# PRE-CLASSIFICATION OF POINTS AND SEGMENTATION OF URBAN OBJECTS BY SCAN LINE ANALYSIS OF AIRBORNE LIDAR DATA

M. Hebel [a], U. Stilla [b]

[a] FGAN-FOM, Research Institute for Optronics and Pattern Recognition, 76275 Ettlingen, Germany - hebel@fom.fgan.de
[b] Photogrammetry and Remote Sensing, Technische Universitaet Muenchen, 80290 Muenchen, Germany - stilla@bv.tum.de

**KEY WORDS:** Laser scanning, LiDAR, airborne remote sensing, on-line processing, classification, segmentation, urban data

**ABSTRACT:**

Currently available laser scanners are capable of performing hundreds of thousands of range measurements per second at a range resolution of a few centimeters. Despite that increasing performance, up to now airborne LiDAR mapping has established itself only in fields of application that do not require on-line data processing. A typical example in this context is urban modeling. We want to point out some other tasks like object recognition, situation analysis and on-line change detection that have come into reach of today's LiDAR technology. Primary goal of the work presented in this paper is the on-line data preparation for subsequent analysis. We present a workflow of real-time capable filter operations to detect the ground level and distinguish between clutter and man-made objects in airborne laser scanner data of urban regions. Based on interpretation of single scan lines, straight line segments are first segmented and then connected, and the resulting surfaces are delineated by a polygon. A preliminary step is done towards fast reconstruction of buildings for rapid city-modeling, co-registration or recognition of urban structures.

## 1. INTRODUCTION

### 1.1 Problem description

Airborne laser scanning (ALS) of urban regions is nowadays commonly used as a basis for 3D city modeling. Typical applications lie in the fields of city planning, tourism, telecommunication, architecture, archeology and environmental protection. A good overview and a thorough description of ALS principles can be found in (Wehr & Lohr, 1999). Laser scanning has several advantages compared to classical aerial photography. It delivers direct 3D measurements independently from natural lighting conditions, and it offers high accuracy and point density.

Despite increasing performance of LiDAR systems, most remote sensing tasks that require on-line data processing are still accomplished by the use of conventional CCD or infrared cameras. Typical examples are airborne monitoring and observation devices that are used for automatic object recognition, situation analysis or real-time change detection. Looking at urban regions, these sensors can support law enforcement, firefighting, disaster management, and medical or other emergency services. At the same time it is often desirable to assist pilots with automatic aircraft guidance in case of poor visibility conditions. Three-dimensional information as provided by the LiDAR sensor technology would ease these tasks, but in many cases the complexity of irregularly distributed laser point clouds prevents an on-line data processing.

In another aspect, on-line pre-processing and reducing the ALS data to the essential information are important for efficient data storage and data transfer in a sensor network. Additionally, when combining different data sets, e.g. showing the same urban region in oblique view from different directions, the pairwise co-registration is even more accurate when dealing with structures of higher order like surfaces instead of the original point clouds (Gruen & Akca, 2005).

In the classical workflow of ALS data processing e.g. for city modelling, the first step is to register all the collected data by using navigational sensors (INS and GPS), resulting in an irregularly distributed 3D point cloud. Automatic processing of these data is quite complex since it is necessary to determine a set of nearest neighbors for each data point to handle search operations within the data set. The common technique for that is the generation of a triangulated irregular network (TIN). This approach leads to most accurate results, but it is not applicable for real-time applications.

### 1.2 Contribution

Most of currently used airborne laser scanners like the RIEGL LMS-Q560 utilize opto-mechanical beam scanning to measure range values in single scan lines. The third dimension is provided by the moving airborne platform. This paper aims at fast pre-classification of LiDAR points and segmentation of buildings in urban environments based on the analysis of these scan lines. Instead of initial georeferencing of all range measurements, the analysis of geometric features in the respective local neighborhood of each data point is performed directly on the 2D regularly distributed scan line data. These operations can be executed comparatively fast and are applicable for online data processing. This paper presents a workflow of real-time capable operations to detect the ground level and distinguish between clutter and man-made objects in airborne laser scanner data of urban regions. A preliminary step is done towards the fast reconstruction of buildings including their facades for rapid city-modeling or object recognition. The detected rooftops can even be used for a fast co-registration of different views of the same urban region. If an accurate city model is already available, the proposed methods could be used for a structural comparison, e.g. for change detection.

### 1.3 Related work

Processing of laser scanner point clouds for automatic filtering, segmentation, classification and modeling of structures has been thoroughly studied by many researchers in recent years. In some parts our work follows or extends the ideas presented in other articles that are especially mentioned in this section. According to (Vosselman et al., 2004), two major classes of segmentation algorithms can be pointed out: surface growing and scan line segmentation. Pointwise growing of planar faces in combination with a three-dimensional Hough transform as described by Vosselman and Dijkman (2001) can be used to reconstruct accurate 3D building models, but since it needs a TIN structure this approach is suited only for off-line data processing.

Fundamental ideas on fast segmentation of range images into planar regions based on scan line analysis have been published by Jiang and Bunke (1994, 1999). Their algorithm first divides each row of a range image into straight line segments, and then it performs a region growing process with these line segments. Since this is the most obvious way of proceeding, we basically adapted that approach to work with our data. Instead of range images we have to deal with continuously recorded scan lines that are not necessarily parallel. Jiang and Bunke originally used the splitting algorithm of Duda and Hart. Axelsson (1999) described a classification of points in a scan line based on the second derivatives of the elevation difference. In contrast our filtering of straight line segments is based on a robust estimation technique (RANSAC). This was also proposed by Hatger and Brenner (2003) for the segmentation of roads in regularized ALS grid data. Since our data is not regularized this way, we had to implement a different method to merge straight line segments into surfaces. Sithole and Vosselman (2003, 2005) used scan line based methods for structure detection in point clouds and filtering of ALS data, but they refer to the term "scan line" in a different manner. Instead of processing hardware generated scan lines like we do, they define scan lines with multiple orientations by slicing the point cloud and connecting 3D points based on height difference and slope.

From an application-oriented point of view, only few articles have yet been published on on-line processing of laser scanner data. Examples are automatic uploading of piled box-like objects (Katsoulas & Werber, 2004) or the exploitation of LiDAR data to obtain traffic flow estimates, described by Toth et al. (2004).

## 2. EXPERIMENTAL SETUP

The sensors that are briefly described here have been attached to a Bell UH1-D helicopter to acquire the data shown in this paper.

### 2.1 Navigational sensors

The APPLANIX POS AV comprises a GPS receiver and a gyro-based inertial measurement unit (IMU), which is the core element of the navigational system. The GPS data are used for drift compensation and geo-referencing, whereas the IMU determines accelerations with high precision. These data are transferred to the position and orientation computing system (PCS), where they are fused by a Kalman filter, resulting in position and orientation estimates for the sensor platform.

### 2.2 Laser Scanner

The RIEGL LMS-Q560 is a laser scanner that gives access to the full waveform by digitizing the echo signal. The sensor makes use of the time-of-flight distance measurement principle with nanosecond infrared pulses. Opto-mechanical beam scanning provides single scan lines, where each measured distance can be geo-referenced according to the position and orientation of the sensor. Waveform analysis can contribute intensity and pulse-width as additional features, which is typically done by fitting Gaussian functions to the waveforms. Since we are mainly interested in fast processing of the range measurements, we neglect full waveform analysis throughout this paper. Range $d$ (expressed in meter) under scan angle $\alpha$ (-30 degree to +30 degree) is estimated corresponding to the first returning echo pulse as it can be found by a constant fraction discriminator (CFD). Positions with none or multiple returns are discarded, and with $x=d\cdot sin(\alpha)$, $y=d\cdot cos(\alpha)$ the scan line data is given in 2D Cartesian coordinates. Since the rotating polygon mirror operates with 1000 scan positions, one scan line can successively be stored in an array $A$ of maximum size $(1000, 2)$. In praxis, navigational data assigned to each range measurement need also to be stored in that array for later georeferencing. Figure 1 illustrates the process of data acquisition and exemplarily shows a scan line measured at an urban area. Buildings appear to be upside down in that representation, because the rooftops are nearer to the sensor than ground level.

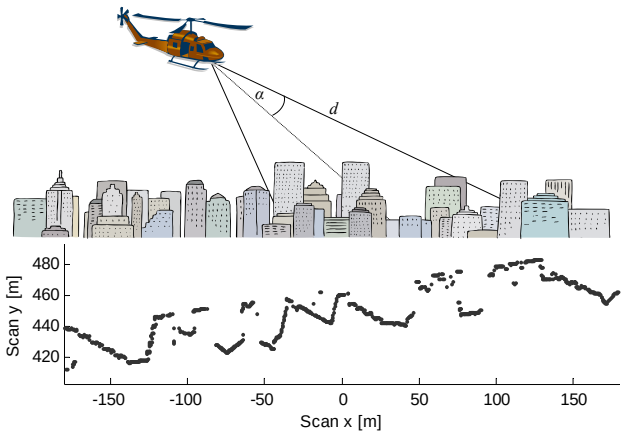



Figure 1. Illustration of data acquisition (top) and exemplary scan line (bottom).

## 3. USED METHODS AND DATA PROCESSING

Most parts of typical buildings will appear as local straight line segments within the scan line data, even if the airborne laser scanner is used in oblique configuration to obtain information concerning the facades of buildings. Our method is intended to filter the data points in each scan line to keep only those points that are most promising to represent parts of buildings. Consequently, points at ground level and those belonging to objects with an irregular shape like trees or other vegetation are removed. To distinguish between clutter and man-made objects, the RANSAC technique is used to fit straight line segments to the scan line data.

### 3.1 Random sample consensus (RANSAC)

The random-sample-consensus paradigm (RANSAC) as described by Fischler and Bolles (1981) is a standard technique to estimate parameters of a mathematical model underlying a set of observed data. It is particularly used in case that the observed data contain data points which can be explained by a set of model parameters (inliers) and such data points that do not fit the model (outliers). To apply the RANSAC scheme, a procedural method has to be available that determines the

parameters to fit the model to a minimal subset of the data. In this paper we use RANSAC to fit line segments to subsets of 2D points in a scan line. If we have a set of $n$ points $\{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n\}$ and we assume that this set mostly contains points that approximately lie on straight line (inliers) and some others that do not (outliers), simple least squares model fitting would lead to poor results because the outliers would affect the estimated parameters. RANSAC estimates a straight line only by taking the inliers into account, provided that the probability of choosing only inliers among the data points is sufficiently high. To compute a straight line, a random sample of two points (the minimal subset) $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ is selected. The resultant line's normal vector $\boldsymbol{n}_0$ can easily be computed by interchanging the two coordinates of $(\boldsymbol{p}_i - \boldsymbol{p}_j)$, altering the sign of one component and normalizing the vector to unit length. This yields the normal vector $\boldsymbol{n}_0$ and with $(\boldsymbol{x}-\boldsymbol{p}_i)\cdot\boldsymbol{n}_0 = 0$ the line's Hessian normal form is given. Given this representation it is easy to check any other point $\boldsymbol{p}$ if it is an inlier or outlier simply by computing the distance $d = |(\boldsymbol{p}-\boldsymbol{p}_i)\cdot\boldsymbol{n}_0|$ to the previously obtained line. If the distance $d$ is below a pre-defined threshold, we assess that point as inlier. The number of inliers and the average distance of all inliers to the line are used to evaluate the quality of the fitted straight line. This procedure is repeated several times in order to converge to the best possible straight line.

### 3.2 Scan line analysis

The following steps are executed to fit straight line segments to the scan line data and to remove irregularly shaped objects:

(1) Choose an unmarked position $i$ at random among the available data in the array $A$ holding the scan line data.
(2) Check a sufficiently large interval around this position $i$ for available data, resulting in a set $S$ of 2D points.
(3) Set the counter $k$ to zero.
(4) If $S$ contains more than a specific number of points (e.g. at least six), continue. Otherwise mark the current position $i$ as discarded and go to step 14.
(5) Increase the counter $k$ by one.
(6) Perform a RANSAC-based straight line fitting with the 2D points in the specified set $S$.
(7) If RANSAC is not able to find an appropriate straight line or the number of inliers is low, mark the current position as discarded and go to step 14.
(8) Obtain the line's Hessian normal form $(\boldsymbol{x}-\boldsymbol{p}_i)\cdot\boldsymbol{n}_0 = 0$ and push the current position $i$ on an empty stack.
(9) Pop the first element $j$ off the stack.
(10) If the counter $k$ has reached a predefined maximum and the number of points in $S$ is high enough, store the 2D normal vector information $\boldsymbol{n}_0$ at position $j$ and mark that position as processed.
(11) Check each position in an interval around $j$ that has not already been looked at whether the respective point lies sufficiently near to the straight line. If so, push its position on the stack. Additionally, include the 2D point in a new set $S'$.
(12) While the stack is not empty, go to step 9. Otherwise continue with step 13.
(13) If the counter $k$ has reached its maximum (e.g. two cycles), set it to zero and continue with step 14. Otherwise go to step 4 with the new set of points $S:=S'$.
(14) Go to step 1 until a certain number of iterations has been performed or no unmarked data is available in the scan line.

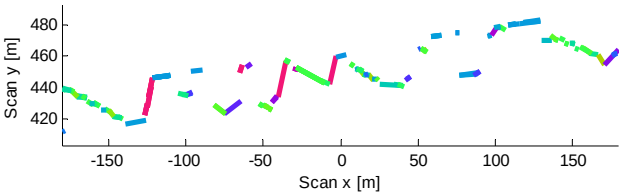



Figure 2. Detected straight line segments in a typical scan line, color-coded according to 2D normal direction.

In each iteration step we randomly select a position in the array $A$ of scan line data points and try to fit a straight line segment to the neighboring data at that position. The described RANSAC technique provides a robust estimation of the line segment's parameters, with automatic evaluation of the quality, e.g. by the number of outliers. If the fitted straight line is of poor quality, the data associated with the current position is assessed as clutter. Otherwise, we try to optimize the line fitting by looking for all data points that support the previously obtained line, which is done in steps (9), (10), (11) and (12). These steps actually represent a line growing algorithm. The local fitting of a straight line segment is repeated once with the supporting points to get a more accurate result. The end points of the resulting line segment can be found as the perpendicular feet of the two outermost inliers. These and the 2D normal direction are stored before the method is repeated until all points in the scan line are either assessed as clutter or part of a line segment. Figure 2 shows detected straight lines for one exemplary scan line, depicted with a suitable color-coding according to the normal direction. The median locations of all detected line segments are used as subset of the seed-points in the next scan line to speed up calculations, so the choice in step (1) is no longer "random" at this stage.

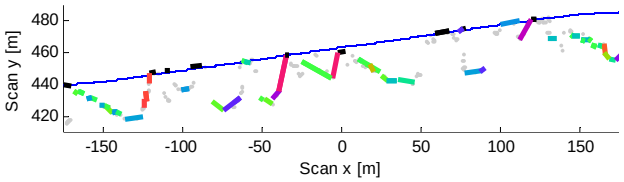


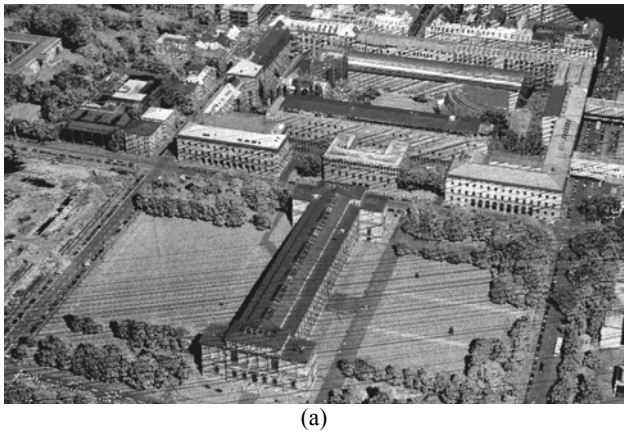

Figure 3. Automatically identified dividing line between buildings and ground level in a scan line.

### 3.3 Detection of the ground level

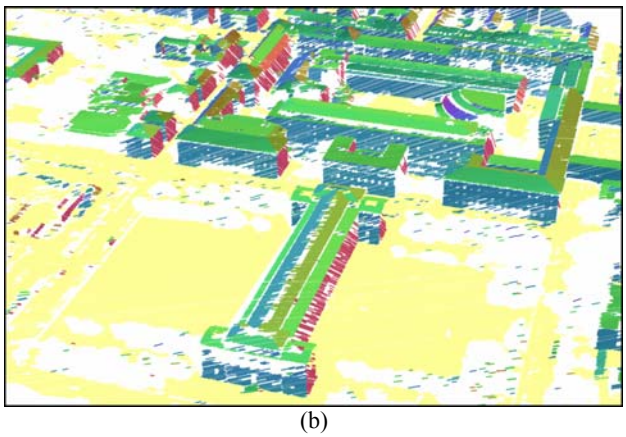
The next step is to identify all points in the scan line that form the ground level. A line segment at ground level can be characterized by the number of points lying beneath it in an appropriate neighborhood with respect to its 2D normal direction (that number should be near zero). In general, this is not a sufficient condition, but it yields enough candidates for an estimation of the dividing line between objects of interest and ground level. The dividing line is formed by estimates of the sensor-to-ground distance in y-direction at each position in the scan line. Each newly found line segment, potentially lying at ground level, contributes to that estimate in a neighborhood of its position with respect to its normal direction. Thus, this approach is permissive to unevenness of the terrain (Figure 3). Finally, all line segments lying completely below the dividing line are assessed as ground level, whereas line segments crossing or lying above are classified as part of a building. For increased robustness, the exponentially smoothed moving averages of the dividing line's parameters are perpetually transferred from the previously processed scan lines.

After most of the clutter and unwanted data points at ground level have been removed, the remaining line segments are likely to belong to planar parts of buildings, e.g. facades or rooftops. With our approach, only the end points of each detected line segment have to be georeferenced to result in correct positioned straight 3D lines. That reduces the amount of arithmetic operations, since only few points need to be converted with respect to the sensor orientation.

To give a comparison, Figure 4 (a) shows a rendered visualization of a point cloud of an urban area. Each range measurement has been georeferenced and every 3D point is depicted with its associated intensity resulting from off-line waveform analysis. The whole data set contains 1.300.000 points. Both full waveform analysis and converting of all points to the global 3D coordinate system are time-consuming. In contrast, Figure 4 (b) shows straight line segments after real-time capable scan line processing. Each line segment is depicted with a color-coding according to its 2D normal direction; the detected ground level is shown in yellow. Here the data set contains only 35.000 line segments classified as building and 15.000 line segments at ground level.



(a)



(b)

Figure 4. (a) Laser data of an urban area scanned in 45° oblique view, (b) same urban region after scan line analysis.

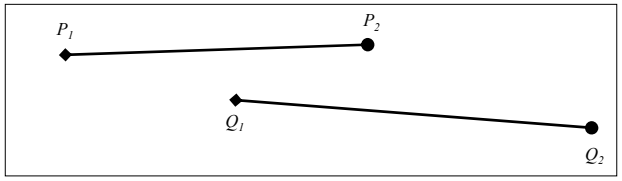



Figure 5. Two line segments within the same scan line.

### 3.4 Merging of line segments within a scan line

Detected straight line segments within a single scan line are often ambiguous and affected by gaps. To merge overlapping or adjacent pieces that are collinear, line-to-line distances have to be evaluated. Let $(P_1, P_2)$ and $(Q_1, Q_2)$ denote two different line segments detected within the same scan line (Figure 5).

$P_1$, $P_2$, $Q_1$, and $Q_2$ are the georeferenced end points in 3D space. Since every range measurement has a time-stamp, $P_1$ and $Q_1$ can be chosen to represent the first recorded end point in the respective line segment, $P_2$ and $Q_2$ are chosen accordingly. Two different distance measures are evaluated to decide whether the two line segments are to be merged or not. The first distance $d_1$ indicates if $(P_1, P_2)$ and $(Q_1, Q_2)$ are overlapping. In that case it would be zero; otherwise it is set to the minimum Euclidean distance between end points of the two line segments. With the abbreviations $\boldsymbol{v}_1 = \boldsymbol{p}_1 - \boldsymbol{q}_1$, $\boldsymbol{v}_2 = \boldsymbol{p}_1 - \boldsymbol{q}_2$, $\boldsymbol{v}_3 = \boldsymbol{p}_2 - \boldsymbol{q}_1$, and $\boldsymbol{v}_4 = \boldsymbol{p}_2 - \boldsymbol{q}_2$, distance $d_1$ is defined as

$$d_1 := \begin{cases} \min\left(\|\boldsymbol{v}_1\|, \|\boldsymbol{v}_2\|, \|\boldsymbol{v}_3\|, \|\boldsymbol{v}_4\|\right) & \text{if } \boldsymbol{v}_i^T \boldsymbol{v}_j \geq 0 \ \ \forall i < j \\ \boldsymbol{0} & \text{else} \end{cases} \tag{3.1}$$

With $\boldsymbol{p}_v = (\boldsymbol{p}_2 - \boldsymbol{p}_1)/\|\boldsymbol{p}_2 - \boldsymbol{p}_1\|$ and $\boldsymbol{q}_v = (\boldsymbol{q}_2 - \boldsymbol{q}_1)/\|\boldsymbol{q}_2 - \boldsymbol{q}_1\|$, the parameters of the perpendicular feet of each end point with respect to the other line segment are given as

$$s_1 = (\boldsymbol{q}_1 - \boldsymbol{p}_1)^T \boldsymbol{p}_v \qquad t_1 = (\boldsymbol{p}_1 - \boldsymbol{q}_1)^T \boldsymbol{q}_v$$
$$s_2 = (\boldsymbol{q}_2 - \boldsymbol{p}_1)^T \boldsymbol{p}_v \qquad t_2 = (\boldsymbol{p}_2 - \boldsymbol{q}_1)^T \boldsymbol{q}_v$$

The second distance $d_2$ is a measure of collinearity. It describes the sum of all minimal Euclidean distances of end points to the other line segment. Using the above parameters $s_1$, $s_2$, $t_1$ and $t_2$, distance $d_2$ can be expressed as

$$d_2 := \|\boldsymbol{p}_1 + s_1 \boldsymbol{p}_v - \boldsymbol{q}_1\| + \|\boldsymbol{p}_1 + s_2 \boldsymbol{p}_v - \boldsymbol{q}_2\| + \|\boldsymbol{q}_1 + t_1 \boldsymbol{q}_v - \boldsymbol{p}_1\| + \|\boldsymbol{q}_1 + t_2 \boldsymbol{q}_v - \boldsymbol{p}_2\| \tag{3.2}$$

Let $L$ denote the list of all detected line segments within the current scan line. The algorithm to find corresponding line segments works as follows:

(1) Initialize the current labeling number $m$ with 1.
(2) Select the next entry $a$ in $L$, starting with the first one.
(3) If $a$ is unlabeled, set its label to $m$ and increase $m$ by 1.
(4) Successively test each line segment $b$ in $L$ following after $a$ if $d_1(a,b)$ and $d_2(a,b)$ are smaller than predefined thresholds. If so, go to step (5), otherwise continue with testing until $b$ reaches the end of the list $L$. In that case, go to step (6)
(5) If $b$ is unlabeled, set its label to the label of $a$. Otherwise set the label of $a$ and $b$ to the minimum of both labels. Continue testing in (4).
(6) Continue with (2) until $a$ reaches the end of the list $L$.
(7) Repeat the procedure until labels do not change anymore.

Roughly spoken, the above procedure first initializes each line segment detected in the scan line with a unique label. Those collinear line segments that are found to overlap or lie adjacent are linked together by labeling them with their minimum labeling number. This process is repeated until the labels reach a stable state. The emerging clusters of line segments with same label are then represented by one single line segment, given by the two outermost end points of that cluster.

### 3.5 Merging of line segments over several scan lines

In principle, merging of 3D line segments over multiple scan lines is performed similar to the methods described in the previous section. In contrast to merging of line segments within the same scan line, we are now interested in coplanarity instead of collinearity. Thus other distance measures have to be evaluated. Let $P_i$, $P_j$, and $P_k$ be three of the four end points of two line segments. The distance of the fourth end point $P_m$ to the plane defined by the three others is a measure of coplanarity. We define distance $d_3$ as the sum of all four possible combinations:

$$d_3 := \sum_4 \frac{\left|(\boldsymbol{p}_i - \boldsymbol{p}_m)^T (\boldsymbol{p}_i - \boldsymbol{p}_j) \times (\boldsymbol{p}_i - \boldsymbol{p}_k)\right|}{\left\|(\boldsymbol{p}_i - \boldsymbol{p}_j) \times (\boldsymbol{p}_i - \boldsymbol{p}_k)\right\|} \qquad (3.3)$$

With the notations of section 3.4, $d_4$ is simply defined as the minimum Euclidean distance between respective first and last end points and the centers of two line segments

$$d_4 := \min\left(\|\boldsymbol{v}_1\|, \|\boldsymbol{v}_4\|, \tfrac{1}{2}\|\boldsymbol{v}_1 + \boldsymbol{v}_4\|\right) \qquad (3.4)$$

Another distance measure can be expressed by the angle between direction vectors $\boldsymbol{p}_v$ and $\boldsymbol{q}_v$

$$d_5 := \left|\arccos\left(\boldsymbol{p}_v^T \boldsymbol{q}_v\right)\right| \qquad (3.5)$$

Labeling of line segments over multiple scan lines works analogous to the labeling procedure within a single scan line. Nevertheless, there are some differences since scan lines are recorded successively. Moreover, simply testing of two line segments for coplanarity would allow the label to be handed over at edges of buildings. To avoid that, the 3D normal direction at each line segment has to be estimated.

- Let $n$ be the number of scan lines to be traced back (e.g. five). $L_0$ denotes the list of line segments in the current scan line, $L_1$ is one scan line behind, $L_2$ is two steps behind and so forth.
- First, test every line segment in $L_0$ if it is near to other line segments in $\{L_1, \ldots, L_n\}$ in terms of distance measures $d_3$, $d_4$, and $d_5$. If line segments $a$ and $b$ correspond this way, store this link in a database.
- After that, line segments in $L_n$ will receive no further links. The 3D normal direction at each line segment in $L_n$ is estimated by RANSAC based plane fitting to the set of associated other line segments. If that set contains too few points or the number of outliers is high, the respective line segment is of class *2*. Those line segments are typically isolated or near to the edge of a building. All other line segments in $L_n$ belong to class *1* and a 3D normal direction can be assigned to them.
- Initialize all line segments of $L_n$ with new labeling numbers. Test every line segment in $L_n$ if it is near to other line segments in $\{L_{n+1}, \ldots, L_{2n}\}$ in terms of distance measures $d_3$, $d_4$ and $d_5$ (these links are already established).
- If line segments $a$ and $b$ are linked, the following cases may occur:
  - $a$ and $b$ are of class *2*: do nothing
  - only one line segment is of class *1*: the class *2* line segment receives the label of the class *1* element
  - $a$ and $b$ are of class *1*: if the angle between the associated normal directions (calculated same as $d_5$) falls below a predefined threshold, set the label of $a$ and $b$ to the minimum of both labels
- Continue comparing $L_n$ to $\{L_{n+1}, \ldots, L_{2n}\}$ until labels reach a stable state.

Just for clarification: the labels that are assigned to the scan lines in section 3.4 are independent from those introduced here. The first $n$ scan lines are required for initialization before the algorithm starts working. At least $2n$ scan lines are needed before the merging of line segments begins. Line segments that form a connected (mostly planar) surface will successively be marked with the same label until no more fitting line segments are recorded (Figure 6).

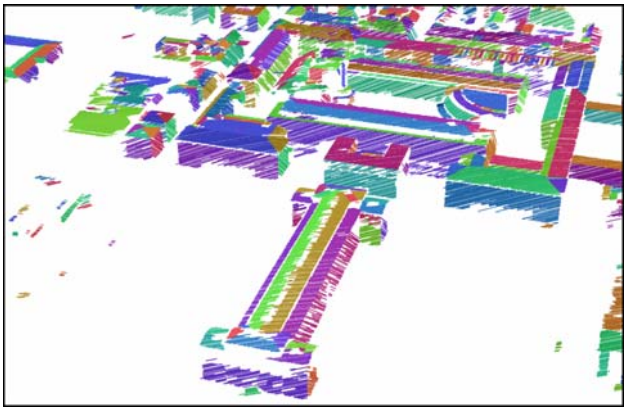


Figure 6. Result of scan line analysis and line segment grouping (color corresponds to label).

### 3.6 Delineation of connected line segments

For data storage and preparation of subsequent analysis, it is convenient to delineate each completed cluster of connected line segments by a polygon (closed traverse). The 3D normal direction $\boldsymbol{n}_c$ of a cluster $C$ can be estimated as the weighted average of normal directions of all line segments (weighted by line length). It is easy to determine an affine transformation $E$ that transforms $\boldsymbol{n}_c$ to the z-axis, such that the points of $E(C)$ roughly lie in the x-y plane. The boundary is derived by determination of a 2D alpha shape (Edelsbrunner et al., 1983) with an alpha corresponding to scan line distance.

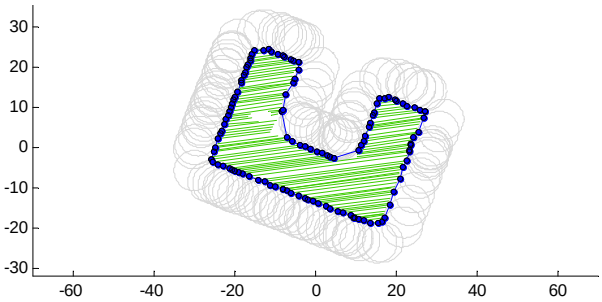



Figure 7. Exemplary alpha shape of a non-convex cluster.

The basic idea behind an alpha shape is to start with the convex hull. Then a circle of radius α is rolled around that convex hull. Anywhere the alpha-circle can penetrate far enough to touch an internal point (a line segment) without crossing over a point at the boundary, the hull is refined by including that interior point. Then the alpha shape is transformed back by application of $E^{-1}$. Finally, we have a set of 3D shapes representing grouped line

segments resulting from scan line analysis (Figure 7, Figure 8). Subsequent analysis depends on the problem at hand. For building reconstruction and model generation, the detected planar faces have to be intersected to find edges and corners. Examples for that procedure can be found in (Vosselman, 1999) or (Rottensteiner, 2005).

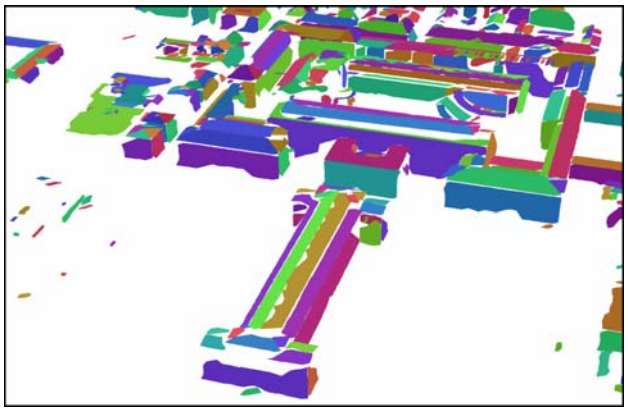


Figure 8. Shapes detected at an urban region.

## 4. DISCUSSION AND CONCLUSION

The preceding sections have described a workflow of operations to detect the ground level and distinguish between clutter and man-made objects in airborne laser scanner data of urban regions. Analysis of ALS data is done by scan line analysis instead of the usual TIN based approach. With the proposed methods, surfaces can be segmented "on-the-fly", thus enabling ALS to be used for applications that require on-line data processing.

The results presented in this paper were obtained with programs that were developed under MATLAB®. With that implementation and the data shown in Figure 4, scan line based filtering, line segment grouping and delineation of the detected surfaces take about 6 minutes on a standard PC, whereas the data recording took only 16 seconds. Nevertheless we feel confident that all computations can be accomplished in real-time if a more efficient implementation would be used. Computation time can get much shorter, since the proposed techniques have high potential for parallel processing. The inherent real-time capabilities of the used techniques are more crucial than computation time. Here it is undesirable to use an algorithm that requires the whole data set to be present before data processing can start. Our method for scan line grouping described in section 3.5 typically needs the data of only ten consecutive scan lines to be available in memory, which are recorded within 0.1 seconds. Scan line analysis is done by RANSAC iterations, combined with a line-growing algorithm. In principle, RANSAC is "any-time" capable, since its number of iterations can be adapted to the requirements. Speed is increased even more by using the locations of all detected line segments as subset of the seed-points in the next scan line. Our future work will focus on on-line change detection at urban environments. We will also use the proposed methods to provide an accurate co-registration of different (oblique) views of the same urban region.

## REFERENCES


Axelsson, P., 1999. Processing of laser scanner data – algorithms and applications. ISPRS Journal of Photogrammetry and Remote Sensing, 54, pp. 138-147.

Edelsbrunner, H., Kirkpatrick, D. G., Seidel, R., 1983. On the shape of a set of points in the plane. IEEE Transactions on Information Theory 29 (4), pp. 551-559.

Fischler, M.A., 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), pp. 381-395.

Gruen, A., Akca, D., 2005. Least squares 3D surface and curve matching. ISPRS Journal of Photogrammetry and Remote Sensing 59 (3), pp. 151-174.

Hatger, C., Brenner, C., 2003. Extraction of Road Geometry Parameters form Laser Scanning and existing Databases. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 34 (3/W13).

Jiang, X., Bunke, H., 1994. Fast Segmentation of Range Images into Planar Regions by Scan Line Grouping. Machine Vision and Applications 7 (2), pp. 115-122.

Jiang, X., Bunke, H., 1999. Edge detection in range images based on scan line approximation. Computer Vision and Image Understanding 73 (2), pp. 183-199.

Katsoulas, D., Werber, A., 2004. Edge Detection in Range Images of Piled Box-like Objects. Proceedings of the 17th International Conference on Pattern Recognition ICPR 2004 (2), pp. 80-84.

Rottensteiner, F., Trinder, J., Clode, S., Kubik, K., 2005. Automated Delineation of Roof Planes from LIDAR Data. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 36 (3/W19), pp. 221-226.

Sithole, G., Vosselman, G., 2003. Automatic Structure Detection in a Point-Cloud of an Urban Landscape. Proceedings of the 2nd Joint Workshop on Remote Sensing and Data Fusion over Urban Areas URBAN 2003, pp. 67-71.

Sithole, G., Vosselman, G., 2005. Filtering of airborne laser scanner data based on segmented point clouds. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 36 (3/W19), pp. 66-71.

Toth, C. K., Grejner-Brzezinska, D., Moafipoor S., 2004. Precise Vehicle Topology and Road Surface Modeling Derived from Airborne LiDAR Data, Proceedings of the ION 60th Annual Meeting.

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

Vosselman, G., Dijkman, S., 2001. 3D Building Model Reconstruction from Point Clouds and Ground Plans. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 34 (3/W4), pp. 37-44.

Vosselman, G., Gorte, B.G.H., Sithole, G., Rabbani, T., 2004. Recognising structure in laser scanner point clouds. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 46 (8), pp. 33–38.

Wehr, A., Lohr, U., 1999. Airborne Laser Scanning – an Introduction and Overview, ISPRS Journal of Photogrammetry and Remote Sensing 54, pp. 68-82.