Student thesis series INES nr 254

Automated Plane Detection and Extraction from Airborne Laser Scanning Data of Dense Urban Areas

Ning Zhang

2012
Department of
Physical Geography and Ecosystems Science
Lund University
Sölvegatan 12
S-223 62 Lund
Sweden

Automated Plane Detection and Extraction from Airborne Laser Scanning Data of Dense Urban Areas

---

Ning Zhang 2012

Master degree thesis, 30 credits in *Geomatics*
Department of Physical Geography and Ecosystems Science, Lund University

Supervisors

Karin Larsson
Department of Physical Geography and Ecosystems Science

Yubin Kuang
Prof. Kalle Åström
Centre for Mathematical Sciences

Lund University

Abstract

Digital 3D city models are used in many GIS applications. Manual digitization of 3D buildings is rather tedious, hence automated approaches are much preferred. In this study, a pipeline for automated extraction and estimation of building roof facets from LIDAR data are devised based on available methods and ideas, which is part of the entire frame work of automatic creating of 3D city models.

In this pipeline, the ground and non-ground points are separated using a graph cuts based method combined with EM algorithm only from elevation data. The tree points and other undesirable clutters are further excluded using the criteria derived from principle component analysis. Finally, the model parameters of roof facets are estimated with a RANSAC based method.

Extensive experiments have been conducted in the different steps of the pipeline. The splitting of data into subtiles of 25m by 25m can effectively reduce the influence of slightly varied topography of the test region. With a tuned spatial regularization parameter, graph cuts method can result in a more smooth classification, with some cars and low shrubs separated from buildings. The quality of extraction of roof points depends on both size of the neighboring radius and the choice of the threshold for the criteria. The kD-tree structure is used for searching neighboring points more efficiently. For the linear threshold classifier, 1m is an ideal radius size for filtering off non-plane points yet preventing the missing of smaller roof facets. In contrast, when applying graph cuts based classifier in detecting planes, even 2m radius does not result in missing points on the joint part between roof facets. The modified RANSAC algorithm can estimate each planar roof facet model from the data containing many potential models robustly and efficiently, however it cannot separate the facets that are fully coplanar. In this method, LIDAR data is the only necessary input. It does not need any other geographical data such as vector data or DEM.

Keywords: Geomatics; Geography; Physical Geography; LIDAR; Pattern recognition; Classification; 3D city modeling; Graph cuts; EM algorithm; kD tree; RANSAC

Acknowledgement

First of all, I would like to express my sincere gratitude to my main supervisor Karin Larsson, for her patient guidance and valuable advices in scientific research; and to my second supervisor, Yubin Kuang for his effective assistance with the mathematics used in this study and leading me out of the most difficult time of this period. And I would like to thank Prof. Kalle Åström for the enlightening discussion and reading of my thesis. And also, I would like to thank Harry Lankreijer, director of studies in dept. of Physical Geography and Ecosystems Science for allowing me to have the second supervisor.

Meanwhile, special thanks are given to Thomas Åkerholm from the government of Lund municipality, who provides the LIDAR data used for this study, and I also thank Dr. Lars Harrie for introducing me to Mr. Åkerholm.

My special thanks are extended to those who have offered me valuable helps during the period: Miso Iric offered me an internship opportunity in the government of Malmö municipality. His patient personal instruction on creating 3D city models was greatly appreciated. Yufeng Cui from SuperSIT Inc, Tianjin, China provided me with materials about 3D GIS and taught me how to develop ArcGIS Engine applications. Gang Li and Heng Zhang from Tianjin Urban Planning and Design Institute gave advices on the choice of topics and introduced how GIS is used in urban planning. Petter Strandmark, a PhD student in Mathematical Imaging Group, inspired me with an important idea. Hongxiao Jin, a PhD student in dept. of Physical Geography and Ecosystems Science, influences me with his great enthusiasm in scientific research. And Helena Eriksson provided me with a lot of administrative helps.

Last but not least, I would like to express my very great appreciation to Fredrik and Nina Lagerberg, and all the other members in the family, with whom I enjoyed the happiest time of my life in Sweden. Without their family-like concern and encourage, I could have not finished the master program.

Abbreviations and glossary

ALS	Airborne Laser Scanning
ASCII	American Standard Code for Information Interchange II
DEM	Digital Elevation Model
DGPS	Differential Global Positioning System
DLT	Direct Linear Transformation
DSM	Digital Surface Model
DTM	Digital Terrain Model
GIS	Geographical Information System
GML	Geography Markup Language
IMU	Inertial Measurement Unit
kD	k dimensional
LIDAR	Light Detection and Ranging
LADAR	Laser Detection and Ranging
LOD	Level of Detail
NaN	Not a Number
NDVI	Normalized Difference Vegetation Index
MAP	Maximum A Posteriori
PCA	Principle Component Analysis
RANSAC	Random Sample Consensus
SVD	Singular Value Decomposition
TIN	Triangulated Irregular Network
XML	Extensible Markup Language

TABLE OF CONTENTS

1. INTRODUCTION

Geographic information system (GIS) is a computer aided system that captures, stores, analyzes, manages and presents various types of geographical data (Worboys, 2004). Compared with conventional information systems, it focuses on solving problems related to spatial issues, such as geography, positioning and navigation. Technically, it is highly interdisciplinary covering computer science, cartography, remote sensing, geodesy, mathematics, information technology and management. However, different applications seem to have distinct forms of GIS, e.g., it can be associated with a research tool in geography, a graphic database on cadastral management or a real-time system on transportation query, etc.

Since it emerged, GIS has for long been mainly used in two-dimensional geographical data. However, in applications such as landscape analysis in urban planning, noise modeling, underground pipeline management, telecommunication (signal range simulation) and real estate marketing (integrate indoor structure into building model), which require immediate accessibility of the third-dimensional information, 2D GIS is not suitable. Many city governments and companies have shown great interest in combining GIS functionalities with 3D geographical data, especially of urban areas. In this context, developing automated methods for creating 3D city models has become a hot research topic in both academia and industry.

1.1 Objectives

Although many researchers have published their novel and cutting-edge methods, perhaps due to space limitations in the articles, or purpose of protecting their core technologies, those articles often fail to describe their methods in sufficient detail. There will be uncertainties when they are applied in real applications. Thus, this study seeks to make the following contributions: 1) developing an automated pipeline for extracting buildings and creating plane models for roof facets based on airborne laser scanning point cloud data; 2) evaluating the extraction effect with regard to different parameters; 3) provide the theory and procedures explicitly to make the method repeatable. The method is in no need of any other auxiliary data; all relevant features are derived solely from the 3D point data itself.

Chapter 2 presents a thorough literature review, from 3D data collection to the existing methods in industries and related researches in academia. Chapter 3 gives an introduction to the data used in this study. The explicit methodology and theory is described in Chapter 4. The experimental results and relative comments are presented in Chapter 5, followed by the discussion and conclusions in Chapter 6 and 7. Finally, some important but lengthy experimental results and some knowledge that are not closely related to the methodology are attached in Appendices.

1.2 Tools

There exist many powerful computer programs for automatically creating massive 3D city models. But they are normally in-house and commercial, which are normally extremely expensive and inaccessible. On the other hand, many open source software can be used for free in research, such as GRASS GIS and LAStools. However, those programs provide only general processing functionalities such as format conversion, rasterization, classifying ground and non-ground points, which greatly limits the extension feasibility, e.g., designing a user defined filter. Especially, although LAStools provides 27 efficient tools covering nearly all the commonly used functionalities, only 12 most basic ones are really open source (Isenburg, 2012). Owing to the obstacles, all the experiments in this study are programmed from scratch. Matlab serves as the main platform for the visualization and organization of different functionalities, where some open source C++ packages of key functionalities are integrated.

1.3 Limitations

Due to the time limit, the study focuses only on extracting roof points and generating plane models for roof facets, which is part of the entire process of automated 3D city modeling. The complete system for automated 3D city modeling is studied but has not been fully implemented. A proposed future work is however described in Appendix A.

1.4 Clarification on some terminologies

The term “modeling” in this context merely refers to creating geometric models for buildings, instead of a mathematical modeling of some process. For example, a plane can be modeled by $\{(x, y, z) \mid ax + by + cz + d = 0\}$, where (a, b, c, d) are the model parameters. And a “roof facet” refers to a planar surface that is part of a complete roof structure. All the terms such as “classification”, “segmentation”, “separation” and “clustering” refer to the same operation, which is merely dividing a set of data into groups according to some attributes of the data.

2. BACKGROUND

As is well known, 3D technology is commonly used in video games, films, and industrial animations and simulations. In 3D GIS, buildings and other urban facilities can be modeled as the graphs composed by 3D polygons. These urban facilities can include constructions, trees, bridges, roads and traffic signs, etc. In order to conduct spatial analysis accurately, the 3D models are normally required to be accurately georeferenced. This requires that the models should be generated from some raw 3D data that is also georeferenced. Section 2.1 introduces the two most fundamental methods for massive and rapid 3D raw data collection. In section 2.2, a Web exchange format for 3D city models is introduced. Section 2.3 focuses on the available methods in both industry and academia.

2.1 3D Data Collection

Airborne laser scanning (ALS) and 3D reconstruction from multi-view images are the two fundamental methods for collecting raw 3D data. The raw data refers to 3D scattered points representing the shapes of the target objects.

2.1.1 3D reconstruction from multi-view images

Given images of the same object taken from different positions, the three-dimensional coordinates of a point can be estimated from the two-dimensional information of the corresponding point in those images. The imaging mechanism of a modern camera is very complex. For mathematical tractability, we normally use *Pinhole Camera* (Figure 1) to model how 3D objects from real world are projected onto 2D images.

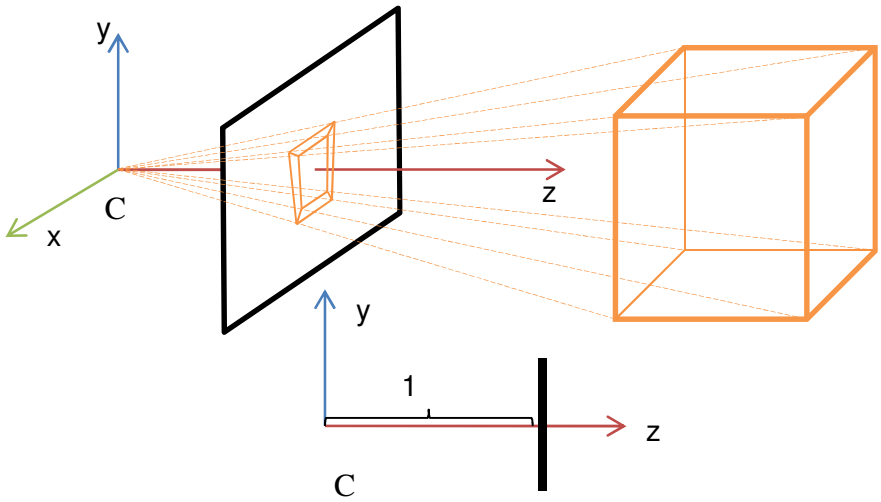


Figure 1. Pinhole camera model.

The pinhole camera model can be formulated by a 3x4 matrix

$$P = KR[\, I \mid -C] \tag{1}$$

where R is a 3x3 rotation matrix determining the camera orientation; C is the 3D coordinate of the camera center; I represents a 3x3 identity matrix; K is a 3x3 matrix consisting of the camera's internal parameters such as focal length, scaling and skewing ratios. Suppose that X is the *homogenous* coordinate of a point in the 3D world and x is the corresponding homogenous 2D image coordinate, i.e., $X = [x, y, z, 1]^T$ and $x = [a, b, 1]^T$, the relation between X and x can be formulated as

$$\lambda x = PX \tag{2}$$

where λ is a normalizing constant. For multi-view images, the projections of the same 3D point are formulated as a system of equations:

$$\begin{cases} \lambda_1 x_{1j} = P_1 X_j \\ \lambda_2 x_{2j} = P_2 X_j \\ \quad\quad ... \\ \lambda_n x_{nj} = P_n X_j \end{cases} \tag{3}$$

The problem of reconstructing $P_i\ (i = 1, \ldots, n)$ and $X_j\ (j = 1, \ldots, m)$ from image points x_{ij} is usually called *structure and motion*. One of the key theories to solving this problem is the epipolar geometry for two cameras, by which the relative pose of the cameras can be obtained for computing the 3D coordinate of X_j. Figure 2 illustrates an intuitive interpretation of epipolar geometry.

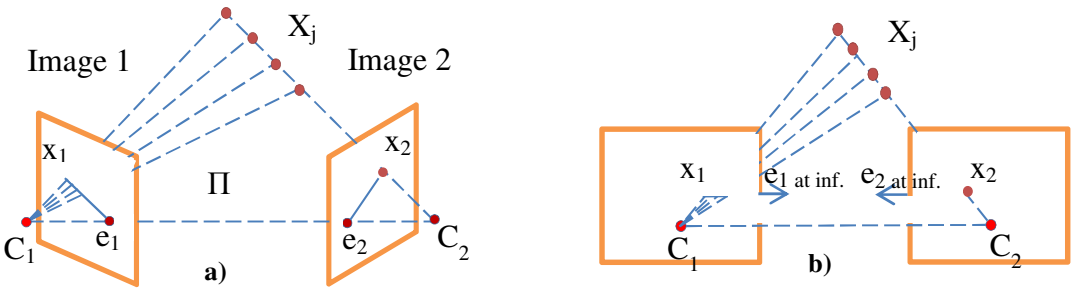


Figure 2. Epipolar geometry of two cameras. a) Converging cameras. b) Parallel cameras.

Suppose there are two cameras with the focal (center) points C_1 and C_2. The x_1 and x_2 are the projection of the 3D point X in the images taken by the two cameras. The e_1 and e_2 are called *epipoles*, which are the projection of C_1 and C_2 on their opponent image planes. Those points are actually coplanar. The imaginary plane Π intersects the two images at x_1e_1 and x_2e_2, which are referred to as *epipolar lines*. For two converging cameras, all the epipolar lines corresponding to different X_j should converge to the corresponding epipoles. On each image, the projected point of X_j must be on its epipolar line. Thus, the corresponding image points forms an *epipolar constrains*, by which the camera matrix P_2 are solvable conditional to an initialized P_1. Afterwards,

the 3D point X_j can be calculated by the intersection of the projection lines C_1x_1 and C_2x_2. This process is called *triangulation*. (Hartley and Zisserman, 2003)

By contrast, if the cameras are parallel, i.e., the two image planes are parallel, the epipolar lines are consequently parallel with each other in an image. The epipoles become vectors pointing to the direction parallel to the epipolar lines. Such a vector, with a form of $[x, y, 0]^T$ in homogeneous coordinate, is called *point at infinity* in the context of prospective geometry. In general, epipole is a point that is not expected to lie inside the area of visible image, even in the case of converging cameras. *"Epipolar geometry depends only on the relative pose (position and orientation) and internal parameters of the two cameras. It does not depend on the scene structure (3D points external to the camera)."*(Zisserman 2004).

Today, both parallel and converging cameras are applied in collecting massive 3D points of urban area. The former case is more common in traditional photogrammetry based methods. Normally, an aircraft mounted with camera and GPS flies back and forth in parallel courses above a certain region. The photos are taken successively during the flight and have a certain degree of overlap both along (forward overlap) and across (side overlap) the flying direction (Figure 3). The 3D coordinate of corresponding points on the two images can be calculated based on the epipolar geometry

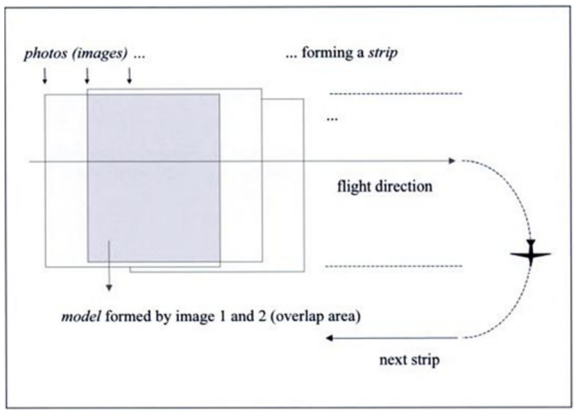


Figure 3. Courses of photography (Linder 2009).

An obvious characteristic of this method is that it cannot collect the same amount of information for vertical faces as for roofs owing to the use of vertical photography. The information on vertical faces can be supplemented by oblique aerial photos, as if the photos were taken from converging cameras.

2.1.2 Airborne Laser Scanning

Airborne laser scanning (ALS), commonly referred to as Light Detection and Ranging (LIDAR) or Laser Detection and Ranging (LADAR), is an optical and active remote sensing technology for generating point clouds that draws the shape of the land surface by emitting laser beams. The terms can be seen equivalent and are interchangeably used in the entire thesis. The necessary devices mounted on board an aircraft for

the scanning include scanner assembly, differential GPS (DGPS) and Inertial Measurement Unit (IMU).

Scanner assembly:

The sensor continuously emits laser pulses to the land surface and receives the reflected signals as the aircraft flies forward. As is shown in Figure 4, the range, which refers to the distance from the sensor to the target, is derived from the elapsed time between the emitting and receiving of a laser pulse. Meanwhile, the laser beams are swung across the direction of flight to a certain angle (Figure 5). Thus, the density of the measured spots depends on speed and flying height of the aircraft.

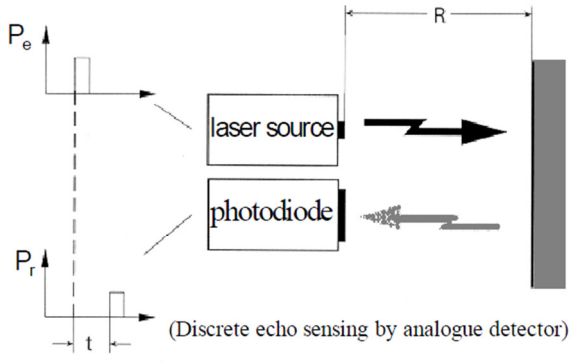


Figure 4. R = ct /2, R = range; c = velocity of light; t = elapsed time; Pe = emitting phase; Pr = receiving phase. (Wehr and Lohr 1999)

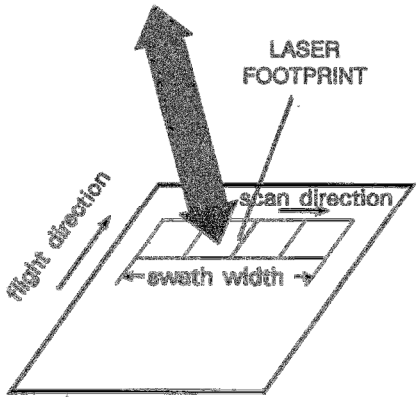


Figure 5. Swath of the scanning. (Wehr and Lohr 1999)

GPS and Inertial Measurement Unit (IMU)

In order to calibrate errors caused by atmosphere, off-line differential GPS (DGPS) is applied for calculating the geographical coordinate of the scanner (WGS84) every time the range is detected, with reference to ground stations on the earth. The IMU device provides necessary parameters to relate the GPS coordinate to the detected spots, such as velocity, acceleration, yaw angle of the aircraft, and swath angle of the laser beams. The coordinate of a detected spot on the land surface is calculated from the range, swath angle and the GPS coordinate. The rest of the parameters are used for compensating the positioning errors. Figure 6 demonstrates how the three devices interact in scanning land surface.

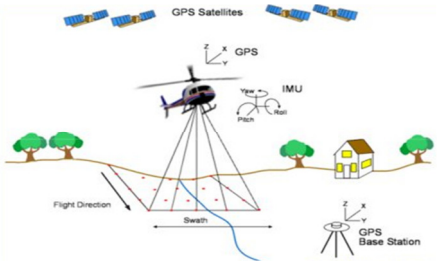


Figure 6. The principle of airborne laser scanning. (Mackinnon)

Compared with image based methods, ALS has two advantages: 1) ALS itself emits signals to detect the target objects, thus it is independent of day light. 2) Laser beam is highly energy-condensed light with very small radius. It can penetrate vegetation and detect the shape of the "bare earth", while traditional aerial photos are impossible to reveal the real terrain under foliaged vegetation canopies (Figure 7).

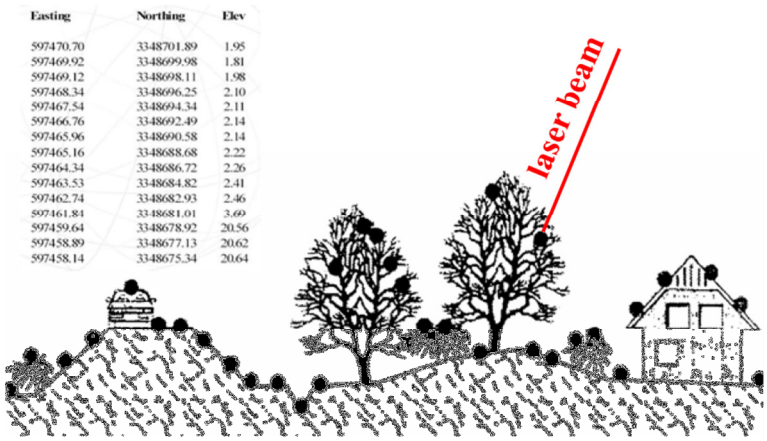


Figure 7. Laser beams can penetrate trees. (Rutzinger 2011)

2.2 Data format of 3D City Models

An evidence of the growing interest in 3D City could be the invention of CityGML (City Geography Markup Language), which is a data exchange format developed by the members of the Special Interest Group 3D (SIG 3D) of the initiative Geodata Infrastructure North-Rhine Westphalia (GDI NRW) in Germany. Based on XML (Extensible Markup Language) format, the CityGML standard combines both geometric and semantic information of city facilities, and defines five levels of details to reduce the unnecessary computations when the map is zoomed out (Figure 8).

Figure 8. The five levels of detail (LoD) defined in CityGML (Kolbe *et al.* 2005)

2.3 Creating 3D city models

In this section, some typical approaches, both automatic and non-automatic for 3D building reconstruction are introduced.

2.3.1 Methods in industry

Manually digitization of 3D city models is still widely used in industry. For instance, Tianjin Free Trade Zone, a district in Tianjin municipality, China, digitizes the local 3D virtual city by combining several desktop 3D modeling programs. First, 3D buildings are roughly modeled as boxes using ArcMap by adding a field of elevation to the attribution table and assigning each building polygon a representative height from existing cadastral data. Afterwards, those 3D boxes are imported to 3Ds Max and "sculptured" into their real appearances according to the photos personally taken by modeling workers. Meanwhile, trees and other urban facilities such as traffic lights and signs are only modeled in approximate sizes. Finally, the modified 3D models are imported to ArcGIS platform and visualized in ArcGlobe or ArcScene. ArcGIS has its own 3D data format named *Multipatch*, where polygons that belong to the same building are grouped into one integrated graphical object by default. A terrain model for the urban area is derived from DEM data. Finally, with trees, traffic lights and signs, and other 2D vector data such as traffic network imported altogether into the ArcGIS, where 3D spatial analysis tasks are able to be implemented with the functionalities offered in ArcGIS environment.

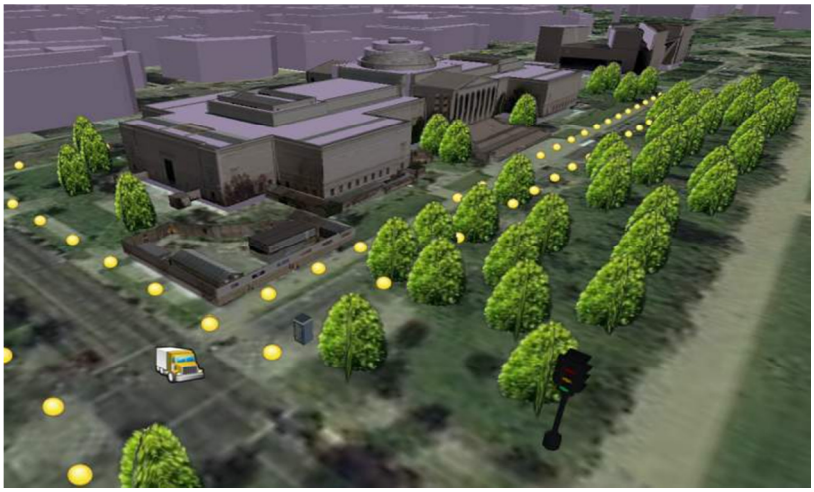


Figure 9. 3D city models in ArcGIS environment.

As another example, the GIS and Land surveying department in the government of Malmö municipality is responsible for collection, purchase and management of geographical data of Malmö. They use different desktop programs such as Espa Systems, AutoCAD Map3D/Civil3D, and Google Sketchup. Espa Systems is a software suite for digital photogrammetry and LIDAR data processing. Roof outlines and ridges can be digitized and automatically transmuted into basic roof models of predefined types in the roof library (Figure 10). The complex roofs that cannot be found in the roof library then have to be drawn individually in AutoCAD Map3D or Civil3D. They completed roof models and exported to Google Sketchup. Roofs of different LODs are saved in different layers. Since vertically directing aerial images have little infor-

mation on building's walls, the walls are simply the extensions of the roof's outline vertically down to the earth. The 3D complete building model ends up with the intersection by the ground model which is TIN derived from existing DEM data. Figure 11 demonstrates the digitization environment and roof models in Google Sketchup.

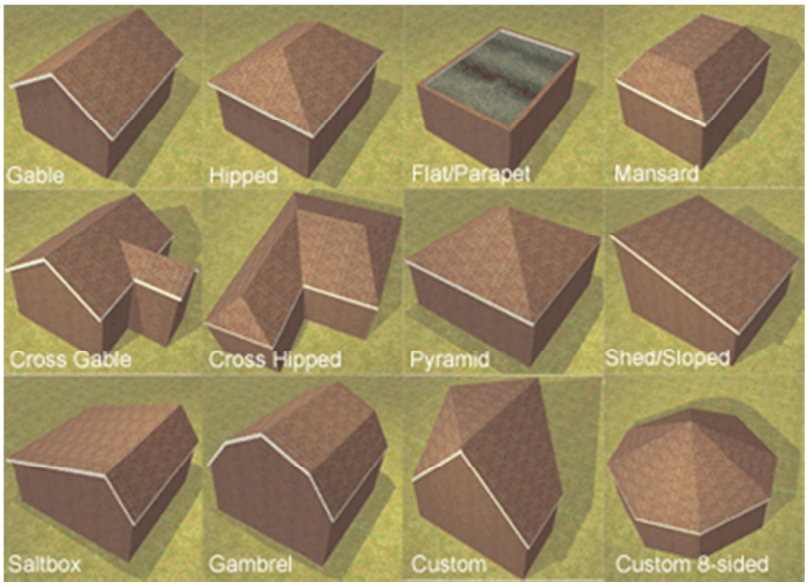


Figure 10. The predefined basic roof types in a digitization program. (Plan3D, 2011)

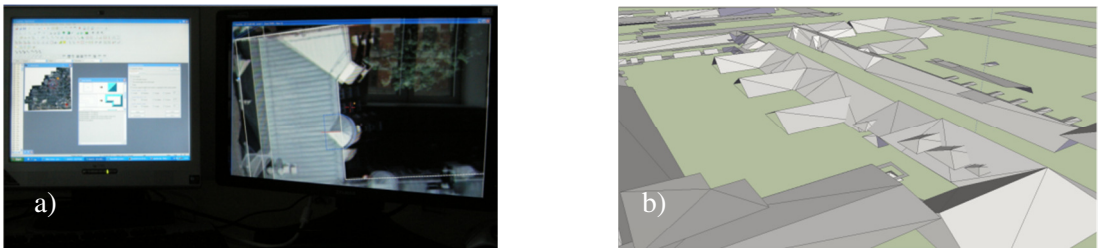


Figure 11. a) Digitization of roofs with stereo-pair images. b) Roof models

There are also automated approaches based on complete multi-view reconstruction pipeline. C3 Technologies AB (has been acquired by a company from the U.S.) is specialized in producing 3D city maps from multi-view aerial photos. Five cameras of different orientations (one vertical to the ground and the other four oblique to opposite directions) are mounted on the aircraft. The corresponding points are detected automatically using Scale-invariant feature transform (SIFT). 3D triangle meshes over the points are automatically and correctly generated with the aid of colors in the images. And finally, all the meshes are rendered by the color of the images. The entire process is said fully automatic. And their models can be highly detailed and photorealistic (Figure 12).

Figure 12. Photorealistic 3D models produced by C3 technologies AB. (Kim 2011)

2.3.2 Related researches

Kada and McKinley (2009) developed a LIDAR based 3D building reconstruction approach with the aid of existing vector data of building footprints, which was applied for 3D reconstruction of the entire cities of East Berlin and Cologne, Germany. They designed an algorithm partitioning a building's footprint into nonintersecting, mostly quadrangular cells. Each cell is then assigned with a parametric roof model that fits the LIDAR points above the piece best. Figure 13 demonstrates the process of their method.

Figure 13. a) Building footprint and its decomposition into cells. b) LiDAR points inside the cell colored according to their local regression plane and the best fitting roof shapes.

Vosselman (1999,2001) applied Hough transform and connected component analysis, to detect and classify buildings. "main building orientation" is derived and used as a constraint for the orientation of build's edges. The building's regularities are controlled by geometric constraints.

Vosselman and Dijkman (2001) introduced vector data of ground plan as auxiliary information to their former approach. The building area of the ground plan is split into smaller rectangular area, where the Hough transformation is performed. The final faces are reconstructed using a "split-and merge" approach.

Mass and Vosselman (1999) predefined a seven parameter model for a simple gabled roof primitive, and solved the parameter using first-order and second-order moments of the original laser points.

Verma *et al*. (2006) devised a novel automatic method based on an adjacency graph representing topological information between segmented roof facets. It first removes all the points that are not distributed as a plane by principle component analysis, and uses connected component analysis to group all the points into individual roofs and ground based on their vicinity. The group with largest number of points is labeled ground points. And then, the points in each entire roof are regrouped according to the planar patch they belong to. The adjacency graph is constructed by the planar patches (vertices) and whether they are connected (edge). The connections (edges) are labeled as O+, O- , S+ and N, where O+ and O- refers to that two roof facets are orthogonal convex and concave corner in 2D view respectively, S+ means the ridge of a roof, and N for all other connecting cases. The entire adjacency graph will contain subgraphs that represents predefined simpler roof structures, such as U-shaped, L-shaped and hipped primitives. The points are then labeled according to the belonging vertices in the graph. Those points that cannot be labeled will be removed. The final model is refined by minimizing a non-linear energy function composed by the sum of the distances of the estimated height of the roof model to all the corresponding points, with constrains on symmetries and right angles.

3. DATA

In this chapter, the general information about the LIDAR data is introduced. An overview of the data is provided in Table 1.

3.1 Area

The LIDAR data used in this study is a 250m by 250m tile covering an area of the city center in Lund, Sweden (Figure 14).

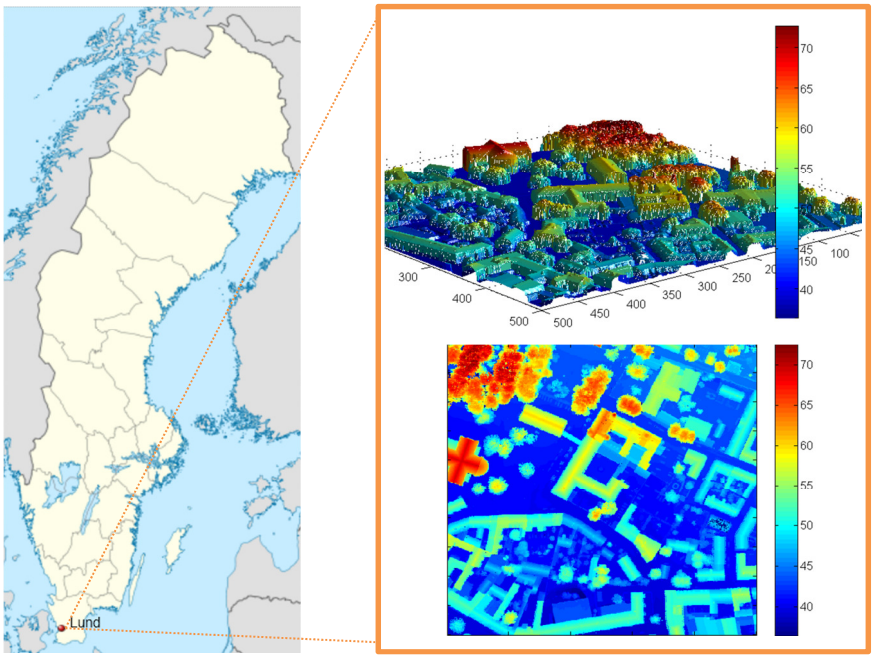


Figure 14. The height map of the experimental region, which is a 250m by 250m data tile within the center of Lund, Sweden. The color scale demonstrates the elevation (meter).

Location	Center of Lund, Sweden
	(55°42'09.40" N, 13 °11'38.08" E)
Area	250m x 250m
Number of Points	2966837
Density	47.4694/m^2
Geodetic referencing system	SWEREF 99 13.30
Height system	RH 2000
Format	ASCII
Attributes	3D coordinates, reflectance

Table 1. The general information about the LIDAR data.

Located in County of Skåne, southern Sweden, Lund is originated from approximetly 990 A.D.. A number of old buildings in the city have typical historical-style roofs, such as gable roof, gambrel roof, mansard roof, hipped roof, shed roof and their combinations (Figure 10).

Resembling many other European cities, Lund is not established on a plain. The topography is slightly varied. The elevations in the entire city contain an almost linear trend, by which the south part of the city is up to 80m lower than the north part (ThomasÅkerholm, personal communication, 26th Sept, 2011).

3.2 Format

The data is in the format of American Standard Code for Information Interchange II (ASCII). Figure 15 demonstrated that the ASCII format is actually a neatly arranged table in only plain text. Each row in the table stands for a data record and each column represents an attribute. The attributes comprise 1) classes of the points (the provided classification is poor and of no use), 2) three-dimensional coordinates and 3) reflectance. An advantage of ASCII format is that the values can be easily browsed and edited by any text editor. The data in ASCII can be imported and parsed using I/O operation by any programmable, graphics-supported software such as ArcGIS, Matlab or Google Sketchup. However, parsing of ASCII data is normally very slow and the file size can be extremely large. On the contrary, LAS format, which is a binary format widely supported by LIDAR processing programs, is more efficient in data processing and need much less storage memory. (ASPRS Online, 2010).

```
1 130037.900 6176200.760 45.770 130
1 130038.960 6176200.830 47.100 20
1 130038.980 6176200.810 46.600 180
1 130039.000 6176200.790 46.100 200
1 130039.040 6176200.780 45.640 70
1 130037.460 6176200.640 42.850 120
1 130037.760 6176200.640 42.830 170
1 130038.310 6176200.670 43.250 330
1 130039.030 6176200.720 43.990 20
1 130039.070 6176200.700 43.530 190
1 130039.080 6176200.680 43.000 180
1 130037.870 6176200.550 40.290 190
1 130037.950 6176200.540 39.880 230
1 130039.090 6176200.620 41.350 20
1 130038.260 6176200.540 39.870 200
1 130039.090 6176200.600 40.810 70
```

Figure 15. LIDAR data in ASCII.

3.3 Inconsistency

The LIDAR data have a certain inconsistency with vector cadastral data in representing the same object that mainly comes from the different data collecting mechanisms. The cadastral vector data are measured according to buildings' footprints, whereas LIDAR measures buildings on the roofs, which are normally larger than the footprints. Figure 16 illustrates a topological test of the LIDAR points of Lund Cathedral com-

pared by polygon measured by GPS. The points outside the polygon are shown in Figure 16 c), forming an outline of the building. The widths of this outline are measured by sampling five segments (red circles) from the points. Table 2 shows that the widths of the extended part of LIDAR data compared to the vector data are smaller than 0.5m. Since not too large, this error could however be negligible in some applications that are not strict with accuracy.

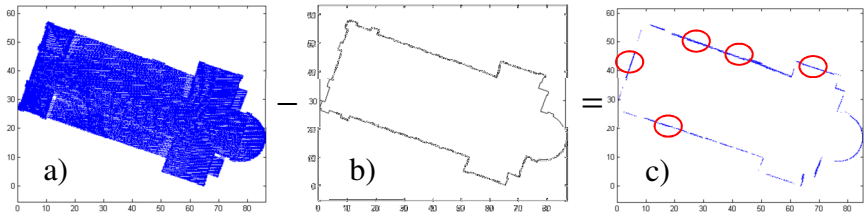


Figure 16. Inconsistency between LIDAR data a) and cadastral vector data b). There are 117298 points belonging to Lund Cathedral, of which 2766 points lie outside the area of the polygon c). The red circles represent the five sampled segments from the points in c).

	Segment 1	**Segment 2**	**Segment 3**	**Segment 4**	**Segment 5**
width (m)	0.2293	0.4660	0.1472	0.1213	0.1151

Table 2. The widths of the 5 sampled segments from the points in Figure 16 c). All of widths are smaller than 0.5m.

4. METHODOLOGY

In this chapter, the pipeline converting the raw 3D points into roof facet polygons is explicitly described. For classification of ground and non-ground, a graph cuts based classifier combined with EM algorithm is described in section 4.1. The roof points extraction based on PCA technology is described in section 4.2. Finally, a modified RANSAC algorithm for automatic generation of roof facet polygons is described in section 4.3.

Figure 17 demonstrates the entire pipeline. Besides the LIDAR data itself, the pipeline does not need any other auxiliary data such as remote sensing image, DEM or vector data of constructions. The pipeline is only part of the process of automatic 3D city modeling. The last arrow implies that there are still some successive procedures before the complete 3D city models can be created.

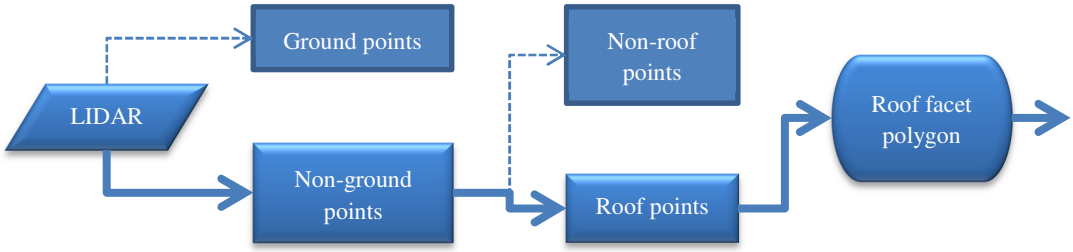


Figure 17. The flow chart of the pipeline.

4.1 Step 1: Classification of ground and non-ground areas

In the first step (Figure 18), the ground points need to be excluded from the data set because they are not used in the following steps. For the urban area established in plain, where ground heights are normally consistent on different areas, the ground and non-ground points can be simply separated by setting a threshold on heights. However, as is mentioned in the previous sections, the test region has a varied topography. Thus, the method applies the graph cuts with necessary parameters automatically found by EM algorithm. The method relating to the graph cuts and its formulation are introduced from Section 4.1.1 and 4.1.3. The EM algorithm and its derivation are described in Section 4.1.4.

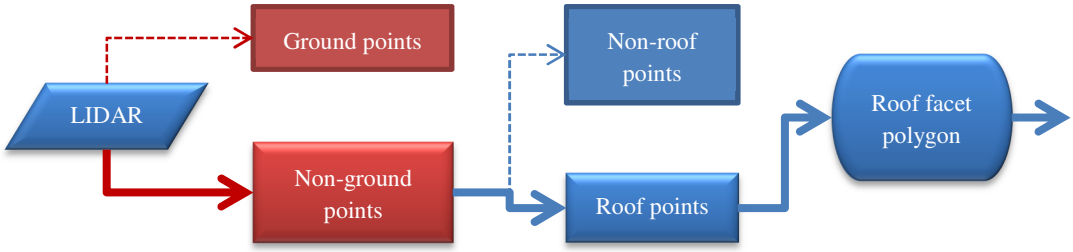


Figure 18. The red blocks and arrows denote the current stage in the pipeline.

4.1.1 Graph cuts and maxflow-mincut theorem

There are a variety of classification methods, from the simple methods, e.g. parallelepiped classifier (Lillesand *et al*, 2004), k-means, to the more sophisticated methods, e.g. parametric method: Gaussian mixture model with Expectation Minimization (EM) classifier, non-parametric: SVM classifier, random forest classifier, etc. A common characteristic of these classification methods is that the decision on the classification of an object is independent from that of its neighboring objects. In result, the classification usually ends up with a "salt and pepper" effect, i.e., there are many isolated single object of a class surrounded by neighboring objects of another class. Another type of classification method takes the spatial influence into consideration. The classification of an object is influenced by the class of its neighbors. Consequently, the classification of a datum depends on both of its own value and the neighbors' classes, so that many isolated objects will therefore be assimilated into their surroundings, which results in smoother boundaries between classes.

4.1.1.1 The mathematical formulation of a classification problem

The principle of graph cuts and its formulation has been described by Kolmogorov *et al.* (2004) in detail. Mathematically, a classification problem comes down to a mapping from a data set *P* to a set of labels (or classifications) *L*. The definition of a datum varies in different contexts: it can refer to a pixel in an image, or a 3D point in point clouds data. The solution of a classification problem can be seen as finding a labeling *f* that minimizes some energy function with regard to the data value and the labeling. For a classification without spatial influence, the energy function in a general form is:

$$E(f) = \sum_{p \in P} D_p(f_p) \tag{4}$$

The data term $D_p(f_p)$ is the penalty or cost of assigning some label f_p to pixel p given the data values. The resulting labeling is a vector of labels for the entire data, e.g., $f = \{1,\ldots,|L|\}^{|P|}$. Solving such a problem is normally straightforward. For each datum p, we need only to choose the label that minimizes $D_p(f_p)$, e.g., assigning the point to the cluster with the closest centroid in k-means algorithm, where $D_p(f_p)$ is the Euclidean distance between p and the centroid of the cluster f_p. In the case where the influence from neighbors is considered, (4) should be supplemented with a second term and become

$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{p,q \in N} V_{p,q}(f_p, f_q) \tag{5}$$

where $p, q \in N$ denotes p and q are neighbors under some user-defined neighboring system (Section 4.1.1.5), and $V_{p,q}(f_p, f_q)$, named smoothness term or *spatial regularization*, which penalizes the assignment of two different labels to the adjacent data p

and q. However, minimizing (5) turns out to be a combinatorial optimization problem which is expensive to solve using exhaustive enumeration. Some general optimization techniques, such as simulated annealing, can solve the problem in theory but is very slow in practical because they require exponential time (Kolmogorov and Zabih, 2004). Fortunately, Greig *et al*. (1989) originally applied the *max-flow min-cut theorem* to obtain the maximum a posteriori (MAP) estimate of a binary image. The resulting estimate is a binary image smoother than the original one. Relevant algorithms based on this idea has been well developed in the last ten years and today they are able to solve the problem in polynomial time (Kolmogorov and Zabih, 2004).

4.1.1.2 Max-flow min-cut theorem

As the prerequisite knowledge, the max-flow min-cut theorem, proved by Ford and Fulkerson (1956), should be introduced. Suppose there is a flow network modeled as a *directed graph* $G = (V, E)$, where V stands for the set of vertices in the graph including a *source* s and a *sink* t; and E stands for the set of mono-directional edges connecting all the adjacent vertices. A flow is generated from the source s and terminated in the sink t, passing through all the nodes and edges. Each vertex except s and t receives and emits always the same amount of flow. Each edge has a capacity $c(u, v)$ that limits the amount of flow passing through. An *s-t* cut refers to the cut of a certain set of edges in such a way that no flow can pass from the source to the sink. More precisely, it partitions all the vertices into two disjoint subsets S and T, where $s \in S$ and $t \in T$. The *capacity* of the *s-t* cut refers to the sum capacities of all the cut edges, i.e.,

$$c(S,T) = \sum_{\substack{u,s\in S, v,t\in T, \\ (u,v)\in N}} c(u,v) \tag{6}$$

The max-flow min-cut theorem can be described as the rule that the maximum amount of the flow passing from the source to the sink is determined by a combination of the edges with the minimum sum capacities meanwhile satisfying the condition that cutting those edges causes an *s-t* cut. This combination can therefore be found by maximizing the flow. Figure 19 gives an intuitive interpretation of the theorem. In a simple case in **a)**, it is obviously that the maximum flow is determined by the capacity of the bottleneck edge. This property also applies to the more complex networks such as in **b)**. But the bottle necks cannot be found by intuition in this case. Instead, the solution should rely on some sophisticated algorithms, of which, Edmonds-Karp algorithm, developed by Ford and Fulkerson, has the complexity of $O(VE^2)$ time, while the push-relabel algorithm (Goldberg *et al*, 1986) achieved a complexity of $O(V^2E)$. The latter is more efficient because vertices are usually far fewer than edges in a complex graph.

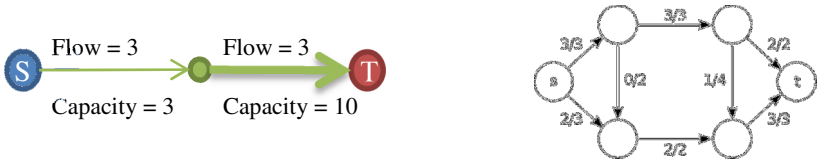


Figure 19. Flow in a graph: a) a simple graph; b) a complex network.

4.1.1.3 Energy minimization using graph cuts

In the theory of Greig *et al*. (1989), an image can be modeled as the aforementioned flow network, where each pixel is denoted by a vertex in the graph, and each edge connects two adjacent pixels under a user-defined neighboring system (section 4.1.1.5). The capacity of an edge is substituted by the cost of the energy function (5) in a certain way. Consequently, the solution for minimizing the energy function (5) is imaginatively associated to the minimum cut in the flow network. Since the energy minimization is modeled by cutting a graph, the method is named "graph cuts". In addition to the high efficiency, graph cuts method computes the global minimum to certain types of energies (Boykov *et al*, 2001), which is superior to some existing algorithms that can only find one of the local minima.

In this study, an open source C++ package MAXFLOW is used in the computation of the max-flow/min-cut of a graph, developed by Boykov and Kolmogorov. The program provides also an API for Matlab. The technical details about the algorithm are described in Boykov *et al.* (2004).

4.1.1.4 Data structure for graph

A graph can be stored in *adjacency list* or *adjacency matrix*. In an adjacency list, each object is associated with a list of all its neighbors. Suppose that there are n objects, each of which has m neighbors, an adjacency list needs to take up $n \times m$ units of memory. While an adjacency matrix is an n by n upper triangular matrix with each entity M_{ij} satisfying

$$M_{ij} = \begin{cases} 1, & i,j \in N, i < j \\ 0, & i,j \notin N, i > j \end{cases} \quad (7)$$

It takes up $O(n^2)$ units of memory, which is very memory consuming and impractical when n is large. On the other hand, it takes only constant time to index a neighboring object, whereas in an adjacent list it requires a searching operation of linear time. Overall, adjacency list is more commonly used in practice. (Cormen *et al.*, 2001)

4.1.1.5 Neighboring system

Graph cuts based method was originally devised for problems in image analysis and computer vision. Thus, the typical neighboring systems are based on image pixels, e.g., 4-connectivity and 8-connectivity (Figure 20). The nature of the neatly arranged pixel structure allows the neighbor indexing taking only a constant time.

For randomly distributed points, the neighbors of a point can be defined by all the points with the Euclidean distance to this point smaller than a radius (Figure 21), as

was used in Lafarge *et al*. (2011). However, such a neighbor system has two potential drawbacks: 1) the neighbors of a point are randomly distributed, so there does not exist a fixed structure to index neighbors in a constant time. Instead, the neighbors have to be searched in a certain way. A naïve algorithm will need up to $O(n^2)$ time to search neighbors for all the points, which is very cumbersome for high resolution LIDAR. 2) The number of neighbors is not consistent and much dependent on the size of the radius. A large radius can guarantee a point having sufficient many neighbors. But this results in too many redundant edges in the graph and consequently may slow down the computation. If the radius is too small, some points will have no neighbors at all. Nevertheless, this neighboring system is used in this study because this is the only choice so far for irregular distributed points. The searching time of neighbors can be reduced to $O(\log n)$ if the kD-tree structure is applied (Worboys *et al.* 2004). The introduction to kD-tree is given in Appendix B.

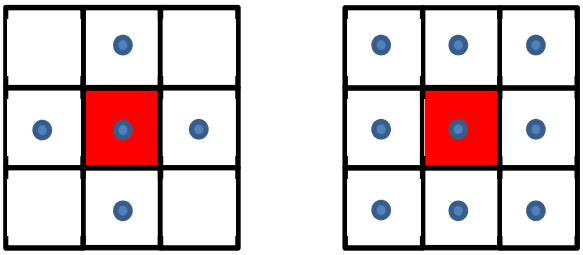


Figure 20. Neighboring systems in image.

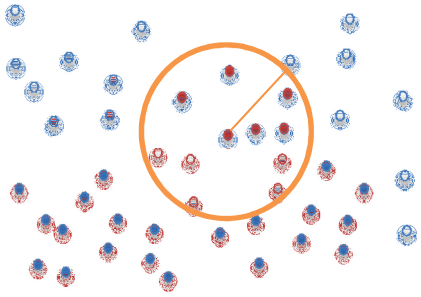


Figure 21. A neighboring system for randomly distributed points.

4.1.2 Split the region into smaller areas

As was mentioned in chapter 3, there is basically a southward linear trend over the entire city of Lund. It is doubtable whether a single mean height is adequate to represent the expectation height of the entire region. A common sense in spatial statistics says that the *stationarity* is always stronger in a subarea than in the entire area, i.e., the elevation difference decreases as the area shrinks. In this sense, by splitting the map and processing them separately, as the assumption for the model above fits better, one might generate a more promising result. To be more specific, the data is split up into subtiles of 25m by 25m (Figure 23), which is the size that can guarantee most of the subtiles include both ground and non-ground points with the assumption that most buildings are smaller than this size. In addition, the sparse urban areas where all the points belong to ground can be judged by putting a threshold on the smallest height difference within the subtile. Figure 22 compares the distributions of the elevation in the entire data (Left) and one of the subtiles (Right). The most dominate peak on the left side of each histogram corresponds to the distribution of ground points. The elevation variance for ground is much smaller in the subtiles than that in the entire data.

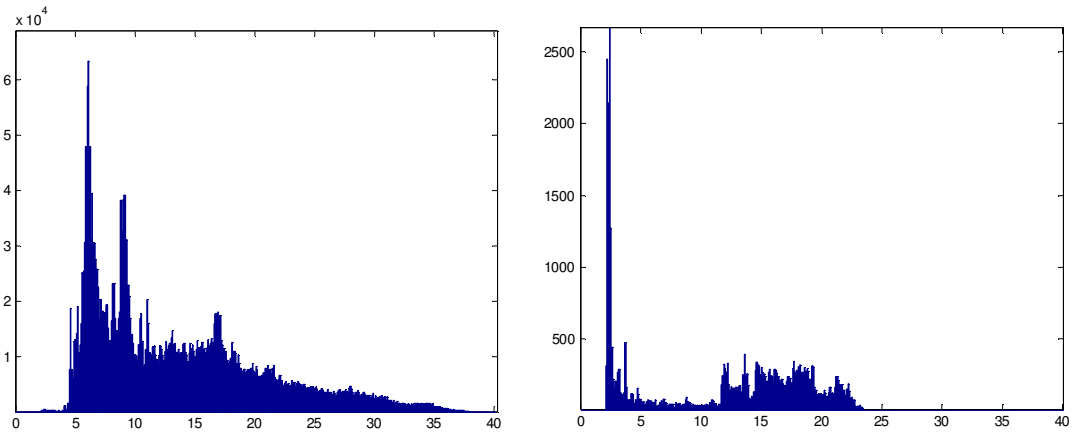


Figure 22. The histograms of the elevation in the entire test region (Left) and one of the subtiles (Right). The most dominate peak on the left side of each histogram corresponds to distribution of ground points. The variance is much smaller in the subtile.

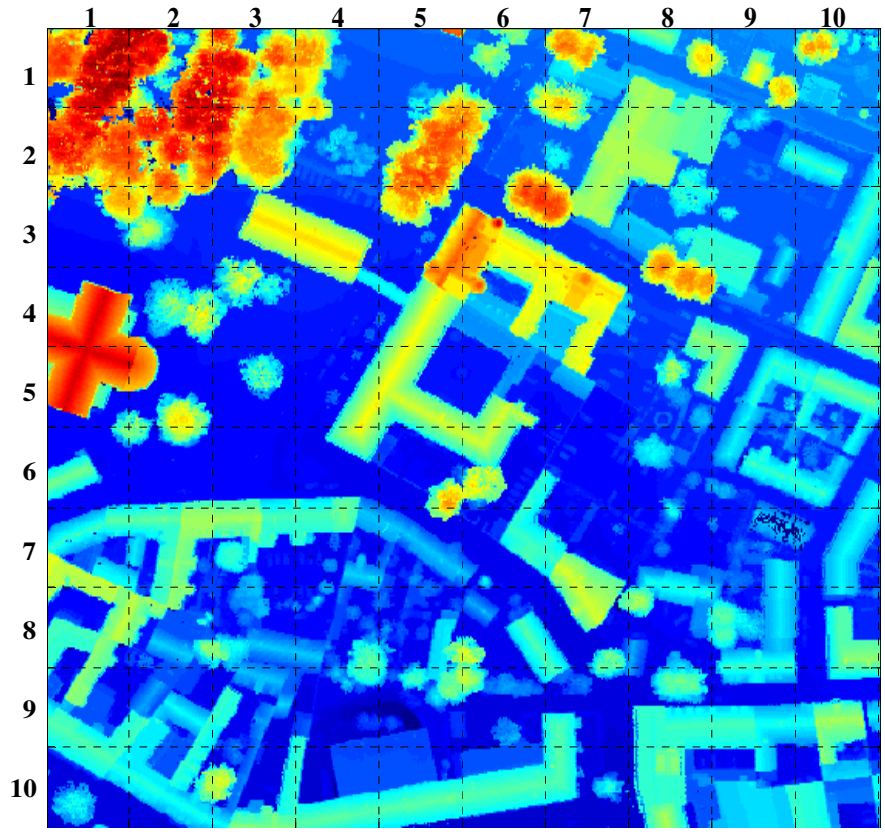


Figure 23. The test region is split into 100 subtiles, each of which has the size of 25m x 25m. The axis labels the relative locations for each subtile.

4.1.3 Defining data terms

The energy function (5) has been specifically defined for separating ground and non-ground points in this study, which is a modified version of Mumford-Shah functional. Section 4.1.3.1 gives a general introduction of Mumford-Shah functional. The

modification of Mumford-Shah functional specifically for the problem is described in section 4.1.3.2.

4.1.3.1 Mumford-Shah functional

A simplified version of Mumford-Shah functional by Mumford and Shah (1989) is formulated by

$$E(\gamma) = \iint_{\Gamma} (I(x) - \mu_1)^2 dx + \iint_{\Omega \setminus \Gamma} (I(x) - \mu_0)^2 dx + \lambda \text{length}(\gamma) \qquad \textbf{(8)}$$

where $I(x)$ denotes height at the position x. The first two terms and the third term corresponds to the data term and smoothness term respectively. The functional reflects the idea that the ground and non-ground points have the mean heights of μ_0 and μ_1 respectively. If a point's height is more close to μ_0, it will be probably labeled as ground, vice versa. The γ denotes the boundary between the ground and non-ground areas. It is not difficult to understand that the smoothness of resulting classification image has a direct relation to the sum of the length of the boundaries, i.e., less "salt and pepper" effect corresponds to γ of a shorter length. Figure 24 demonstrates the notations in (8).

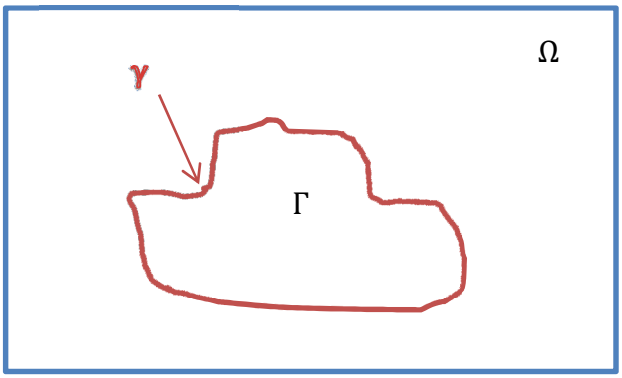


Figure 24. A demonstration to the notations corresponding to image segmentation using Mumford-Shah functional.

Since either a digital image or a set of LIDAR data is discrete, the functional is converted into the discrete form:

$$E(\theta) = \sum_{i=1}^{n} \theta_i (I(i) - \mu_1)^2 + \sum_{i=1}^{n} (1 - \theta_i)(I(i) - \mu_0)^2 + \frac{\lambda}{2} \sum_{i=1}^{n} \sum_{j \in N_i} \mathbb{1}(\theta_i \neq \theta_j) \qquad \textbf{(9)}$$

Where and $I(i)$ denotes the height value of the ith point and θ_i is defined as:

$$\theta_i = \begin{cases} 1, & \text{if pixel } i \text{ is nonground} \\ 0, & \text{if pixel } i \text{ is ground} \end{cases}$$

4.1.3.2 Changing data terms in Mumford-Shah functional

The distribution of the heights in a subtile is shown in Figure 22 (Right), where the prominent peak to the left corresponds to the ground heights and the rest of the histogram represents non-ground points. This height pattern can be modeled by a mixture

statistical model consisting of the two Gaussian distributions with different means and variances:

$$\begin{cases}\Theta_0 = (\mu_0, \sigma_0^2) \\ \Theta_1 = (\mu_1, \sigma_1^2)\end{cases}$$

that govern the heights of ground and non-ground points separately, where μ_i and σ_i^2 denote the mean and variance for the model i. The probability density of a height value h_i belonging to model k is formulated by

$$f(h_i|\Theta_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{(h_i - \mu_k)^2}{2\sigma_k^2}\right) \quad \textbf{(10)}$$

In this study, the data term of Mumford-Shah functional defined in (9) is replaced with (10). Notice that when using probability density function as the data term, a point with height value h is regarded more likely to belong to ground if $f(h|\Theta_0) > f(h|\Theta_1)$, which implies the new data terms should be *maximized.* To convert the data terms to be consistent with the *minimization* of the energy function, we can however inter-change the data terms, i.e., assigning $f(h|\Theta_0)$ to the data term of non-ground points and vice versa, as is shown in (11). In addition, the new data term can potentially re-sult in a more accurate classification since both mean and variance of the heights are considered in the model.

$$E(\theta) = \sum_{i=1}^{n} \theta_i f(I(i)|\Theta_0) + \sum_{i=1}^{n} (1 - \theta_i) f(I(i)|\Theta_1) + \frac{\lambda}{2} \sum_{i=1}^{n} \sum_{j \in N_i} \mathbb{1}(\theta_i \neq \theta_j) \quad \textbf{(11)}$$

The Gaussian mixture model can be automatically estimated by EM algorithm, which is explicitly described in section 4.1.4.

4.1.4 Finding mean and variance automatically using EM algorithm

The most ideal mixture model for the heights can be automatically found by Expecta-tion-Minimization (EM) algorithm. In brief, EM algorithm is an iterative method that implements the following two steps alternatively, which are 1) reclassify data using the latest model parameters (means and variances) and 2) updates the parameters based on the classification of the data until the parameters converge. The model pa-rameters are often initialized empirically or based on the solutions of other methods. In this section, the principle and mathematical formulation of EM algorithm are ex-plicitly described based on Lindgren (2005), combined with the context of separating ground and non-ground points. A Matlab package written by Lindgren is used in the programming.

4.1.4.1 Introduction to EM algorithm

Let $\mathrm{x} = \{x_1, \dots, x_n\}$ stands for the n relative height values and $x_i \geq 0$, $\mathrm{z} = \{z_1, \dots, z_n\}, z_i \in \{0,1\}$ for the labeling for ground points (0) and non-ground points

(1), Ψ for the universal set containing all the parameters, $\pi_{z_i} = p(z_i|\Theta_{z_i})$ for the prior probability of pixel *i* belonging to class z_i and $\Theta_{z_i} = (\mu_{z_i}, \sigma_{z_i}^2)$ for the parameters of the Gaussian model of class z_i. Thus, the joint distribution for both the observations and labeling is formulated by:

$$p(\mathrm{x}, \mathrm{z}|\Psi) = \prod_{i=1}^{n} p(x_i, z_i|\Theta_{z_i}) = \prod_{i=1}^{n} p(x_i|z_i, \Theta_{z_i})\pi_{z_i} \tag{12}$$

In each iteration, the parameters of interest are the ones maximizing the expectation of the log-likelihood of $p(\mathrm{x}, \tilde{\mathrm{z}}|\Psi)$ conditional to the old parameters estimated from the previous iteration, i.e.,

$$Q\left(\Psi, \Psi^{(t)}\right) = E\left(\ln p(\mathrm{x}, \tilde{\mathrm{z}}|\Psi)|\mathrm{x}, \Psi^{(t)}\right) \tag{13}$$

$$\Psi = \arg\max Q\left(\Psi, \Psi^{(t)}\right) \tag{14}$$

This is the reason why the algorithm is named after EM. And equation (14) is actually the backbone of the entire algorithm. The explicit algorithm can be described as the following steps:

1) Initialize $\Psi^{(0)}$ empirically. The initialization of $\pi_{z_i}^{(0)}$ and $\Theta_{z_i}^{(0)}$ are briefly described in Section 4.1.4.3.
2) Iterates the Expectation (E) and Maximization (M) steps until the parameters converge.

   i. E-step:

   Calculate the posterior distribution $p_{i,k}^{(t)} = p\left(\tilde{z}_i = k|x_i, \Psi^{(t)}\right)$, $k = \{0, 1\}$ according to Bayes' rule:

$$\begin{aligned} p_{i,k}^{(t)} &= \frac{p\left(x_i, \tilde{z}_i = k|\Psi^{(t)}\right)}{p(x_i|\Psi^{(t)})} \\ &= \frac{p\left(x_i|\tilde{z}_i = k, \Theta_k^{(t)}\right)\pi_k^{(t)}}{\sum_{j=0}^{1} p\left(x_i|\tilde{z}_i = j, \Theta_j^{(t)}\right)\pi_j^{(t)}} \end{aligned} \tag{15}$$

   ii. M-step:

   Update the parameters in sequence by

$$\begin{cases} \pi_k^{(t+1)} = \frac{1}{n}\sum_{i=1}^{n} p_{i,k}^{(t)} \\ \mu_k^{(t+1)} = \sum_{i=1}^{n}\left(\frac{p_{i,k}^{(t)}}{\sum_{i=1}^{n} p_{i,k}^{(t)}} x_i\right) \\ \sigma_k^{2\,(t+1)} = \sum_{i=1}^{n}\left(\frac{p_{i,k}^{(t)}}{\sum_{i=1}^{n} p_{i,k}^{(t)}}\left(x_i - \mu_k^{(t+1)}\right)^2\right) \end{cases} \tag{16}$$

Equation (16) is derived from (14). The relation between (16) and (14) is explicitly described in the following section.

4.1.4.2 A detailed derivation of the equations in EM algorithm

Bilmes (1998) elaborates clearly on how to formulate (14). Taking the function $h(\theta, x)$ for example, where θ a constant and x is a random variable with the distribution $f(x)$, its expectation can be formulated by

$$E\big(h(\theta,x)\big) = \int h(\theta,x) f(x) dx \tag{17}$$

in the continuous form and

$$E\big(h(\theta,x)\big) = \sum h(\theta,x_i) p(x_i) \tag{18}$$

in the discrete form, where $p(x_i)$ is the *probability mass function* with regard to each pixel value. By analogy, (14) includes the arguments of x, $\tilde{z}$, Ψ and $\Psi^{(t)}$, where x and $\Psi^{(t)}$ are always constant; Ψ is adjusted in M-step but fixed in E-step. This makes $\tilde{z}$ the only random variable in the function, which is governed by the distribution (15). Thus, (14) can be formulated by

$$Q\big(\Psi,\Psi^{(t)}\big) = \sum_{i=1}^{n}\sum_{k=0}^{1} \ln p(x_i,\tilde{z}_i = k|\Psi) p_{i,k}^{(t)} \tag{19}$$

Since

$$\begin{aligned} \ln p(x_i,\tilde{z}_i = k|\Psi) &= \ln(p(x_i|\tilde{z}_i = k,\Theta_k)\pi_k) \\ &= \ln p(x_i|\tilde{z}_i = k,\Theta_k) + \ln(\pi_k) \end{aligned} \tag{20}$$

(19) can be rewritten as

$$\begin{aligned} Q\big(\Psi,\Psi^{(t)}\big) &= \sum_{i=1}^{n}\sum_{k=0}^{1} \big(\ln p(x_i|\tilde{z}_i = k,\Theta_k) + \ln(\pi_k)\big) p_{i,k}^{(t)} \\ &= \sum_{i=1}^{n}\sum_{k=0}^{1} p_{i,k}^{(t)} \ln p(x_i|\tilde{z}_i = k,\Theta_k) + \sum_{k=0}^{1}\left(\ln(\pi_k)\sum_{i=1}^{n} p_{i,k}^{(t)}\right) \end{aligned} \tag{21}$$

The new parameters Ψ maximizing (21) are found by the non-linear optimization technology:

Estimate $\boldsymbol{\pi}$:

Introduce the Lagrange multiplier λ and the linear constraints $\sum_{k=0}^{1}\pi_k = 1$ and $\pi_k \geq 0$ to the partial differential function

$$\frac{\partial}{\partial \pi_k}\left(Q\big(\Psi,\Psi^{(t)}\big) + \lambda\left(\sum_{k=0}^{1}\pi_k - 1\right)\right) = 0$$

$$\Rightarrow \frac{\partial}{\partial \pi_k}\left(\sum_{k=0}^{1}\left(\ln(\pi_k)\sum_{i=1}^{n} p_{i,k}^{(t)}\right)+\lambda\left(\sum_{k=0}^{1}\pi_k-1\right)\right)=0$$

$$\Rightarrow \frac{1}{\pi_k}\sum_{i=1}^{n} p_{i,k}^{(t)}+\lambda=0$$

$$\Rightarrow \pi_k^{(t+1)}=\frac{1}{-\lambda}\sum_{i=1}^{n} p_{i,k}^{(t)} \quad \textbf{(22)}$$

Substituting (22) into $\sum_{k=0}^{1}\pi_k=1$, we obtain $\lambda=-n$. Thus, the prior distribution of class *k* for the new iteration is

$$\pi_k^{(t+1)}=\frac{1}{n}\sum_{i=1}^{n} p_{i,k}^{(t)} \quad \textbf{(23)}$$

Estimate Θ:

The parameters from different models are independently estimated. For each model, all the parameters of other models are fixed. Thus, maximizing *Q* with regard to the model *k* is equivalent to maximizing

$$Q_k=\sum_{i=1}^{n}\ln p(x_i|\tilde{z}_i=k,\Theta_k)p_{i,k}^{(t)}$$

$$=\sum_{i=1}^{n}\left(-\frac{1}{2}\ln(2\pi)-\frac{1}{2}\ln\sigma_k^2-\frac{(x_i-\mu_k)^2}{2\sigma_k^2}\right)p_{i,k}^{(t)} \quad \textbf{(24)}$$

Solving $\partial Q_k/\partial\mu_k=0$ and $\partial Q_k/\partial\sigma_k^2=0$ separately, we will obtain the same equations in (16).

4.1.4.3 The method to initialize the EM algorithm

In this section, we describe the method to initialize the EM algorithm. As is mentioned in the previous sections, the relative heights of the data are used in the data term. For each subtile, the relative height $H_{relative} = H_{absolute} - \min(H_{absolute})$. Thus, the range of $H_{relative}$ is always above 0. We first assume that all the points with $H_{relative}$ below a threshold of 2m should belong to ground. Then the means, variances and the percentage of each class derived from this classification are used as the initial parameters in EM algorithm. For simplicity, this threshold is named *initial height* in the following paragraphs.

4.2 Step 2: Extracting roof points

This step aims at extracting roof points from the non-ground points that are extracted from the previous step (Figure 25). Principal component analysis (PCA) is a com-

monly used technique in mining the key indices for detecting roof points, which has been applied in Verma *et al.* (2006), Matei *et al.* (2008), Carlberg *et al.* (2009) and Lafarge and Mallet (2011). In this section, the method of extraction of roof points using PCA is explicitly described. Especially, the principle of PCA is introduced in Section 4.2.2. The key indices for extracting roof points are introduced in Section 4.2.3. Section 4.2.4 elaborates on the different classifiers and choice of parameters.

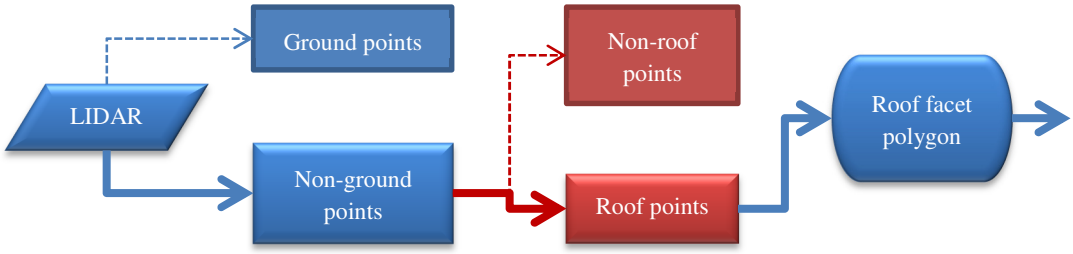


Figure 25. The red blocks and arrows denote the current stage in the pipeline.

4.2.1 Reflectance

It is plausible that vegetation has a significantly lower reflectance than those concrete materials do, such as road or construction, thus we can extract points of vegetation based on the differences in reflectance. Thus, the reliability of reflectance will be tested first. But it is not be used in this study since the reliability is unknown.

4.2.2 Principal Component Analysis (PCA)

Principal component analysis (Pearson, 1901) is a mathematical method that converts a set of observations denoted by correlated or uncorrelated variables into a set of values of uncorrelated variables, named *principal components*. In this study, the variables of the data refer to the 3D coordinate of the points. The procedure of PCA can be briefly described as follows: 1) Let X be an n by 3 matrix consisting of row vectors representing 3D coordinates of the points and $\overline{\mathrm{X}}$ the mean coordinate, i.e., the centroid of the points. A 3 by 3 covariance matrix Σ is approximated by

$$\Sigma = \frac{1}{n}\left(\mathrm{X} - \overline{\mathrm{X}}\right)^{\mathrm{T}}\left(\mathrm{X} - \overline{\mathrm{X}}\right) \tag{25}$$

which can be either positive-definite or positive-semidefinite. 2) Decompose Σ by

$$\Sigma = P\Lambda P^{\mathrm{T}} \tag{26}$$

where Λ denotes a 3 by 3 diagonal matrix containing the 3 eigenvalues of Σ; and each column in P is an eigenvector corresponding to the eigenvalue of the same column in Λ. The 3 eigenvectors are pairwise orthogonal. The eigenvalues and eigenvectors together describe the spatial pattern of the points. In theory, if the points are exactly distributed on a hyperplane of $\mathbb{R}^2$, Σ will have rank 2, i.e. the smallest eigenvalue is

zero. In practice, due to the existence of noise in LIDAR data, Σ has normally the full rank. Nonetheless, the smallest eigenvalue should be significantly smaller than the other ones. The corresponding eigenvector is the normal vector of the fitting plane of the points. Figure 26 demonstrates how PCA is associated with the points, where (V_1, V_2, V_3) is the eigenvectors. Since those eigenvectors are orthogonal, they can also serve as a new coordinate system for the points as needed. The coordinates under the new coordinate system are computed by

$$\tilde{X} = XP \tag{27}$$

If we calculate the variances of the new coordinates for each dimension in the rotated coordinate system, we can find that they are actually identical to the aforementioned eigenvalues. This reveals the physical meaning of the eigenvalues derived from a covariance matrix (Figure 27).

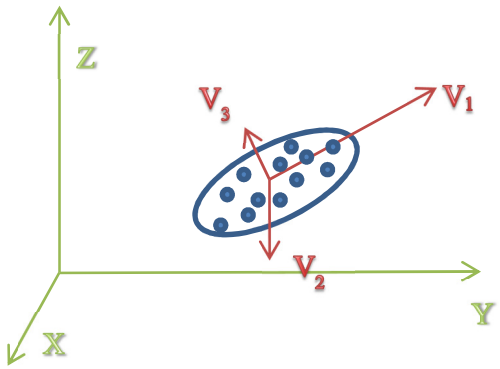


Figure 26. The transformation of basis. v1, v2 and v3 are the three eigenvectors indicating the directions along which the data is distributed.

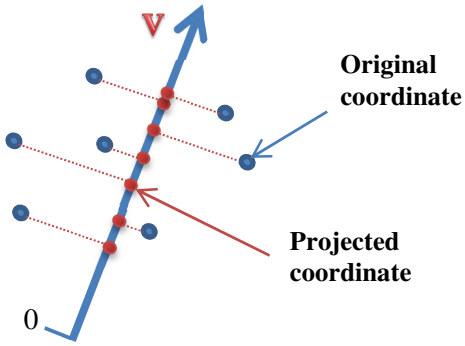


Figure 27. Projected coordinates of the points to an eigenvector. The variance of the coordinates is equivalent to the eigenvalue corresponding to this eigenvector derived by PCA from the covariance matrix.

4.2.3 Mining of indices for roof points recognition

In this section, some key indices for the recognition of roof points are introduced. For each point p_i, PCA is first implemented on all the neighboring points with the Euclidean distances smaller than a radius, by which we seek to obtain the three eigenvalues $\lambda_{i1}<\lambda_{i2}<\lambda_{i3}$ and the normalized eigenvector $\mathbf{n} = (a_i, b_i, c_i)$ corresponding to the smallest eigenvalue λ_{i1}. The neighboring system and the kd-tree structure used in the points processing have been introduced in Section 4.1.1.5.

Based on these eigenvalues and the eigenvector, some commonly used key indices for roof point recognition can be mined for distinguishing the roof points and other points from different respects, which are *planarity index*, *linearity index*, *vertical index* and *density index*.

1) **Planarity index:**

   As is introduced in the previous section, if a set of points are distributed like a plane, for each point with its neighbors processed by PCA, the eigenvalues satisfy $\lambda_1 << \lambda_3$. Thus, the planarity index can be defined by the ratio of λ_1 / λ_3, a smaller value of which suggests that the point is more likely to belong to a roof.

2) **Linearity index:**

   In the set of points satisfying $\lambda_1 << \lambda_3$, there exist some collinear points, which means the points are distributed as a line. Those points are also undesirable and therefore need to be removed. For each of those points, their eigenvalues satisfy $\lambda_2 << \lambda_3$. Thus, the linearity index can be defined by the ratio of λ_2 / λ_3, a larger value of which suggests that the point should be retained.

3) **Verticality index**

   The points of vertical walls need to be removed as well because they are normally irregularly and sparsely distributed due to the mechanism of airborne laser scanning. The existence of those points may potentially affect the robustness for the following process (Verma *et al*. 2006, Matei *et al.* 2008, Carlberg *et al*. 2009 and Lafarge and Mallet 2011). The verticality index is simply defined by $|c|$, i.e., the absolute value of the third dimension of the normalized eigenvector

4) **Density index:**

   Considering that the spatial distribution of roof points normally keeps a certain high level of density, while some non-roof points, especially the remaining noises after the exclusion of ground points, are very sparse and isolated, we can use an index relating to points density to distinguish them. Since the neighboring radius is constant for all the points, the density index is simply defined by the number of neighbors within the radius for each point. The fewer neighbors a point has, the more likely that this point should be discarded. Furthermore, since the input of PCA requires at least 3 points, this index is of importance in preventing the program from crashing.

4.2.4 Using different classifiers based on the indices

Based on the above indices, we can use some classifiers to recognize the roof points. A simple linear threshold classifier is first used.

1) The sparse and isolated points can be removed by the criterion that the number of neighbors smaller than a threshold d, which is assigned with $15R$. The R denotes the neighboring radius of the points. The value of 15 is chosen based

on a statistics of the minimum number of neighbors of the points on a manually selected roof.

2) Most of the tree points and other non-planar clutters can be filtered off by $\lambda_1/\lambda_3 < r_1$. Collinear points can be filtered off by the constraint $\lambda_2/\lambda_3 < r_2$. In addition, points of vertical walls are filtered off by $|c_i| > r_3$. These three thresholds can be manually tuned based on the histograms of their distributions and the results of the extractions.

Meanwhile, the graph cuts based classifier combined with EM algorithm is experimented on the planarity index. Unlike elevations, the distribution of λ_1/λ_3 for roofs should be relatively consistent and independent of locations. Thus, it is possible to apply the EM algorithm in the entire data to estimate the global parameters of the distribution of λ_1/λ_3. This classifier serves as a comparison to the simple linear threshold classifier.

4.3 Step 3: A RANSAC based method for plane fitting

The plane model for each roof facet can be computed from the roof points extracted in the previous step (Figure 28). The buildings can be first separated by applying *Connected Component Labeling* on all the points within the test region, assuming the distance between every two buildings is at least v meters (Verma 2006). Ideally, one should rasterize the roof points into an image with the resolution of 3m and implements the connected component labeling based on the pixels. Due to the time limit, this step is omitted in this study and only briefly introduced in Appendix C. The test region is the roof points of Lund cathedral extracted by the linear threshold classifier.

In this study, a RANSAC based technique is independently developed and implemented for automatically computing plane model and extracting points of each roof facet, which can be visualized as 3D polygons. In section 4.3.1, the ordinary RANSAC algorithm for single plane fitting is introduced. Sections 4.3.2 give the detailed description for multi-plane detection. Section 4.3.3 described the method of determining the parameters. And Section 4.3.4 describes the method for evaluating the robustness of the algorithm.

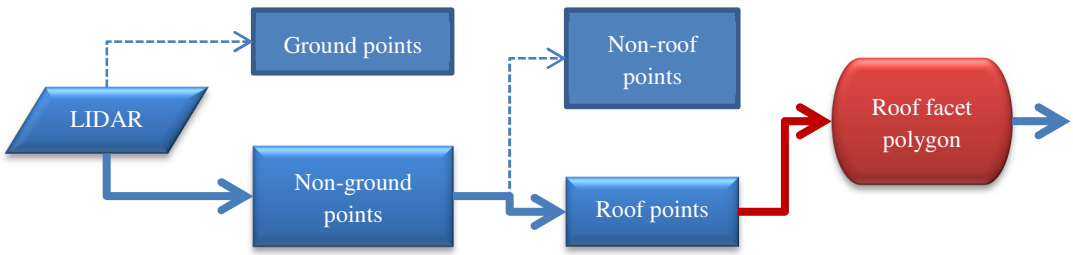


Figure 28. The red block and arrow denote the current stage in the pipeline.

4.3.1 Ordinary RANSAC

Model fitting refers to estimating the most ideal parameters of a model so that the model gives the greatest extent of coincidence with the data. The most widely used method for fitting data with Gaussian noise is least squares. However, using least squares directly will treat each datum equally, even including outliers. When outliers exist in a data set, a more robust method should be used. (Szeliski 2010).

Rousseeuw (1984) proposed the *least median of squares* (LMS). The key change of LMS from ordinary least square method is that the objective function to be minimized is the median of the squared residuals instead of the sum.

Another robust approach for model fitting is Random Sample Consensus (RANSAC), proposed by Fischler and Bolles (1981). The fundamental idea of this approach is each time selecting a certain amount of samples randomly from the data to generate candidate models in a certain way, and applying some scoring system to determine the best model when all the iterations finish.

4.3.1.1 Algorithm outline

Given a data set P, the algorithm iterates for k times. For each iteration:

1) Randomly select n samples from P to compute a hypothetic model M, where n is the minimum required number of data for determining a model, i.e., $n = 2$ for a line and $n = 3$ for a plane.
2) For each $p_i \in P$, calculate its deviation from M. The data with the deviation smaller than an error tolerance t are added in the *Consensus Set S* of the current iteration.
3) According to the assumption that the outliers are supposed to be much fewer than inliers, the S with the number of data smaller than a threshold d will be discarded. This threshold eliminates those obviously unqualified consensus sets in advance of the following model estimations, so that the unnecessary computations are avoided.
4) The model of the current iteration is estimated by the data in S in a least squares sense.
5) The estimated model is compared with the saved best model of previous iterations according to some scoring system measuring how well the model fits the data, which is introduced in section 4.3.1.3. If the new model is better, it will be saved as the best model by far. Otherwise, the model will be discarded.
6) The algorithm goes to 1) for the next iteration.

In this study, the error to be minimized is the sum of squared Euclidean distances from the points to the model (Figure 29), i.e., in the *Total Least Squares* sense.

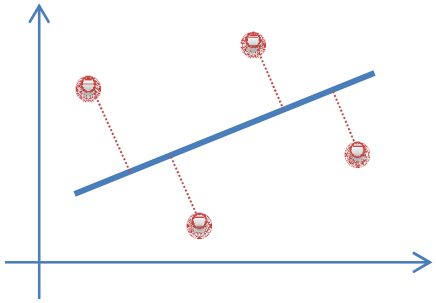


Figure 29. The total least squares used in line fitting for 2D points

4.3.1.2 Properties of the thresholds

The error tolerance t and the minimum required number of iterations k are the two critical parameters in RANSAC. In theory, t should depend on the distribution of the noises in the data, e.g., one or two standard deviations plus the average errors to the hypothetic model (Fischler *et al.*, 1984). However, it is not practical to evaluate the effect of noise analytically from the n random samples, as a result, t is normally determined empirically (Vosselman *et al.*, 2010).

The minimal number of iterations depends on the *a priori* probability of sampling outliers in data, which can be simply seen as the proportion of outliers in the data. Let w be the approximated proportion of outliers, and the model is determined by n independently selected samples. For each iteration, the probability that all the n samples are inliers is $(1\text{-}w)^n$. Thus, $1\text{-}(1\text{-}w)^n$ means the probability that at least one of the n samples is an outlier, in which case the estimated model is definitely unqualified. After k iterations, the algorithm might end up with having never selected such n samples that all of them are inliers. The probability of this case is $(1\text{-}(1\text{-}w)^n)^k$. A sufficiently large k can limit this probability down to 5%:

$$(1-(1-\mathrm{w})^n)^k \leq 5\%$$
$$\Rightarrow k \geq \frac{\log(1-95\%)}{\log(1-(1-w)^n)} \quad \textbf{(28)}$$

where k is the minimum required number of iterations. Since RANSAC is based on random samplings over a data set, it is not a deterministic method. Figure 30 demonstrates the relationship between the percentage of outliers and the runtime efficiency. The number grows exponentially as the outliers increase in a linear ratio. A data set with about 20% outliers will need only 5 iterations to get a qualified result, while the data with 80% outliers will need up to approximately 600 iterations.

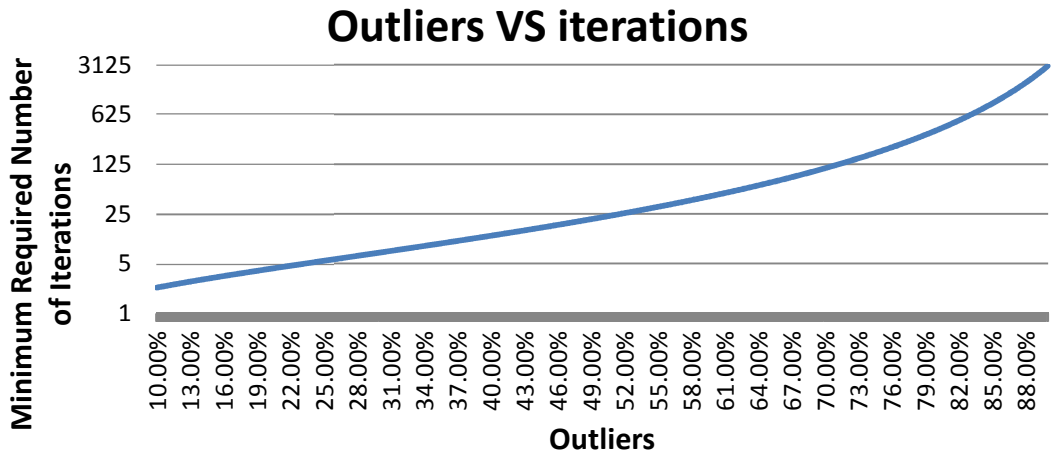


Figure 30. The relation between the percentage of outliers and minimum required number of iterations.

4.3.1.3 The scoring system

As to the scoring system voting for the best model, there are two different methods that can be applied. Formally, the scoring system can be viewed as finding the model that minimizes the energy function:

$$\widetilde{M} = \operatorname{argmin} \sum_{p \in P} s(M, p) \tag{29}$$

where $s(M, p)$ denotes the specific scoring method with regard to model M and point p. The simplest method is to choose the model with the largest consensus set. This corresponds to

$$s(M, p) = \begin{cases} 0, & distance(M, p) < t \\ 1, & distance(M, p) \geq t \end{cases} \tag{30}$$

This method fits for the case when t is properly set. When t has to be large for some reason, the resulting consensus set may therefore include many outliers and then affect the correctness of the model. To solve the potential problem caused by a large t, Torr and Zisserman (1998) introduced the MSAC (M-estimator Sample Consensus) algorithm, which modifies the $s(M, p)$ in RANSAC as

$$\tilde{s}(M, p) = \begin{cases} distance^2(M, p), & distance(M, p) < t \\ t^2, & distance(M, p) \geq t \end{cases} \tag{31}$$

The robustness is still guaranteed in that: 1) since t^2, which is the penalty given to outliers, is always larger than $distance^2(M, p)$, this scoring system will also prefer to choose the model with a larger consensus set; 2) When the consensus set is large enough but includes many outliers, The $\sum_{p \in P} distance^2(M, p)$ will be large and therefore the bad models are distinguishable (Vosselman *et al.* 2010).

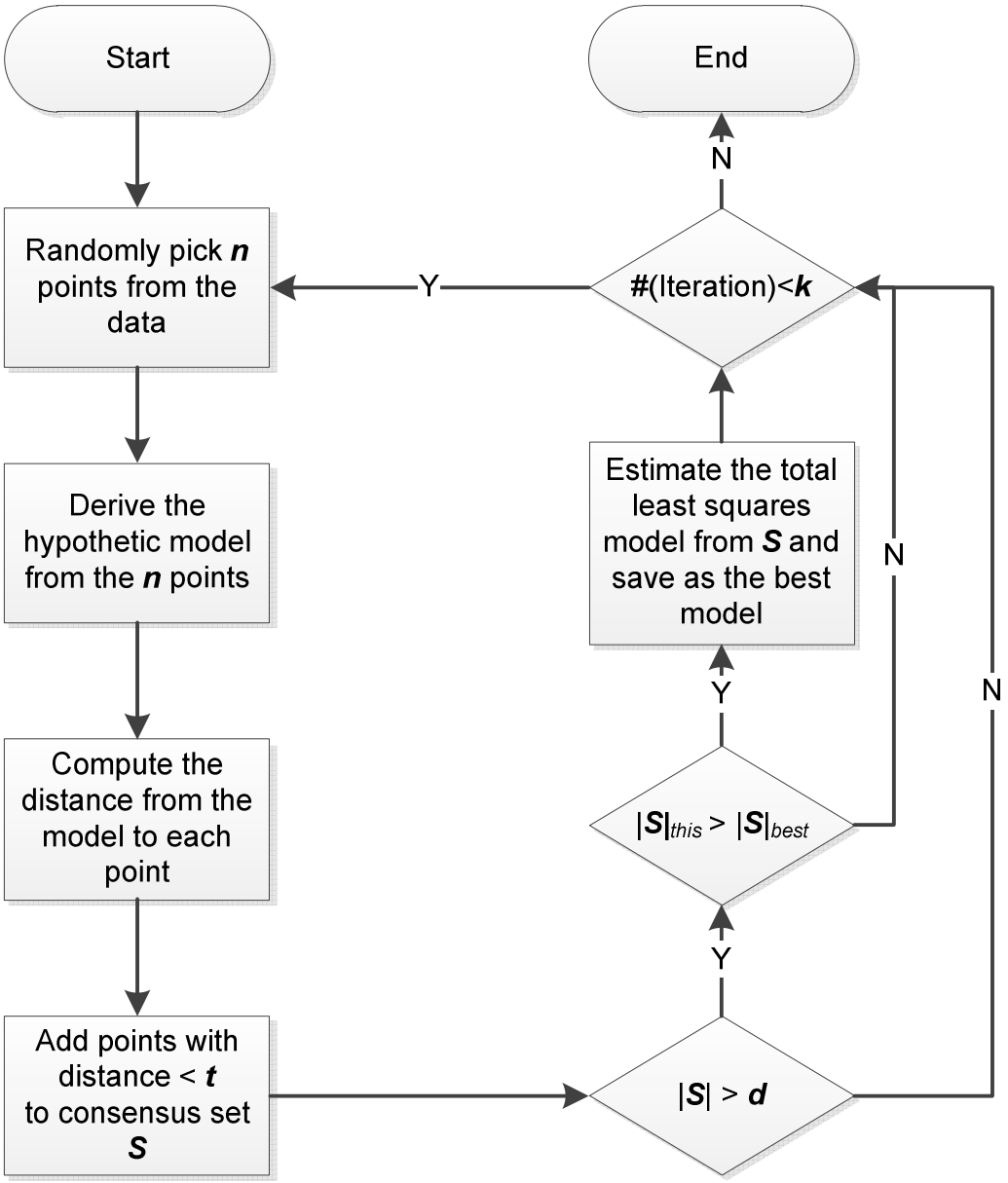


Figure 31. The flow chart of the RANSAC paradigm for single model fitting, where |*S*| denotes the number of elements in set *S*.

4.3.2 Multi-plane fitting

The original RANSAC is normally not applicable for the data potentially containing several models, because for each model, all the points belonging to a different model are outliers. According to Figure 30, this will require thousands of iterations. In this section, a modified RANSAC is designed for solving the problem. Section 4.3.2.1 and 4.3.2.2 introduce the necessary supplements to the ordinary RANSAC. Section 4.3.2.3 describes the modified RASNAC algorithm.

4.3.2.1 Sampling from neighboring points

It is observable that if the samplings are forced within a small range, the *n* selected points will be more likely from the same roof facet. This idea is predicated on the as-

sumption that all the roof facets are not curved, so that sampling points from any local part of a facet is equivalent.

Based on this idea, the n samples are selected from the *neighboring points* of a randomly selected point p_i. Let A stand for the event of successfully finding a model and B for the event that the neighboring points of p_i are from the same roof facet. According to Bayes' theorem, the probability of the occurrence of A is formulated by

$$P(A) = \frac{P(B)P(A|B)}{P(B|A)} \tag{32}$$

As a common sense, A is a sufficient condition to the occurrence of B. Thus, $P(B|A) = 1$ and

$$P(A) = P(B)P(A|B) \tag{33}$$

The event A|B, i.e., finding the model given that all the samples are selected on the same roof facet, is equivalent to the single model fitting problem using ordinary RANSAC. Its number of iterations, denoted by $k(A|B)$, obeys the curve demonstrated in Figure 30. By analogy, $k(B)$ is determined by $P(B)$, which can also be seen as the probability of finding a qualified point p_i. The total number of iterations is the product of the two terms:

$$k(A) = k(B) \times k(A|B) \tag{34}$$

And $k(B)$ is the only term to reduce.

4.3.2.2 Considering local normal as a criterion

In addition, a predicable drawback of ordinary RANSAC based methods is that it cannot separate points from two facets that share the same plane. If some of the points of a roof facet are coplanar with another facet, those points are subject to misclassification. To solve this problem, the local normal vector of each point, which has been obtained from the previous step, is introduced as an auxiliary criterion. Figure 32 demonstrates that some of the points in question will become separable.

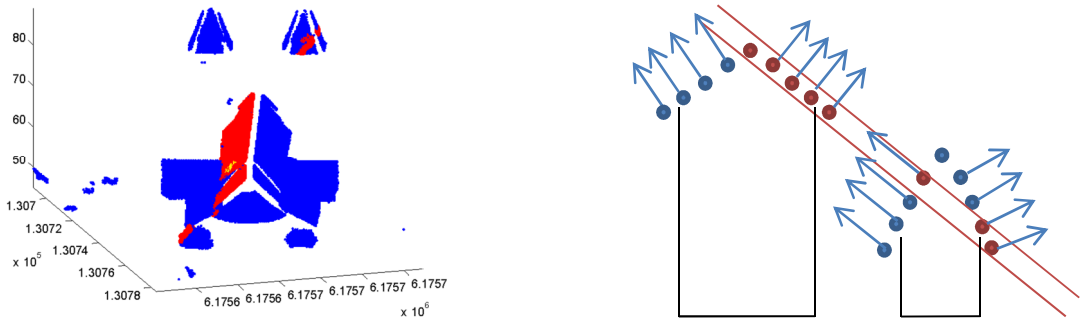


Figure 32. The red points belong to different facets but are coplanar, so they are subject to be misclassified. However, they can be separable by introducing the local normal for each point that is derived from the previous step.

4.3.2.3 The modified RANSAC algorithm

The modified algorithm comprises two layers of iterations. The inner layer is similar to the ordinary RANSAC, but is supplemented with the local normal. The outer level is the random selection of a qualified point p_i. Every time the algorithm finds the model for a roof facet, the points of this roof facet will be discarded from the data. Thus, an UNEXTRACTED list U is used for recording the indices of the remaining points. The algorithm is implemented as follows:

1) Randomly select a point p_i from the U. If U is empty, the algorithm terminates. Otherwise, if all its neighbors N are within the same roof facet, and the number of neighbors is larger than a threshold d_1, the algorithm continues to 2); otherwise, the point should be resampled. If the number of neighbors is smaller than d_1, remove p_i and their neighbors from U.

2) Run the inner iteration for k times:

   i. Randomly select 3 points from the neighbors and estimate a hypothetic model.

   ii. Calculate the deviations from the model to each point in U. Save all the points with the error smaller than an error tolerance t_1 to a set S_1.

   iii. Calculate the mean normal of those neighboring points. And calculate the angles between the mean normal to each local normal of the points in S_1.

$$\alpha_i = \arccos\left(\frac{\mathbf{n}_{mean}^T \mathbf{n}_i}{|\mathbf{n}_{mean}||\mathbf{n}_i|}\right) \tag{35}$$

   where $\mathbf{n}$ stands for a normal vector in the form of $(a, b, c)^T$. And save all the points in S_1 with the angle smaller than a threshold α to the Consensus Set S_2.

   iv. If the number of points in S_2 is smaller than threshold d_2, the algorithm returns to i. Otherwise, estimate the total least squares model from S_2.

   v. (30) is applied as the scoring system for voting for the best model, i.e., if the number of points in S_2 is larger than that from the best current model, save this model as the best model. Otherwise, return to i. for the next inner iteration.

3) Calculate the deviations from the best model to each point in S_2 of the model, and add the points in U with the error smaller than another error tolerance t_2 into a set S_3, from which the convex hull of the roof facet is extracted. Finally, remove S_3 from U. So far the model of a roof facet and its convex hull has been successfully estimated. The algorithm returns to 1) for the next outer iteration.

Figure 32 demonstrates the entire process of the modified RANSAC. The setting of the thresholds is specified in Section 4.3.3.

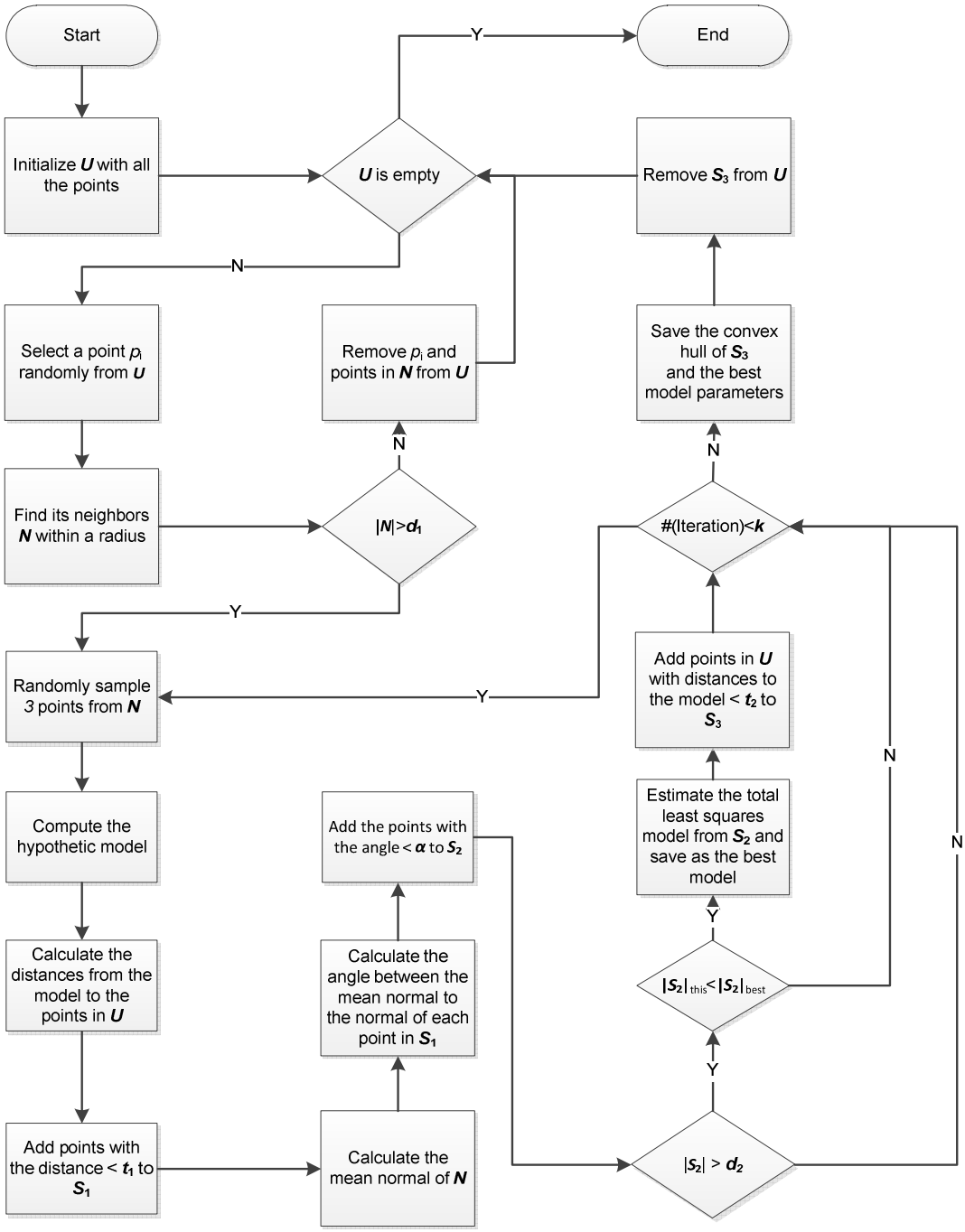


Figure 33. The flow chart of the modified RANSAC for multi-model fitting

4.3.3 Setting of thresholds

In this section, the selection of parameters is described in detail. First of all, t_1 is the threshold for determining the inliers of the points to a model. Assuming all the roof points are distributed with the similar thickness, the error tolerance t_1 is empirically assigned with the thickness of a manually selected roof facet. To visualize the

cross-section of the roof facet, PCA is applied on those points for deriving the eigenvectors to which the coordinate system is rotated by applying (27). Figure 34 displays the cross-section of the roof facet. In this coordinate system, t_1 is assigned with the variance of the coordinates that are projected on the vertical axis.

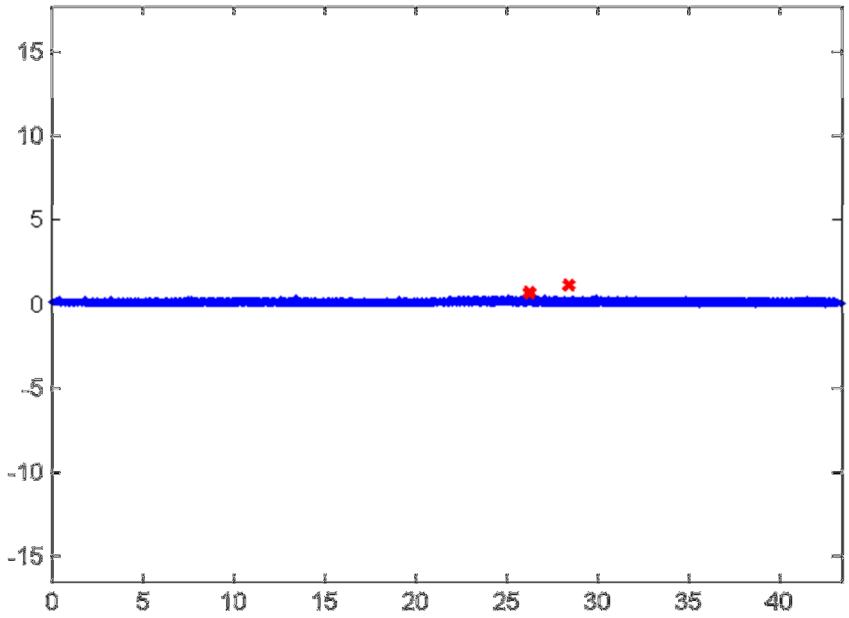


Figure 34. A cross section of a roof facet displayed in the new coordinate system, where the outliers are highlighted.

The threshold t_2 is used for extracting the points of an entire roof facet, from which the convex hull is derived. Thus, t_2 should be set larger than t_1. In this study, the t_2 is adjusted according to the experiments.

The number of iteration k is determined by equation (28). Assuming in every roof facet 40% of the points are outliers, the value of k is therefore 13 with the success rate of 95%.

There are two purposes for using threshold d_1. First, the number of neighbors should be sufficiently large (at least larger than 3), so that they can generate enough hypothetic models. Second, it can filter off sparse points, which are normally some undesirable isolated points remaining from the last step. d_1 is empirically chosen as $10R^2$, where R is the neighboring radius. The d_2 is also empirically set to 50.

The angle tolerance α is set as the mean of the angles between the mean normal to each local normal within the neighbors plus m times standard deviations of those angles. The value of m is also tuned by experiments.

4.3.4 Evaluating the Robustness of RANSAC on Single Model Fitting

If the algorithm is applied to all the points for hundreds of times, because the order of estimating the specific roof facet is totally random, it is difficult to find a way to sort them in a uniform order, which will be problematic for collecting statistical results on each model. Consequently, in the first test, the algorithm is simply applied 200 times

on only one of the roof facets for evaluating the accuracy of the algorithm, which is measured by the variance of the deviation angles (35) between each of the 200 results and their mean.

In the second test, the algorithm is tested on the cathedral points for 10 times (Appendix F). The quality of the roof facet plane models is evaluated from visual inspection.

5. RESULTS

The pipeline is extensively tested on the LIDAR data. In this section, the experimental results relating to the three steps, which are classification of ground and non-ground points, extraction of roof points and generation of roof facet polygons are presented respectively in Section 5.1, 5.2, and 5.3.

5.1 Classification of ground and non-ground areas using graph cuts

In this section, the results of the experiments in classification of ground and non-ground points are presented. The EM estimates of the Gaussian mixture model are demonstrated in Section 5.1.1. In Section 5.1.2, the classification on the lower-right 36 subtiles using graph cuts are presented with regard to different spatial regularizations. Section 5.1.3 demonstrates a comparison between the classifications with and without splitting the data into subtiles.

5.1.1 Parameter estimation using EM algorithm

The first experiment is the estimation of Gaussian mixture model using EM algorithm based on elevation data. The initial height for ground heights is set 2m. Figure 35 demonstrates a typical Gaussian mixture model automatically estimated by EM algorithm in one of the subtiles. The EM estimations of 36 lower right subtiles of the area are presented in Appendix D.

From those plots, we can see that in most of the subtiles, the estimated model parameters fit the data well in general. This should give credit to the fact that in most of the subtiles the distributions of ground and non-ground points are sufficiently different so that it is easy to find the correct convergence to separate them. Especially they will always converge to the same values no matter if the initial height is set 2m or 3m. However, the estimation is problematic if there are several dominant peaks in the subtile, such as the subtiles demonstrated in Figure 36 and Figure 37. The convergence in those subtiles is sensitive to the choice of the initial height. In this sense, 3m is a more ideal value for the initial height. In addition, Figure 38 illustrates a distribution of the number of iterations when using EM algorithm on the 36 subtiles, where we can see that most of the subtiles can converge in approximately 10 iterations.

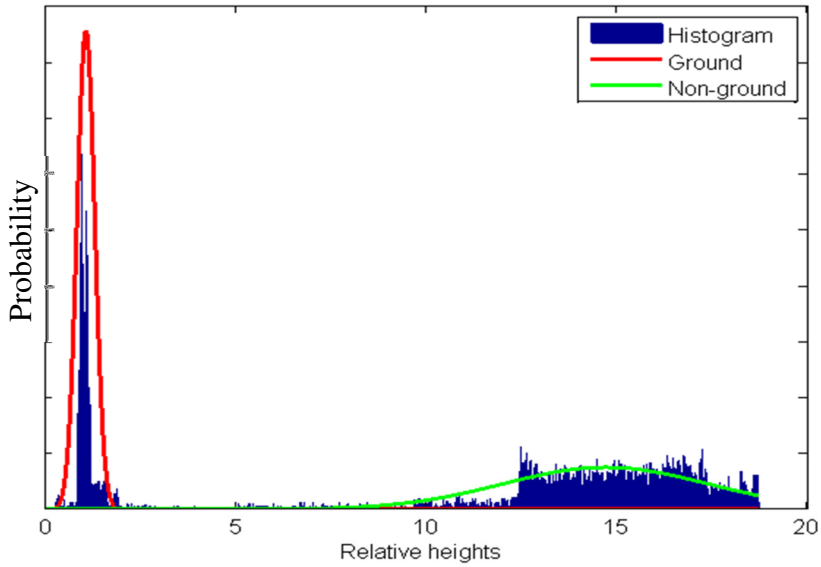


Figure 35. The two Gaussian models estimated from the data using EM algorithm

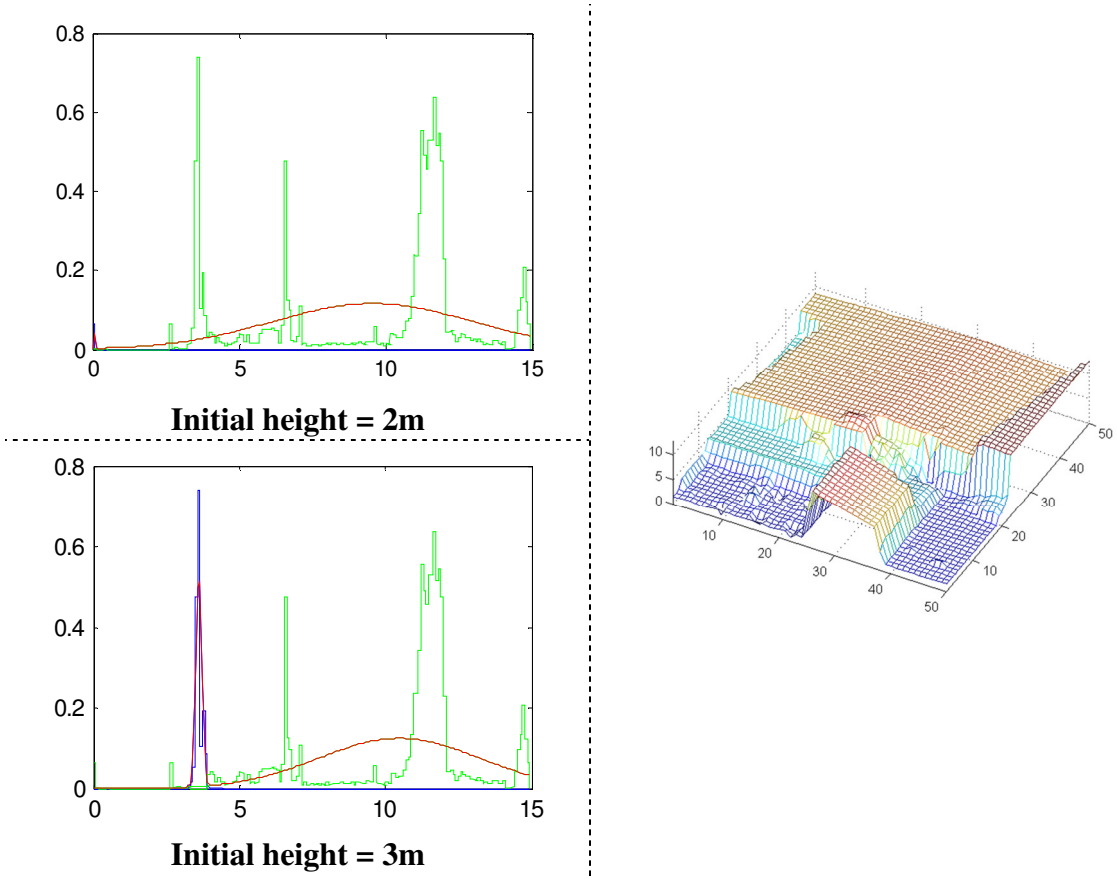


Figure 36. The EM estimation on the subtile (10, 9), where different initial heights are used and compared. When using the 2m initial height, the parameters converge to a bad result

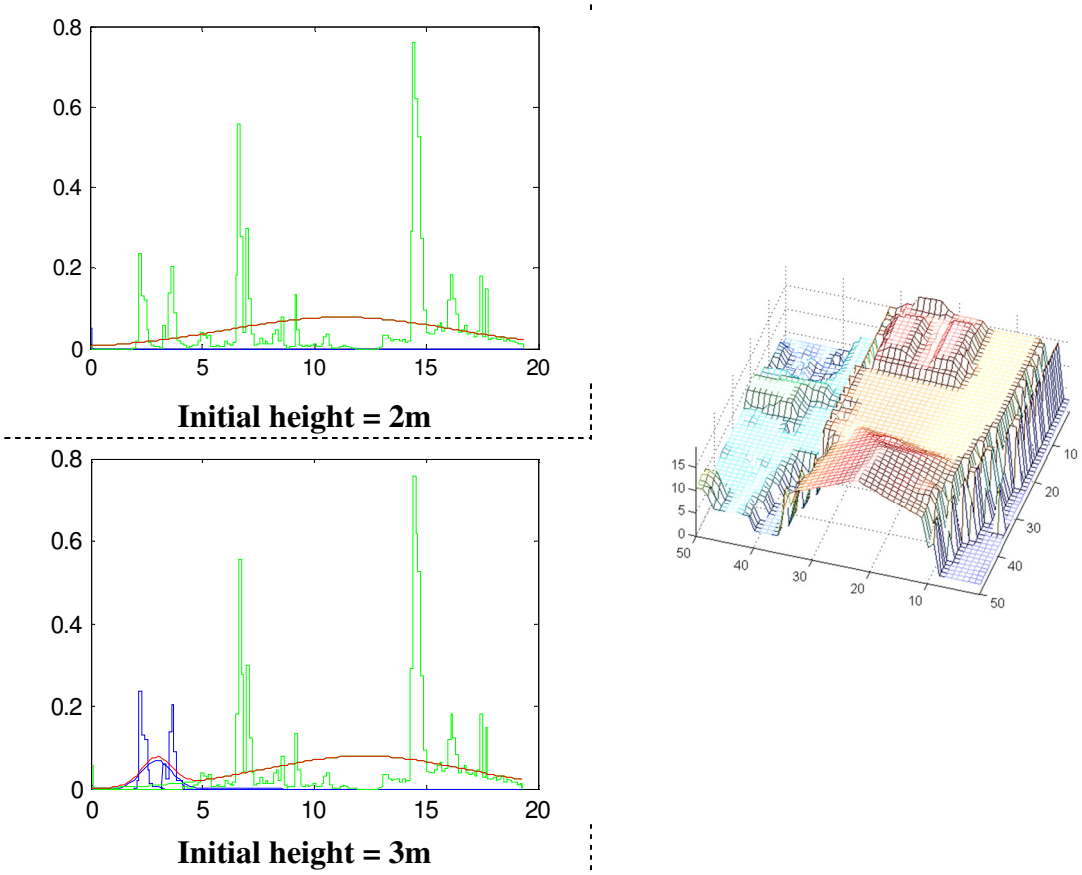


Figure 37. The EM estimation on the subtile (10, 8), where different initial heights are used and compared. When using the 2m initial height, the algorithm converges to a seemingly bad result.

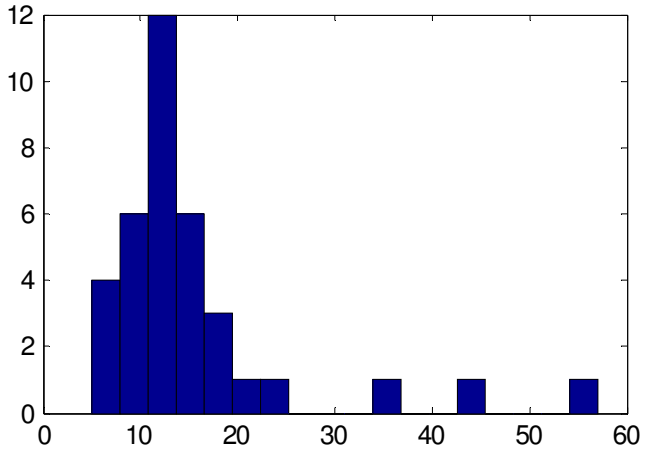


Figure 38. The distribution of the number of iterations in the EM algorithm applied on the 36 subtiles.

5.1.2 Graph cuts with different spatial regularization

The effects of spatial regularization are studied in this section. Figure 39 demonstrates the non-ground points extracted using graph cuts method. For the first figure, the spatial regularization factor is set to 0. In contrast, the second figure is the result when a

proper spatial regularization factor is applied, where we can see that some cars and low shrubs are classified as ground. For the third figure, the spatial regularization is set very large, consequently some roofs are disappeared. This suggests that the spatial regularization should be carefully tuned.

Spatial regularization = 0 **Spatial regularization = 2** **Spatial regularization = 10**

Figure 39. The classification of ground and non-ground areas with respect to different factor values of spatial regularizations. The differences are presented on those red circles.

5.1.3 The effect of splitting the data into subtiles

Figure 40 shows the comparison of the classifications with and without splitting the data into subtiles. In this test, the data is rasterized into a height map because the required RAM for processing the entire LIDAR data is beyond the ability of the computer used in this study. We can find that some of the building areas in the first image are not present in the second image, while in the second image, the northeast part is misclassified as building. The third image represents the elevation trend for the ground of the region, where the northeast part is higher than its southwest counterpart.

Classify with splitting **Classify without splitting** **Elevation Trend**

Figure 40. The comparison of classifications of ground and non-ground areas with and without being split beforehand, and the estimated elevation trend.

5.2 Extracting roof points

5.2.1 The reliability of provided reflectance

First, we studied whether the provided reflectance can be seen as an index for extracting roof points. Figure 41 demonstrates the test comparing the points with high reflectance (>200) and those with low reflectance (<20). A remarkable phenomenon can be seen in **a)** that some of the roofs have the similarly low reflectance to that of vegetation; for some of the gabled roofs, the two opposite roof facets have even sharply different reflectances. This suggests the reflectance is not a reliable index for extracting roof points.

a) Intensity < 20 **b) Intensity > 200**

Figure 41. The points with different thresholds in intensity.

Some of the roof points have distinct reflectances. This could be ascribed to the following reasons: 1) some materials on the roofs absorb more signals than others, or 2) some signals were reflected in a large angle of incidence. In fact, for the data obtained from traditional passive remote sensing, a robust classification method based on reflectance usually employs the reflectances of multi-spectrum. For instance, the Normalized Difference Vegetation Index (NDVI), which is the ratio between the sum and difference of reflectance in near infrared and red, has been proved to be a reliable index for detecting vegetation. A single spectrum would normally not be competent alone. Nevertheless, there could exist some way of combining the reflectance into the framework as a supplemental feature, which could be a topic in the future.

5.2.2 Applying the linear threshold classifier

The quality of a classification relies greatly on the choice of parameters in the classifier. In this section, different thresholds are tested for the proposed linear threshold classifier and the graph cuts based classifier. First, the distributions of the classification indices are studied because they can always provide intuitive information about the suitable value ranges. From Figure 42, we can find that in general, the distribu-

tions under the 2m neighboring radius look more concentrated, which implies the roof points are easier to extract with 2m radius. The transparent pink regions demonstrate the reasonable value ranges for the thresholds.

Figure 43 shows the extraction results for the two representative subtiles, in which buildings and trees take up the entire area respectively. This test aims at investigating the extraction quality under the 0.5m neighboring radius, with respect to different thresholds on the planarity index. From the figure we can see that if the threshold is set as 0.08, roof facets can be well extracted. Meanwhile, since 0.08 is relatively "loose", many tree points cannot be thoroughly excluded. As the threshold goes down, the number of undesirable tree points decreases. But the roof areas shrink at the same time due to the increasingly stricter criterion.

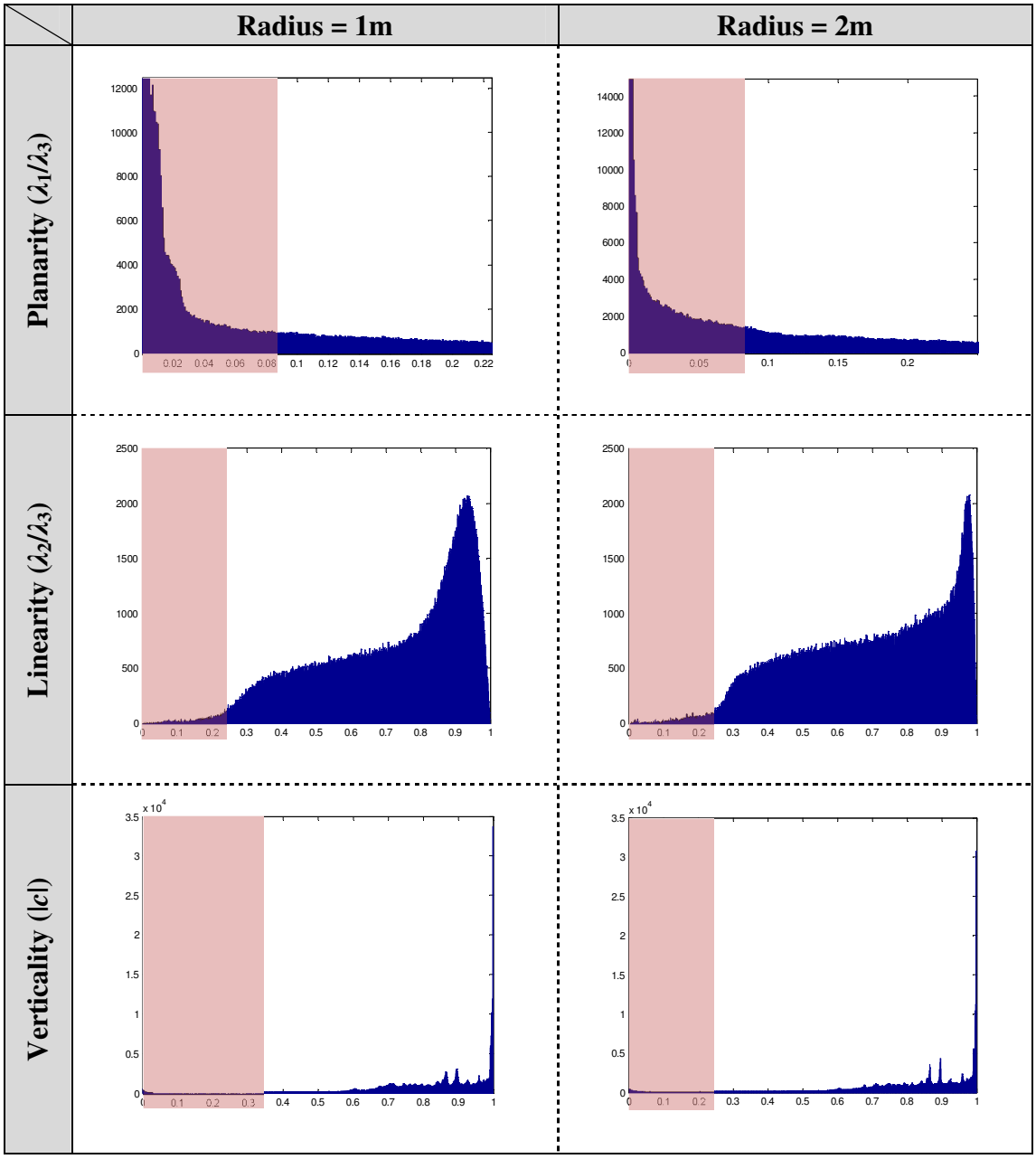
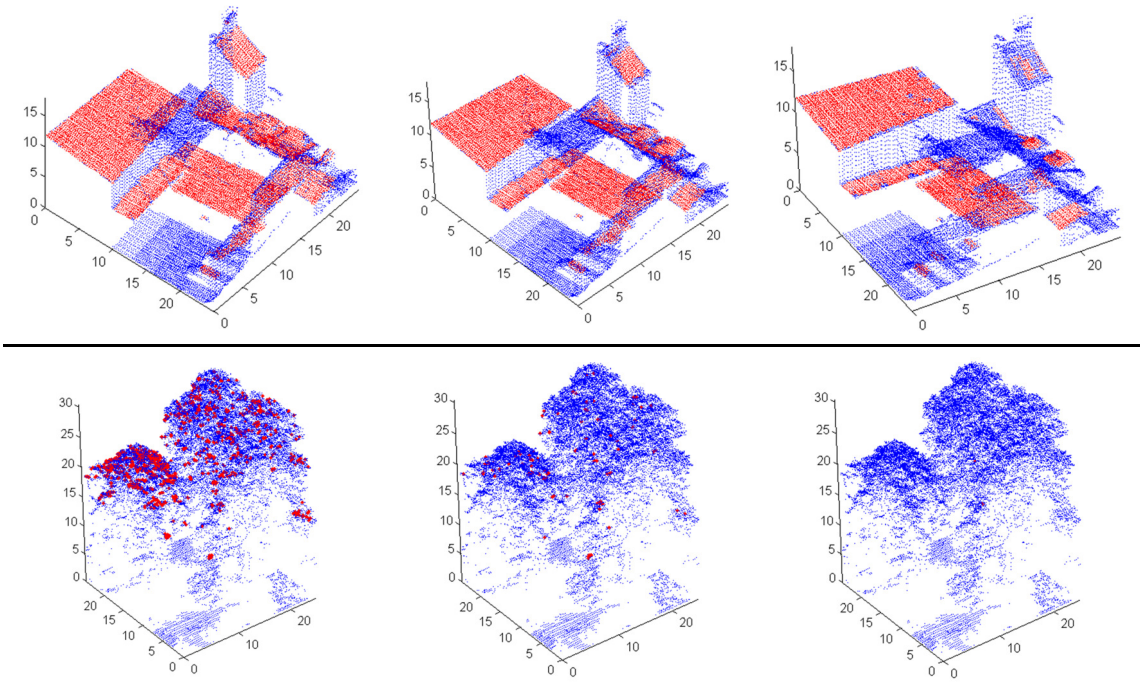


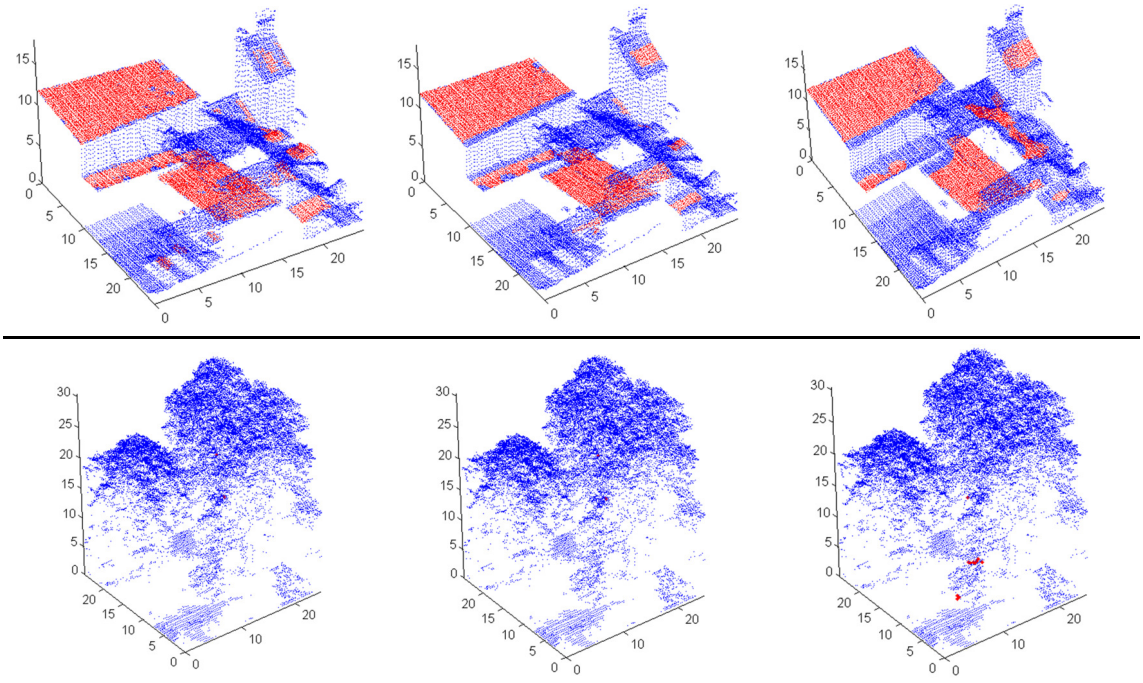
Figure 42. The distribution of the three indices for extracting roof points with respect to two different sizes of neighboring radiuses: 1m and 2m. The red transparent regions denote the range of the appropriate thresholds.

In comparison, Figure 44 illustrates the extraction results with respect to different radius sizes, which are 0.5m, 1m and 2m. For each size, the thresholds on the planarity index are properly set so that the undesirable tree points can be almost cleared. Observe the buildings points and we can find that as the radius size increases, more points on the joint part between adjacent roof facets are missing.



$r_1 = 0.08$; $r_2 = 0.2$; $r_3 = 0.3$	$r_1 = 0.03$; $r_2 = 0.2$; $r_3 = 0.3$	$r_1 = 0.01$; $r_2 = 0.2$; $r_3 = 0.3$

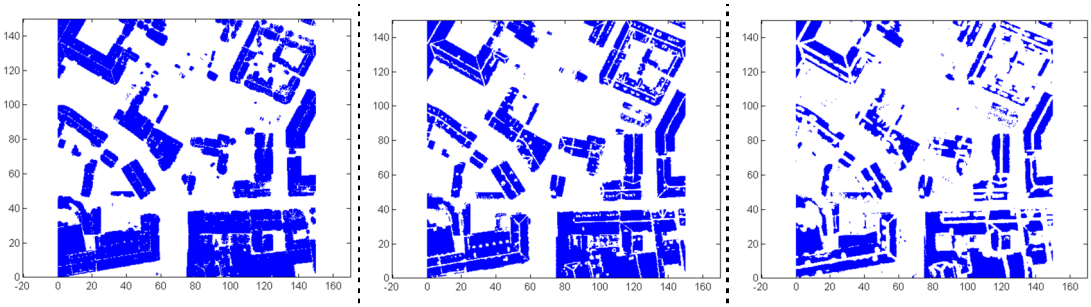
Figure 43. A comparison on the extraction of buildings under different criteria of λ_1/λ_3 when the radius of the kernel is 0.5m. Red and blue points denote roof points and other undesired points respectively.



Radius = 0.5m	Radius = 1m	Radius = 2m

Figure 44. A comparison on the extraction of buildings by different radius sizes, under the condition that the tree points are minimally misclassified by assigning the appropriate values for r1. Red and blue points denote roof points and other undesired points respectively.

Figure 45 demonstrates the extraction of a larger area including different buildings when applying different radius sizes and a less-strict threshold on r_1. This means some noises are allowed to exist so that the roof points can be well retained. Afterwards, the noises are erased by a threshold on each point with less than 10 neighbors. By comparison, we can find that applying the 2m radius will cause massive disappearance of roof points. When radius is 1m, only the roof facets of LOD 3 (Figure 8) are missing. When radius is 0.5m, roof facets of LOD 3 can be retained, but in expense of remaining more undesirable noises.



Radius = 0.5m; r_1= 0.08; r_2= 0.2; r_3 = 0.3 | **Radius = 1m, r_1= 0.08; r_2 = 0.2; r_3 = 0.3** | **Radius = 2m, r_1= 0.04; r_2 = 0.2; r_3 = 0.3**

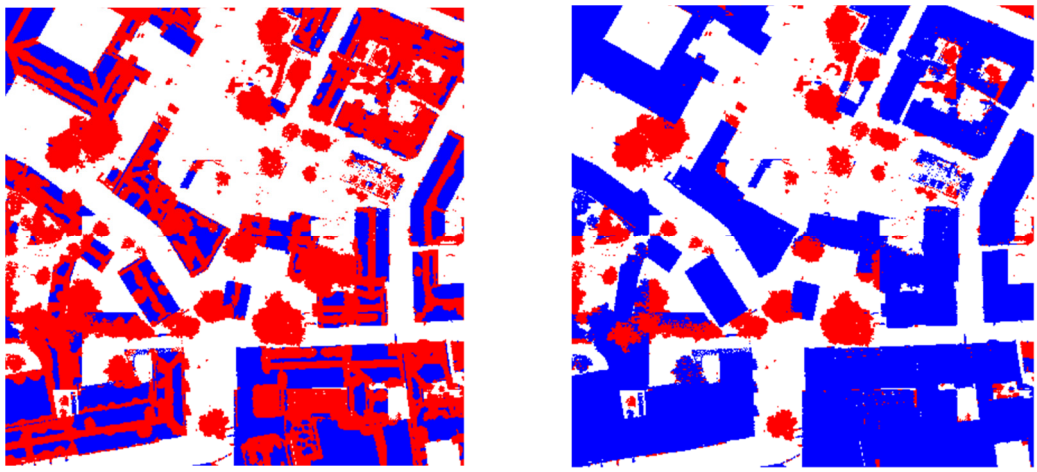
Figure 45. Extracting roof points in a larger area, where different radius are used. The threshold of r_1 is not very strict. The remaining noises are removed by a filter of the density index.

It seems that when using the simple linear threshold classifier, the quality of extraction is sensitive to the size of neighboring radius and strictness of the thresholds. The distributions of the indexes will have smaller variances if the neighboring radius is larger, which improves the quality of the extraction. Meanwhile, more points on the joint part between adjacent roof facets are subject to be misclassified to non-plane points. On the other hand, a smaller neighboring radius can limits the points missing on the joint part between adjacent roof facets. But the consequent larger variances in the indices distribution require stricter criteria to exclude undesirable points, which will also result in points missing on the roof facets. Thus, a good choice for the thresholds in this simple classifier should be based on the balance between the influences from the size of radius and the strictness of the criteria. For the dense urban area where exist small roof facets, the simple thresholding classifier seems not a good strategy for plane detection.

5.2.3 Applying the graph cuts based classifier

Since the determination of the optimal parameters is very sensitive in the simple thresholding classifier for plane detection, applying the graph cut based classifier is motivated due to the assumption that the spatial regularization can help the correct classification of the points on the joint part of adjacent roof facets. Figure 46 demonstrates the comparison of detecting planes using the simple linear threshold classifier (Left) and the graph cuts based classifier (Right), conditional to the neighboring radius of 2m. From the left figure, we can see clearly that many points on joint part between the roof facets are misclassified. In the right figure, however, the graph cuts

classifier merges those points into their surrounding roofs. But this causes some tree points closed to buildings are misclassified. The EM estimate for the parameters of the distribution of the planarity index is demonstrated in Figure 47.



The simple linear threshold classifier; **The graph cuts based classifier;**

Figure 46. The comparison of plane detections with (right image) and without (left image) using graph cuts. The radius of neighboring structure is 2m for both cases. Blue and red denote the classified plane and non-plane respectively.

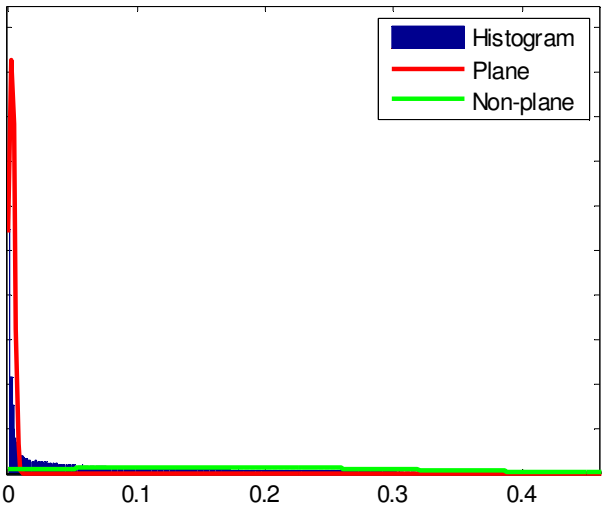


Figure 47. The Gaussian mixture model estimated from the planarity index in the entire 250m x 250m region using EM algorithm. The model parameters are used as arguments of Mumford-Shah functional in the graph cuts method.

5.3 The modified RANSAC for multi-plane fitting and visualization

In this section, the results for modified RANSAC are presented. The roof points are extracted by the simple linear threshold classifier, where there are gaps between coupled roof facets. For all the experiments, the value of t_2 is 1.5m; the value of m for the

angle threshold α is set as 5, i.e., 5 times standard deviations. The detailed explanation on the choice of those thresholds has been introduced in section 4.3.3.

5.3.1 Accuracy tests

The modified RANSAC algorithm is implemented 200 times on a large roof facet. The standard deviation of the angles from the mean normal of the 200 results to each of those normals is 0.1131°. The explicit results for 200 times are listed in Appendix E.

5.3.2 Visual inspection

Figure 48 shows the models estimated from the building points using the modified RANSAC, each of which is rendered as a 3D polygon bounded by the convex hull of the roof facet. We can find that some of actually separated roofs are mistakenly modeled as a whole 3D polygon (Red circle). The algorithm takes 10s on average in the processing of 73229 points. The explicit plots for the ten results can be found in Appendix F, eight of which are satisfactory. In each of the two results in question, there is only a small roof facet that is not properly modeled. They can however be manually modified afterwards in practice.

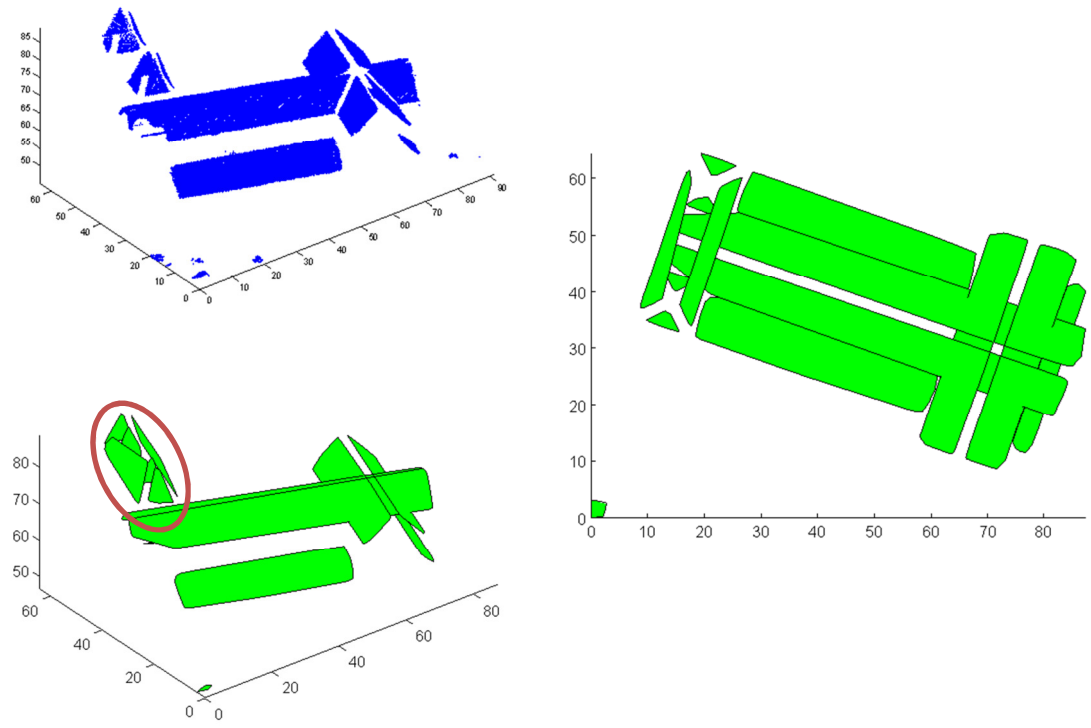


Figure 48. The visualization of roof facet polygons (Green) estimated from the roof points extracted without using graph cuts (Blue). Some of the facets are not separated (the red circle).

6. DISCUSSION

Although many experiments relating to points extraction and plane modeling have been conducted, due to the limitations on time, hardware and software, there are some respects in this study that can be further improved. For example:

1) The quality of the extraction has not been numerically evaluated because the correct referencing classification is not provided. Because of the data inconsistency mentioned in Section 3.3, the vector data of constructions is not suitable for evaluating the extraction.

2) The connected component labeling is not conducted. This step aims at labeling points with different building numbers, so that the points belonging to the same building are processed as a group.

3) The robustness of the kd-tree used in this study has not been investigated, e.g., when finding the neighbors of a point within a radius, whether really all the points within the radius are included.

4) The open source kd-tree package used in this study cannot handle the data of too large quantity. According to the experiments, the program crashes when building a kd-tree for millions of points. This is another reason for splitting the data into subtiles.

5) Due to the less effort put in developing memory-efficient codes, the memory often overflows when processing millions of points at the same time even though the data are split, which causes a great inconvenience in applying a method on all the subtiles. The workaround is applying the method on some of the subtiles. When the entire data has to be considered, the data are rasterized for saving memory. The neighboring structure in a LIDAR data is normally heavier than those in an image, which is the main reason for the program consuming so many memories and running time.

6) The most time-consuming part of the program is repeatedly constructing and destroying kd-tree in different steps. This is because the open source kd-tree package does not provide the function for removing points. One of the solutions could be designing a dynamically scalable kd-tree in such a way that the tree need only be built-up once and the points can be removed from the tree while keeping the tree balanced for the remaining points. Since the time limit, this task is left for the future.

7) The classification of LIDAR data comes down to the problem of finding discriminable indices and designing robust classifiers. The indices in this study are only the most commonly used ones. However, there could be many other better designed indices or combinations of indices that can potentially improve

the quality of classification. For example, Lafarge and Mallet (2011) uses a *Scatter* index that is defined as the *minimal principal curvature* mean of a considering point and its neighbors, in which a high value corresponds to trees and other undesirable urban components. Meanwhile, the rapid development of the LIDAR technology today may provide more and more useful physical features to improve classification accuracy.

8) The modified RANSAC is only tested on a building with large roof facets. When applying the method to the smaller roofs, some parameters may have to be changed. The roof points used in this test are extracted by the simple linear threshold classifier, where there are gaps between coupled roof facets. The algorithm can make use of the gaps to reduce the $k(B)$ in equation (34) by setting the searching radius smaller than the gap. Whereas, for the roof points extracted by the graph cuts based classifier, where there are no gaps between coupled roof facets, the algorithm will need an extra procedure to check whether the selected points are within the same roof facet.

9) This version of modified RANSAC is only limited to planar roof facets. However in reality, some planar roof facets have a certain degree of curvature on their tails. This is the main error source from which some undesired fitting polygons are generated.

7. CONCLUSIONS

In this study, a pipeline of detecting plane and creating polygons for roof facets has been devised and implemented based on some existing ideas and methods. Some technical details omitted by other literatures, such as choices of parameters, choices of energy functions have been explicitly described. The framework can be useful in processing LIDAR data limited in dense urban area. But the codes will need further optimizations if used in the real industry.

For separation of ground and non-ground areas, classifying separately on the smaller areas and merging them together afterwards is an effective way of reducing the influence of slightly varied topography. However, the splitting does not guarantee every subtile can achieve a good classification. Instead, it improves the classification in such a way that the bad classification is limited within the local subtile where the height distribution is too complex and therefore will not affect the classification of the neighboring subtiles. EM algorithm can effectively and efficiently find the most appropriate parameters automatically. When the spatial regularization is properly set, graph cuts method can make the classification smoother. Especially, some cars and low shrubs can be separated from buildings.

With regards to the quality of the extraction, the simple linear threshold classifier depends on both the strictness of the criteria and the radius size. When the radius is set small, some of the undesirable points cannot be well excluded. When the radius is set large, undesirable points will remains fewer, but there will be more points missing on the joint part between roof facets. This can results in some smaller roof facets can be totally missing. One of the solutions is set a less strict threshold and filtering off the noises afterwards. According to the test, the radius of 1m gives the generally best result. Another solution is using the graph cuts based classifier, which can guarantee no points disappearance on the joint part between roof facets even though the criteria on the indices are strict and the size of neighboring radius is of up to 2m.

Finally, the modified RANSAC gives a robust way of estimating multiple models. The variance of the estimates is sufficiently small, although it cannot generate a unique solution. A drawback of this RANSAC base method is that it cannot separate the points that are totally coplanar. The separation of such roof facets will perhaps need some other information, e.g., adjacency between points.

In addition, there are also a few smaller polygons generated from small undesirable points that fail to be filtered off. They can be removed afterwards.

Overall, this study contributes to clarifying some technical details in one of the possible methods for detecting and extracting roof facets in dense urban area.

8. REFERENCES

ASPRS Online. (2010). Common Lidar Data Exchange Format - .LAS Industry Initiative. Available from <http://www.asprs.org/a/society/committees/lidar/lidar_format.html>. [Retrieved 21 Nov 2011].

Bilmes, J. (1997). A Gentle Tutorial on the EM Algorithm and Its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. (Technical Report ICSI-TR-97-021). ICSI.

Boykov, Y., Veksler, O. and Zabih, R. (2001), Fast Approximate Energy Minimisation via Graph Cuts, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29, 1222–1239.

Boykov, Y. and Kolmogorov, V. (2004). An Experimental Comparison of Min-cut/Max-flow Algorithms for Energy Minimization in Vision. *IEEE Trans. On Pattern Analysis and Machine Intelligence (TPAMI)*, 29(9): 1124-1137.

Carlberg, M., Gao, P. Chen, G. and Zakhor, A. (2009). Classifying Urban Landscape in Aerial Lidar Using 3D Shape Analysis. *IEEE International Conference on Image Processing*.

Plan3D. (2011). *Roof Types*. Digital image. Available from: <http://www.plan3d.com/p3dhelp/Roofs.htm>. [Retrieved 21 Nov 2011].

Connected-Component Labeling. In *Wikipedia: The Free Encyclopedia*. Wikimedia Foundation Inc. Encyclopedia on-line. 18 May 2012. Available from: <http://en.wikipedia.org/wiki/Connected-component_labeling>. Internet. [Retrieved 21 Nov 2011].

Cormen, T. H., Leiserson, C. E., Rivest, R. L. and Stein C. (2001). *Introduction to Algorithms*, Second Edition. MIT Press and McGraw-Hill. ISBN 0-262-03293-7.

Fischler, M. A. and Bolles, R. C. (1981). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395.

Forsyth, D. A. and Ponce, J. (2003). *Computer Vision, A Modern Approach*. Prentice Hall. ISBN 0-13-085198-1.

Goldberg, A. V., Tarjan, R. E. (1986). A New Approach to the Maximum Flow Problem. *Annual ACM Symposium on Theory of Computing, Proceedings of the eighteenth annual ACM symposium on Theory of computing*, 136–146. ISBN 0-89791-193-8, 1986.

Greig, D. M., Porteous, B. T., and Seheult, A. H. (1989). Exact Maximum A Posteriori Estimation for Binary Images. *Journal of the Royal Statistical Society*: *Series B*, 51, 271–279.

Hartley, R. and Zisserman, A (2003). *Multiple View Geometry in computer vision*. Cambridge University Press. ISBN 0-521-54051-8

Horn, B. K. P (1986). *Robot Vision*. MIT Press. pp. 69–71. ISBN 0-262-08159-8.

Image Analysis, Laboratory session 3 (2010). Laboratory instruction. Course material in Image Analysis, Centre for Mathematical Sciences, Lund University, Sept. 2010. Available from :< http://www.maths.lth.se/media/FMA170/2011/lab3en.pdf >. [Retrieved 21 Nov 2011].

Isenburg. (2012). *License Agreement*. Available from <http://www.cs.unc.edu/~isenburg/-lastools/download/LICENSE.txt>. [Retrieved 21 Nov 2011].

Kada, M. and McKinley, L. (2009). 3D Building Reconstruction from LIDAR Based On a Cell Decomposition Approach. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXVIII, 3 / W4: pp. 47-52.

Kolbe, T.H., Gröger, G. and Plümer, L. (2005). CityGML – Interoperable Access to 3D City Models. *Proceedings of the 1st International Symposium on Geo-information for Disaster Management*. Springer Verlag, 883-889.

Kolmogorov, V. and Zabih, R. (2004). What Energy Functions Can Be Optimized via Graph Cuts. *IEEE TPAMI*, 26(2):147–159.

Lafarge, F. and Mallet, C. (2011). Building Large Urban Environments from Unstructured Point Data. *IEEE International Conference on Computer Vision*, 978-1-4577- 1102-2/11.

Lillesand, T.M., Kiefer, R.W. and Chipman J.W.(2004). *Remote Sensing and Image Interpretation*. 5th edn. New York : Wiley, cop. 2004.

Lindgren, F. (2005). *Image Modelling and Estimation, A Statistical Approach*. 3rd edn. Mathematical Statistics, Centre for Mathematical Sciences, Lund University.

Linder, W. (2006). *Digital Photogrammetry: A Practical Course*. 2nd edn. Springer. ISBN-10: 3540291520.

Mackinnon, E. *Three fundamental devices of LIDAR*. Digital image. Available from: <http://tmackinnon.com/lidar.php>. Internet. [Retrieved 21 Nov 2011].

Kim, A. (2011). 3D Mapping Company C3 Technologies Acquired by... Someone. *Macrumors*. Available from: <http://www.macrumors.com/2011/08/01/3d-mapping-company-c3-technologies-acquired-by-someone/>. Digital image. [Retrieved 21 Nov 2011].

Mass, H.G. and Vosselman, G. (1999). Two Algorithms for Extracting Building Models from Raw Laser Altimetry Data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54(2-3), 153-163.

Mumford, D. and Shah, J. (1989). Optimal Approximations by Piecewise Smooth Functions and Variational Problems. *Comm. Pure Appl. Math.*, XLII(5):577–685.

Pearson, K. (1901). On Lines and Planes of Closest Fit to Systems of Points in Space. *Philosophical Magazine* 2 (6): 559–572.

RANSAC. In *Wikipedia: The Free Encyclopedia*. Wikimedia Foundation Inc. Encyclopedia on-line. 22 April 2012. Available from: <http://en.wikipedia.org/wiki/RANSAC>. Internet. [Retrieved 21 Nov 2011].

Rousseeuw, P. J. (1984). Least median of squares regression. Journal of the American Statistical Association, 79:871–880.

Rutzinger, M. (2011). *Introduction to LIDAR*. Presentation notes in the guest lecture in dept. of Physical Geography and Ecosystem Science, Lund University, Sweden.

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Microsoft Research. ISBN-13: 978-1848829343.

Torr, P. and Zisserman, A. (1998). Robust Computation and Parametrization of Multiple view relations. *Proceedings of the Sixth International Conference of Computer Vision (ICCV' 98)*, 4-7 January 1998, Bombay, India, IEEE Computer Society, Washington, DC, USA, 727.

Verma, V., Kumar, R. and Hsu, S., (2006). 3D Building Detection and Modeling from Aerial Lidar Data. *Proceedings of the IEEE computer Society Conference on Computer Vision and Pattern Recognition* (CVPR'06), 2213-2220.

Vosselman, G., (1999). Building Reconstruction Using Planar Faces in Very High Density Height Data. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 32(Part 3/2W5), 87-92.

Vosselman, G. and Dijkman, S., (2001). 3D Building Model Reconstruction from Point Clouds and Ground Plans. *International Archives of Photogrammetry and Remote Sensing*, 35(Part 3/W4), 37-44.

Vosselman, G. and Mass, H-G. (2010). *Airborne and Terrestrial Laser Scanning*. Whittles Publishing. ISBN 978-1-904445-87-6.

Wehr, A. and Lohr, U. (1999). Airborne Laser Scanning - An Introduction and Overview. *ISPRS Journal of Photogrammetry & Remote Sensing*. 54 (1999). pp. 68-82.

Worboys, M and Duckham, M (2004). *GIS, A Computing Perspective*. CRC Press LLC. ISBN 0-415-28375-2.

Zisserman, A. (2004). *Epipolar Geometry*. Lecture notes in the course: Light and Geometry in Vision, Spring 2009. Available from <http://www.ics.uci.edu/~dramanan/teaching-/cs217_spring09/lec/stereo.pdf>. [Retrieved 21 Nov 2011].

APPENDIX A. Future Work

According to the method by Verma *et al*. (2006), the estimated roof facet model parameters will be refined by non-linear optimization with constraints. For a gabled roof composed by two planar facets, let (x_{ridge}, y_{ridge}, z_{ridge}) denote a point on the roof ridge approximated by the intersection line between the two facets. The coupled facets can be formulated with regard to the ridge point:

$$\Pi_k = \{x, y, z | a_k(x - x_{ridge}) + b_k(y - y_{ridge}) + c_k(z - z_{ridge}) = 0\} \quad \textbf{(36)}$$

where $k = \{1,2\}$ and $c_k > 0$. Let (x_{ik}, y_{ik}, z_{ik}) be the coordinate of ith point belonging to facet k. The estimated height from the plane with regard to the point p_{ik} is therefore:

$$\hat{z}_{ik} = -\frac{1}{a_k}(a_k x_{ik} + b_k y_{ik} - a_k x_{ridge} - b_k y_{ridge}) + z_{ridge} \quad \textbf{(37)}$$

Let $s_k = -\sqrt{a_k^2 + b_k^2}/c_k$ and $o_k = a_k/\sqrt{a_k^2 + b_k^2}$ denote the slope and 2D orientation of facet k (Figure 50), we can get

$$\begin{aligned} \hat{z}_{ik} &= -\frac{\sqrt{a_k^2 + b_k^2}}{c_k}\left(\frac{a_k x_k + b_k y_k - a_k x_{ridge} - b_k y_{ridge}}{\sqrt{a_k^2 + b_k^2}}\right) + z_{ridge} \\ &= s_k\left(o_k x_{ik} + \sqrt{1 - o_k^2}\, y_{ik} - o_k x_{ridge} - \sqrt{1 - o_k^2}\, y_{ridge}\right) + z_{ridge} \end{aligned} \quad \textbf{(38)}$$

which is demonstrated in Figure 49.

The two planar facets of a regular gabled roof are normally symmetric. Thus, the constraints of $s_1 = s_2$ and $o_1 = -o_2$ are set so that the two facets are forced to have the same slope and opposite orientations.

As a result, the refinement is actually minimizing the following energy function

$$E = \sum_{k=1,2}\sum_{i} \varepsilon_{ik}{}^2 = \sum_{k=1,2}\sum_{i} (\hat{z}_{ik} - z_{ik})^2 \quad \textbf{(39)}$$

subject to $s_1 = s_2$ and $o_1 = -o_2$.

This formulation implies that the optimization is in *ordinary* least squares sense, which means only the sum squares of z-coordinates are minimized. If we want to change the formulation into *total* least squares, the error function has to be replaced by the sum squares of point-plane distances. First the plane model Π_k can be changed to the expression with regard to s_k and o_k:

$$o_i(x - x_{ridge}) + \sqrt{1 - o_i^2}(y - y_{ridge}) + \frac{1}{s_i}(z - z_{ridge}) = 0$$

$$\Longrightarrow s_i o_i(x - x_{ridge}) + s_i\sqrt{1 - o_i^2}(y - y_{ridge}) + (z - z_{ridge}) = 0 \quad (40)$$

Then the error function becomes:

$$\acute{\varepsilon}_{ik} = \frac{\left\| s_k o_k(x_{ik} - x_{ridge}) + s_k\sqrt{1 - o_k^2}(y_{ik} - y_{ridge}) + (z_{ik} - z_{ridge}) \right\|}{\sqrt{{s_k}^2 + 1}}$$

Unfortunately, due to the time limit, the refinement has not been implemented by programming. In addition, although the roof polygons are created for some algorithms that automatically assemble the refined planar facets into complete roof structures, they can also facilitate manual digitization, since digitizing directly on LIDAR points might be boring and daunting for human beings. The proposed process of digitization will only need the following two steps: 1) select planar facets belonging to a complete roof manually, 2) convert them into a predefined roof structure, such as gabled or heaped roof. The pipeline of the remaining tasks is given in Figure 51.

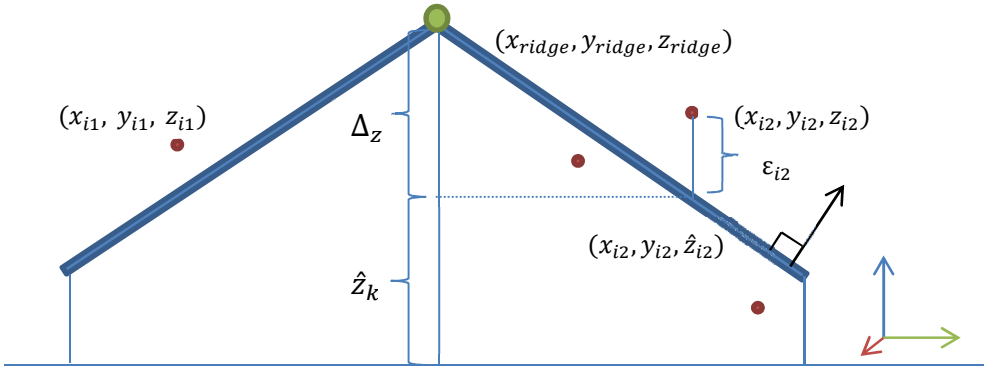


Figure 49. The refinement of roof model parameters.

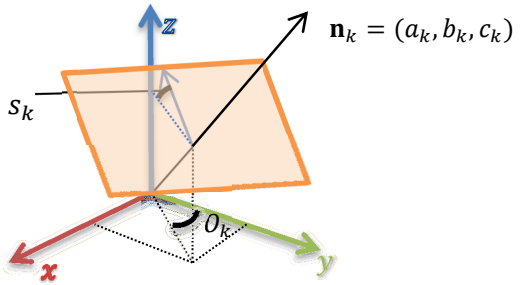


Figure 50. The slope and orientation of a planar roof facet.

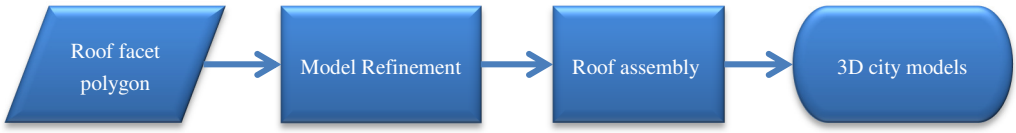


Figure 51. The rest of the procedures to create 3D city models.

APPENDIX B. kD tree

kD-tree is a binary tree structure for storing multi-dimensional data in such a way that the data can be more efficiently queried than using a naïve algorithm. Each datum is stored as a node in the tree. Each node has a left descendent and right descendent representing the two data which are respectively smaller and larger than the datum on the node in one of the dimensions. "kD-tree" is only a general name for cases of arbitrary dimensions. It can be also specified according to the dimension of the data, which means that for 2D data, it is called 2D-tree while 3D-tree for 3D data. Considering that the LiDAR data is distributed in 3D Euclidean space, we will use the name "3D-tree" for clarity in the following context.

A complete 3D-tree is built through inserting the points into an empty 3D-tree one after another. Each point is inserted by comparing its x, y or z coordinate periodically to the coordinate of its node at different depths of the tree. If the inserting point has a smaller x coordinate than its node, it will take up the left descendent of this node, *vice versa*. Since for a binary tree structure, the searching efficiency is directly determined by its depth, a balanced 3D-tree is preferred. Thus, the *median* of the remaining points is selected as the point to be inserted every time. The medians can be efficiently found using *Heapsort* algorithm with the $O(n\log n)$ time (Cormen *et al.* 2001). And considering that inserting a new datum into a balanced 3D tree takes $O(\log n)$ time, building up a 3D tree takes $O(n\log^2 n)$ time.

The algorithm of building up a 3D tree is given below. (Worboys *et al*. 2004)

Input: Point *p* and 3D-tree T

1: if T is an empty 3D-tree then
2: T←a new 3D-tree with root p and two null tree descendants
3: node *n*←the root of T
4: tree level *l*←0
5: repeat
6: if *l* mod 3 == 0 then
7: *a* ← *x*-coordinate of *p*
8: *b* ← x-coordinate of *n*
9: elseif *l* mod 3 == 1 then
10: *a* ← y-coordinate of *p*
11: *b* ← y-coordinate of *n*
12: else
13: *a* ←z-coordinate of *p*
14: *b* ←z-coordinate of *n*
15: if *a* < *b* then *n'* ← the LEFT node from *n*
16: else *n'* ← the RIGHT node from *n*
17: *n* ← *n'*
18: *l* ← *l* + 1
19:until *n* = null tree
20:insert new node *p* at position of *n* with two null tree descendant

This study applies an open source kD-tree package published by Tagliasacchi (2008).

APPENDIX C. Connected Component Labeling

The connected component labeling aims at separating non-adjacent pixels in an image into different classes. Figure 52 demonstrates that the two disjoint areas are labeled differently. The algorithm is given below.

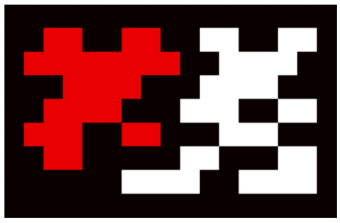


Figure 52. An example of connected component labeling applied on an image, where black denotes NaN.

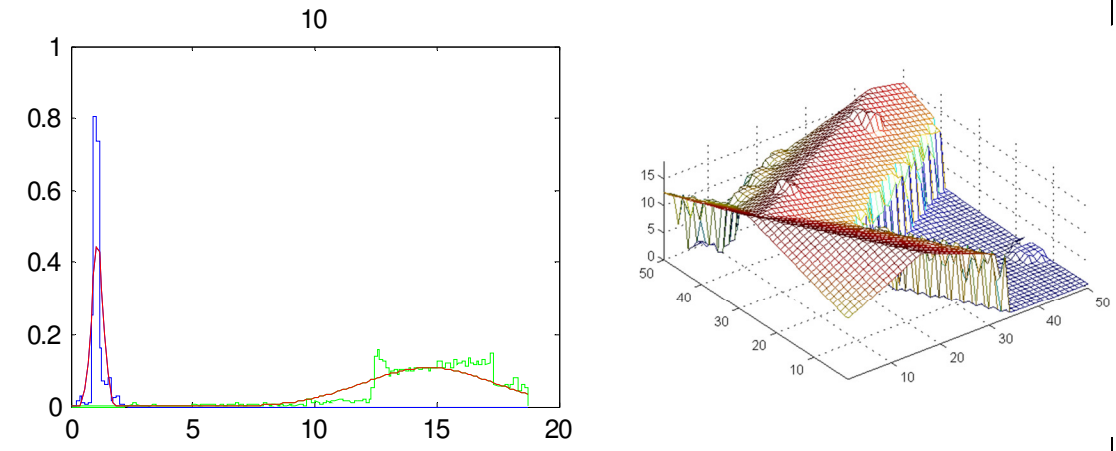
```
algorithm TwoPass (data)
Linked ← []
Labels ← structure with dimensions of data, initialize
First Pass
for i ← 1 to Num_of_Row
    for j ← 1 to Num_of_Column
        if data[i][j] is not Background
            neighbors ← connected elements (whose value = current element's value)
            if neighbors is empty
                linked[NextLabel] ← set containing NextLabel
                labels[i][j] ← NextLabel
                NextLabel ← NextLabel + 1
            else
                Find the smallest label
                L ← neighbors_labels
                labels[i][j] ← min(L)
                for label in L
                    linked[label] ← union(linked[label], L)
Second pass
for i ← 1 to Num_of_Row
    for j ← 1 to Num_of_Column
        if data[i][j] is not Background
            labels[i][j] ← find(labels[i][j])
return labels
```

(Horn, 1986)

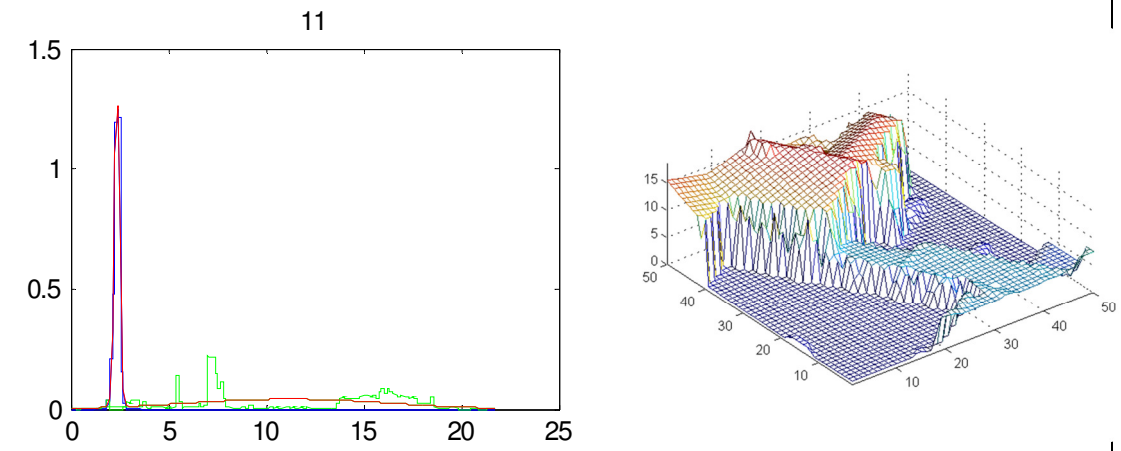
APPENDIX D. Gaussian Mixture Model of Elevations Estimated by EM Algorithm

The plots below demonstrate the Gaussian mixture models estimated by EM algorithm for each subtile. The black curve stands for the mixture model for all the heights. The blue and green curves denote the Gaussian models of ground and non-ground respectively. 36 of 100 subtiles are chosen. The (x,y) represents the location of a subtile in the entire data, which correspond to the grids in Figure 23. The numbers above the figures are the number of iterations before the parameters converge.

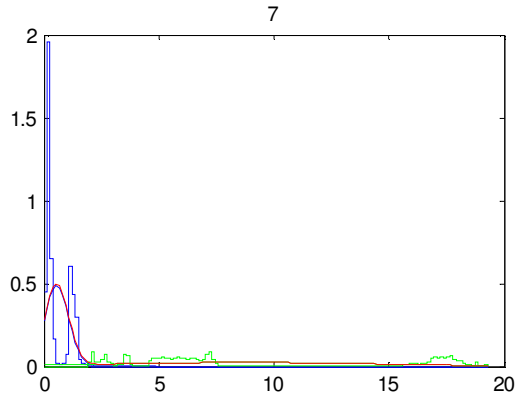
(5,5)

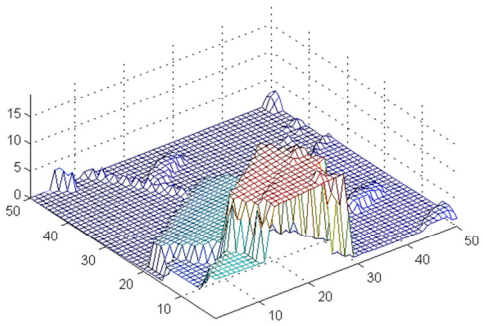


(5,6)

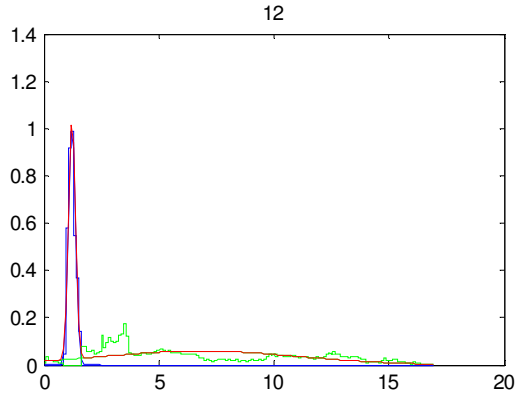


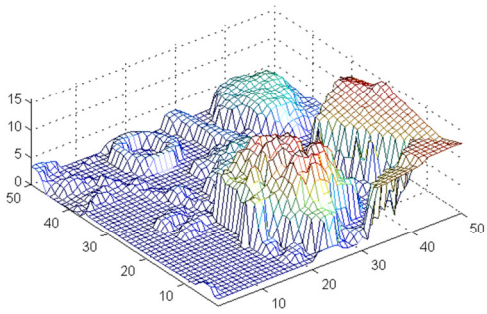
(5,7)


7
2
1.5
1
0.5
0
0
5
10
15
20

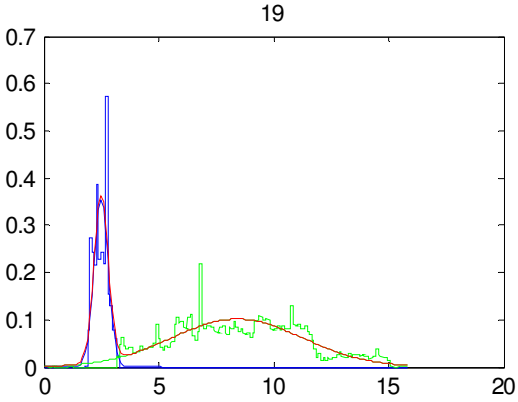

15
10
5
0
50
40
30
20
10
10
20
30
40
50

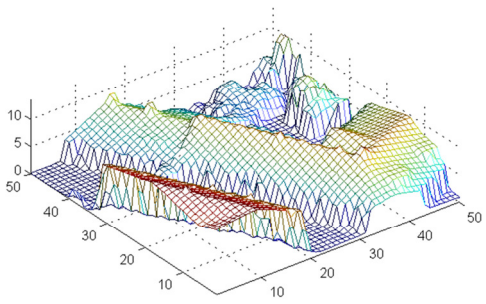
(5,8)


12
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20

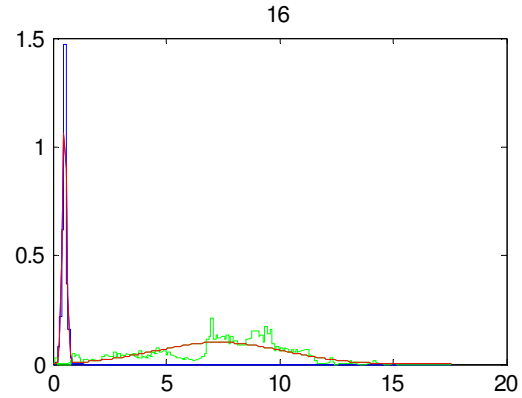

15
10
5
0
50
40
30
20
10
10
20
30
40
50

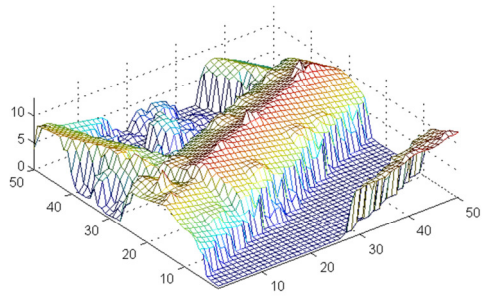
(5,9)


19
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5
10
15
20

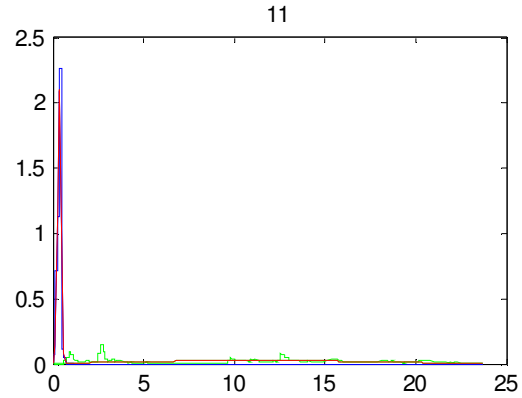

10
5
0
50
40
30
20
10
10
20
30
40
50

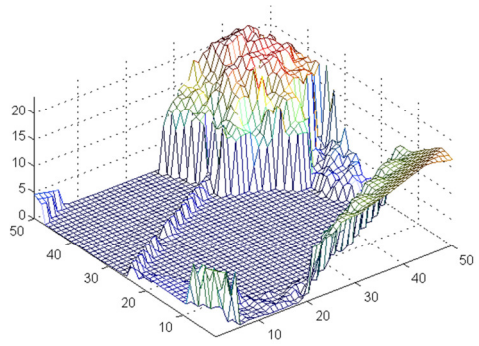
(5,10)


16
1.5
1
0.5
0
0
5
10
15
20

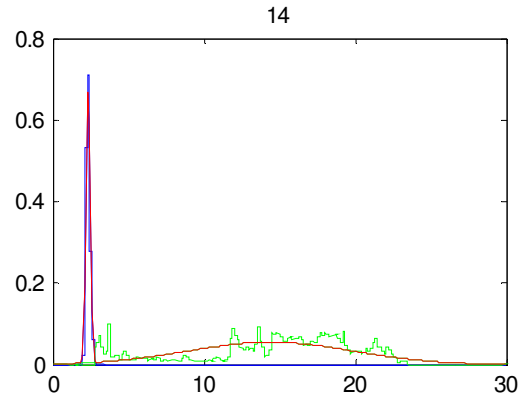

10
5
0
50
40
30
20
10
10
20
30
40
50

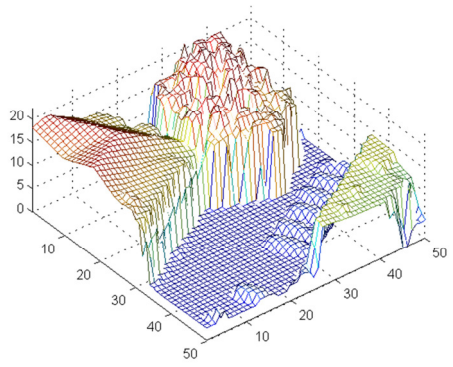
(6,5)


11
2.5
2
1.5
1
0.5
0
0
5
10
15
20
25

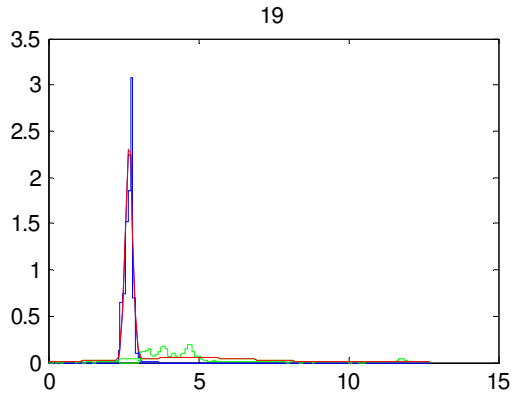

20
15
10
5
0
50
40
30
20
10
10
20
30
40
50

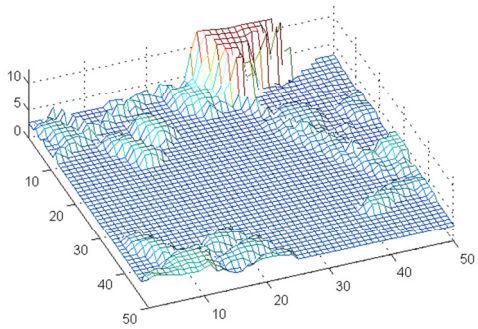
(6,6)


14
0.8
0.6
0.4
0.2
0
0
10
20
30

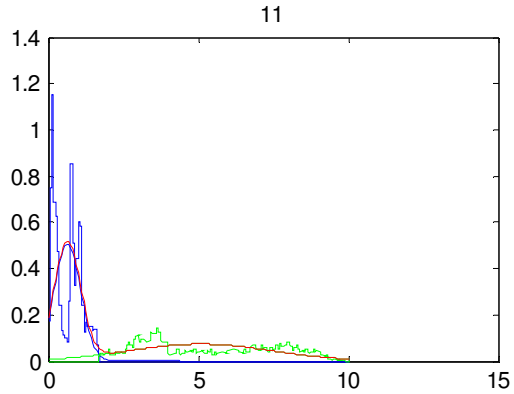

20
15
10
5
0
10
20
30
40
50
10
20
30
40
50

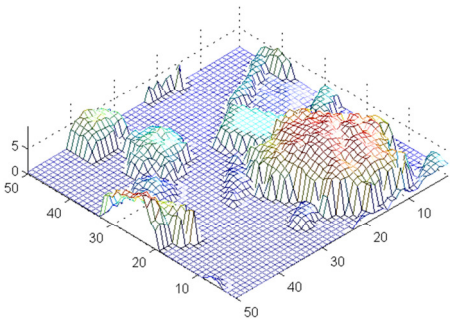
(6,7)


19
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15

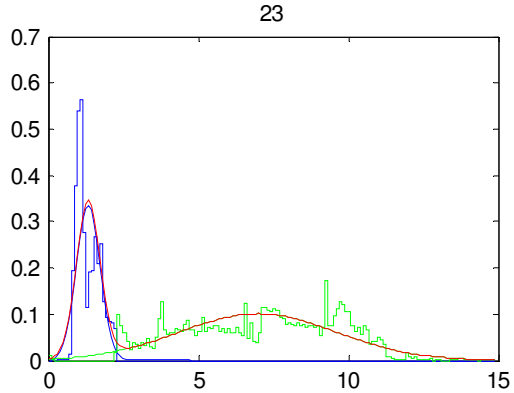

10
5
0
10
20
30
40
50
10
20
30
40
50

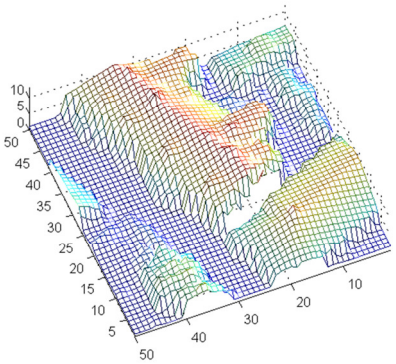
(6,8)


11
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15

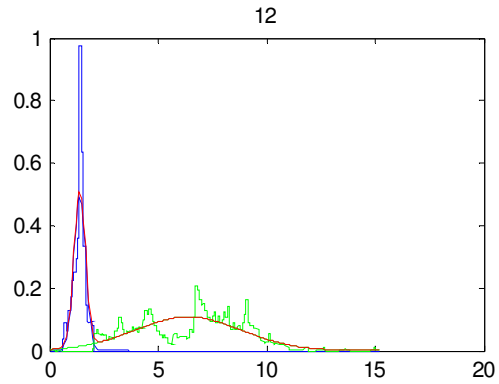

5
0
50
40
30
20
10
10
20
30
40
50

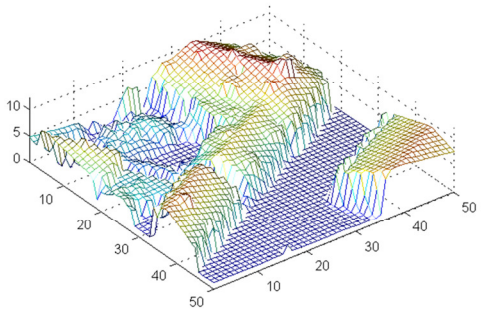
(6,9)


23
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5
10
15

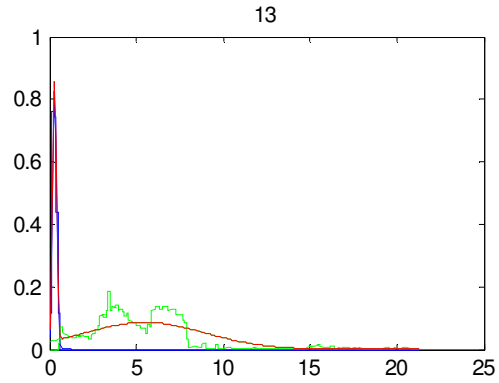

10
5
0
50
45
40
35
30
25
20
15
10
5
10
20
30
40
50

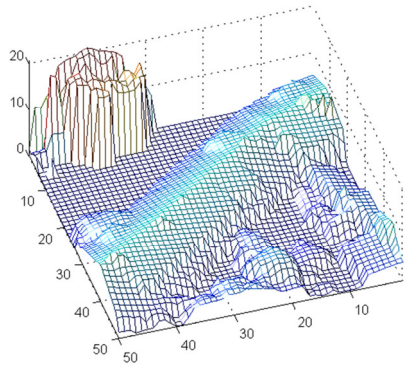
(6,10)


12
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20

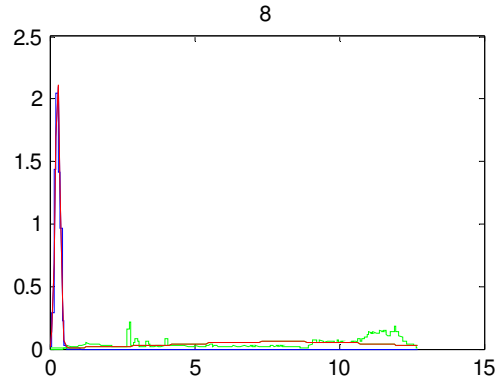

10
5
0
10
20
30
40
50
10
20
30
40
50

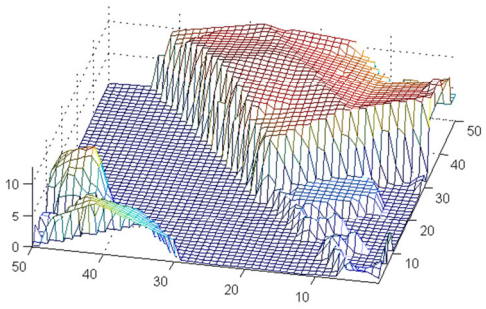
(7,5)


13
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20
25

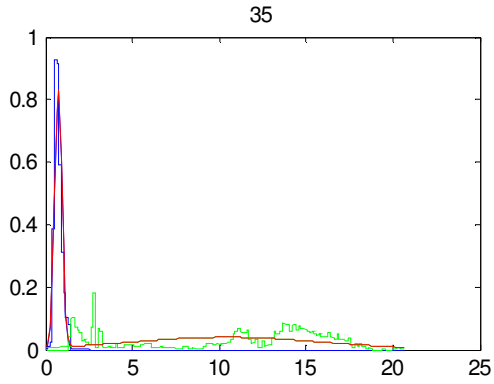

20
10
0
10
20
30
40
50
50
40
30
20
10

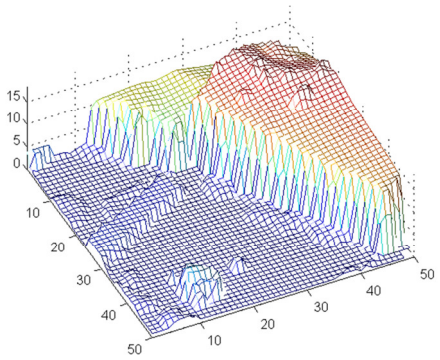
(7,6)


8
2.5
2
1.5
1
0.5
0
0
5
10
15

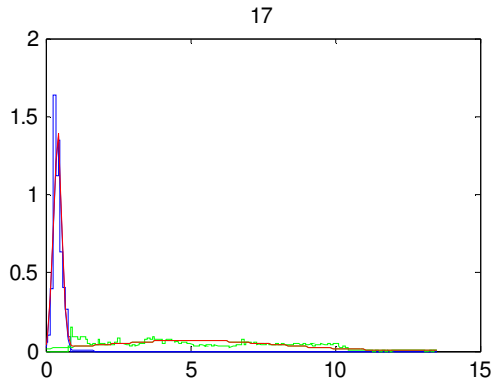

10
5
0
50
40
30
20
10
50
40
30
20
10

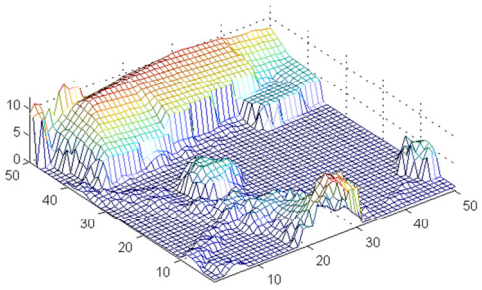
(7,7)


35
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20
25

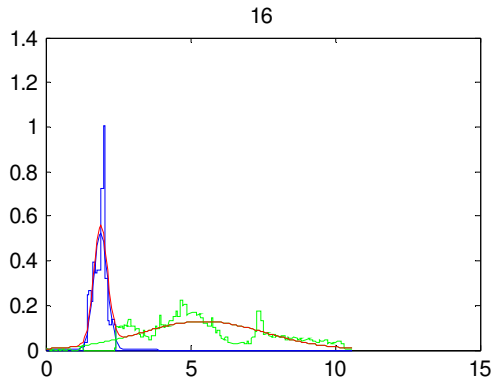

15
10
5
0
10
20
30
40
50
10
20
30
40
50

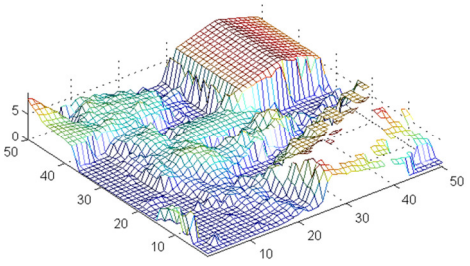
(7,8)


17
2
1.5
1
0.5
0
0
5
10
15

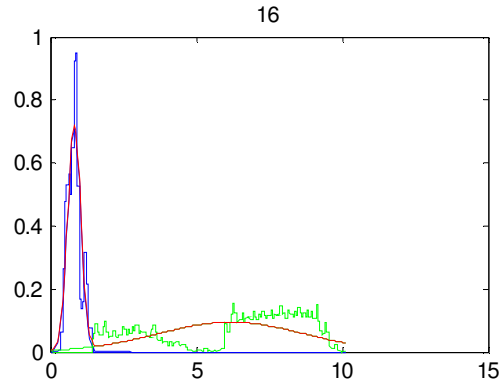

10
5
0
50
40
30
20
10
10
20
30
40
50

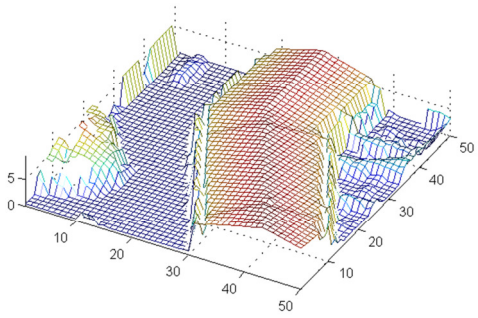
(7,9)


16
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15

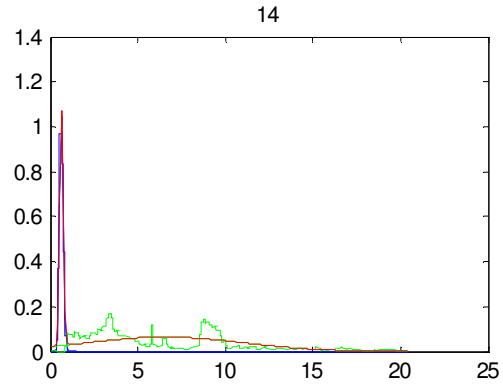

5
0
50
40
30
20
10
10
20
30
40
50

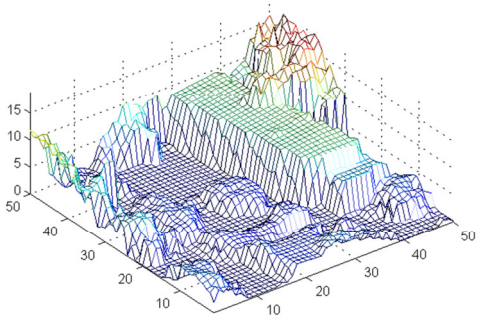
(7,10)


16
1
0.8
0.6
0.4
0.2
0
0
5
10
15

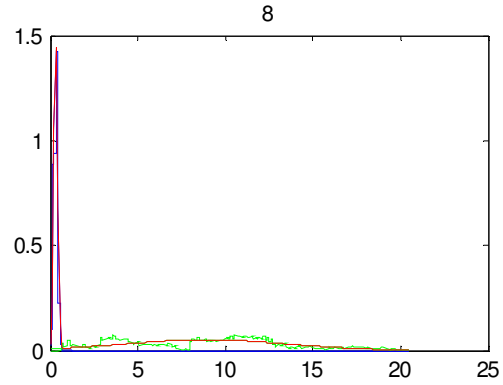

5
0
10
20
30
40
50
10
20
30
40
50

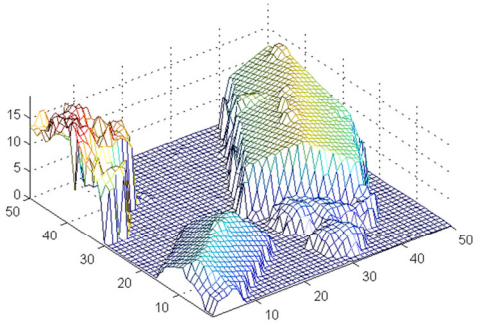
(8,5)


14
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20
25

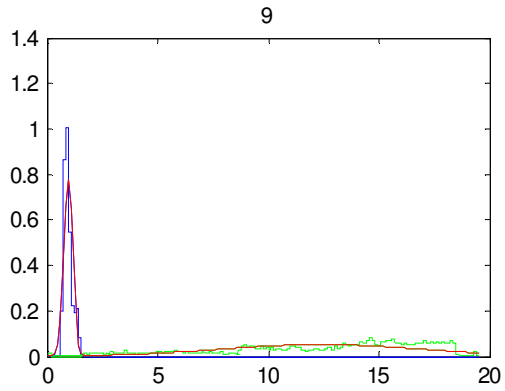

15
10
5
0
50
40
30
20
10
10
20
30
40
50

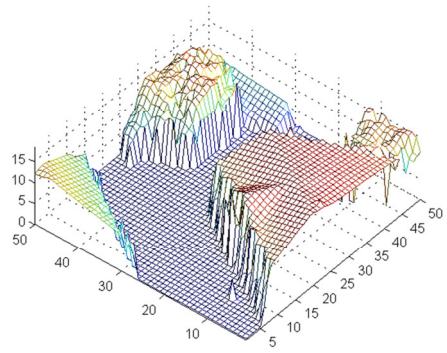
(8,6)


8
1.5
1
0.5
0
0
5
10
15
20
25

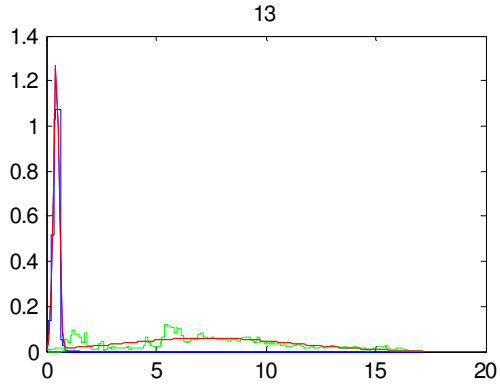

15
10
5
0
50
40
30
20
10
10
20
30
40
50

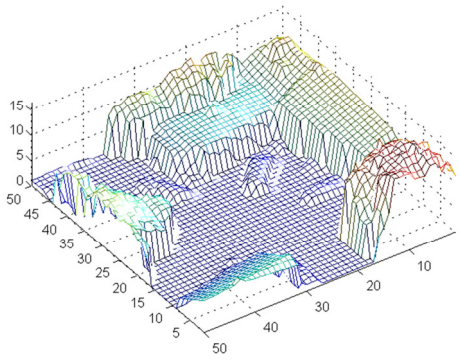
(8,7)


9
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20

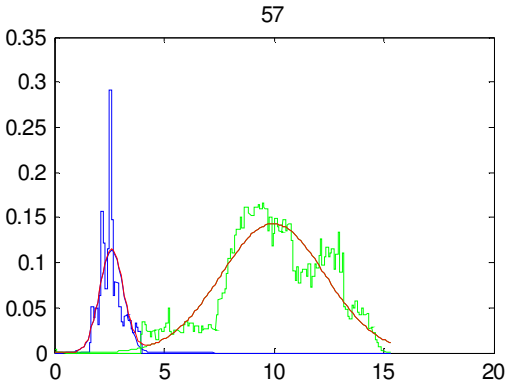


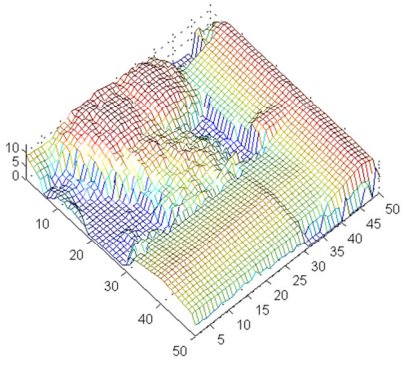
(8,8)


13
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20

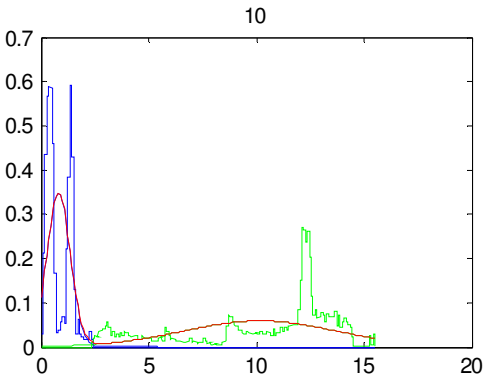


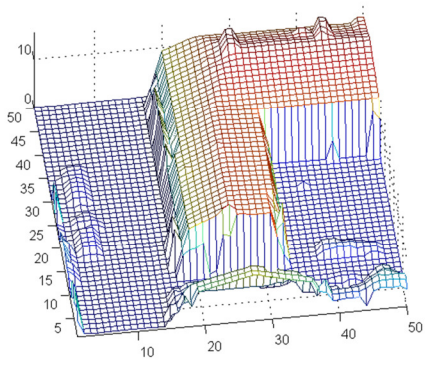
(8,9)


57
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
5
10
15
20

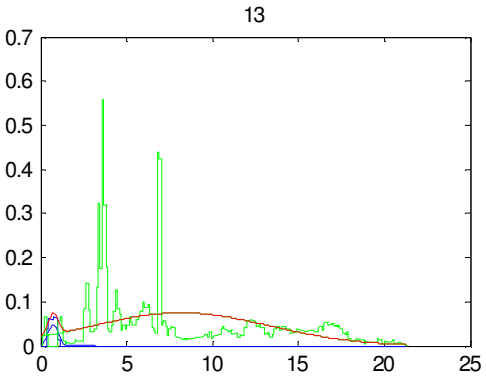


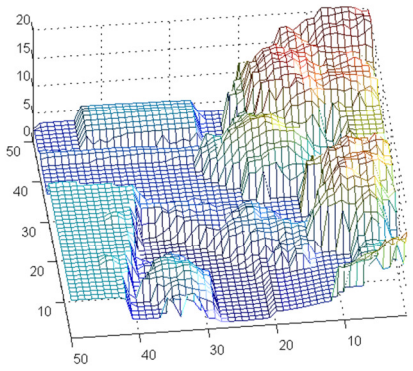
(8,10)


10
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5
10
15
20

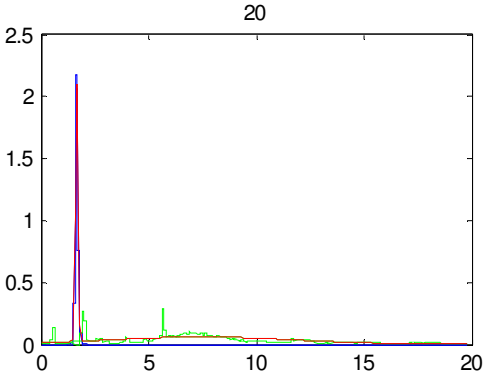


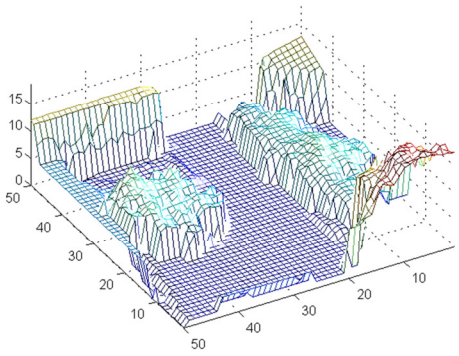
(9,5)


13
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5
10
15
20
25

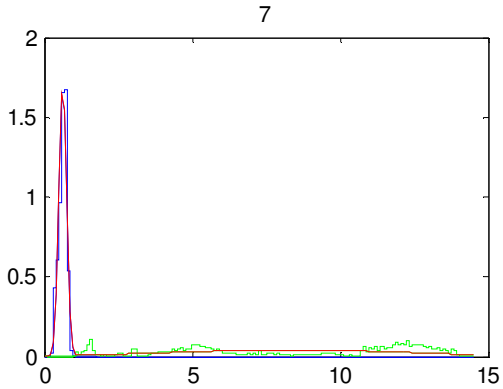


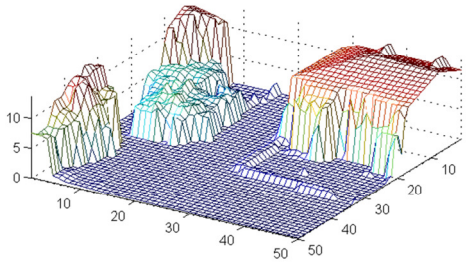
(9,6)


20
2.5
2
1.5
1
0.5
0
0
5
10
15
20

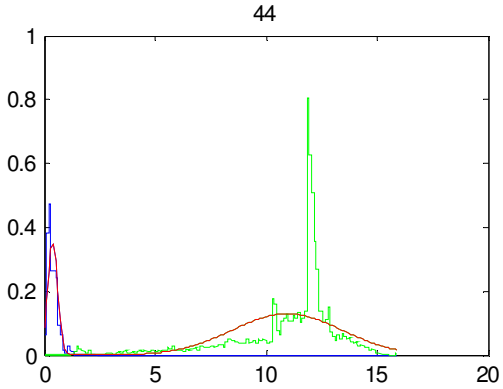


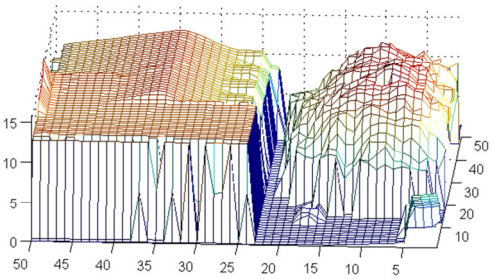
(9,7)


7
2
1.5
1
0.5
0
0
5
10
15

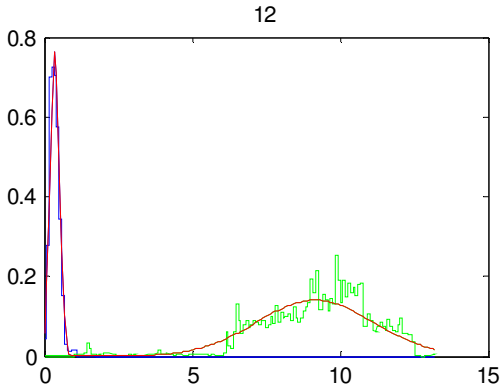

10
5
0
10
20
30
40
50
10
20
30
40
50

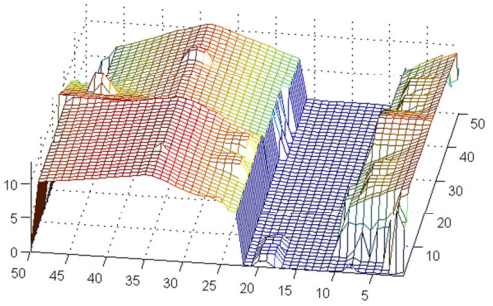
(9,8)


44
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20

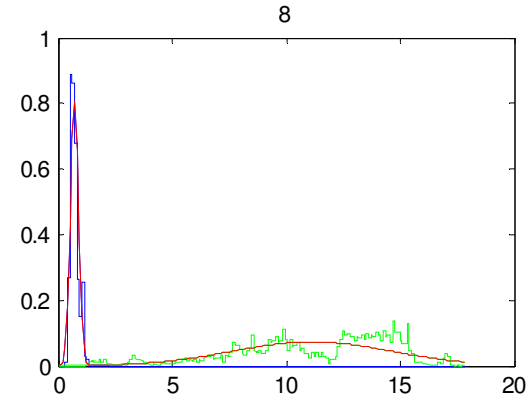

15
10
5
0
50
45
40
35
30
25
20
15
10
5
50
40
30
20
10

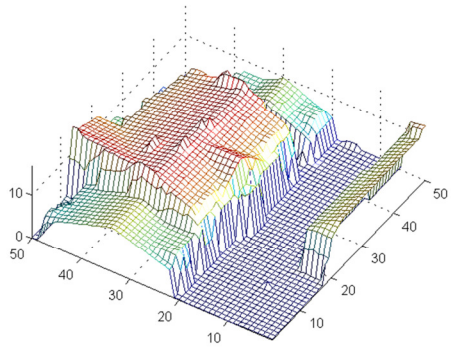
(9,9)


12
0.8
0.6
0.4
0.2
0
0
5
10
15

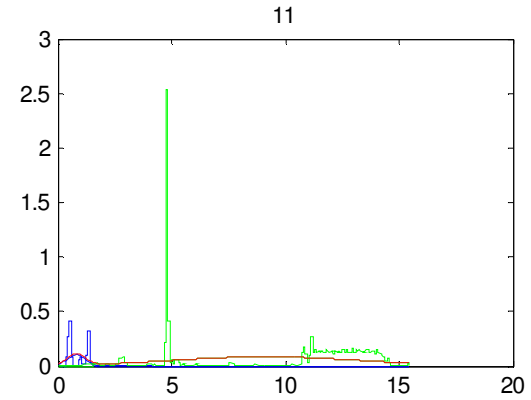

10
5
0
50
45
40
35
30
25
20
15
10
5
50
40
30
20
10

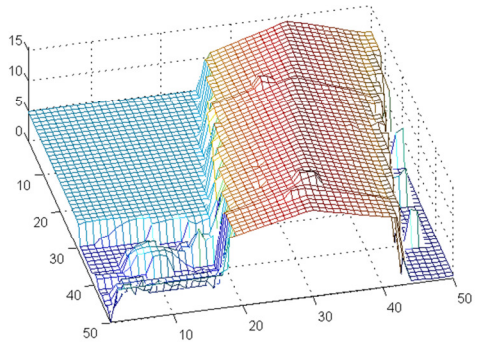
(9,10)


8
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20

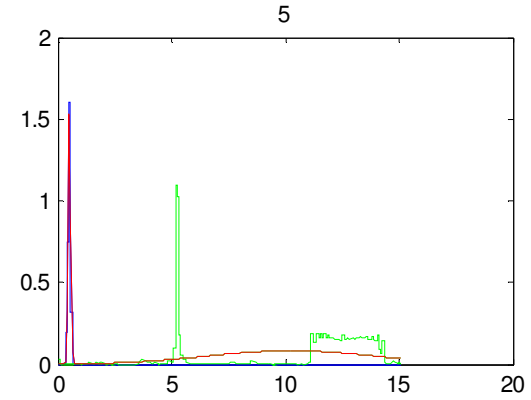

10
0
50
40
30
20
10
10
20
30
40
50

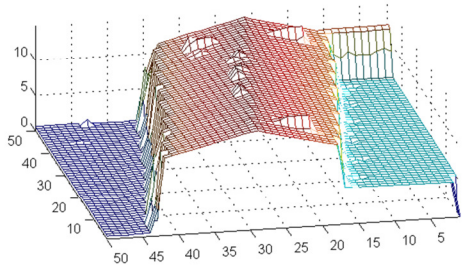
(10,5)


11
3
2.5
2
1.5
1
0.5
0
0
5
10
15
20

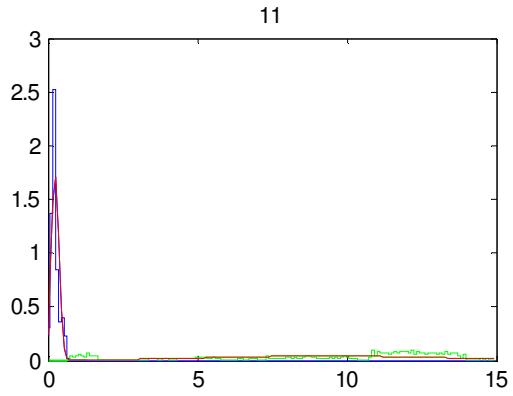

15
10
5
0
10
20
30
40
50
10
20
30
40
50

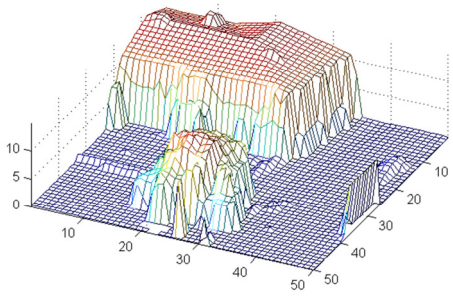
(10,6)


5
2
1.5
1
0.5
0
0
5
10
15
20

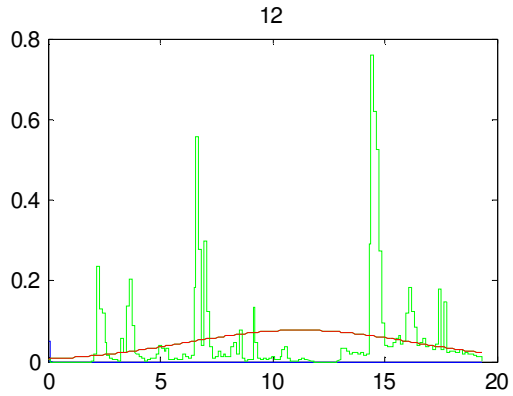

10
5
0
50
40
30
20
10
50
45
40
35
30
25
20
15
10
5

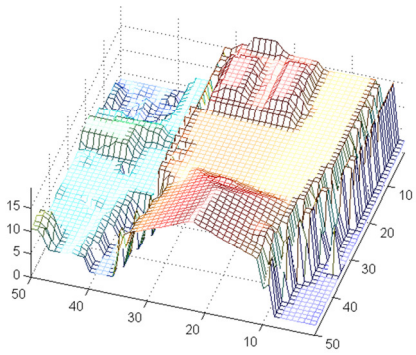
(10,7)


11
3
2.5
2
1.5
1
0.5
0
0
5
10
15

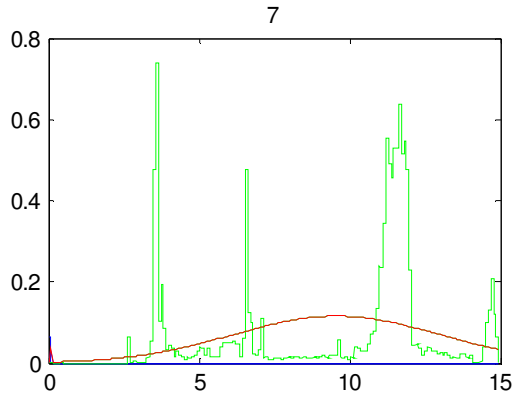

10
5
0
10
20
30
40
50
10
20
30
40
50

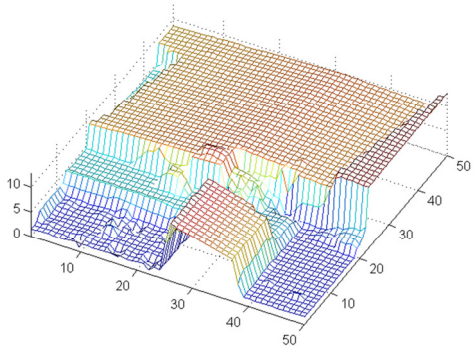
(10,8)


12
0.8
0.6
0.4
0.2
0
0
5
10
15
20

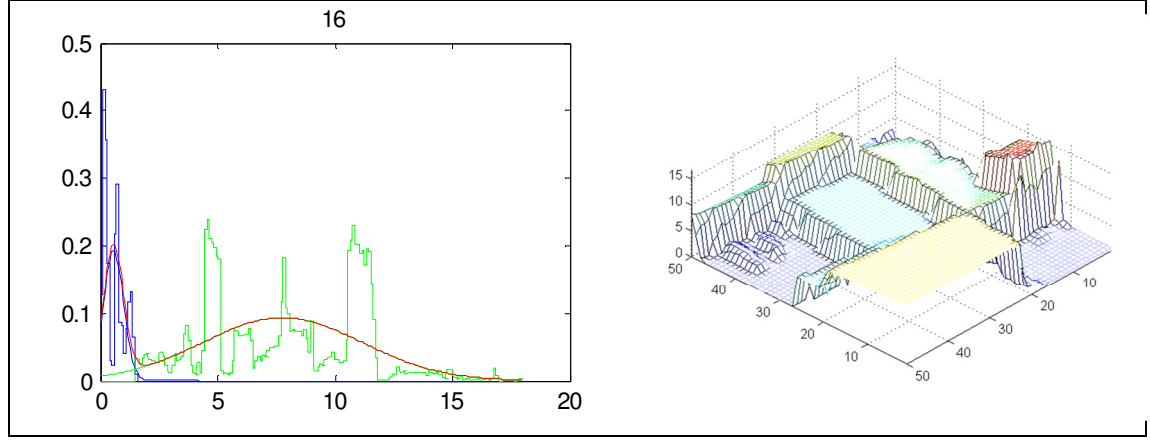

15
10
5
0
50
40
30
20
10
10
20
30
40
50

(10,9)


7
0.8
0.6
0.4
0.2
0
0
5
10
15


10
5
0
10
20
30
40
50
50
40
30
20
10

(10,10)



APPENDIX E. Robustness of the Modified RANSAC

Let {x, y, z | ax + by + cz + d = 0} be a plane model, the table below lists the 200 results of the estimated model parameters.

Column One (100 x 4)			
a	**b**	**c**	**d**
-0.00693	-0.0201	0.035737	-0.99914
-0.00674	-0.01955	0.03528	-0.99916
-0.00664	-0.01926	0.035045	-0.99918
-0.0067	-0.01947	0.035209	-0.99917
-0.0068	-0.01973	0.035437	-0.99915
-0.00675	-0.0196	0.035325	-0.99916
-0.00663	-0.01938	0.035104	-0.99917
-0.00671	-0.01954	0.035257	-0.99916
-0.00685	-0.01977	0.035497	-0.99915
-0.00681	-0.01979	0.035479	-0.99915
-0.00681	-0.01991	0.035536	-0.99915
-0.00671	-0.01955	0.03526	-0.99916
-0.00689	-0.01998	0.035641	-0.99914
-0.00654	-0.01911	0.034879	-0.99919
-0.00673	-0.01955	0.035278	-0.99916
-0.00686	-0.01984	0.035542	-0.99915
-0.00673	-0.01959	0.035309	-0.99916
-0.00668	-0.01951	0.035238	-0.99917
-0.00672	-0.01955	0.03528	-0.99916
-0.00665	-0.01941	0.03514	-0.99917
-0.00673	-0.0196	0.035305	-0.99916
-0.00677	-0.0197	0.035389	-0.99916
-0.00679	-0.01967	0.035394	-0.99916
-0.00682	-0.01985	0.035527	-0.99915
-0.00676	-0.01965	0.035362	-0.99916
-0.00679	-0.01968	0.035406	-0.99916
-0.00662	-0.01929	0.035044	-0.99918
-0.00679	-0.01973	0.035432	-0.99915
-0.00678	-0.01965	0.035372	-0.99916
-0.00688	-0.01995	0.03562	-0.99914
-0.00683	-0.01986	0.035535	-0.99915

Column Two (100 x 4)			
a	**b**	**c**	**d**
-0.00675	-0.01958	0.035319	-0.99916
-0.00671	-0.01952	0.035237	-0.99917
-0.00668	-0.01938	0.035145	-0.99917
-0.00672	-0.01951	0.035249	-0.99917
-0.00679	-0.01962	0.035361	-0.99916
-0.00674	-0.01962	0.035328	-0.99916
-0.00671	-0.01956	0.035278	-0.99916
-0.00653	-0.01914	0.034899	-0.99919
-0.0067	-0.01946	0.035211	-0.99917
-0.00664	-0.01928	0.035064	-0.99918
-0.00668	-0.01931	0.035108	-0.99917
-0.00669	-0.01949	0.035213	-0.99917
-0.00675	-0.01951	0.035268	-0.99916
-0.00666	-0.01945	0.035175	-0.99917
-0.00658	-0.01938	0.035072	-0.99918
-0.00665	-0.01934	0.035102	-0.99917
-0.00669	-0.01929	0.035097	-0.99918
-0.00672	-0.01953	0.03526	-0.99916
-0.00671	-0.01953	0.035255	-0.99916
-0.00675	-0.01961	0.035323	-0.99916
-0.00676	-0.01966	0.035365	-0.99916
-0.00668	-0.01936	0.03513	-0.99917
-0.00684	-0.01987	0.035553	-0.99915
-0.00674	-0.0196	0.035316	-0.99916
-0.00677	-0.01962	0.035353	-0.99916
-0.00669	-0.01951	0.035229	-0.99917
-0.00655	-0.01924	0.034967	-0.99918
-0.00657	-0.01915	0.03494	-0.99918
-0.00637	-0.0188	0.034579	-0.9992
-0.00676	-0.01964	0.035357	-0.99916
-0.00631	-0.01854	0.034394	-0.99922

-0.00671	-0.0195	0.035235	-0.99917
-0.00664	-0.01924	0.035032	-0.99918
-0.00677	-0.01969	0.035387	-0.99916
-0.00664	-0.01932	0.035091	-0.99918
-0.00671	-0.01945	0.03521	-0.99917
-0.0067	-0.01953	0.03525	-0.99917
-0.00691	-0.01999	0.035671	-0.99914
-0.00654	-0.01925	0.034955	-0.99918
-0.0067	-0.01935	0.035141	-0.99917
-0.00668	-0.01951	0.035226	-0.99917
-0.00675	-0.01964	0.035349	-0.99916
-0.00671	-0.01943	0.0352	-0.99917
-0.00667	-0.01962	0.035263	-0.99916
-0.00692	-0.02	0.035681	-0.99914
-0.00673	-0.01959	0.035303	-0.99916
-0.00659	-0.01927	0.035016	-0.99918
-0.00674	-0.0196	0.035327	-0.99916
-0.00681	-0.01968	0.035414	-0.99916
-0.00672	-0.01949	0.035234	-0.99917
-0.00673	-0.01964	0.035325	-0.99916
-0.00664	-0.01939	0.035124	-0.99917
-0.00665	-0.01925	0.035053	-0.99918
-0.00672	-0.01936	0.035158	-0.99917
-0.00658	-0.01912	0.034932	-0.99919
-0.00683	-0.01982	0.035502	-0.99915
-0.00666	-0.01929	0.035075	-0.99918
-0.00666	-0.01928	0.035079	-0.99918
-0.00669	-0.01938	0.035159	-0.99917
-0.00665	-0.01924	0.035055	-0.99918
-0.00675	-0.01958	0.035311	-0.99916
-0.00668	-0.01949	0.035208	-0.99917
-0.00676	-0.01963	0.035345	-0.99916
-0.00674	-0.0196	0.035315	-0.99916
-0.00672	-0.01963	0.035312	-0.99916
-0.00684	-0.01975	0.03548	-0.99915
-0.00658	-0.01921	0.03498	-0.99918
-0.00649	-0.01874	0.034638	-0.9992
-0.00672	-0.01955	0.035275	-0.99916
-0.00646	-0.01889	0.034706	-0.9992
-0.00675	-0.01959	0.035315	-0.99916

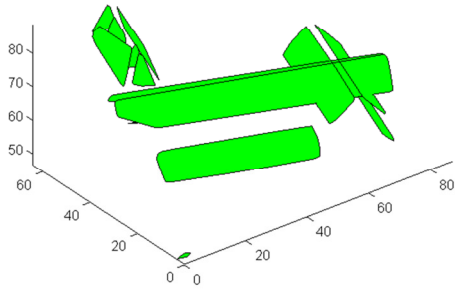
-0.00679	-0.01977	0.035445	-0.99915
-0.00674	-0.0196	0.035318	-0.99916
-0.00682	-0.01973	0.03546	-0.99915
-0.00669	-0.01942	0.035175	-0.99917
-0.00672	-0.01975	0.035366	-0.99916
-0.00683	-0.01972	0.035448	-0.99915
-0.0066	-0.01921	0.034984	-0.99918
-0.0066	-0.0192	0.034991	-0.99918
-0.00683	-0.01979	0.035485	-0.99915
-0.00685	-0.0199	0.035577	-0.99915
-0.00657	-0.01924	0.034986	-0.99918
-0.00667	-0.0194	0.035147	-0.99917
-0.00665	-0.0193	0.035078	-0.99918
-0.00682	-0.0198	0.035495	-0.99915
-0.00676	-0.0196	0.035338	-0.99916
-0.0066	-0.01929	0.035045	-0.99918
-0.00682	-0.01975	0.035461	-0.99915
-0.00661	-0.01925	0.035026	-0.99918
-0.00661	-0.01931	0.035052	-0.99918
-0.00658	-0.01921	0.034975	-0.99918
-0.00679	-0.0197	0.035408	-0.99916
-0.00677	-0.01956	0.035318	-0.99916
-0.00675	-0.01969	0.035366	-0.99916
-0.00688	-0.01989	0.035594	-0.99914
-0.00672	-0.01956	0.035285	-0.99916
-0.00685	-0.01985	0.035549	-0.99915
-0.00655	-0.01911	0.0349	-0.99919
-0.00668	-0.01949	0.035209	-0.99917
-0.00663	-0.01942	0.035125	-0.99917
-0.00683	-0.0198	0.035504	-0.99915
-0.00679	-0.01983	0.03548	-0.99915
-0.00673	-0.01961	0.035319	-0.99916
-0.00684	-0.01987	0.035542	-0.99915
-0.00672	-0.01943	0.035204	-0.99917
-0.00673	-0.01951	0.035259	-0.99917
-0.00679	-0.01971	0.035426	-0.99915
-0.00672	-0.01959	0.035301	-0.99916
-0.00677	-0.01969	0.035389	-0.99916
-0.00685	-0.01989	0.035571	-0.99915
-0.00655	-0.01919	0.034933	-0.99918

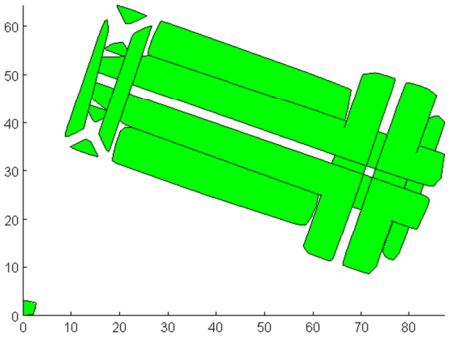
-0.00684	-0.01983	0.035525	-0.99915
-0.00668	-0.0193	0.035107	-0.99917
-0.00672	-0.01956	0.035285	-0.99916
-0.0067	-0.01938	0.035148	-0.99917
-0.00664	-0.01936	0.035105	-0.99917
-0.00674	-0.01958	0.035307	-0.99916
-0.00685	-0.01987	0.035553	-0.99915
-0.00678	-0.0197	0.035406	-0.99916
-0.00673	-0.0196	0.035311	-0.99916
-0.00667	-0.01942	0.035168	-0.99917
-0.00661	-0.01915	0.034962	-0.99918
-0.00663	-0.01936	0.035097	-0.99917
-0.00666	-0.01905	0.034933	-0.99919
-0.0069	-0.01997	0.035655	-0.99914
-0.00667	-0.01924	0.035062	-0.99918
-0.00682	-0.01982	0.035504	-0.99915
-0.00661	-0.01944	0.035111	-0.99917
-0.00665	-0.01931	0.035085	-0.99918
-0.00668	-0.01946	0.035199	-0.99917
-0.0067	-0.01986	0.035409	-0.99915
-0.00673	-0.01958	0.035299	-0.99916
-0.0068	-0.01974	0.035444	-0.99915
-0.00686	-0.01991	0.035587	-0.99914
-0.00662	-0.01927	0.035042	-0.99918
-0.0068	-0.01977	0.035462	-0.99915
-0.00676	-0.01965	0.035363	-0.99916
-0.0067	-0.01951	0.035244	-0.99917
-0.00674	-0.01961	0.035318	-0.99916
-0.00676	-0.01967	0.035368	-0.99916

-0.00671	-0.01955	0.035265	-0.99916
-0.00676	-0.01966	0.035368	-0.99916
-0.00674	-0.01967	0.035357	-0.99916
-0.00678	-0.01961	0.035349	-0.99916
-0.0067	-0.01954	0.035254	-0.99916
-0.00671	-0.01952	0.035247	-0.99917
-0.00677	-0.01989	0.035486	-0.99915
-0.00681	-0.01971	0.035441	-0.99915
-0.0067	-0.01945	0.035196	-0.99917
-0.00666	-0.01947	0.035183	-0.99917
-0.00676	-0.0196	0.035335	-0.99916
-0.00676	-0.01966	0.03537	-0.99916
-0.00655	-0.0193	0.034997	-0.99918
-0.00683	-0.0198	0.035499	-0.99915
-0.00666	-0.01943	0.03516	-0.99917
-0.00681	-0.01969	0.03542	-0.99916
-0.00664	-0.01932	0.03509	-0.99918
-0.00663	-0.01934	0.035082	-0.99918
-0.0068	-0.01967	0.035409	-0.99916
-0.00673	-0.01961	0.035314	-0.99916
-0.00677	-0.01961	0.035337	-0.99916
-0.00677	-0.01955	0.035316	-0.99916
-0.00697	-0.02101	0.036221	-0.9991
-0.00665	-0.01933	0.035098	-0.99917
-0.00668	-0.01948	0.035209	-0.99917
-0.00673	-0.01952	0.035261	-0.99916
-0.0066	-0.01919	0.034984	-0.99918
-0.0066	-0.01927	0.035015	-0.99918
-0.00675	-0.01958	0.035307	-0.99916

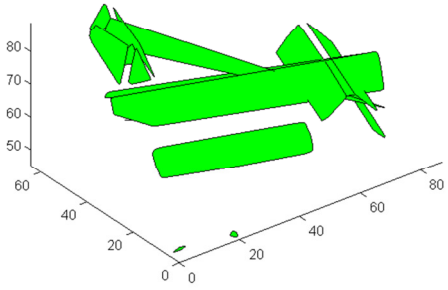
APPENDIX F. Running the Modified RANSAC for 10 Times

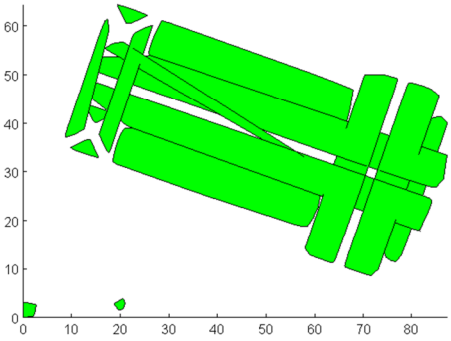
Implementation 1:
Elapsed time = 10.5856s



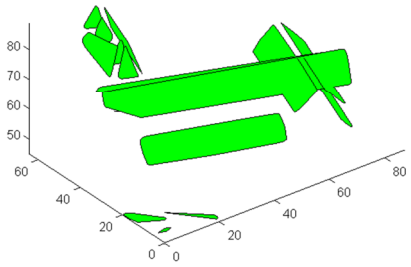


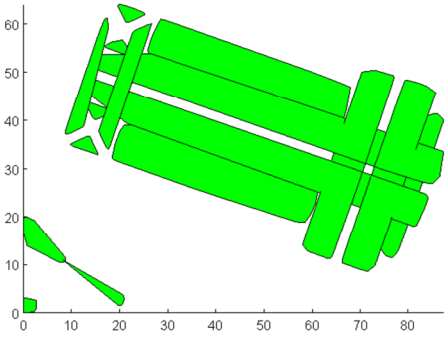
Implementation 2:
Elapsed time = 10.1564s



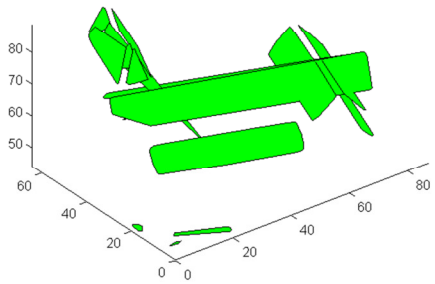
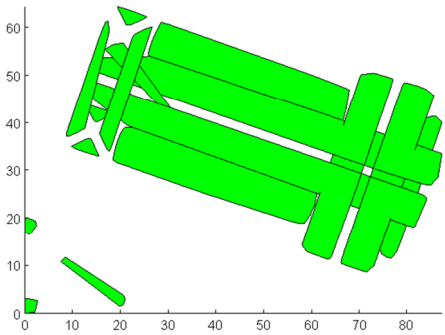


Implementation 3:
Elapsed time = 9.8941s

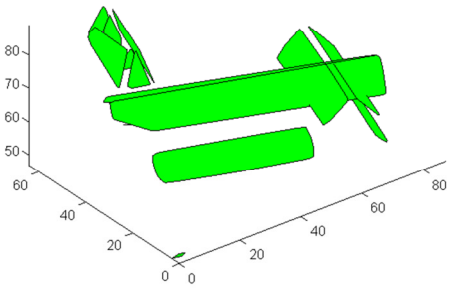
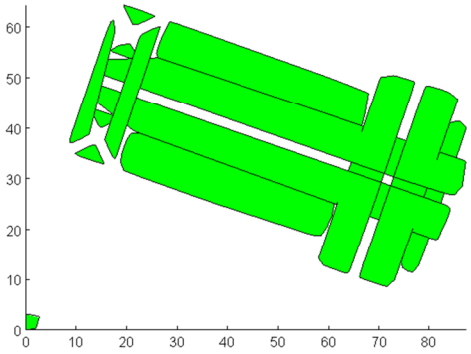




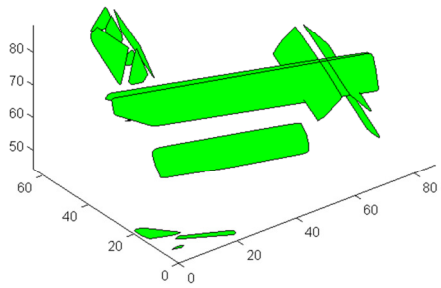
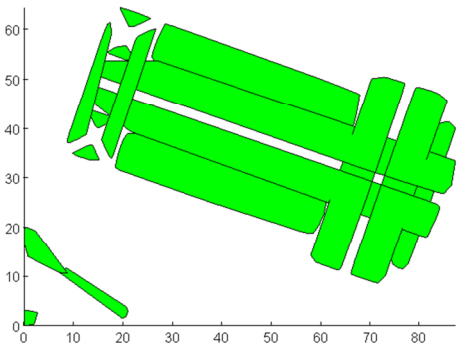
Implementation 4:
Elapsed time = 10.3007s

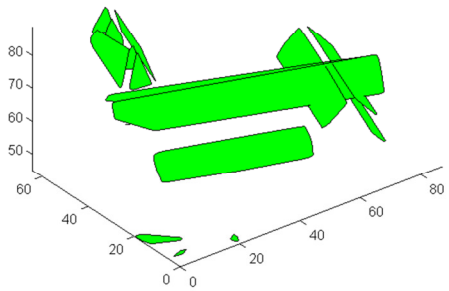
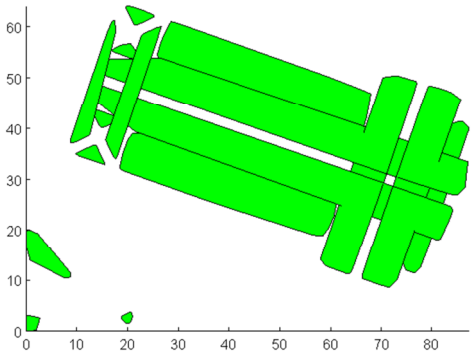
Implementation 5:
Elapsed time = 10.0485s

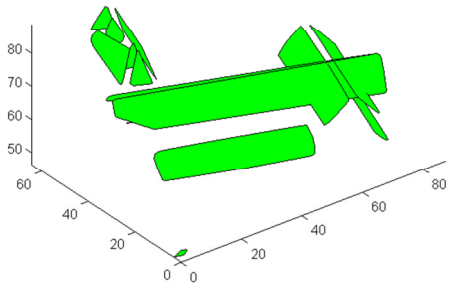
Implementation 6:
Elapsed time = 10.3277s

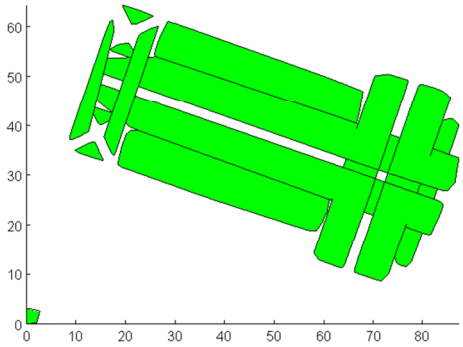



Implementation 7:
Elapsed time = 9.7276s

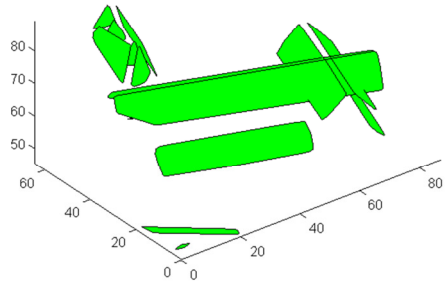



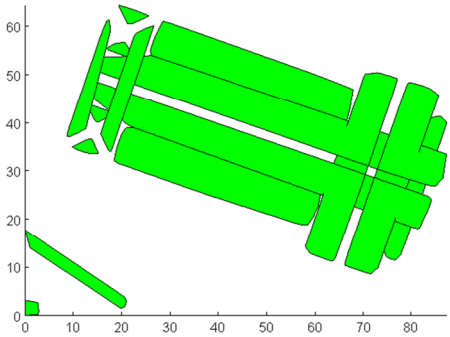
Implementation 8:
Elapsed time = 10.7447s



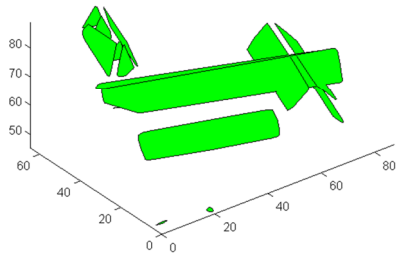


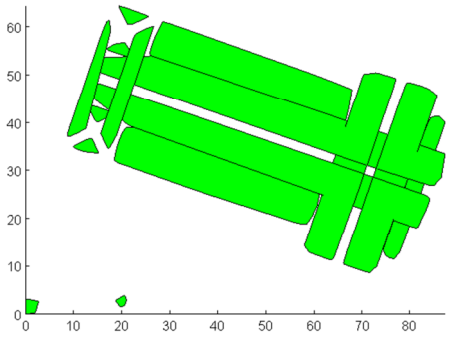
Implementation 9:
Elapsed time = 9.9212s





Implementation 10:
Elapsed time = 10.6762s





Institutionen för naturgeografi och ekosystemvetenskap, Lunds Universitet.

Student examensarbete (Seminarieuppsatser). Uppsatserna finns tillgängliga på institutionens geobibliotek, Sölvegatan 12, 223 62 LUND. Serien startade 1985. Hela listan och själva uppsatserna är även tillgängliga på LUP student papers (www.nateko.lu.se/masterthesis) och via Geobiblioteket (www.geobib.lu.se)

The student thesis reports are available at the Geo-Library, Department of Physical Geography, University of Lund, Sölvegatan 12, S-223 62 Lund, Sweden.
Report series started 1985. The complete list and electronic versions are also electronic available at the LUP student papers (www.nateko.lu.se/masterthesis) and through the Geo-library (www.geobib.lu.se)

199 Herbert Mbufong Njuabe (2011): Subarctic Peatlands in a Changing Climate: Greenhouse gas response to experimentally increased snow cover

200 Naemi Gunlycke & Anja Tuomaala (2011): Detecting forest degradation in Marakwet district, Kenya, using remote sensing and GIS

201 Nzung Seraphine Ebang (2011): How was the carbon balance of Europe affected by the summer 2003 heat wave? A study based on the use of a Dynamic Global Vegetation Model; LPJ-GUESS

202 Per-Ola Olsson (2011): Cartography in Internet-based view services – methods to improve cartography when geographic data from several sources are combined

203 Kristoffer Mattisson (2011): Modelling noise exposure from roads – a case study in Burlövs municipality

204 Erik Ahlberg (2011): BVOC emissions from a subarctic Mountain birch: Analysis of short-term chamber measurements.

205 Wilbert Timiza (2011): Climate variability and satellite – observed vegetation responses in Tanzania.

206 Louise Svensson (2011): The ethanol industry - impact on land use and biodiversity. A case study of São Paulo State in Brazil.

207 Fredrik Fredén (2011): Impacts of dams on lowland agriculture in the Mekong river catchment.

208 Johanna Hjärpe (2011): Kartläggning av kväve i vatten i LKAB:s verksamhet i Malmberget år 2011 och kvävets betydelse i akvatiska ekosystem ur ett lokalt och ett globalt perspektiv

209 Oskar Löfgren (2011): Increase of tree abundance between 1960 and 2009 in the treeline of Luongastunturi in the northern Swedish Scandes

210 Izabella Rosengren (2011): Land degradation in the Ovitoto region of Namibia: what are the local causes and consequences and how do we avoid them?

211 Irina Popova (2011): Agroforestry och dess påverkan på den biofysiska miljön i

Afrika.

212 Emilie Walsund (2011): Food Security and Food Sufficiency in Ethiopia and Eastern Africa.

213 Martin Bernhardson (2011): Jökulhlaups: Their Associated Landforms and Landscape Impacts.

214 Michel Tholin (2011): Weather induced variations in raptor migration; A study of raptor migration during one autumn season in Kazbegi, Georgia, 2010

215 Amelie Lindgren (2011) The Effect of Natural Disturbances on the Carbon Balance of Boreal Forests.

216 Klara Århem (2011): Environmental consequences of the palm oil industry in Malaysia.

217 Ana Maria Yáñez Serrano (2011) Within-Canopy Sesquiterpene Ozonolysis in Amazonia

218 Edward Kashava Kuliwoye (2011) Flood Hazard Assessment by means of Remote Sensing and Spatial analyses in the Cuvelai Basin Case Study Ohangwena Region –Northern Namibia

219 Julia Olsson (2011) GIS-baserad metod för etablering av centraliserade biogasanläggningar baserad på husdjursgödsel.

220 Florian Sallaba (2011) The potential of support vector machine classification of land use and land cover using seasonality from MODIS satellite data

221 Salem Beyene Ghezahai (2011) Assessing vegetation changes for parts of the Sudan and Chad during 2000-2010 using time series analysis of MODIS-NDVI

222 Bahzad Khaled (2011) Spatial heterogeneity of soil CO2 efflux at ADVEX site Norunda in Sweden

223 Emmy Axelsson (2011) Spatiotemporal variation of carbon stocks and fluxes at a clear-cut area in central Sweden

224 Eduard Mikayelyan (2011) Developing Android Mobile Map Application with Standard Navigation Tools for Pedestrians

225 Johanna Engström (2011) The effect of Northern Hemisphere teleconnections on the hydropower production in southern Sweden

226 Kosemani Bosede Adenike (2011) Deforestation and carbon stocks in Africa

227 Ouattara Adama (2011) Mauritania and Senegal coastal area urbanization, ground water flood risk in Nouakchott and land use/land cover change in Mbour area

228 Andrea Johansson (2011) Fire in Boreal forests

229 Arna Björk Þorsteinsdóttir (2011) Mapping *Lupinus nootkatensis* in Iceland using SPOT 5 images

230 Cléber Domingos Arruda (2011) Developing a Pedestrian Route Network Service (PRNS)

231 Nitin Chaudhary (2011) Evaluation of RCA & RCA GUESS and estimation of vegetation-climate feedbacks over India for present climate

232 Bjarne Munk Lyshede (2012) Diurnal variations in methane flux in a low-arctic fen in Southwest Greenland
233 Zhendong Wu (2012) Dissolved methane dynamics in a subarctic peatland
234 Lars Johansson (2012) Modelling near ground wind speed in urban environments using high-resolution digital surface models and statistical methods
235 Sanna Dufbäck (2012) Lokal dagvattenhantering med grönytefaktorn
236 Arash Amiri (2012)Automatic Geospatial Web Service Composition for Developing a Routing System
237 Emma Li Johansson (2012) The Melting Himalayas: Examples of Water Harvesting Techniques
238 Adelina Osmani (2012) Forests as carbon sinks - A comparison between the boreal forest and the tropical forest
239 Uta Klönne (2012) Drought in the Sahel – global and local driving forces and their impact on vegetation in the 20th and 21st century
240 Max van Meeningen (2012) Metanutsläpp från det smältande Arktis
241 Joakim Lindberg (2012) Analys av tillväxt för enskilda träd efter gallring i ett blandbestånd av gran och tall, Sverige
242 Caroline Jonsson (2012) The relationship between climate change and grazing by herbivores; their impact on the carbon cycle in Arctic environments
243 Carolina Emanuelsson and Elna Rasmusson (2012) The effects of soil erosion on nutrient content in smallholding tea lands in Matara district, Sri Lanka
244 John Bengtsson and Eric Torkelsson (2012) The Potential Impact of Changing Vegetation on Thawing Permafrost: Effects of manipulated vegetation on summer ground temperatures and soil moisture in Abisko, Sweden
245 Linnea Jonsson (2012). Impacts of climate change on Pedunculate oak and Phytophthora activity in north and central Europe
246 Ulrika Belsing (2012) Arktis och Antarktis föränderliga havsistäcken
247 Anna Lindstein (2012) Riskområden för erosion och näringsläckage i Segeåns avrinningsområde
248 Bodil Englund (2012) Klimatanpassningsarbete kring stigande havsnivåer i Kalmar läns kustkommuner
249 Alexandra Dicander (2012) GIS-baserad översvämningskartering i Segeåns avrinningsområde
250 Johannes Jonsson (2012) Defining phenology events with digital repeat photography
251 Joel Lilljebjörn (2012) Flygbildsbaserad skyddszonsinventering vid Segeå
252 Camilla Persson (2012) Beräkning av glaciärers massbalans – En metodanalys med fjärranalys och jämviktslinjehöjd över Storglaciären
253 Rebecka Nilsson (2012) Torkan i Australien 2002-2010 Analys av möjliga orsaker och effekter
254 Ning Zhang (2012) Automated plane detection and extraction from airborne laser scanning data of dense urban areas

255 Bawar Tahir (2012) Comparison of the water balance of two forest stands using the BROOK90 model