighboring points with similar eigenvectors we can estimate the roof segments and use that to generate 3D models of the buildings.

![Image](a) ![Image](b)

**Figure 5.15.** An interpolated $z_{max}$-image is shown to the left and to the right the smallest eigenvalues are shown. White corresponds to high values.

In Figure 5.15 the smallest calculated eigenvalues are shown for each point. Points that are part of vegetation get very high values. Edges of buildings also generate high values. To use this as a classifier for the points a segmentation method could be used. Points with similar normals should be grouped together to form segments. By removing segments that consist of very few points the flat surfaces sporadically found in vegetation can be removed easily. Segments that are part of vegetation appear sometimes but these segments consist of very few points. The segmentation algorithm just described is not yet implemented and is therefore subject for future work.
Chapter 6

Results

In this chapter we compare our developed algorithms with the currently used algorithms at FOI. The methods that are compared are the algorithms to detect outliers, the ground estimation algorithms and the classification algorithms.

6.1 Outlier detection

At FOI there are at the moment one method to remove outliers implemented. In this section a comparison is done between this algorithm and the one we have developed. The method to remove outliers which is in use today is described in Section 4.3.

6.1.1 Test regions

Two different regions are processed. Region 1 covers $60 \times 30$ m and contains mainly outliers of type 1. Region 2 covers $42 \times 39$ m and contains some outliers of type 2.

6.1.2 Test results

In region 1 the new algorithm identified 36 points as outliers. All of them are identified correctly. The result can be seen in Figure 6.1 where $z_{h2}$ and $z_{l2}$ are shown as horizontal lines. We have used a square, $s_1$, with sides of length 30 m. The old method only identified 2 points as outliers. The old algorithm adds 30 m to the calculated maximum surface and in this case that was too much. If we add a smaller value to the max surface we would increase the risk of identifying high points from the trees as outliers. The problem with this method is that the filtering which is done on the images removes a lot of information.

In region 2 the new algorithm found 3 outliers. The old one removed the same 3 points so in this case there were no differences. Figure 6.2 shows that there is one outlier that has not been identified. This is due to the fact that the square in our algorithm we have used is too large, in this case 42 m. If we decrease the size of the square the point will be identified as an outlier. Our method gives good
results in general while working on quite large squares but in rough and steep terrain it is a good idea to decrease the size of the square.

6.1.3 Computation time

When we look at the calculation time we see an enormous improvement. For a 120×120 m large region the old algorithm needed 75 seconds and the new algorithm needed less than 1 seconds.

Figure 6.1. The points from region 1 with $z_{h2}$ and $z_{l2}$ shown as horizontal lines.

Figure 6.2. The points in region 2. The diamonds are the points with the $z$-values $z_{l1}$ and $z_{h1}$, respectively. The horizontal lines represent $z_{l2}$ and $z_{h2}$.
6.2 Ground estimation

In this section the two ground estimation algorithms developed during this thesis are compared with the methods that have previously been developed and used at FOI. The new methods are here referred to as Active contours interpolated and Cone selection. The Cone selection algorithm does not create a surface like the others. Instead it just classify the points as ground or non-ground points. To illustrate the results the points classified as ground points are triangulated to get a visual ground surface. Both methods are explained in Section 5.3. The older algorithms are Active contours original and Watershed ground estimation. They are explained in Section 4.4.1 and Section 4.4.2, respectively.

6.2.1 Test regions

We will test the algorithms on some small difficult areas. The regions are chosen to illustrate different types of terrain. The 3 regions are:

Region 1 contains some big trees
Region 2 contains some houses and vegetation.
Region 3 contains a hill.

To simplify for the reader each area is shown together with the results from the ground estimation algorithms.

6.2.2 Test results

The results from the ground estimation methods can be seen in Figures 6.3, 6.4 and 6.5. By looking at the results from region 1 and 2 we can conclude that both algorithms that are based on active contours has a tendency to let the ground surface grow up in trees and buildings. In these two regions both Watershed ground estimation and Cone selection give a better result.

We can see that the Watershed ground estimation has failed to find a ground segment in the left part of region 2 and therefore just made an extrapolation in this area. Region 3 contains a small hill to illustrate this problem even better. The error occurs when the terrain is hilly because the Watershed ground estimation has problems to climb up the steep hills. Instead it will just cut the hill and interpolate. When estimating ground in real forest terrain this error occur quite often.

The Cone selection method has overall shown a good result. It does not grow up in trees and it finds the ground everywhere where the ground has been hit by the laser scanner. Due to the fact that we only use the points it will also be more exact even in easy open terrain and show ground details much better. This can be seen on the roads and the close surroundings of the roads in region 2. The Cone selection method does have problem to classify points on steep hills as ground points. These kind of steep hills do exist but we have not had the opportunity to analyze any laser data that included any such regions.
Figure 6.3. Ground estimation of region 1. Axes are in meters.
6.2 Ground estimation

Interpolated $z_{\text{max}}$

Figure 6.4. Ground estimation of region 2. Axes are in meters.
Figure 6.5. Ground estimation of region 3. Axes are in meters.
6.3 Classification

Some results from the classification algorithm developed during this thesis are shown in this section. A comparison with two, at FOI, existing algorithms is also found here. More information about these two algorithms can be found in Section 4.5. Figure 6.6 shows how the classification images should be interpreted.

![Class annotation](image)

**Figure 6.6.** Class annotation.

6.3.1 Test regions

Three different regions will be compared which includes areas where the older algorithms had problems and where they have succeeded well. The first two regions illustrates how we solve the problem with separating trees which are close to buildings. The last region is larger and illustrates how our classification algorithm performs in urban areas. Together with the results a correct, manually made, classification is shown.

6.3.2 Test results

As can be seen in Figure 6.7 both previously existing methods have problems to separate the tree from the building in the upper left corner. The tree height is about the same height as the building and they are standing very close to each other. Our classification algorithms uses PCA and analyzes the points to find the surfaces. It has shown to be a successful method and it has separated the two objects correctly. Another hard problem for the earlier algorithms is to find where to separate the long building from the big trees that stand close to this building. These trees are much higher than the building and with our method it is therefore easy to separate the trees from the building. In both of those cases our algorithm has done a better job. Otherwise our new algorithm and Watershed classification gives about the same results in region 1.

In Figure 6.8 the classification results from region 2 are shown. In region 2 both previous algorithms had problems to separate trees from the building in the middle of the scene. The new algorithm has shown a much better result in this area. It has failed to classify some small parts of the building correctly though.

Region 3 is a larger region, 400×600 m. The image classification results can be seen in Figure 6.9 and Figure 6.10. The main idea with this region is to see how our algorithm works in urban terrain. This region includes many small buildings which is time consuming for our algorithm to process and these kinds of regions
are most difficult to classify correctly. We can see here that the result is slightly better than than the result from the old algorithms.

Neural net classification has overall shown the worst result. This algorithm is based on neural nets and to give the best results for this algorithm it should be trained in an area similar to the one that is about to be classified. Usually a standard data training set is used and this is how it has been used during this comparison.

### 6.3.3 Computation time

So far the evaluation of the classification algorithms have focused on the level of correctness. In this section we will look at the computation time for each algorithm. The absolute times are dependent of the capacity of the computer. By running the algorithms on the same computer and on the same dataset the times can be compared. The computation times for the different algorithms are summarized in Table 6.1.

<table>
<thead>
<tr>
<th></th>
<th>Thread classification</th>
<th>Watershed classification</th>
<th>Neural net classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Region 1</td>
<td>0:15</td>
<td>0:16</td>
<td>0:31</td>
</tr>
<tr>
<td>Region 2</td>
<td>0:30</td>
<td>0:40</td>
<td>0:42</td>
</tr>
<tr>
<td>Region 3</td>
<td>16:00</td>
<td>23:00</td>
<td>27:00</td>
</tr>
</tbody>
</table>

*Table 6.1.* Computation times for three different classification algorithms evaluated in three regions (minutes:seconds).
Figure 6.7. The classification images for region 1.
Correct classification

Thread classification

Watershed classification

Neural net classification

Figure 6.8. The classification images for region 2.
Figure 6.9. The classification images for region 3.
Figure 6.10. The classification images for region 3 (continued).
Chapter 7

Concluding Remarks

In this chapter a summary of the master thesis is given and we present some of the ideas we have for future research.

7.1 Conclusions

During this master thesis we have implemented a new format to store laser point data. Instead of storing rasterized images of the data this new format stores the original location of each point. The format is referred to as the LPS format and it is possible to add additional information about the points.

We have implemented different methods that use the information from the LPS format. One method is used to detect outliers and we have shown that it is much faster and more flexible than the previous algorithm.

Two ground estimation methods have been implemented. One is based on the previous method that uses active contours and we have made modifications to decrease the processing time. Another method to estimate the ground surface that we have implemented is the cone ground estimator. It is faster and easier to understand how the parameters affect the result, compared to the previous ground estimation methods. It also gives the most accurate results in our tests.

In this thesis we have implemented a method to classify the points measured by the laser scanner. The classes used are buildings and vegetation. By giving each point a class instead of each pixel in a rasterized image, it is possible to correctly classify areas with trees that partly cover buildings. In our tests this method is faster and gives a more accurate result than the previous methods. This method works on both rasterized data and on the points directly. We have also been looking at a method that works on the points directly. It uses PCA to calculate an estimate of the local region around each point. This measure tells if the local neighborhood can be well approximated by a line, by a plane, or if the points are distributed in each dimension.

In this thesis we have shown that it is possible to get more accurate results by analyzing the points directly instead of only using rasterized images and image
processing algorithms. We have shown that these methods can be implemented without increasing the processing time.

7.2 Future research

The ground estimation method that uses cone selection has shown very good results. There are still areas where it fails, for example bridges and steep slopes. These situations have been briefly studied in Section 5.3.2, but must be further analyzed to generate a good result. The classification that uses PCA on the points in a sphere has also shown to give a lot of information. This method can in most cases separate trees from buildings and it has been able to find very small flat objects that all earlier methods failed to detect. How points should be grouped together to create segments of for example buildings and powerlines, must be analyzed more. We have also been thinking about changing the sphere to other objects. This can be needed to find the powerlines as the point density on the lines can be very low and the points in the sphere can be too few to use PCA. One simple solution can be to grow the sphere, but that will include the risk that a neighbor powerline is included in the sphere. Another way could be to use a cylinder or an ellipsoid instead of a sphere to decrease the risk that another powerline is included.

By using PCA it is also possible to put a certainty on the classified points. A point could for example be classified as 80% building and 20% vegetation. In the future several classifiers methods can be used together. A voting system can assign a final class to the points. The process that detects outliers could also add a certainty value to the points telling at which probability this is a correct point.

Classification and ground estimation are today two separated operations. When analysis is performed on point sets instead of images, we have seen that they are well connected. Therefore we propose that in the future ground estimation and classification should be merged into one step. The ground class is just another class of the points and when a ground surface is needed some kind of interpolation method should be performed on the points classified as ground points. One example of an interpolation method that we think would give a realistic and correct ground surface is spline interpolation. If a future classification method adds a certainty measure to the points this information could be used to give a different weight to the points when interpolating the surface.

Some of the newer airborne laser systems are able to not only measure several returns, but also to store the sampled waveform of the received reflected intensity. It would be interesting to see if classification methods can increase accuracy by using that information. At FOI, progress in this field has recently taken a giant step forward, and the results can be seen in [Nor06].
Bibliography


På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommerciell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Johan Nordin, Jakob Ruhe