

Article

## Comprehensive Utilization of Temporal and Spatial Domain Outlier Detection Methods for Mobile Terrestrial LiDAR Data

Michael Leslar <sup>1,2,\*</sup>, Jian-guo Wang <sup>1</sup> and Baoxin Hu <sup>1</sup>

<sup>1</sup> Department of Earth and Space Science, York University, 4700 Keele Street, Toronto, ON M3J 1P3, Canada; E-Mails: jgwang@yorku.ca (J.W.); baoxin@yorku.ca (B.H.)

<sup>2</sup> Industrial and 3D Imaging Department, Optech Incorporated, 300 Interchange Way, Vaughan, ON L4K 5Z8, Canada

\* Author to whom correspondence should be addressed; E-Mail: mikel@optech.ca; Tel.: +1-905-660-0808.

Received: 25 June 2011; in revised form: 29 July 2011 / Accepted: 8 August 2011 /

Published: 16 August 2011

---

**Abstract:** Terrestrial LiDAR provides many disciplines with an effective and efficient means of producing realistic three-dimensional models of real world objects. With the advent of mobile terrestrial LiDAR, this ability has been expanded to include the rapid collection of three-dimensional models of large urban scenes. For all its usefulness, it does have drawbacks. One of the major problems faced by the LiDAR industry today is the automatic removal of outlying data points from LiDAR point clouds. This paper discusses the development and combined implementation of two methods of performing outlier detection in georeferenced point clouds. These methods made use of the raw data available from most time-of-flight mobile terrestrial LiDAR scanners in both the temporal and spatial domains. The first method involved a moving fixed interval smoother derived from the well-known position velocity acceleration Kalman Filter. The second method fitted a quadratic curved surface to sections of LiDAR data. The combined use of these routines is discussed through examples with real LiDAR data.

**Keywords:** outlier detection; mobile terrestrial LiDAR; curved surface fitting; Kalman Filter

---

## 1. Introduction

LiDAR (Light Detection and Ranging) is a tool, which allows for the fast and efficient capture of three-dimensional spatial information from real world targets. This ability has allowed both terrestrial based and airborne LiDAR to be used in a variety of applications [1,2]. Until recently, terrestrial based time of flight LiDAR has been relegated to stationary tripod mounts with comparatively low scanning speeds (10,000 points per second) when compared to airborne LiDAR systems (300,000 points per second). With the advent of mobile terrestrial LiDAR, this is no longer the case. Terrestrial based scenes can now be collected faster than ever, firstly because they are being collected from a moving platform and secondly, because collection speed has greatly increased (500,000 points per second). This increase in the number of terrestrial based data points collected during a survey means that greater and greater amounts of data are being produced faster. To complicate matters, due to the fact that the scanners are now immersed in the scene being scanned instead of flying high above it, the geometry contained in these massive data files is more complex than those encountered previously. This makes filtering of the data harder than previously encountered but even more necessary. Specifically, detecting and eliminating erroneously collected points or outliers becomes critical.

Simply stated, an outlier is a point which differs from its neighbors or neighborhood significantly [3]. The determination of what the term significantly means is, of course, up to the individual user of the data. Outliers in LiDAR data occur due to a variety of reasons. Some of these reasons such as boundaries of occlusion, surface reflectance and multi-path reflection are described in [4]. To this list can be added moving objects which pass through the scan area faster than they can be captured and particulate matter such as snow, rain, dust, *etc.*, in the air, which reflect the laser energy.

Several strategies exist for dealing with these outliers [4-9]. They can be classified as univariate (single variable), multivariate (multiple variable), parametric (statistic based) and non-parametric (non-statistic based) methods [5]. The non-parametric methods can further be broken down into distance-based, depth-based, density-based and clustering techniques [5,7,8]. Despite this large amount of research into outlier detection, correctly finding outliers in spatial data remains a vexing problem. Most of the strategies listed above fail when confronted with obstacles such as surface discontinuities, poor statistical distributions and varying local densities within the LiDAR point cloud [4]. Improving outlier detection for LiDAR requires the development of algorithms which make use of as much of the data, inherently output in LiDAR point clouds as is practical. This includes using the precise timings available from time-of-flight LiDAR systems, raw polar coordinate observations, calculated Cartesian coordinates and the intensity data if possible. Once developed, these algorithms employing different detection strategies can be combined to improve overall detection results.

In this study two different algorithms, one in the temporal domain (Moving Fixed Interval Smoother) and the other in the spatial domain (Curved Surface Fitting), were developed and utilized in a novel way for comprehensive outlier detection in real mobile terrestrial LiDAR data. The algorithm in the temporal domain detects outliers by testing the difference between the computed coordinates of a point and a prediction of these values based on the surrounding data. The predicted values were obtained through a modified version of the moving fixed interval smoother derived from the position, velocity and acceleration (PVA) version of the Kalman Filter [10]. Utilization of the precise timings available from LiDAR either mobile or static time-of-flight in a PVA Kalman Filter for outlier

detection is a new concept to literature. The second algorithm uses a best-fit quadratic curved-surface to spatially measure each point in the cloud and compare it to the points in its neighborhood. Their performance was numerically studied both individually and in tandem. While quadratic curved surface fitting is not new in literature [11-15], the sequential use of the spatial domain algorithm after the temporal domain algorithm is a new idea and will provide for an overall better result.

## 2. Moving Fixed Interval Smoother (MFIS)

### 2.1. The MFIS Algorithm

Given a discrete time series  $z(1), z(2), \dots, z(n)$  of a continuous time signal with their standard deviations  $\sigma_1, \sigma_2, \dots, \sigma_n$  and the associated precise timings  $t_1, t_2, \dots, t_n$ , a second order polynomial of  $z$  can be used to model the time series. By using a random variable  $x(k)$  to denote the state of  $z(k)$  at time instant  $t_k$ , the second order polynomial was given as follows:

$$x(k) = x(k-1) + \dot{x}(k-1) \cdot (t_k - t_{k-1}) + \frac{1}{2} \ddot{x}(k-1) \cdot (t_k - t_{k-1})^2 \quad (1)$$

$$\dot{x}(k) = \dot{x}(k-1) + \ddot{x}(k-1) \cdot (t_k - t_{k-1}) \quad (2)$$

$$\ddot{x}(k) = \ddot{x}(k-1) \quad (3)$$

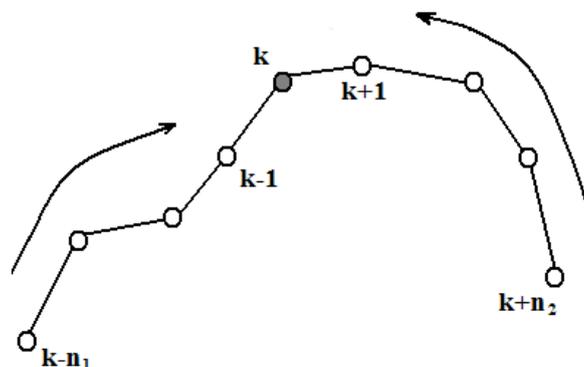
wherein the process noise is not included, and the measurement equation was:

$$z(k) = x(k) + \Delta(k) \quad (4)$$

where  $\Delta(k)$  was the white noise with zero expectation and a variance of  $\sigma_k^2$ . The filter algorithm based on Equation (1) is called the PVA filter in position tracking applications, and also called the  $\alpha$ - $\beta$ - $\gamma$  filter if it is time invariant [16].

The proposed moving fixed-interval smoother was developed on the basis of Equations (1–4) that estimate the states for a time instant  $k$  using the measurements over a specified window  $(t_{k-n_1}, t_{k+n_2})$  (Figure 1).

**Figure 1.** The Fixed Interval Smoother.



With the measurements  $z(k-n_1), \dots, z(k-1), z(k), \dots, z(k+n_2)$ , the unbiased linear smoother was derived based on the principle of minimal variance. The smoothed solution for the states at  $k$  was given by [10].

$$\hat{x}(k) = \sum_{i=-n_1}^{n_2} a_i(k) \cdot z(k+i) \tag{5}$$

$$\hat{\hat{x}}(k) = \sum_{i=-n_1}^{n_2} b_i(k) \cdot z(k+i) \tag{6}$$

$$\hat{\hat{\hat{x}}}(k) = \sum_{i=-n_1}^{n_2} c_i(k) \cdot z(k+i) \tag{7}$$

with their variances

$$\sigma_x^2(k) = \sum_{i=-n_1}^{n_2} a_i^2(k) \cdot \sigma_{k+i}^2 \tag{8}$$

$$\sigma_{\hat{x}}^2(k) = \sum_{i=-n_1}^{n_2} b_i^2(k) \cdot \sigma_{k+i}^2 \tag{9}$$

$$\sigma_{\hat{\hat{x}}}^2(k) = \sum_{i=-n_1}^{n_2} c_i^2(k) \cdot \sigma_{k+i}^2 \tag{10}$$

where

$$a_i(k) = \frac{1}{\sigma_{k+i}^2} \left\{ \lambda_1(k) + \lambda_2(k) \cdot \delta_{k,k+i} + \lambda_3(k) \cdot \delta_{k,k+i}^2 \right\} \tag{11}$$

$$b_i(k) = \frac{1}{\sigma_{k+i}^2} \left\{ \mu_1(k) + \mu_2(k) \cdot \delta_{k,k+i} + \mu_3(k) \cdot \delta_{k,k+i}^2 \right\} \tag{12}$$

$$c_i(k) = \frac{1}{\sigma_{k+i}^2} \left\{ \eta_1(k) + \eta_2(k) \cdot \delta_{k,k+i} + \eta_3(k) \cdot \delta_{k,k+i}^2 \right\} \tag{13}$$

$$\begin{pmatrix} \lambda_1(k) & \mu_1(k) & \eta_1(k) \\ \lambda_2(k) & \mu_2(k) & \eta_2(k) \\ \lambda_3(k) & \mu_3(k) & \eta_3(k) \end{pmatrix} = \begin{pmatrix} \sum_{i=-n_1}^{n_2} \frac{1}{\sigma_{k+i}^2} & \sum_{i=-n_1}^{n_2} \frac{\delta_{k,k+i}}{\sigma_{k+i}^2} & \sum_{i=-n_1}^{n_2} \frac{\delta_{k,k+i}^2}{\sigma_{k+i}^2} \\ \sum_{i=-n_1}^{n_2} \frac{\delta_{k,k+i}}{\sigma_{k+i}^2} & \sum_{i=-n_1}^{n_2} \frac{\delta_{k,k+i}^2}{\sigma_{k+i}^2} & \sum_{i=-n_1}^{n_2} \frac{\delta_{k,k+i}^3}{\sigma_{k+i}^2} \\ \sum_{i=-n_1}^{n_2} \frac{\delta_{k,k+i}^2}{\sigma_{k+i}^2} & \sum_{i=-n_1}^{n_2} \frac{\delta_{k,k+i}^3}{\sigma_{k+i}^2} & \sum_{i=-n_1}^{n_2} \frac{\delta_{k,k+i}^4}{\sigma_{k+i}^2} \end{pmatrix}^{-1} \tag{14}$$

with  $\delta_{k,k+i} = t_{k+1} - t_k$ .

By rearranging Equation (5), one can predict the point  $z(k)$ :

$$z_p(k) = \frac{1}{1-a_0(k)} \cdot \left\{ \sum_{i=-n_1}^{-1} a_i(k) \cdot z(k+i) + \sum_{i=1}^{n_2} a_i(k) \cdot z(k+i) \right\} \tag{15}$$

Using the measurement  $z(k)$  and predicted value  $z_p(k)$ , the difference  $\delta_z(k) = z(k) - z_p(k)$  was computed for the purpose of outlier detection. The variance of this difference is as follows:

$$\sigma_{\delta_z}^2(k) = \sigma_k^2 + \frac{1}{(1-a_0(k))^2} \cdot \left\{ \sum_{i=-n_1}^{-1} a_i^2(k) \cdot \sigma_{k+i}^2 + \sum_{i=1}^{n_2} a_i^2(k) \cdot \sigma_{k+i}^2 \right\} \tag{16}$$

Accordingly, the standardized difference was assumed to be normally distributed as:

$$\frac{\delta_z(k)}{\sigma_{\delta_z}(k)} \sim N(0,1) \tag{17}$$

under the Null hypothesis  $H_0 : \hat{\varepsilon}(k) = 0$  against the alternative hypothesis  $H_1 : \hat{\varepsilon}(k) \neq 0$ .

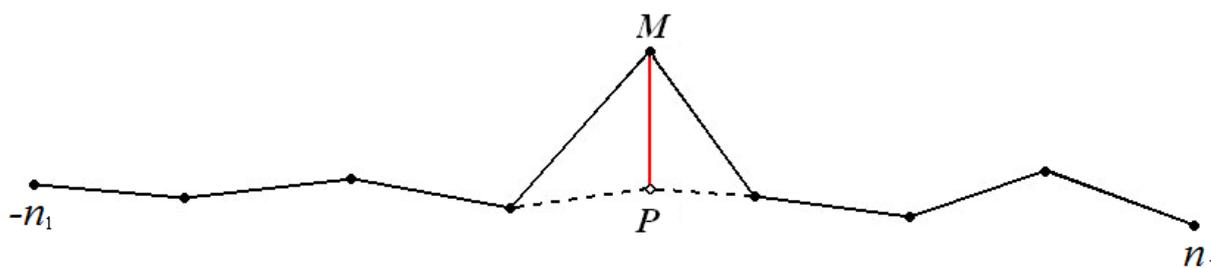
A time series was investigated where the position of each point in an example point cloud was predicted through the use of an appropriate interval  $(-n_1, n_2)$  and the statistic test was performed. Outliers were identified through the use of Equation (17) as the predictor.

## 2.2. Outlier Detection in Time Domain

In theory, data from a rotating prism mobile terrestrial LiDAR can be regarded as a set of discrete observations from a single continuous line of data. In practice, discrete observations provided by all mobile terrestrial systems include raw angle-range measurements which were adjusted by the calibration model and used to compute the local east, north, up or  $x, y, z$  coordinates. This means that we were provided with a choice of raw angle-range, adjusted angle-range or coordinates when extracting discrete observations from the output data. Most importantly, because of the nature of mobile terrestrial LiDAR data, each discrete observation (whichever is chosen) was paired with an accurate timestamp indicating when the observation was made.

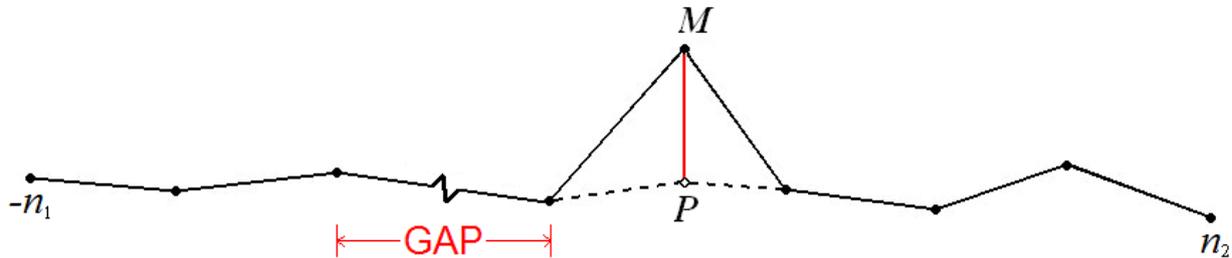
Utilizing the available information in MFIS requires that a moving window be created to extract small samples of the point cloud for analysis. This window would centre around each discrete observation in turn and use the data immediately preceding the observation as well as the data immediately succeeding the observation to calculate a predicted value. Comparing this predicted value with the observed value allows outlying data to be identified and removed from the LiDAR data. Figure 2 shows a sketch of a typical window of data that might be extracted from a point cloud and how the distance between the observed measurement ( $M$ ) and the predicted measurement ( $P$ ) was used to identify points which lie outside their neighborhood.

**Figure 2.** Time series of points used to generate predictions ( $P$ ) for measured points ( $M$ ).



The moving fixed interval prediction recognizes the fact that the point cloud can be treated as a series of lines of point data. Forming windows out of these lines of data requires care since any significant gap in the data has the potential to produce erroneous predictions (Figure 3). As the gap shown in Figure 3 was allowed to increase, the likelihood that the predicted point ( $P$ ) falls close to the true position decreased. Once the predicted measurement has strayed from the true value, any comparison between the measured value and the predicted value was meaningless.

**Figure 3.** Time series of points with an appreciable gap between two neighboring points. As the gap increases, the likelihood that the predicted measurement (P) represents the true value, decreases.



In practice, gaps caused by occlusions, reflections and/or drop-out readings effectively segmented the continuous line of data into smaller sections. In addition, since a significant portion of any terrestrial LiDAR scan is likely to include portions of the sky, numerous LiDAR points were expected to be missing from the point cloud. These missing shots effectively segment the continuous line being followed by the scanner’s optics into multiple smaller line segments. Treating these smaller line segments as independent entities allowed us to apply the PVA filter to each of these subset lines from the point cloud. Allowances had to be made for lines shorter than the window size  $(-n_1, n_2)$  and the window size had to be adjusted to accommodate points at the start and end of each line.

Great care was also taken when interpreting the test results for a given point. If an outlier is included in a window of data then the likelihood that the predicted point (P) falls close to the true position was again decreased. One way to counter this possible scenario was to form the same data section into three windows  $(-n_1, 0)$ ,  $(0, n_2)$  and  $(-n_1, n_2)$ . Computing and testing the predicted points  $(P_1, P_2, P_3)$  as shown in Equation (17), we were able to conclude that if one of the three predicted points pass, then the observed measurement (M) passes and was not treated as an outlier. This strategy effectively deals with the situation where more than one outlier existed in a given window  $(-n_1, n_2)$ . Outliers which occurred either immediately before the observed measurement (M) or immediately after the observed measurement (M) in the discrete time series did not cause a false detection to occur. When outliers existed both immediately before the observed measurement (M) and immediately after the observed measurement (M), a false detection still likely occurred.

### 3. Curved Surface Fitting (CSF)

#### 3.1. The CSF Algorithm

The generic model of a quadric curved-surface was given by

$$\begin{aligned}
 & f(a_1, \dots, a_{10}, x, y, z) \\
 & = a_1 \cdot x^2 + a_2 \cdot y^2 + a_3 \cdot z^2 + a_4 \cdot x \cdot y + a_5 \cdot x \cdot z + a_6 \cdot y \cdot z + a_7 \cdot x + a_8 \cdot y + a_9 \cdot z + a_{10} = 0
 \end{aligned}
 \tag{18}$$

where  $(x, y, z)$  is the coordinate of a point on the surface and  $a_j$  ( $j = 1, \dots, 10$ ) are the parameters. Due to the ambiguity in the surface determination introduced by the parameter  $a_{10}$ , it was necessary to constrain the ten parameters by

$$C = a_1^2 + a_2^2 + a_3^2 + a_4^2 + a_5^2 + a_6^2 + a_7^2 + a_8^2 + a_9^2 + a_{10}^2 = 1 \tag{19}$$

Given the measurements  $(x_i, y_i, z_i)$  of point  $i$  with its 3x3 variance matrix  $D_{ii}$  and the approximate values  $(a_1^{(0)}, \dots, a_{10}^{(0)})$  of the ten parameters, one obtains the linearized form of Equation (18) as

$$A_i v + B_i \delta a + w_i = 0 \tag{20}$$

where

$$v = (\dots \dots \dots v_{x_i} \ v_{y_i} \ v_{z_i} \ \dots \ \dots \ \dots)^T \tag{21}$$

$$A_i = (\dots \dots \dots a_{x_i} \ a_{y_i} \ a_{z_i} \ \dots \ \dots \ \dots) \tag{22}$$

$$B_i = (x_i^2 \ y_i^2 \ z_i^2 \ x_i y_i \ x_i z_i \ y_i z_i \ x_i \ y_i \ z_i \ 1) \tag{23}$$

$$w_i = F_i(a_1^{(0)}, \dots, a_{10}^{(0)}, x_i, y_i, z_i) \tag{24}$$

The values  $(a_{x_i} \ a_{y_i} \ a_{z_i})$  in Equation (22) represent the partial derivatives of Equation (18) with respect to the given measurements of point  $i$ . Under the assumption that all of the measurement points are not correlated to each other, one defines

$$v_i = -A_i v = -\{a_{x_i} v_{x_i} + a_{y_i} v_{y_i} + a_{z_i} v_{z_i}\} \tag{25}$$

to create an equivalent single measurement to the measured three coordinate components of a point so that Equation (25) was simplified to

$$v_i = B_i \delta a + w_i \tag{26}$$

for  $i = 1, 2, \dots, n$ .

As it can be seen, the combination of the linearized form of Equations (18), (19) and (26) is a standard parametric adjustment model with a constraint. There is no need to provide further detail for its solution. For more details, refer to [17].

Due to the likelihood that one or more outliers may creep into the point cloud sample being used to form the polynomial surface, it was a good idea to provide a statistic check on the goodness-of-fit for each calculated surface. By comparing the a posteriori variance with the a priori variance, we produced such a statistic which follows the Chi-Squared distribution, as shown in Equation (27):

$$\frac{V^T \cdot P_{LL} \cdot V}{\sigma_0^2} \sim \chi^2(n - 10 + 1) \tag{27}$$

In order to test if a point (i) is a potential outlier, two different test statistics can be constructed. When point i was included in the polynomial surface fitting, the Tau distribution [18] was used to test the residual of point i as shown in Equation (28) (test statistic 1).

$$T_i^{(1)} = \frac{|v_i|}{\hat{\sigma}_0 \cdot \sqrt{q_{v_i}}} \sim \tau(n - 10 + 1) \tag{28}$$

where  $v_i$  was the residual of point i,  $\hat{\sigma}_0$  was the posterior variance of unit weight and  $q_{v_i}$  was the cofactor of  $v_i$ . The Tau test was used in test statistic 1 due to the fact that  $v_i$  and  $\hat{\sigma}_0$  are dependant variables.

Alternatively, when point  $i$  was not included in the polynomial surface fitting, the Student  $t$  distribution was used to test the discrepancy between point  $i$  and the surface as shown in Equation (29) (test statistic 2).

$$T_i^{(2)} = \frac{w_i}{\sigma_{w_i}} \sim t(n - 1 - 10 + 1) \quad (29)$$

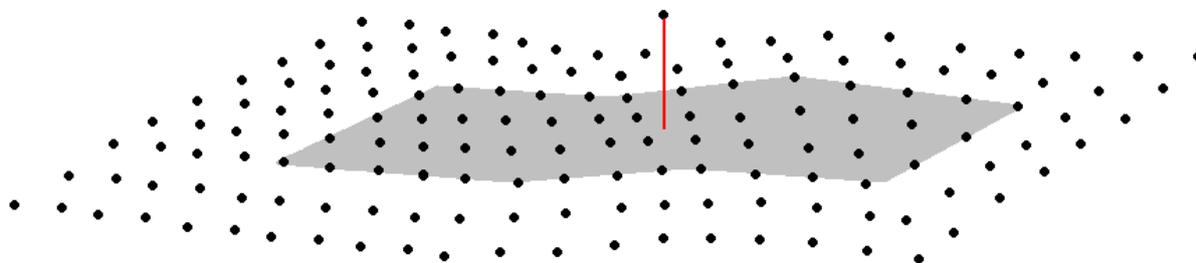
where  $w_i$  was computed by plugging point  $i$  into Equation (18) after the surface parameters have been determined and  $\sigma_{w_i}$  was the estimated standard deviation of  $w_i$ .

Using the specified patch size, data surrounding each individual point in an example point cloud was used to create a curved surface. The surfaces were validated using Equation (27) and outliers were spatially detected using Equation (29) as the predictor.

### 3.2. Outlier Detection in the Spatial Domain

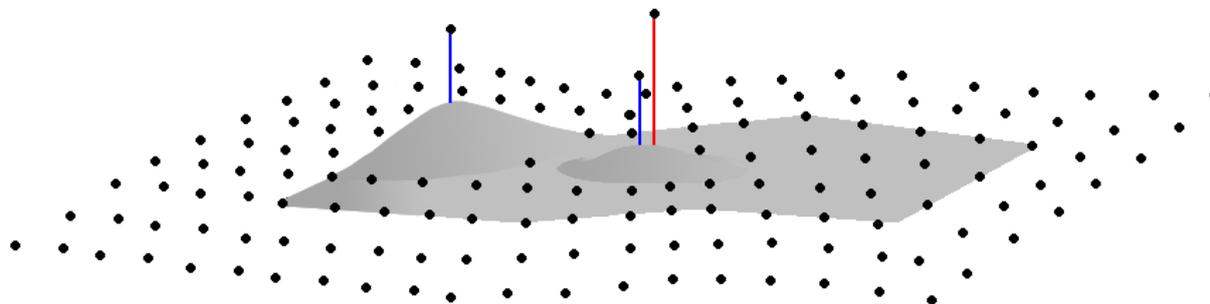
Viewing the LiDAR data as a strictly spatial entity, the relative position of a point with respect to its neighbors was used to identify outliers. Conducting a search around each point in a point cloud, a representative sample of neighboring points was obtained. This representative sample was used to form a surface. Comparing these points to the surface, outliers were identified by their spatial separation from the aforementioned surface. This concept is illustrated in Figure 4.

**Figure 4.** Polynomial surface patch in the immediate neighborhood of the point being tested.



The quadratic curved-surface fitting algorithm generates small surface patches in the neighborhood of each point (Figure 4). This is an outlier detector in the spatial domain which relies on the assumption that the points immediately adjacent to an outlier will themselves lie on the surface and not be outliers as well. The number of points to use in the polynomial patch fitting was a variable that needs to be determined. On one hand, at least 10 points are required to derive the best fit surface. On the other hand, the larger the number of coordinates used the greater the probability that other outliers will be incorporated into the calculation of the surface (Figure 5). In fact, when discussing LiDAR, the conditions which cause an outlier will also greatly increase the likelihood that other outliers lie close by. Therefore, care had to be taken when setting a patch size. The test statistic given in Equation (27) by giving us a measure of the fit of the surface to the data patch was used as an aid to determining whether outliers are included within the selected patch data.

**Figure 5.** Polynomial surface patch generated using a data section containing outlying points. Blue lines indicate outliers used to compute the surface. The Red line indicates the point being tested as an outlier.



## 4. Tests and Results

### 4.1. The Lynx Mobile Mapper: Hardware and Software

MFIS and CSF were tested using parts of two data sets collected with the Lynx Mobile Mapper. The Lynx Mobile Mapper, consisted of two LiDAR sensors, two calibrated passive imaging cameras, and the Applanix POS LV 420 during data collection. This system is designed to collect rich survey-grade LiDAR and image data from a vehicle moving at traffic speeds.

For each test site, calibration of the Applanix POS system was accomplished immediately before the data collect. GAMS (GPS azimuth measurement subsystem) parameters for the Applanix POS system were determined the day of the data collection to ensure accuracy. For each test area the data sections used in testing were selected from the same data used to determine the LiDAR system boresight values. The system lever arms and boresight values were obtained using the LiDAR manufacturer's recommended procedure.

Processing the raw POS data proceeded using the software package POSPAC. The result of this processing was an SBET (Smoothed Best Estimated Trajectory) file. This SBET file was then used in conjunction with the boresight values previously obtained to process the raw LiDAR data. This was accomplished using the software package Dashmap. The processed LiDAR data was output into ASCII format.

### 4.2. Description of Data

Four sections of Lynx point clouds (A, B, C and D) shown in Figures 6 and 7 were selected for testing. Table 1 gives specifics about the contents of these point clouds.

**Table 1.** Specifications for point clouds used in algorithm testing.

Point Cloud	A	B	C	D
Total No. of Points	295,147	495,345	120,092	216,228
Total No. of Outliers	872	280	1,308	226
Total % of Points Which are Outliers	0.30	0.06	1.09	0.10

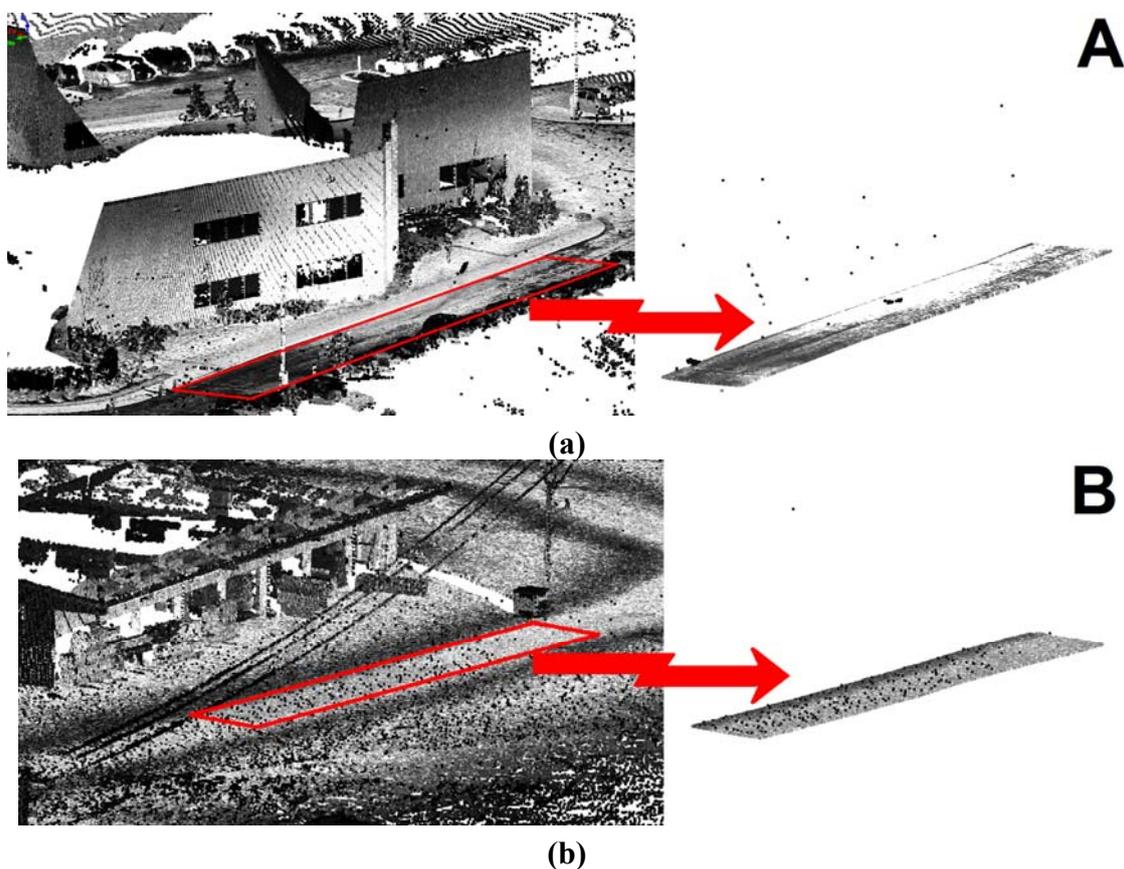
#### 4.2.1. Data with Simple Geometry

Point cloud A was obtained on a section of asphalt from a generic parking lot over which multiple drive passes were performed. Point cloud B was collected in a dirt lot where multiple passes were also performed.

Point cloud A contained numerous outliers in two large groups with other outliers spread throughout the data. As shown in Table 1, the outliers made up 0.30% of the total point cloud. This data was collected on a day where the asphalt was wet but the temperature was just above 0° Celsius. The prevailing cold wet conditions caused condensation from the vehicle's exhaust pipe to combine with varying high and low intensity returns from the standing pools of water. This caused multiple laser reflections to be recorded above the asphalt surface.

In contrast, point cloud B was collected in a lumber yard with an unpaved, rough finished, mostly native clay driving area that had been pitted and grooved by the passage of heavy vehicles. The ambient temperature during the collect was about 15° Celsius and the ground surface was dry. These conditions produced a point cloud with comparatively few outliers (0.06% from Table 1). Many of the outliers which did exist in this data set were within centimeters of the ground surface. It has also been observed that several of the outliers in this point cloud were collected from a different position than the ground surface and therefore lie out of temporal series with much of the data.

**Figure 6.** Point clouds of simple geometry taken from two separate parking lots used during testing of the two outlier algorithms previously described. **(a)** Point cloud A contains numerous outliers evenly distributed above an asphalt surface; **(b)** Point cloud B contains outliers distributed evenly across a bare soil surface.

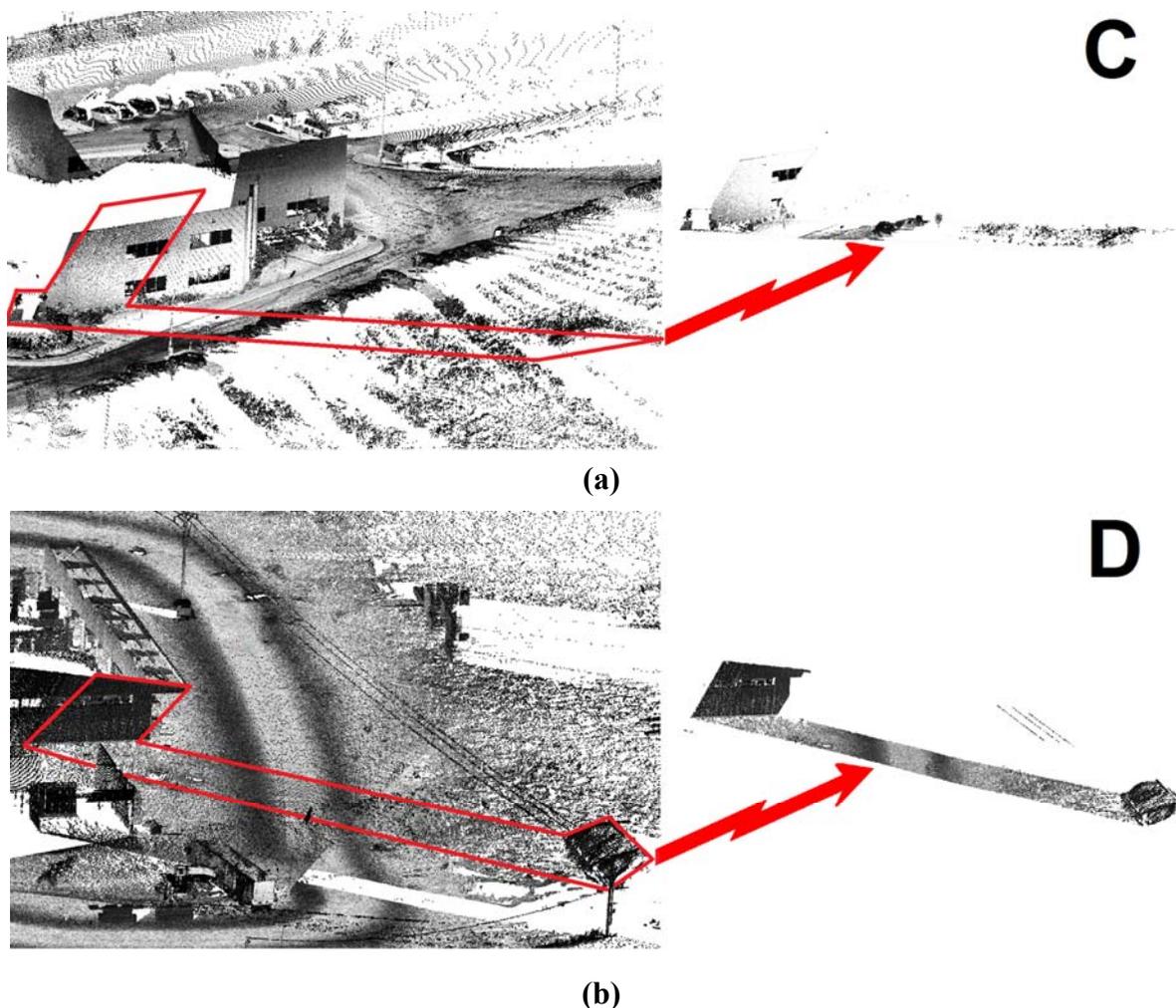


#### 4.2.2. Data with Complex Geometry

Point cloud C was taken from the same data set as point cloud A. A two second section of the vehicle trajectory was isolated and all the LiDAR data collected during that time was extracted to form point cloud C. Similarly, point cloud D was taken from the same data set as point cloud B. Again a two second section of the vehicle trajectory was isolated and all LiDAR data collected during that time was extracted to form point cloud D.

Point cloud C contains a complex scene including part of a building with windows, cars, asphalt road, sidewalk, curb, two small trees, bushes and part of a natural field. As Table 1 indicates, the number of outliers in point cloud C comprise about 1.09% of the data. Point cloud D contains another complex scene including part of a building with windows, overhead wires, a bare earth driving surface and a bare earth mound. Unlike point cloud C, point cloud D contains no vegetation, and as such, the number of points that can be classified as outliers is much lower (0.10% of the data, Table 1).

**Figure 7.** Point clouds of complex geometry taken from two separate parking lots used during testing of the two outlier algorithms previously described. **(a)** Point cloud C contains numerous outliers along with a building wall, sidewalk, curb, road, field and vegetation; **(b)** Point cloud D contains outliers along with a building wall, overhead wires, bare soil surface and soil mound.

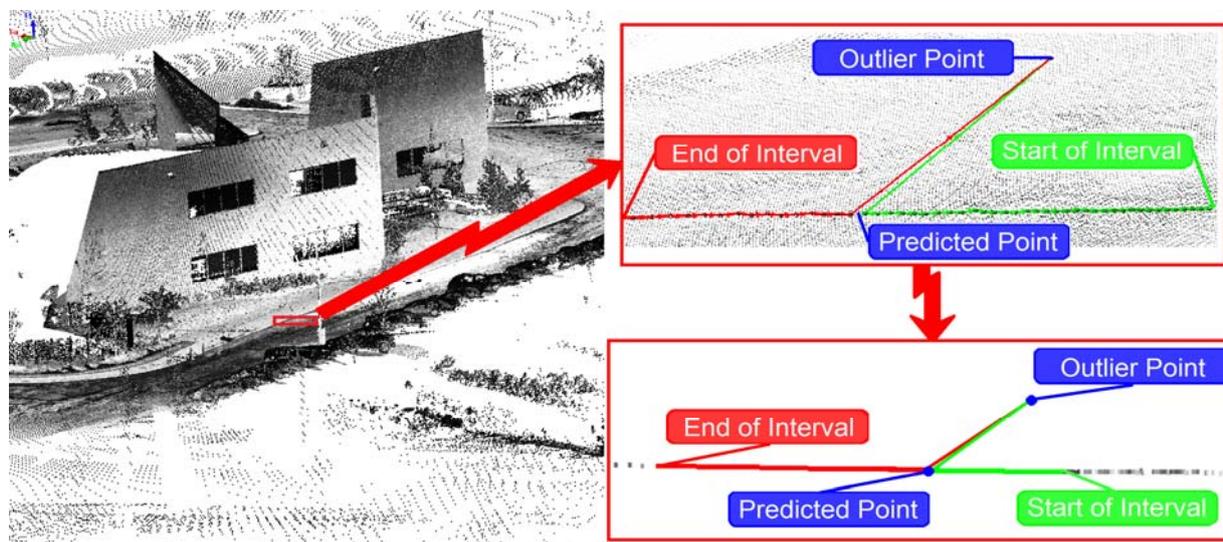


### 4.3. Analysis of Outlier Detection Utility in Real Data

Both of the algorithms described in Sections 2 and 3 were implemented under Microsoft Visual C++ 6.0.

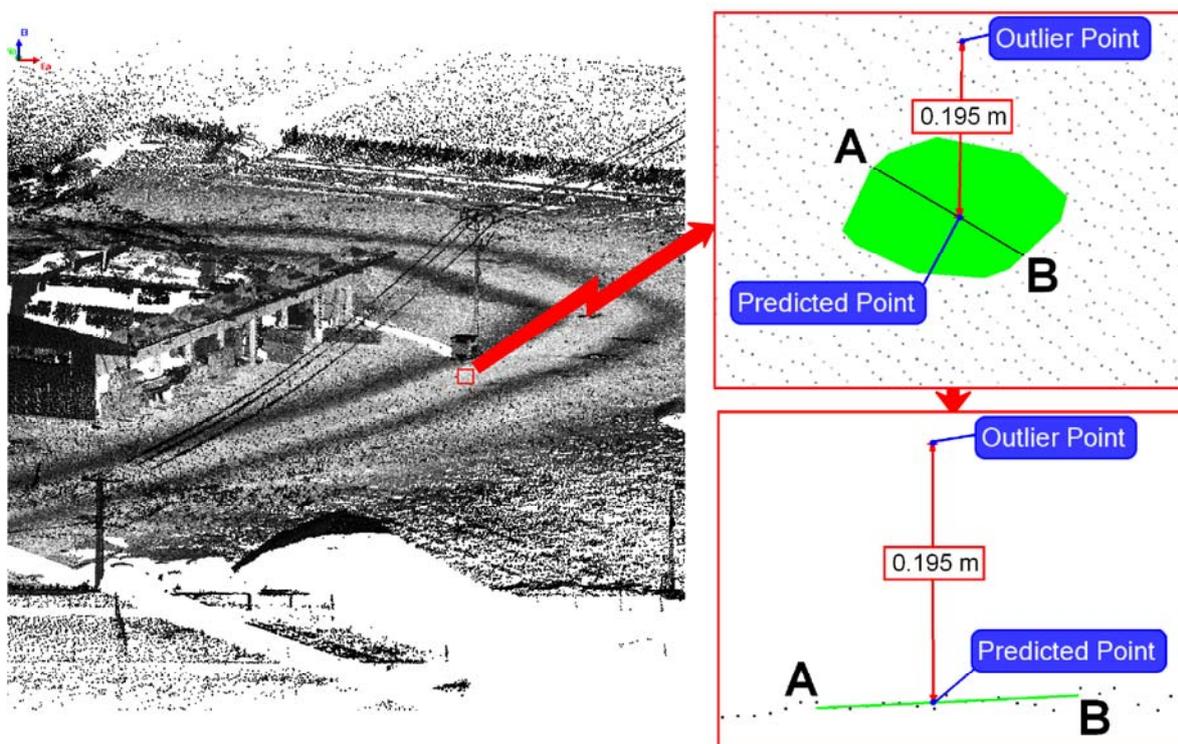
To ensure that the routine was working as expected and to assess the effectiveness of the routine in realistic data, a line of data was extracted from point cloud A (Figure 8). Figure 8 shows a line of data with one of the points lying far out of spatial position from the rest. The labels ‘Start of Interval’ and ‘End of Interval’ give an indication of the order in which the points were collected. Those points, colored green were collected before the outlier point and those points colored red were collected after the outlier point.

**Figure 8.** The moving fixed interval smoothing method applied to a section of a mobile terrestrial LiDAR point cloud collected with the Lynx Mobile Mapper.



As with the moving fixed interval smoother, quadratic polynomial surface fitting was used on a section of point cloud B (Figure 9). As before, this was done to ensure that the routine was working as expected and to assess the effectiveness of the routine in realistic data. The results are illustrated in Figure 8. Selecting a patch size of 100 points around the point in Figure 8 identified as ‘Outlier Point’, a polynomial surface was calculated. Computing the shortest distance between ‘Outlier Point’ and the surface, we find that the point lies 0.195 m from the surface. Generating the statistics described in Section 2.1, we find that the surface does fit the data well but the point is far outside the allowable deviation from the surface. Looking at a cross section of the surface and data (Figure 9), the deviations of the points around the calculated surface become clear. The deviation of the point labeled ‘Outlier Point’, is obviously far greater than the deviation of any other point.

**Figure 9.** Section of a mobile terrestrial LiDAR point cloud collected with the Lynx Mobile Mapper. Object is being viewed from the South.



#### 4.4. Results

As previously mentioned in Section 2.2, the output from the mobile terrestrial LiDAR allowed for raw angle-range measurements, angle-range measurements adjusted by the system calibration information and/or coordinate values to be input into the MFIS algorithm as the discrete observations. During testing, all three options were tried with data of both simple and complex geometry, however, the results obtained from the raw angle-range data were unimpressive and they were excluded from this section.

Several trials were conducted to find the optimum window size for MFIS in each point cloud. Table 2 gives the results of the best trial running the MFIS algorithm using the LiDAR range values after they had been adjusted for the scanner's constant range offset value and the individual range corrections for varying intensity returns. Table 3 gives the results of the best trial, running the MFIS algorithm using the output easting, northing and up coordinates.

Being that two test statistics were derived for the CSF routine, one requiring the point being tested to be included in the data used to create the surface patch Equation (28) and one requiring the point being tested to be excluded from the data used to create the surface patch Equation (29), both scenarios were tested. Several trials were conducted to find the optimum patch size for CSF in each point cloud for both test statistic 1 Equation (28) and test statistic 2 Equation (29). The results of the best trial from tests conducted using point clouds A, B, C and D are given in Table 4 for test statistic 1 and in Table 5 for test statistic 2.

**Table 2.** Results from trials conducted using adjusted LiDAR ranges in the Moving Fixed Interval Smoother (MFIS) on point clouds A, B, C and D.

Point Cloud	A	B	C	D
Window Size (points)	20	60	20	15
No. of Outliers Identified	519	54	1,200	202
No. of Non-Outliers Identified (False Identification)	5,709	17,112	24,502	35,195
No. of Outliers Missed	353	226	108	24
% of Outliers Identified	59.52	19.29	91.74	89.38
% of Point Cloud Identified	2.11	3.47	21.40	16.37
% of Point Cloud Identified Incorrectly (False Identification Rate)	1.93	3.45	20.40	16.28

**Table 3.** Results from trials conducted using XYZ coordinates in the Moving Fixed Interval Smoother (MFIS) on point clouds A, B, C and D.

Point Cloud	A	B	C	D
Window Size (points)	15	35	15	15
No. of Outliers Identified	742	244	1,183	206
No. of Non-Outliers Identified (False Identification)	4,130	7,847	22,555	37,255
No. of Outliers Missed	130	36	125	20
% of Outliers Identified	85.09	87.14	90.44	91.15
% of Point Cloud Identified	1.65	1.63	19.77	17.32
% of Point Cloud Identified Incorrectly (False Identification Rate)	1.40	1.58	18.78	17.22

**Table 4.** Results from trials conducted using Quadratic Polynomial Surface Fitting (CSF) on point clouds A, B, C and D using test statistic 1 in Equation (28).

Point Cloud	A	B	C	D
Patch Size (points)	500	100	500	400
No. of Outliers Identified	38	15	118	28
No. of Non-Outliers Identified (False Identification)	616	96	986	784
No. of Outliers Missed	834	265	1,190	198
% of Outliers Identified	4.36	5.36	9.02	12.39
% of Point Cloud Identified	0.54	0.02	0.92	0.38
% of Point Cloud Identified Incorrectly (False Identification Rate)	0.51	0.02	0.82	0.36

**Table 5.** Results from trials conducted using Quadratic Polynomial Surface Fitting (CSF) on point clouds A, B, C and D using test statistic 2 in Equation (29).

Point Cloud	A	B	C	D
Patch Size (points)	500	100	500	400
No. of Outliers Identified	253	199	335	188
No. of Non-Outliers Identified (False Identification)	0	1	7,029	5,193
No. of Outliers Missed	619	81	973	38
% of Outliers Identified	29.01	71.07	25.61	83.19
% of Point Cloud Identified	0.09	0.04	6.13	2.49
% of Point Cloud Identified Incorrectly (False Identification Rate)	<0.01	<0.01	5.85	2.40

The combination of the MFIS and CSF methods was performed, where the reduced point cloud produced by the MFIS method was then analyzed with the CSF method. This combined MFIS/CSF routine was performed using the easting, northing, up coordinate version of the MFIS method and the test statistic 2 version of the CSF method. The results for this test conducted using data strips A, B, C and D are given in Table 6.

**Table 6.** Results from trials conducted using the Moving Fixed Interval Smoother (MFIS) preceding Quadratic Polynomial Surface Fitting (CSF) on point clouds A, B, C and D.

Point Cloud	A	B	C	D
MFIS Window Size (points)	15	35	15	15
CSF Patch Size (points)	100	100	500	400
No. of Outliers Identified	777	245	1,197	220
No. of Non-Outliers Identified (False Identification)	4,130	7,851	24,760	38,874
No. of Outliers Missed	95	35	111	6
% of Outliers Identified	89.11	87.50	91.51	97.35
% of Point Cloud Identified	1.66	1.63	21.61	18.08
% of Point Cloud Identified Incorrectly (False Identification Rate)	1.40	1.58	20.62	17.98

As a further control on the results, point clouds A, B, C and D were imported into the Polyworks ([www.innovmetric.com](http://www.innovmetric.com)) IMSurvey module and the built-in outlier detection routine was used. This outlier routine is a precursor to the wrap mesh function, used in the creation of triangular irregular network (TIN) models. The outlier routine in Polyworks required us to set values for the maximum point to point distance and the maximum cluster size. Several attempts were made to optimize these inputs to the commercial routine. The results of the best trial using the Polyworks software are summarized in Table 7.

**Table 7.** Results from trials conducted using Polyworks IMSurvey's (Version 11.0.30) Reject Outliers routine on point clouds A, B, C and D.

Point Cloud	A	B	C	D
Max Spot Space Measured (m)	0.053	0.040	0.100	0.536
Min Spot Space Measured (m)	0.023	0.012	0.024	0.012
Max Point-to-Point Distance Used (m)	0.080	0.050	0.080	0.080
Maximum Cluster Size Used (m)	5.000	1.000	5.000	1.000
No. of Outliers Identified	717	221	1,281	170
No. of Non-Outliers Identified (False Identification)	15,846	30,645	28,252	42,774
No. of Outliers Missed	155	59	27	56
% of Outliers Identified	82.22	78.93	97.94	75.22
% of Point Cloud Identified	5.61	6.23	24.59	19.86
% of Point Cloud Identified Incorrectly (False Identification Rate)	5.37	6.19	23.53	19.78

#### 4.5 Analysis and Discussions

From the trials of the MFIS routine, it was clear that for simple geometry, using the coordinate values consistently produced better results than using the adjusted ranges. This was most clearly

shown from the trial with point cloud B, where 19.29% of the outliers were found by using ranges as opposed to the 87.14% outliers found by using coordinates. The trials also showed that the coordinate version of the MFIS routine had a much smaller failure rate, identifying 1.58% of point cloud B incorrectly as outliers. The range version of the MFIS routine incorrectly identified 3.45% of point cloud B, nearly double the amount of the coordinate version. The results using coordinates from point cloud B were interesting because many of the outliers in point cloud B occur temporally after the initial scan, and therefore do not follow the time series pattern. Looking more closely at the results we found that many of these outliers in point cloud B fail in the easting or northing components and not, as is the case in point cloud A, the up component. It is significant that the easting and northing components were able to save the MFIS routine in point cloud B, even though many of the outlying data points lie out of temporal sequence with other spatially close data.

Upon introducing complex geometry to the MFIS routine (point clouds C and D), we find that the false identification rate dramatically increases to between 16% and 20% of the total point cloud. Interestingly enough, the use of adjusted ranges in the MFIS routine was able to match the performance when the coordinate values were used in both point clouds C and D. The coordinate method still shows a slightly smaller amount of false identifications (about 1–2%) than when the adjusted ranges were used in point clouds C and D; however, the difference in the number of correct identifications is not as stark as it had been when using simple geometry.

The non-outliers that were wrongly identified by the MFIS routine occur in areas of the point cloud where the regularity of the time series was disrupted by rough objects such as vegetation or manholes. Where a manhole was encountered, the data density was insufficient to model the raised surfaces on the manhole's lid. Also, areas of the point cloud where the system was collecting data while stationary, occlusions and changes in surface direction seemed to cause false detections.

CSF performed best on point clouds B and D, identifying 5.36% and 12.39% of the outliers, respectively, when test statistic 1 was used and 71% and 83% of the outliers, respectively when test statistic 2 was used. Point clouds A and C had only 4.36% and 9.02% of their outliers identified respectively when test statistic 1 was used and 29% and 25% of their outliers identified respectively when test statistic 2 was used. Overall, test statistic 2 consistently produced better results than test statistic 1. Test statistic 2 always found more outliers than test statistic 1. It is interesting to note that the number of false identifications from test statistic 2 did significantly increase with complex geometry, something that cannot be said about the results obtained from test statistic 1.

The CSF routine performed exceptionally poorly on data where the outliers were clumped closely together as was the case in point clouds A and C. Examining the chi squared test statistic, we see that in areas where the data is clumped together, the fit of surrounding polynomial surfaces are quite poor. Additionally, when complex geometry was introduced by point clouds C and D, the chi square statistic shows us that the polynomial surfaces fit very poorly in areas where the point cloud transitioned from one object to another. This routine had much greater success on point clouds B and D, where the outliers are more spread out on the road surface and building. In fact, the spatial nature of the method showed itself as being quite well suited for point clouds where outlying data points may not lie in a sequential temporal order.

Combining MFIS and CSF using simple geometry showed that in both point clouds A and B we found almost 90% of the outliers with relatively few false alarms (about 1.5% of the total point cloud

size). The results proved slightly better in point cloud A when the routines were combined, since a small number of outliers not identified by MFIS were identified by CSF. On the other hand, point cloud B saw much less benefit from the combination of the methods. The vast majority of points found in point cloud B were found by the MFIS routine. In fact only 1 new outlier was identified by the CSF routine when the combined MFIS/CSF routine was applied to point cloud B. With the introduction of complex geometry in point clouds C and D, we find that again we could identify greater than 90% of the outliers, however the cost in false identifications increased to around 20%.

The results from the commercial software package Polyworks show that it was able to identify fewer outliers in point clouds A and B. The cost to identify these outliers was much greater than that from either the MFIS algorithm or the CSF algorithm. Where MFIS identified around 1.5% and CSF identified less than 0.01% of the point cloud incorrectly as outliers, Polyworks identified as much as 6.19% of the point cloud incorrectly as outliers. However, when complex geometry was introduced in point cloud C, we find that the commercial software achieved comparable results with the MFIS routine. The commercial software identified about 97% of the known outliers while wrongly eliminating another 24% of the total point cloud. The MFIS routine found over 90% of known outliers, while eliminating about 20% of the point cloud incorrectly. The complex geometry of point cloud D again separates the commercial software from the combined MFIS/CSF routines. With point cloud D, the commercial software identified around 75% of the outliers in the point cloud while again identifying about 20% incorrectly. The combined MFIS/CSF routine was able to find 97% of the outliers in point cloud D with a similar failure of approximately 20% of the point cloud being identified incorrectly. Like the results obtained from point clouds A and B, the results from point cloud D demonstrate a significant improvement over the commercial software.

## 5. Conclusion

This paper presented the results of combining two different types of algorithms for the detection and filtering of outliers in point clouds collected using mobile terrestrial LiDAR. These routines have taken advantage of the extra information that is generally available from this type of equipment. The two mathematical models presented here allowed for the creation of two computer routines which perform outlier detection in mobile terrestrial point cloud data.

It has been shown that individually, each method of outlier detection has difficulty detecting outliers when certain conditions are met. Under real world conditions, variations in surface quality from objects such as vegetation or manholes can cause problems. The possibility also exists that not all points which lie spatially close to each other will have been collected at relatively similar times. Due to the mobile nature of this LiDAR variant, it is possible that the vehicle carrying the LiDAR has looped back on its previous position or that the scanners have seen the same point from different parts of its trajectory. Equally, the conditions which cause outliers tend to create multiple outliers within a particular region. Creating a surface patch from data that contains outliers, results in a poor fitting surface and a problematic outlier detection outcome.

While each method has proven to have its own strengths and weaknesses, they have each proven capable of detecting and removing outliers from actual LiDAR data. Whether the point cloud contains relatively simple geometry or contains data comprising a more complex scene, these methods have

successfully identified outliers in real LiDAR data. When compared to commercial software, the combined routines have proven to exceed the commercial routine in performance. More work is needed to optimize the inputs to the routines, specifically, to determine accurate error estimates for the point cloud coordinates.

### Acknowledgements

The authors would like to thank the Natural Sciences and Engineering Research Council (NSERC) of Canada for the financial support.

### References

1. Antova, G. Precise Mapping with 3D Laser Scanning. In *Proceedings of the International Conference on Cartography and GIS*, Borovets, Bulgaria, 25–28 January 2006.
2. Miller, M.M.; Meertens, C.; Phillips, D.; Rubin, C.; Ely, L.; Pratt-Sitaul, B. *Collaborative Research MRI: Acquisition of Terrestrial Laser Scanning Systems for Earth Science Research*; Proposal Submitted to EAR Major Research Instrumentation; UNAVCO: Boulder, CO, USA, 2009. Available online: [http://www.unavco.org/pubs\\_reports/proposals/2009/MRI\\_TLS\\_EAR2009\\_UNAVCO-CWU.pdf](http://www.unavco.org/pubs_reports/proposals/2009/MRI_TLS_EAR2009_UNAVCO-CWU.pdf) (accessed on 7 March 2010).
3. Lu, C.T.; Chen, D.; Kou, Y. Algorithms for Spatial Outlier Detection. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03)*, Melbourne, FL, USA, 19–22 November 2003.
4. Sotoodeh, S. Outlier Detection in Laser Scanner Point Clouds. In *Proceedings of the ISPRS Commission V Symposium, Image Engineering and Vision Metrology*, Dresden, German, 25–27 September 2006; Volume XXXVI, Part 5, pp. 297-302.
5. Ben-Gal, I. Outlier detection. In *Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researcher*; Maimon, O., Rockach, L., Eds.; Springer: New York, NY, USA, 2005; Volume 1, Chapter 1, pp. 1-13.
6. Breuing, M.; Kriegel, H.; Ng, R.; Sander, J. LOF: Identifying Density-Based Local Outliers. In *Proceedings of the International Conference on Management of Data (ACM SIGMOD 2000)*, Dallas, TX, USA, 16–18 May 2000.
7. Last, M.; Kandel, A. Automated Detection of Outliers in Real-World Data. In *Proceedings of the Second International Conference on Intelligent Technologies*, Bangkok, Thailand, 27–29 November 2001.
8. Papadimitriou, S.; Kitagawa, H.; Gibbons, P.B.; Faloutsos, C. LOCI: Fast Outlier Detection Using the Local Correlation Integral. In *Proceedings of the 19th International Conference on Data Engineering*, Bangalore, India, 5–8 March 2003.
9. Zheng, M.-Q.; Chen, C.-C.; Lin, J.-X.; Fan, M.-H.; Jansc , T. An Algorithm for Spatial Outlier Detection Based on Delaunay Triangulation. In *Proceedings of the International Workshop on Computational Intelligence in Security for Information Systems (CISIS'08)*, Genova, Italy, 23–24 October 2008.
10. Wang, J.G. *Pre-Processing of INS-Data with the Help of the  $\alpha$ - $\beta$ - $\gamma$ Filter*; Internal Report; Institute of Geodesy, UniBw Munich: Neubiberg, Germany, July 1997; (in German).

11. Ahn, S.J.; Rauh, W.; Cho, H.S.; Wanecke, H.J. Orthogonal distance fitting of implicit curves and surfaces. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 620-638.
12. Gasca, M.; Sauer, T. Polynomial interpolation in several variables. *Adv. Comput. Math.* **2000**, *12*, 377-410.
13. Xu, Y. Polynomial interpolation in several variables, cubature formulae, and ideals. *Adv. Comput. Math.* **2000**, *12*, 363-376.
14. Gruen, A.; Akca, D. Least squares 3D surface and curve matching. *ISPRS J. Photogramm. Remote Sens.* **2005**, *59*, 151-174.
15. Akca, D. Matching of 3D surfaces and their intensities. *ISPRS J. Photogramm. Remote Sens.* **2007**, *62*, 112-121.
16. Chui, C.K.; Chen, G. *Kalman Filtering with Real-Time Applications*, 4th ed.; Springer-Verlag: Berlin/Heidelberg, German, 2009.
17. Wang, J.G. Least Squares Quadric Surface Fitting with the help of Statistical Tests—A Case Study in Industrial Surveying. In *Proceedings of International Geomatics Forum*, Qingdao, China, 29–30 May 2009.
18. Caspary, W.F. *Concepts of Network and Deformation Analysis*; Monograph 11, School of Surveying, UNSW, Sydney, NSW, Australia, August 2000.

© 2011 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution license (<http://creativecommons.org/licenses/by/3.0/>).