


## Article

# High-Quality Contour Line Generation from LiDAR Point Clouds for the Area of Forests

Zongyue Wang <sup>1</sup> , Hongchao Ma <sup>2</sup>, Cuiling Zheng <sup>1,\*</sup> and Rongxin Chen <sup>1</sup>

<sup>1</sup> School of Computer Engineering, Jimei University, Xiamen 361021, China; wangzongyue@jmu.edu.cn, zhengcuiling@jmu.edu.cn, ch2002star@163.com

<sup>2</sup> School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China; hchma@whu.edu.cn

\* Correspondence: zhengcuiling@jmu.edu.cn; Tel.: +86-592-6181948

Version February 1, 2018 submitted to

**Abstract:** A methodology for both accurate and smooth contour line generation from Light Detection and Ranging (LiDAR) point clouds is proposed in this paper. In order to improve the accuracy of contour lines in the area of forests, constrained triangulation networks with break lines are then constructed to generate contour lines. In break line extraction, a bi-threshold method for edge line detection is used to extract both complete and reliable break lines. A point clouds elevation adjustment with constrain of break lines and an interpolator considering a contour interval is proposed to improve the smoothness of contour lines. The proposed interpolator is also can avoid contour line intersection when contour lines are interpolated. Statistical parameters and shape index are then used to evaluate quantitatively the accuracy and smoothness of the resultant contour lines, which fill in the blank of contour lines evaluation in theory. The experiments show that high-quality contours in terms of smoothness and accuracy can be generated from LiDAR point clouds.

**Keywords:** LiDAR; smooth contour line; break line; point cloud; forests

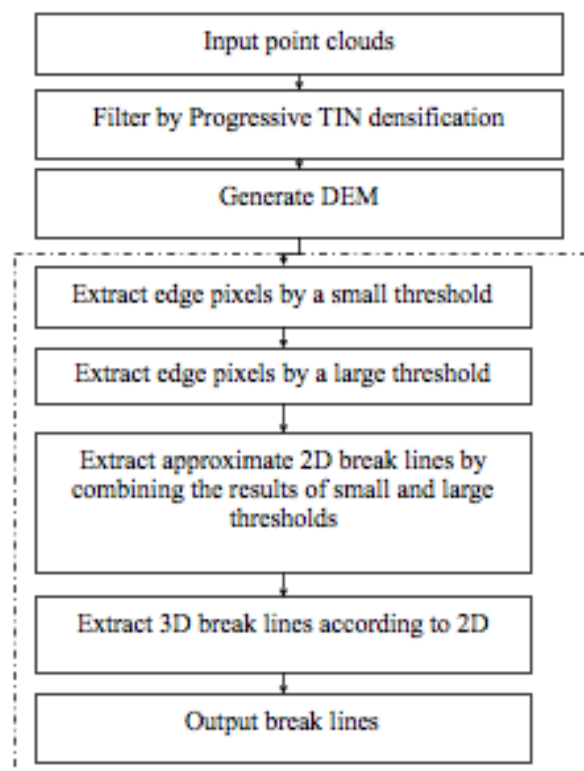
## 1. Introduction

Traditionally, contour lines can be generated by photogrammetric methods[1–3]. However, despite their common use, these methods are quite expensive and low efficiency. Furthermore, the accuracy of the contour lines they generate is not high enough in some areas such as densely vegetation covered forest, deserts, and coastal zones, among others. In the past decade, a new generation of active surveying technique, airborne light detection and ranging (LiDAR), has been utilized due to its high accuracy in 3D spatial data acquisition, high efficiency, and relatively lower cost. One of the primary use of the LiDAR system is for DEM acquisition with a reasonably high accuracy[4]. However, a critical step in contour line generation from LiDAR data is data filtering, which involves distinguishing ground from non-ground points. Almost all filtering algorithms only depend on changes in slope and elevation [5–8], because LiDAR data are recorded in terms of 3D geographic coordinate values rather than the spectral information of ground objects. Although most modern LiDAR systems record the intensity value of a returned signal, these values are un-calibrated. More importantly, distinguishing between ground and non-ground points only by intensity values is difficult [9,10]. Therefore, the ground points in the vicinity of break lines are usually classified as non-ground points. This is due to the sudden changes in elevation or slope over these areas[5,11,12], which leads to the distortion of the contour lines around them. In addition, if contour lines are extracted directly according to the point clouds that have been filtered, the resultant contour lines are prone to be saw toothed, which should be avoided because this destroys the smoothness of a contour line. Therefore, for a smoother contour line generation, the interpolation of the contour points extracted from the point clouds is necessary. This paper presents a new approach for smooth contour line generation from LiDAR point clouds with a reasonably high accuracy. The main idea behind the approach is the consideration of break lines as the constraints to overcome the distortions of the resulting contour lines. After the break lines are extracted,

they are used as the constrained vectors to establish constrained irregular triangular networks[13,14]. Meanwhile, point clouds are elevation-adjusted with consideration of the break lines. Smooth and high accuracy contour lines can then be generated by this method. The next section presents a method for extracting break lines. Then the method of contour line generation with the constraint of break lines, with which smooth contour lines can be generated, is proposed. The method to evaluate the quality of contour lines is proposed in Section 4. This is followed by experiments showing the different methods for break lines extraction and the resultant contour lines. The discussion and the conclusions are described in the final section.

## 2. Extraction of break lines

Break lines are usually located along the steep zones, so this makes them easily detected by commonly used edge detectors. However, traditional edge detectors usually adopt a single threshold to distinguish edge pixels from non-edge ones, such that the extracted edge lines are either broken into too many small fragments, or too many pseudo-edge lines are formed if the threshold value is not accurately determined. Considering that a high threshold value can extract more reliable edge lines, whereas a smaller threshold value can extract more complete ones, a bi-threshold value method for extracting both reliable and complete edge lines from point clouds is proposed. The complete workflow for the extraction of break lines is shown in Figure 1.



**Figure 1.** The process of extracting break lines.

As discussed in the previous section, break lines are always located in zones where the elevation value changes dramatically[15]. Therefore, borrowing edge detectors from the image processing field for break line extraction from the DEM data set is reasonable. In this paper, the LoG edge detector[16] as shown in Eq.1 is used to extract the approximate break lines. LoG is a type of second-order gradient edge detector which combines the Gaussian smooth filter and the Laplacian sharp filter into one filter

that smoothens the data set first and then detects the edge lines. It reduces the influence of noise for edge detection.

$$\log = \nabla^2 G(c, r) = (c^2 + r^2 - 2s^2)/s^4 \quad (1)$$

where  $c$  and  $r$  refer to the number of column and row, respectively, and  $s = \sigma_{Gaussian}$ . When LoG is performed for edge detection, the initial step is to obtain those pixels with zero Digital Number (DN) values after the grid DEM is convoluted by LoG, which is closely related to the threshold of gradient variance. A small threshold leads to many false but complete-edge lines, whereas a large threshold results in segmented and incomplete but less-false-edge lines. A bi-threshold strategy is adopted to combine the merits of both small and large threshold values, and the approximate edge lines are extracted by this strategy. The details are as following.

#### 2.1. A small threshold value is used to extract complete break lines

A second-order gradient image can be obtained after the LoG operator convolutes the grid DEM. Edge pixels correspond to those with zero DN values. Therefore, closed and connected schematic edges can be extracted by delineating pixels with zero DN values. However, large connected areas with zero DN values can be delineated in the gradient image if the point clouds are acquired over flat or very small-slope areas. This is because the gradient tends to be zero in these areas. Although complete edges can be extracted, many of these are pseudo ones because they are actually pixels where very small gradient changes occurred. A true break line should be an elongated edge. Therefore, the eight-neighboring connectivity algorithm is used to cut off those edge lines with small connected domains. In this manner, schematic break lines are successfully extracted.

#### 2.2. A large threshold value is applied to extract high-confident edges

Large variations in gradient occur along the vicinity of a true edge, so a large threshold value is applied to extract high-confident edges. However, small variations in gradient normally exist around some true edges. In this case, a large threshold value can lead to fragmented break lines.

#### 2.3. Combination of break lines by small and large threshold values

A small threshold value extracts more complete break lines, whereas a large threshold value detects more confident ones. Combining them results in the extraction of complete and confident break lines. Let  $l_b$  be the edge pixels detected by a large threshold and  $l_a$  be the edge pixels detected by a small one. They are superimposed. If the overlapping ratio in Eq.2 is greater than the predefined value, then  $l_a$  is identified as a break line.

$$ratio = \frac{l_a \cap l_b}{l_a} \quad (2)$$

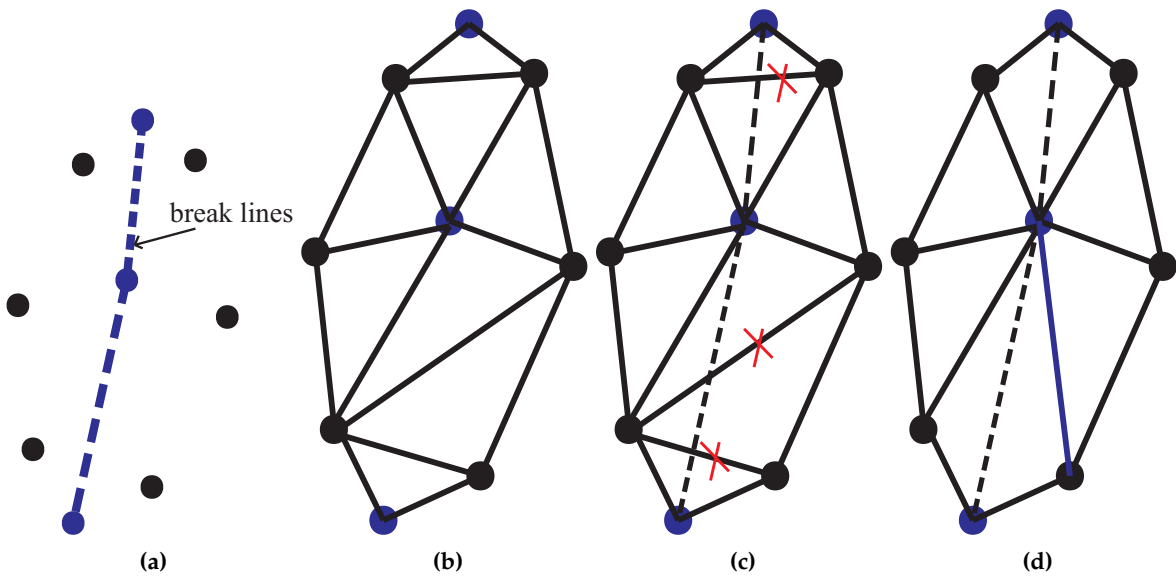
### 3. Contour lines generation with the constraints of break lines

To generate high accuracy and smoothness contour lines is always purchased by cartography community. Three strategies are considered in order to improve the accuracy and smoothness when contour lines are generated from LiDAR point clouds: (1) high accurate break lines are used as constraints for contour line generation; (2) point clouds are filtered with the constraint of break lines; and (3) contour lines are smoothed with the constraint of contour interval.

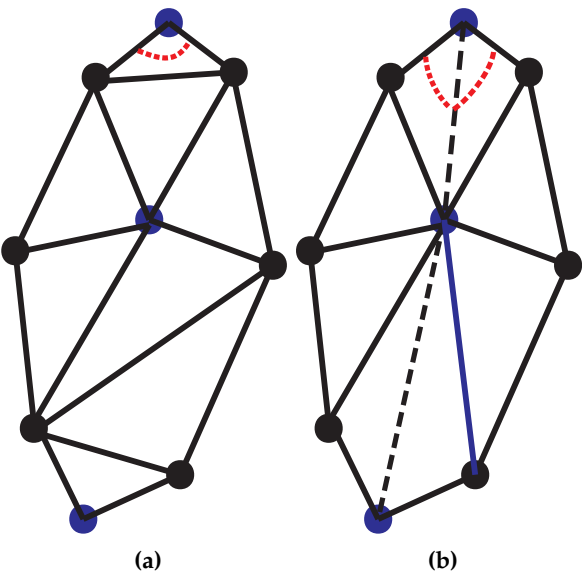
#### 3.1. Break lines constrained triangulation networks

In the paper, break lines are used as constrained lines to be interpolated in the triangulation networks constructed from filtered point clouds in order of expressing more micro-geomorphologic details. To do so, mainly two algorithms can be employed, that is, Sloan's[17] and Floriani's[18]. In Sloan's algorithm, constrained and unconstrained points are not distinguished. Constrained lines are inserted in the adjustment process followed. Floriani's algorithm is more flexible compared to Sloan's.

87 It firstly constructs triangulation networks only by unconstrained points, and then constrained lines are  
88 inserted into the initial networks. In Sloan's algorithm, a constrained line is inserted by interchanging  
89 diagonal lines continuously, which requires the quadrangle with interchangeable diagonal lines to be  
90 strictly convex. This requirement is hard to satisfy in reality. While the core of Floriani's algorithm  
91 is Delaunay triangulation on simple polygons, the implementation of the algorithm is much simpler  
92 and straightforward. Therefore, Floriani's algorithm is adopted in the paper. The scheme indicating  
93 Floriani's algorithm is shown in Figure 2, while Figure 3 shows a resultant contour line with break  
94 lines constrained.



**Figure 2.** The procedure of constructing triangulation networks constrained with break lines. (a) Ground points and structural lines, (b) Construct D-TIN with the key points of break lines, (c) Insert break lines, and (d) Construct D-TIN constrained with break lines.



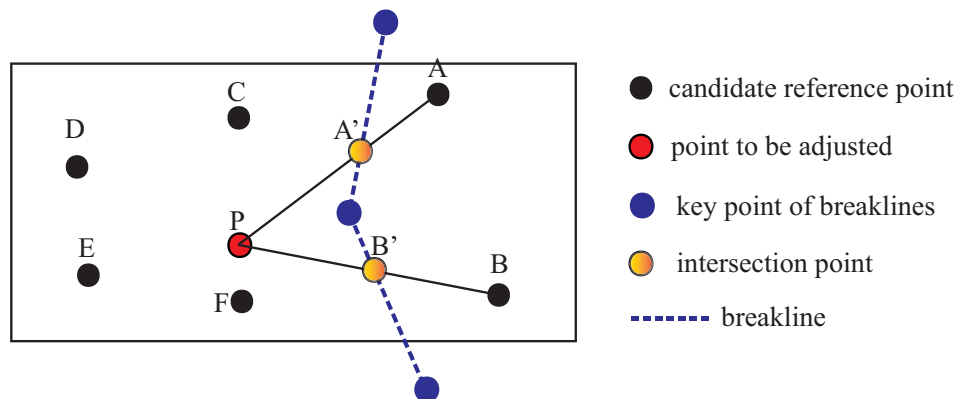
**Figure 3.** Resultant contour line without/with break lines constrained. (a) Without break lines constrained, and (b) With break lines constrained.

### 3.2. Point clouds elevation adjusting with the constraint of break lines

Since constrained triangulation networks constructed contains each break line(see Subsection 3.1), topographic structures can be preserved well. However, if such a triangulation networks is used for contour lines tracing directly, some contour lines would be fragmented or even saw-toothed. This is mainly caused by the following two factors: (a) the ground points obtained from filtering process contain noises, that is, non-ground points, which could generate fragmented contour lines. (b) though high density point clouds are necessary for expressing details of micro-geomorphology, it also causes saw-toothed effect of some of the result contour lines traced from such a dataset. A strategy to overcome these shortcomings is to adjust the elevation values of the ground point dataset in order to make it more suitable for fitting more smooth ground surfaces, while maintaining reasonably high accuracy. Especially in the vicinity of break lines, the result contour should be smoothly connected while preserving geomorphologic properties as much as possible. Bearing all these in mind, a break lines constrained point clouds adjusting method is proposed in the paper, whose details are described in the following.

Step 1: set accumulation error  $s = 0$ , for each point which is not located in a break lines, do the following:

suppose the current point to be adjusted is  $P$ (as shown in Figure 4). Search all the neighboring points  $A, B, C, D, E, F$  within a given range, and make them as reference points. If a break line exists between  $P$  and a reference point, then find the intersection point between the break lines and the line connecting  $P$ . If they are intersected, eg.  $PA$  intersect break lines on Point  $A'$ , then Point  $A$  do not set as reference points. That 's because  $A$  may be located below a cliff while  $P$  is located above the cliff of  $A$  above while  $P$  below. Therefore in this case, the reference point  $A$  can be involved in the process of adjustment. Moving surface fitting algorithm is used for adjustment in the paper.



**Figure 4.** Select reference points for adjustment. Point  $C, D, E, F$  can be set as reference point for processing adjustment, while  $A, B$  can not.

$$z = ax^2 + by^2 + cxy + dx + dy + f \quad (3)$$

Now suppose there are  $m$  reference points around  $P$ . Then for each point, we have the error equation:  $v_i = a\bar{x}_i^2 + b\bar{y}_i^2 + c\bar{x}_i\bar{y}_i + d\bar{x}_i + e\bar{y}_i + f - z_i$ , where  $\bar{x}_i = p_i(x) - P(x)$  and  $\bar{y}_i = p_i(y) - P(y)$ . For  $m$  points we have the error equations in matrix format shown in Eq.4.

$$V = MX - Z \quad (4)$$

where,  $V = \begin{bmatrix} v_1 & v_2 & \dots & v_m \end{bmatrix}$ ,  $Z = \begin{bmatrix} z_1 & z_2 & \dots & z_m \end{bmatrix}$ ,  $X = \begin{bmatrix} a & b & c & d & e & f \end{bmatrix}$ . If the  $i$ th point is an intersection one,  $w_i$  is 0.5, otherwise, 1.

After standard adjustment computation, coefficients vector can be expressed as following:

$$X = (M^T W M)^{-1} M^T P Y \quad (5)$$

120 where,  $W = \{w_i, i = 1, 2, \dots, m\}$ , and  $M = \begin{bmatrix} \bar{x}_1^2 & \bar{y}_1^2 & \bar{x}_1 \bar{y}_1 & \bar{x}_1 & \bar{y}_1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \bar{x}_m^2 & \bar{y}_m^2 & \bar{x}_m \bar{y}_m & \bar{x}_m & \bar{y}_m & 1 \end{bmatrix}$ .

121 from which the elevation value of P is calculated as Figure 4.

122 Calculate the deviation  $\Delta h$  between the estimated elevation and the elevation of check points.

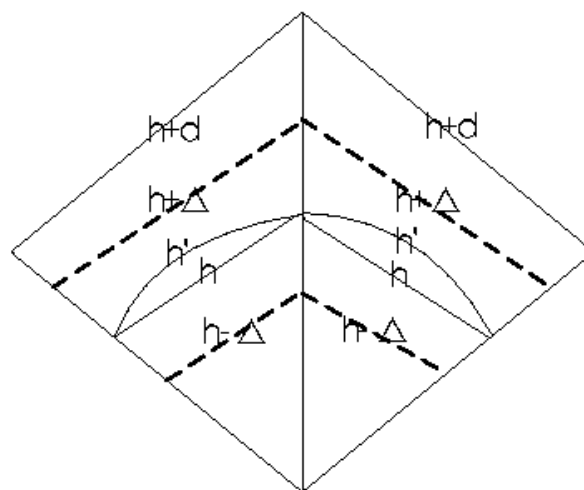
123 The new elevation of a check point is re-calculated by  $Z_p = (Z_p + Z'_p)/2$  and use the new value for  
124 checking if further check is required, record accumulation error  $s = s + \Delta h$ .

125 Step 2: If  $s < c$ , then the fitted ground surface is smooth enough, stop the iteration. Otherwise, go  
126 to Step 1 for the next iteration.  $c = n * r$ , where  $n$  is the number of points,  $r$  is a predefined constant. A  
127 smaller  $r$  results more smooth ground surface.

### 128 3.3. Contour lines smoothing with the constraint of contour interval

129 Contour points can be extracted from the point clouds dataset after it is elevation adjusted.  
130 However, interpolation between contour points is a must in order to generate smooth and continuous  
131 contour lines. Many interpolators have been used in literature, among which commonly used are:  
132 linear interpolation, piece-wise cubic polynomial interpolation[19], tension spline function, cubic B  
133 spline interpolation, etc. Though all these interpolators can smooth the result contour lines in some  
134 sense, none of them performs perfectly. For instance, self-intersection or inter-intersection of contour  
135 lines cannot be avoided completely. A novel tension spline function interpolator is proposed in order  
136 to guarantee smooth but no intersection contour lines, which the basic idea is that tension coefficients  
137 are adaptively setup in according to contour intervals.

138 Traditional method for determining whether neighbouring contour lines are intersected is  
139 calculation the topological relationship after all the contour points are connected with each other. Since  
140 there are large number of contour points in smoothed contour lines, this manner is computational  
141 expensive. We proposed a simple but effective method for the determination of whether two  
142 neighbouring contour lines are intersected. Combined with the coefficients adaptive tension spline  
143 interpolator, the following is the description of the details of the algorithm:



**Figure 5.** Contour lines smoothing with the constraint of contour interval.

144 As shown in Figure 5, let the elevation value of a given contour line be  $h$ ,  $d$  is the contour interval.

145 Let the initial coefficient of the tension spline corresponding to the contour be  $\delta$ . Calculate all the planar



coordinates of the points and estimate their elevation values  $H' = \{h'_1, h'_2, \dots, h'_n\}$  by linear interpolator through inserting them into the triangulation networks constructed for extracting contour points. The deviation between  $H'$  and  $h$  is denoted by  $\Delta H = \{\Delta h'_1, \Delta h'_2, \dots, \Delta h'_n\}$ , where  $\Delta h'_i = |h - h'_i|$ . In order to guarantee the contour not intersect with its neighbors, we first give a deviation threshold  $\Delta < d/2$ , and then check the deviation point by point. Any point with  $\Delta h_i > \Delta$  is viewed as the contour having the trend to intersect with its neighbors, since its curvature is too large. Such a contour line must be smoothed by using larger tension coefficient  $\sigma = \sigma + \Delta\sigma$  and repeat the procedure until all the points in the smoothed contour line satisfy  $\Delta h_i > \Delta$ , or if the contour lies in a too steep zone to satisfy for  $\Delta h_i > \Delta$ , then  $\sigma > \text{Max}\sigma$  and stop the iteration instead, where  $\text{Max}\sigma$  is a predefined constant. For each contour line, the workflow of the algorithm is described in the following.

Step1: set the initial tension coefficient  $\sigma = 1$ .

Step2: calculate all the planar coordinates of points in the smoothed contour line.

Step3: estimate the elevation values of the points by inserting them into the triangulation networks constructed by structural lines constrained filtering point clouds. if  $\Delta h_i > \Delta$  ( $\Delta h_i \in \Delta H$ ) and  $\sigma \leq \text{Max}\sigma$ , then increase  $\sigma = \sigma + \Delta\sigma$ , go to Step 2, otherwise, the current  $\sigma$  is used as the tension coefficient of the contour line.

#### 4. Evaluation of the quality of contour lines

Quality evaluation is an indispensable step in the generation of contour lines from the LiDAR data set. The traditional method for quality evaluation of contour lines is based on ground truth, a method similar to other spatial data quality assessment approaches which is labour consuming in terms of collecting ground truth. Furthermore, the method can assess the quality only by point-based truth. Collecting ground truth corresponding to each point of a contour line is impossible. In this paper, statistical and shape-index based models are proposed for quality evaluation. The former is used for accuracy assessment, whereas the latter is used for smoothness measurement.

##### 4.1. Accuracy assessment of contour lines by the statistical-based method

After contour lines are generated by the process described in this paper according to the given map scale, the following steps are proposed to evaluate the accuracy of the result:

(1) Each contour line with a 2 mm interval is sampled, in which the interval is the length measured in the map sheet. The real coordinates of the point being sampled should be converted from the interval and the given map scale. The sampled points is set as Point Set I.

(2) A triangulation network is constructed with Point Set I.

(3) The filtered LiDAR data set is input, which can be taken as ground check points, and the elevations of the check points are estimated by linear interpolation using the triangulation network constructed in Section 3.3 Step 2. The elevation deviations between the interpolated points and the check points are then calculated.

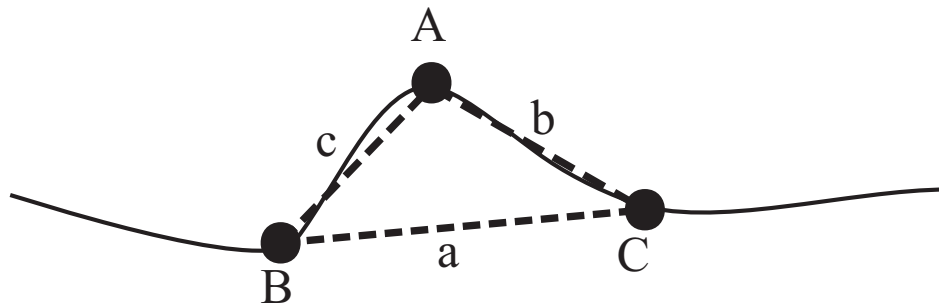
(4) The following statistical parameters are calculated: maximum, minimum, average error, and root mean square error (RMSE).

##### 4.2. Smoothness measurement of contour lines by the shape-index based method

The concept involved in smoothness measurement based on shape index is that the smoothness of a curve increases with its chord length, whereas it decreases with its curvature. Given contour points B, A and C (shown in Figure 6) and a triangle constructed by them, the larger the  $\angle A$  and the longer the chord length  $a$ , the smoother the curve. From a local point of view,  $\angle A$  alone is enough to measure local smoothness. However, chord length  $a$  effects the overall smoothness of the curve. From this point of view, the formula for smoothness measurement is proposed as Eq.6.

$$\eta_i = a(1 - \cos A) = a(1 - \frac{b^2 + c^2 - a^2}{2bc}) \quad (6)$$

when  $\angle A = 0^\circ$ ,  $\eta_i = 0$  which refers to the most unsmooth case, whereas  $\angle A = 180^\circ$ ,  $\eta_i = 2a$ . Such a change is in agreement with visual interpretation.



**Figure 6.** smoothness measurement of contour line by shape-index based method.

For a whole contour line which is segmented by contour points, the ratio of the summation of smoothness degree by each segment to the summation of each chord length is used to evaluate its smoothness measurement as Eq.7.

$$\eta = \frac{\sum_{i=1}^m \eta_i}{\sum_{i=1}^m a_i} \quad (7)$$

where  $m$  denotes the number of triangles constructed by the contour points of the line. If the contour line is a closed one,  $m$  is equal to the number of contour points, whereas  $m$  is equal to the number of contour points minus 2 if it is unclosed.

## 5. Experiments and analysis

The procedure for contour line generation from the LiDAR data set as described in this paper has been validated by a real data set collected by Leica ALS50, the second-generation LiDAR system. The data set covers Changyang area, Hubei Province, China, a hilly area with a relative relief of more than 1000 m. The main system parameters for data collection are as follows: minimum flying height of 1300 m above average ground, scanning angle of 49 degrees, and an overlapping rate of 30%. Spacing along the flight is around 1.3 m, whereas it is 1 m across the flight. The results are in 1.2 points per square meter point density of the final point clouds. The total data volume is around 2.3 gigabytes.

### 5.1. Experiment 1. Break lines extracted using the bi-threshold value method

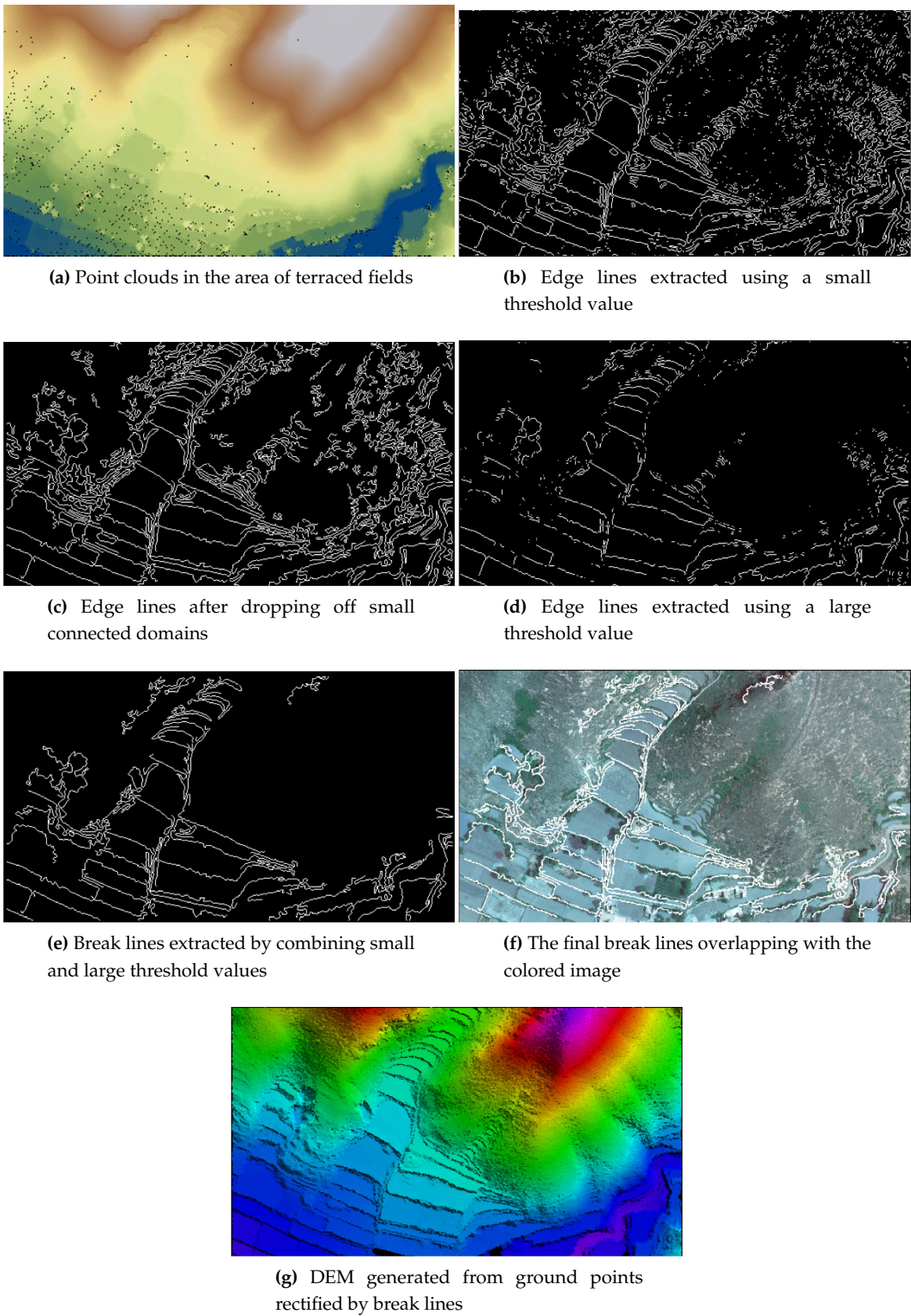
The experiment tests the efficiency of extracting break lines using the bi-threshold method. Point clouds are rendered according to their elevation values, as shown in Figure 7a. The area covers several terraced fields.

The procedure for this experiment is as follows.

- (1) Filter by progressive TIN densification and then generate grid DEM by linear interpolation.
- (2) Convolute the grid DEM by LoG operator, with the size of the convolution set to  $25 \times 25$  and the Gaussian variance set to 2.5.
- (3) Extract edge lines using a small threshold value (5% of the average gradient). The result is shown in Figure 7b. Then those edge lines with small connected domains are cut off, as shown in Figure 7c.
- (4) Extract edge lines using a large threshold value (75% of the average gradient). The result is shown in Figure 7d.
- (5) Extract schematic break lines by combining small and large threshold values. In the experiment, the overlap ratio(see Eq. 2) is 0.4, and the schematic break lines are shown in Figure 7e.
- (6) Perform 3D break line extraction using the piece-wise construction algorithm. The final break lines overlapping with the colored image are shown in Figure 7f.



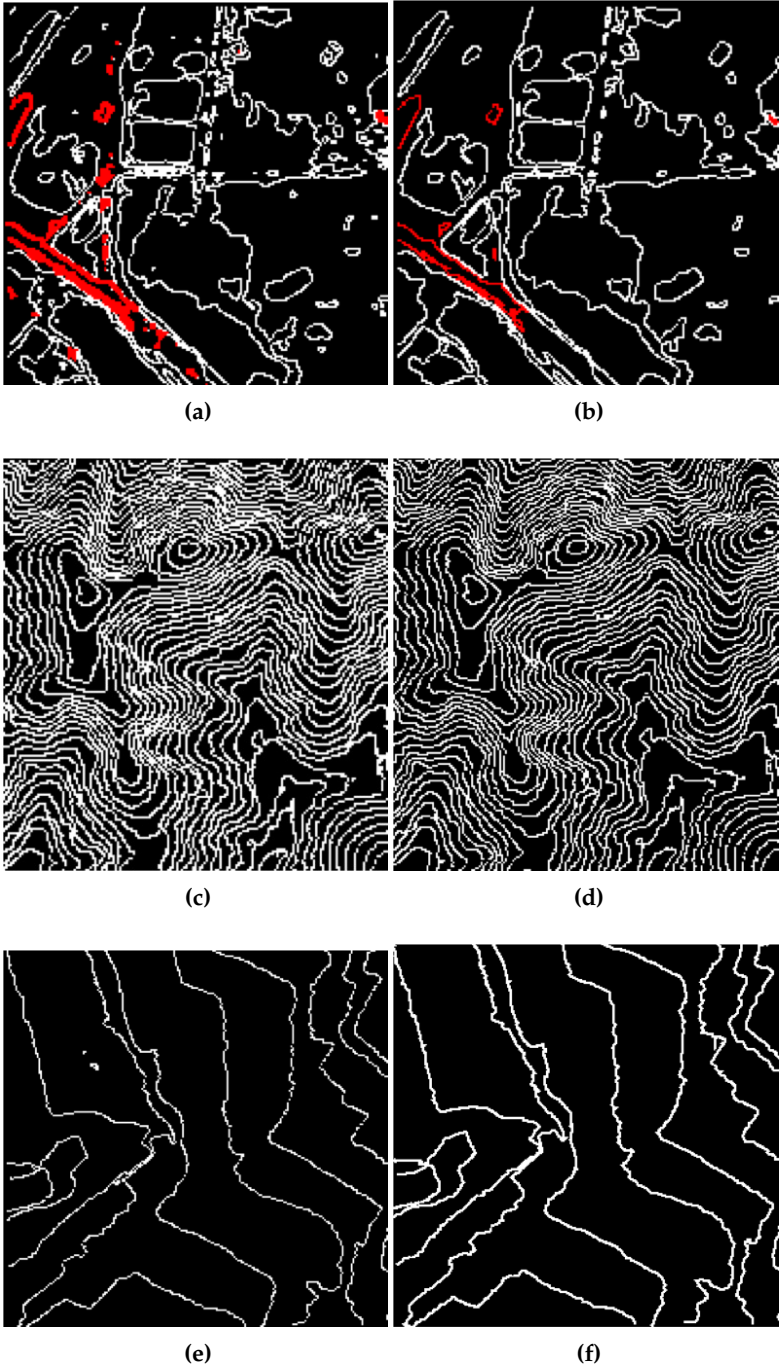
214 (7) When performing 3D break line extraction using the piece-wise construction algorithm, the  
215 LiDAR points near break lines are reclassified. Therefore, more accurate ground points are obtained.  
216 The DEM generated from these ground points is shown in Figure 7g.



**Figure 7.** The procedure of break lines extraction by bi-threshold method.

5.2. Experiment 2. Contour accuracy effected by different topographic conditions and with/without constraint of break lines

Data sets from three types of topographic conditions, namely, flat area, hilly area, and terrace field, are selected for generating contour lines at a scale of 1:500. The size of each area is about 0.5 km x 0.5 km. The resultant contour lines with and without the adjustment of point clouds are displayed in Figure 8. The smoothing parameter  $r$ , as discussed in Subsection 3.2, is 0.15. Contour lines are smoothed with the constraint of contour interval, so none of the contour lines intersect.



**Figure 8.** Contour lines generated from adjustment and without adjustment of point clouds: (a, b) in a flat area , (c,d) in a hilly area, and (e,f) in a terrace field; (a, c, e) Point clouds without elevation adjustment, and (b,d,f) Point cloud elevation adjustment with the constraint of structural lines.

Each contour line with a 2 mm interval is sampled, in which the interval is the length measured in the map sheet. The points form Point Set I. A triangulation network is then constructed with Point Set I. The check points are input, and the elevations of the check points are estimated by linear interpolation from the triangulation network. The elevation deviations are calculated between the interpolated points and the check points. Then statistical parameters such as maximum error (ME), average error (AE), and root mean square error (RMSE) are computed, as shown in Table 1.

Table 1. Accuracy of the contour lines.

Types of topographic condition	No elevation adjustment			Elevation adjustment without the constraint of structural lines			Elevation adjustment with the constraint of structural lines		
	AE	ME	RMSE	AE	ME	RMSE	AE	ME	RMSE
Flat area	0.45	0.12	0.13	0.52	0.12	0.17	0.45	0.12	0.14
Hilly area	0.36	0.09	0.15	0.42	0.09	0.21	0.40	0.09	0.15
Terrace field	0.47	0.10	0.18	0.56	0.11	0.24	0.41	0.11	0.18

Table 1 shows that the average error in flat areas is the largest in the group. This is because when the scale is set to 1:500, the interval of the half-interval contour is 0.5 m, so the contour lines are quite sparse. If the elevation of numerous check points is far from that of the contour lines, then a larger average error exists.

Notably, the RMSEs of the contour lines generated from the points after elevation adjustment with the constraint of structural lines are smaller than those without constraint, especially in areas with dramatic topographic changes such as terrace fields.

Table 2. Smoothness of contour lines in different processes (Scale is 1:500).

Types of topographic condition	No elevation adjustment		Elevation adjustment without the constraint of structural lines		Elevation adjustment with the constraint of structural lines	
	Num of contour lines	Smoothness of contour lines	Num of contour lines	Smoothness of contour lines	Num of contour lines	Smoothness of contour lines
Flat area	1154	1.34	157	1.86	275	1.78
Hilly area	1775	1.52	941	1.90	1156	1.86
Terrace field	949	1.30	282	1.87	348	1.83

In Table 2, the number of contour lines generated from points with elevation adjustment is much less than that without any adjustment. This is primarily due to the decrease in the number of fragment contours. Smoothness with elevation adjustment is improved evidently compared with smoothness without elevation adjustment, whereas it is similar when adjustment is carried out with or without the constraints of structural lines.

5.3. Experiment 3. Quality of contour lines in different scales

The data set in this experiment is the same as that used in Experiment 3, but contour lines are generated in different map scales. The RMSEs in different scales are shown in Table 3.

Table 3. RMSEs of contour lines in different scales.

Types of topographic condition	RMSE of 1:500			RMSE of 1:1000			RMSE of 1:2000		
	Method I	Method II	Method III	Method I	Method II	Method III	Method I	Method II	Method III
Flat area	0.13	0.17	0.14	0.15	0.18	0.15	0.16	0.18	0.17
Hilly area	0.15	0.21	0.15	0.17	0.23	0.17	0.22	0.23	0.22
Terrace field	0.18	0.24	0.18	0.18	0.24	0.19	0.24	0.25	0.24

Table 3, Method I: Contour lines are generated from points without elevation adjustment. Method II: Contour lines are generated from points with elevation adjustment but without the constraint of structural lines. Method III: Contour lines are generated from points with elevation adjustment and with the constraint of structural lines.

Table 4. Smoothness in different scale modes.

Types of topographic condition	Average smoothness of 1:500		Average smoothness of 1:1000		Average smoothness of 1:2000	
	no adjustment	adjusted with constraints	no adjustment	adjusted with constraints	no adjustment	adjusted with constraints
Flat area	1.34	1.78	1.58	1.81	1.79	1.88
Hilly area	1.52	1.86	1.82	1.90	1.90	1.92
Terrace field	1.30	1.83	1.68	1.86	1.85	1.89

From Table 4, smoothness evidently increases with a corresponding decrease in the map scale. In a small map scale scenario, the difference in smoothness between with adjustment and without adjustment becomes negligible.

5.4. Experiment 4. Relationship of accuracy and smoothness

In areas with dramatic topographic changes, the accuracy decreases with a corresponding increase in smoothness if the contour lines are generated for a large map scale. An experiment is then designed to test the relationship of accuracy and smoothness. A terrace field is selected to represent an area with a sharp topographic change. The smooth parameter  $c$  is tuned to generate different contour lines with different smoothness and accuracy values, where  $c = n \times r$  (see details in Section 3.2). Contours with different smoothness values are generated by adjusting parameter  $r$ , and then the RMSE and average smoothness values are calculated. The results are shown in Table 5. The variation in the trend of RMSE and average smoothness (AS) is shown in Figure .

Table 5. Accuracy and smoothness in different processes.

Tolerance error(r)	Elevation adjustment without the constraint of structural lines		Elevation adjustment with the constraint of structural lines	
	RMSE	AS	RMSE	AS
0.08	0.251	1.92	0.203	1.92
0.10	0.248	1.91	0.191	1.90
0.12	0.243	1.87	0.185	1.83
0.15	0.239	1.79	0.181	1.75
0.18	0.198	1.69	0.178	1.66
0.20	0.181	1.58	0.176	1.53
0.25	0.175	1.47	0.175	1.41



Table 5 shows that a larger  $r$  signifies little adjustment in the point clouds. Although the resultant contours can better approximate the real topography in this case, the smoothness of the contour becomes low.

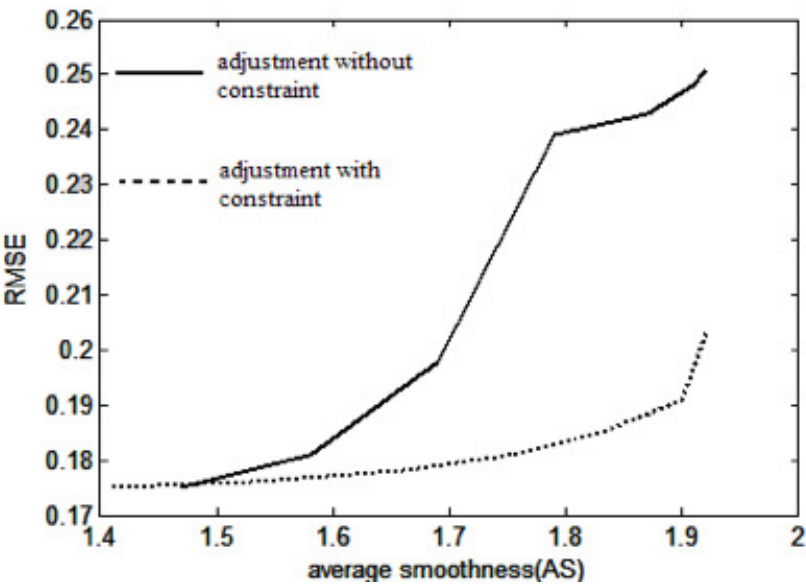


Figure 9. Relationship of accuracy and smoothness.

Figure 9 shows the relationship of smoothness and accuracy of the contour lines. Evidently, RMSE increases with a corresponding increase in smoothness. Further, by comparing the solid line with the dotted line, the contours from cloud points adjusted with the constraint of structural lines can maintain more topographic features compared with those from cloud points adjusted without the constraint of structural lines.

6. Discussion and conclusion

In the paper, a methodology for both accurate and smooth contour line generation from the LiDAR data set is described. First, highly accurate break lines are extracted. Second, filtered point clouds are then adjusted by the constraints of these break lines. Finally, constrained triangulation networks are constructed. In break line extraction, a bi-threshold method for edge line detection is used to extract both complete and reliable break lines. In contour line interpolation, an interpolator, with consideration of the contour interval, is proposed to overcome contour line intersection. Finally, statistical parameters and shape index are adopted to evaluate quantitatively the accuracy and smoothness of the resultant contour lines. The experiments show that in areas with serious topographic changes, the contour lines generated from LiDAR points after adjustment with the constraints of structural lines are more accurate than those from LiDAR points after adjustment without the constraints of structural lines.

In our proposed break lines extraction strategy, if the size of a grid mesh is too small, i.e., if the resolution of the grid DEM is too high, then more time is consumed for the extraction of break lines than low resolution. This is mainly due to a piece-wise construction algorithm is used to refine break lines extraction. This is an iterative process and is thus a time-consuming task, especially when a small step is adopted for the iteration. To overcome this shortcoming, the following are proposed in future research undertakings: (1) identification of the empirical optimal parameters of the size of the grid mesh and iteration step in terms of the efficiency of the program, and (2) further development of the programming method from the traditional single core CPU to the multicore CPU environment, or the use of another parallel programming environment such as cluster or grid computing to improve the running efficiency of the program.



**Acknowledgments:** This work is partially supported by the National Key R&D Program of China under Grand No. 2016YFC0502902, the National Natural Science Foundation of China under Grant No. 41201462, the Key Technical Project of Fujian Province under Grant No. 2014H0034 and 2017H6015, the Natural Science Foundation of Fujian Province under Grant No. 2016J01310 and 2016J01309, the Scientific Research Funding of Educational Department of Fujian Prov. China (JA13168, JA13182) and the Foundation for Young Professors of Jimei University, China.

**Author Contributions:** Zongyue Wang and Cuiling Zheng conceived and designed the experiments; Rongxin Chen performed the experiments; Hongchao Ma and Rongxin Chen analyzed the data; Zongyue Wang and Hongchao Ma wrote the paper.

**Conflicts of Interest:** The authors declare no conflict of interest.

References

1. Khotanzad, A.; Zink, E. Contour Line and Geographic Feature Extraction from USGS Color Topographical Paper Maps. *Pattern Analysis & Machine Intelligence IEEE Transactions on* **2003**, *25*, 18–31.
2. Ai, T. The drainage network extraction from contour lines for contour line generalization. *Isprs Journal of Photogrammetry & Remote Sensing* **2007**, *62*, 93–103.
3. Song, J.; Wang, P.; Miao, Q.; Liu, R.; Huang, B. The Reconnection of Contour Lines from Scanned Color Images of Topographical Maps Based on GPU Implementation. *IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing* **2017**, *PP*, 1–9.
4. Qian, L.I.; Xiao, C.; Chen, J.; Yang, D. Method for constructing DSM based on building contour line and airborne LiDAR data. *Remote Sensing for Land & Resources* **2013**, *25*, 95–100.
5. Shao, Y.C.; Chen, L.C. Automated Searching of Ground Points from Airborne Lidar Data Using a Climbing and Sliding Method. *Photogrammetric Engineering & Remote Sensing* **2008**, *74*, 625–635.
6. Vége, C.; Durrieu, S.; Morel, J.; Allouis, T. A sequential iterative dual-filter for Lidar terrain modeling optimized for complex forested environments. *Computers & Geosciences* **2012**, *44*, 31–41.
7. Zhang, K.; Chen, S.C.; Whitman, D.; Shyu, M.L. A progressive morphological filter for removing nonground measurements from airborne LIDAR data. *IEEE Transactions on Geoscience & Remote Sensing* **2003**, *41*, 872–882.
8. Wang, X.; Glennie, C.; Pan, Z. An Adaptive Ellipsoid Searching Filter for Airborne Single-Photon Lidar. *IEEE Geoscience & Remote Sensing Letters* **2017**, *14*, 1258–1262.
9. Brzank, A.; Heipke, C.; Goepfert, J.; Soergel, U. Aspects of generating precise digital terrain models in the Wadden Sea from lidar–water classification and structure line extraction. *Isprs Journal of Photogrammetry & Remote Sensing* **2008**, *63*, 510–528.
10. Höfle, B.; Pfeifer, N. Correction of laser scanning intensity data: Data and model-driven approaches. *Isprs Journal of Photogrammetry & Remote Sensing* **2007**, *62*, 415–433.
11. Chen, Q.; Gong, P.; Baldocchi, D.; Xie, G. Filtering Airborne Laser Scanning Data with Morphological Methods. *Photogrammetric Engineering & Remote Sensing* **2007**, *73*, 175–185.
12. Sithole, G.; Vosselman, G. Experimental comparison of filter algorithms for bare-Earth extraction from airborne laser scanning point clouds. *Isprs Journal of Photogrammetry & Remote Sensing* **2004**, *59*, 85–101.
13. Deng, S.G.; Liu, G. Study of algorithm for Delaunay triangular irregular network of constrained data field with reverse fault. *Science of Surveying & Mapping* **2006**, *31*, 98–99.
14. Zheng, J.; Zhang, T.; He, H.; Zhu, J. Embedding of a constrained line into a triangulated irregular network. *Journal of Tsinghua University* **2014**, *54*, 1555–1559.
15. Abdullah, Q.A.; Abdullah, Q.A. Mapping Matters—Breaklines for Lidar Data, Do We Really Need Them. *Photogrammetric Engineering & Remote Sensing* **2017**, *83*, 599–602.
16. Rashmi.; Kumar, M.; Saxena, R. Algorithm and Technique on Various Edge Detection : A Survey. *Signal & Image Processing* **2013**, *04*, 65–75.
17. Sloan, S.W. A fast algorithm for constructing Delaunay triangulations in the plane. *Advances in Engineering Software* **1987**, *9*, 34–55.
18. Floriania, L.D.; Puppob, E. An on-line algorithm for constrained Delaunay triangulation. *Cvgip Graphical Models & Image Processing* **1992**, *54*, 290–300.
19. Zheng, F.J. Contour line smooth combined with segmental cubic polynomials with Akima. *Engineering of Surveying & Mapping* **2012**, *21*, 9–12.