Analysis of Airborne LiDAR Point Clouds
With Spectral Graph Filtering

Eda Bayram®, Pascal Frossard®, Fellow, IEEE, Elif Vural®,
and Aydin Alatan®, Senior Member, IEEE

Abstract—Separation of ground and nonground measurements is an essential task in the analysis of light detection and ranging (LiDAR) point clouds; however, it is challenge to implement a LiDAR filtering algorithm that integrates the mathematical definition of various landforms. In this letter, we propose a novel LiDAR filtering algorithm that adapts to the irregular structure and 3-D geometry of LiDAR point clouds. We exploit weighted graph representations to analyze the 3-D point cloud on its original domain. Then, we consider airborne LiDAR data as an irregular elevation signal residing on graph vertices. Based on a spectral graph approach, we introduce a new filtering algorithm that distinguishes ground and nonground points in terms of their spectral characteristics. Our complete filtering framework consists of outlier removal, iterative graph signal filtering, and erosion steps. Experimental results indicate that the proposed framework achieves a good accuracy on the scenes with data gaps and classifies the nonground points on bridges and complex shapes satisfactorily, while those are usually not handled well by the state-of-the-art filtering methods.

Index Terms—Airborne laser scanning, graph signal processing, light detection and ranging (LiDAR) filtering, spectral graph filtering, unorganized 3-D point cloud.

I. INTRODUCTION

AIRBORNE light detection and ranging (LiDAR) systems are very popular in urban applications and geospatial analysis, since they provide quite dense and highly accurate, yet unorganized 3-D point cloud descriptions of earth surface. Due to the nature of LiDAR data, its analysis poses important research challenges. For example, the important task of distinguishing ground and nonground points, called filtering in the LiDAR literature, is not fully solved in general settings. Generally, the elevation is an important criterion to determine whether a measurement belongs to a ground/bare-earth (BE) or nonground/object (OBJ) point. In particular, the LiDAR filtering methods often build on the assumption that the depth change between an object and the BE is very large, whereas such a change is rather slow on terrain measurements. Such filtering algorithms can mostly be classified into four families [1], namely the methods based on: 1) interpolation strategies [2], [3]; 2) slope analysis [4], [5]; 3) morphology operators [6], [7]; and 4) segmentation methods [8], [9]. Most of the existing filtering algorithms unfortunately rely on raster-based representation in which the data are fit on a 2-D grid, instead of directly processing them in the original 3-D domain. This approach leads to inaccuracies in the evaluation of local elevation changes due to resampling and interpolation on the raster data [4], [10], so that the filtering result becomes sensitive to the grid resolution [11]. Therefore, the most robust methods are the few algorithms that directly analyze the original data points, such as the adaptive triangulated irregular network (TIN) densification by Axelsson [12]. Axelsson’s algorithm first creates a sparse TIN on a set of relatively low points, which are the ground seed points, and then refines this reference ground surface by densifying the TIN progressively. Although the performance of this approach is a compelling argument against the raster-based algorithms, it leads to a relatively large misclassification rate for erroneous low-altitude points [13].

In this letter, we propose a new framework to filter LiDAR data in the actual 3-D domain of airborne point clouds. Conventional signal filtering operations work upon the uniformly sampled data, such as speech signals in 1-D or images in 2-D regular lattices, but not irregular LiDAR data. Hence, we consider aerial LiDAR data as a nonuniformly sampled elevation signal on a graph representation defined by the coordinates of the 3-D points and establish our filtering scheme by the help of the graph signal processing framework [14]. This framework is also adopted in [15] to describe different types of edge features occurring in LiDAR data.

The proposed filtering algorithm begins with a novel outlier removal step, where we conduct a simple graph operation on the elevation signal to detect the local isolated points. Then, we directly distinguish OBJ points from BE points through a spectral graph filtering procedure applied on the elevation signal. Specifically, we exploit the fact that a high-pass filter uncovers the scattered points, such as trees and shrubs, or the jump edge points, such as the outlines of the buildings. Therefore, it directly indicates the elevation change between an object and BE, where the large positive filter output arises on the object part of the elevation change, while the large negative response belongs to the ground part. Our framework is inspired by morphology-based recursive filtering algorithms [6], [7]: at each iteration, the detection of OBJ points is followed by an erosion process where the elevation
value at the detected OBJ point is set to the one of their BE neighbors. Such a process permits capturing nonground objects starting from their contour toward their center, as illustrated in Fig. 1.

We have tested our algorithm on a benchmark data set provided by the International Society for Photogrammetry and Remote Sensing (ISPRS) [16]. The results show that the proposed algorithm performs well for particular features that are considered to be challenging for the state-of-the-art filtering methods, such as long linear structures, vegetation on slope, and data gaps. Furthermore, due to the outlier removal step, our algorithm is not trapped by the low outliers, on the contrary to many filtering algorithms including the Axelsson’s method [12].

The rest of this letter is organized as follows. Section II explains the algorithm flow, and Section III introduces the utilized data set and presents the experimental results. Finally, Section IV concludes this letter.

II. Spectral Graph Filtering Algorithm

The algorithm workflow is illustrated in Fig. 2. First, we construct a weighted graph to represent the LiDAR data as an elevation signal on this graph structure, which is described in Section II-A. Then, the algorithm starts with outlier removal as a preprocessing step, which is briefly explained in Section II-B. The main body of the algorithm consists of a procedure that iterates between detection with graph signal filtering and erosion, which are, respectively, explained in Sections II-C and II-D.

A. Graph Signal Representation

Weighted graph representations are very convenient to conduct operations on the irregular structured data types, such as LiDAR point clouds. In general, weights $W$ of the graph, $G(V,E,W)$, store the distances between neighbor points, namely the vertices in $V$ that are connected by the edges in $E$. In this letter, the neighborhood topology of the graph is determined by a $k$-nearest neighbor (k-NN) structure and the weights are computed by a Gaussian function acting upon the distance between the neighboring points. For instance, let $d(i,j)$ be the distance between vertex $i$ and vertex $j$, then, the corresponding index of the weight matrix, $W_{ij}$, can be expressed as

$$W_{ij} = \begin{cases} \exp \left( -\frac{d(i,j)^2}{2\sigma^2} \right), & \text{if } (i,j) \in E \\ 0, & \text{otherwise} \end{cases}$$

In our LiDAR filtering algorithm, each point in the point cloud corresponds to a vertex in the graph representation, then we select the graph signal as the elevation value existing on each vertex, in other words, the $z$ readings of the $(x,y,z)$ 3-D point cloud. Hence, the graph signal can be indicated by a vector $f \in \mathbb{R}^N$, where $N$ is the size of the point cloud.

At this point, we can introduce the graph Laplacian matrix, $L$, which acts as a difference operator on the graph signal. The graph Laplacian is simply computed as $L = D - W$, where $D$ is the diagonal degree matrix whose elements are given by the sum of the weights for the associated vertices. Hence, $L$ is an $(N \times N)$ matrix. By multiplying $L$ with the graph signal vector, $f$, one can measure how the graph signal differs at a vertex with respect to its neighbors. This formulation can be described for vertex $i$ as follows:

$$(Lf)(i) = \sum_{j \in N_i} W_{ij}(f(i) - f(j))$$

where $N_i$ is the set of vertices in the neighborhood of vertex $i$. The graph signal can be analyzed in the spectral domain through the eigendecomposition of graph Laplacian [17]. The eigenvectors, $\{e_i\}_{i=0,1,...,N-1}$, form a Fourier basis, whereas the eigenvalues, $0 = \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_{N-1}$, carry a notion of frequency. The graph Fourier transform is then defined as follows:

$$\hat{f}(\lambda_i) = \sum_{i=1}^{N} f(i)e_i^*(i)$$

where $\hat{f}(\lambda)$ is the spectral domain representation of the graph signal $f$. One can then realize filtering of a signal $f_{in}$ with
a given graph spectral kernel $\hat{g}$, by applying that kernel to the Laplacian matrix, which results in the following filtering operation:

$$ f_{out} = \hat{g}(L) f_{in}. \quad (4) $$

The details about graph signal filtering operation can be examined in [14]. Efficient implementation of this operation is also possible through the polynomial approximation of graph kernels [18].

B. Outliers Removal Using the Laplacian Operator

Outliers mislead many filtering approaches and have significant effects on the classification result. For instance, the low-altitude outliers in LiDAR signals, which are also referred as “negative blunders”, may be confusing for many grid-based algorithms initializing their filtering procedure by picking the lowest points in a grid cell as the ground seed point [1]. Moreover, they may create problems for the morphological filtering methods due to their inclination to label the neighboring points of a low-altitude point as OBJ, which is prompted by the erosion behavior [19]. The outliers can be described as local isolated points on the elevation data. Therefore, we can simply employ the graph Laplacian as a difference operator on the graph signal, as in (2), so that the outliers are detected as the local maxima and minima of the Laplacian of the elevation signal. The graph topology and the weight matrix are constructed based on the 3-D Euclidean distances between the points. The goal of this representation is to discover the points that are significantly different from their neighbors in terms of the 3-D geometry of the point cloud. However, the points on the edges of the objects are not conceived as outliers, since the graph structure is incorporated with the elevation signal. This makes the graph structure be aware of the edge information; hence, the difference operation is suppressed at the edge points. Ultimately, the detected outliers are excluded from the iterative filtering process as in [13] and [19] and generally labeled as nonground points as stated in [9] and [19].

C. Detection of Edge Points by High-Pass Filtering

In the graph signal filtering task, for detection, the OBJ points we adopt in a graph representation where the k-NN operation:

$$ f_{out} = \hat{g}(L) f_{in}. \quad (4) $$

The details about graph signal filtering operation can be examined in [14]. Efficient implementation of this operation is also possible through the polynomial approximation of graph kernels [18].

B. Outliers Removal Using the Laplacian Operator

Outliers mislead many filtering approaches and have significant effects on the classification result. For instance, the low-altitude outliers in LiDAR signals, which are also referred as “negative blunders”, may be confusing for many grid-based algorithms initializing their filtering procedure by picking the lowest points in a grid cell as the ground seed point [1]. Moreover, they may create problems for the morphological filtering methods due to their inclination to label the neighboring points of a low-altitude point as OBJ, which is prompted by the erosion behavior [19]. The outliers can be described as local isolated points on the elevation data. Therefore, we can simply employ the graph Laplacian as a difference operator on the graph signal, as in (2), so that the outliers are detected as the local maxima and minima of the Laplacian of the elevation signal. The graph topology and the weight matrix are constructed based on the 3-D Euclidean distances between the points. The goal of this representation is to discover the points that are significantly different from their neighbors in terms of the 3-D geometry of the point cloud. However, the points on the edges of the objects are not conceived as outliers, since the graph structure is incorporated with the elevation signal. This makes the graph structure be aware of the edge information; hence, the difference operation is suppressed at the edge points. Ultimately, the detected outliers are excluded from the iterative filtering process as in [13] and [19] and generally labeled as nonground points as stated in [9] and [19].

C. Detection of Edge Points by High-Pass Filtering

In the graph signal filtering task, for detection, the OBJ points we adopt in a graph representation where the k-NN topology of the graph and the weight matrix are constructed based on 2-D Euclidean distances that are calculated in terms of the $(x, y)$ coordinates of the points. Then, in order to filter the graph signal, we employ a Mexican-hat kernel [20]

$$ \hat{g}(\lambda) = \lambda \exp(-t\lambda). \quad (5) $$

On this graph kernel, we can adjust parameter $t$ to generate a high-pass filter, since a high-pass filter designed in this graph representation has a large response on abrupt elevation changes between neighboring ground and nonground points, such as the ones on the boundaries of the man-made structures as well as highly scattered points, such as the ones on vegetation. Hence, applying a proper threshold on the filter output directly reveals these points. At the end of one filtering procedure, we can detect the edge points that lie along the outlines of the typically man-made objects and most of the vegetation points, where the filter response is above threshold. Then, we continue with an iterative filtering process to detect the object points inside the object outlines and the remaining points on vegetation.

D. Erosion of the Edge Points

At the end of each filtering process, the erosion of the detected edge points leads the inner OBJ points to be entitled as edge points in the next iteration. The erosion procedure is accomplished by pulling the elevation of the detected edge points to the level of local minima points within their neighborhood. To give an illustration, in Fig. 1(a), the detected edge points are shown in red points and the local minima of the filter output are shown in blue points. The eroded point cloud is displayed in Fig. 1(b), where the elevation value of the red points is changed to the one of their neighboring blue points.

In the proposed algorithm, the value of the threshold applied at the filter output is an important parameter in terms of capturing different OBJ points. In general, the jump edges existing at the outline of the buildings respond higher than the scattered vegetation points, which do not constitute as sharp elevation changes as the boundary points of the buildings do. Hence, the points on the trees or shrubs require a lower threshold to be detected; in addition, they need a small number of iterations to be captured as a whole. For instance, in Fig. 1(b), a substantial amount of the vegetation points are reached at once, whereas the edge points on the houses are obtained solely on the boundaries at one filtering step. Coupled with the fact that the buildings are extracted in each iteration by beginning from their boundaries toward their center, the large objects might require a larger number of iterations. In hierarchical morphological approaches, the problem of conforming different sizes for objects is addressed by increasing progressively the size of the structuring element [6], [7]. Likewise, our algorithm is optimized by raising the detection threshold in each iteration. Thus, we initiate the iterative filtering procedure with a low threshold to remove the vegetation points in the first iterations. On the next iterations, to keep the erosion of the man-made objects, we increase the threshold linearly on a range that is determined with respect to the scene characteristics.

Generally speaking, the rural scenes do not contain buildings; hence, they require a small number of iterations where the threshold grows within a medium narrow range. On the contrary, the urban scenes contain various sizes of man-made objects as well as vegetation; therefore, a broader range of threshold changing from low to high values over a large number of iterations is suitable for them.

III. Evaluation of the Proposed Method

We conduct experiments on the data set published by the ISPRS Working Group III/3 [16]. It includes 15 samples from seven test sites, each of which exhibits different types of features for challenging the various approaches in the LiDAR data analysis as summarized in Table I. For the assessment of the proposed method, several metrics have been used. Type I and Type II errors on the classification results are demonstrated in blue and red points, respectively, on the top view of the point cloud samples in Fig. 3. Type I errors stand for the BE
TABLE I
CHARACTERISTICS OF THE TEST SAMPLES

<table>
<thead>
<tr>
<th>Environment</th>
<th>Sample</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Urban</td>
<td>Samp11</td>
<td>Mixed vegetation and buildings on hillside</td>
</tr>
<tr>
<td></td>
<td>Samp12</td>
<td>Mixed vegetation and buildings</td>
</tr>
<tr>
<td></td>
<td>Samp21</td>
<td>Road with bridge</td>
</tr>
<tr>
<td></td>
<td>Samp22</td>
<td>Bridge and terrain discontinuity</td>
</tr>
<tr>
<td></td>
<td>Samp23</td>
<td>Complex buildings, terrain discontinuity</td>
</tr>
<tr>
<td></td>
<td>Samp24</td>
<td>Ramp and terrain discontinuity</td>
</tr>
<tr>
<td></td>
<td>Samp31</td>
<td>Complex building and negative blunders</td>
</tr>
<tr>
<td></td>
<td>Samp41</td>
<td>Large gaps and outliers in data, courtyard</td>
</tr>
<tr>
<td></td>
<td>Samp42</td>
<td>Trains in railway yard</td>
</tr>
<tr>
<td>Rural</td>
<td>Samp51</td>
<td>Data gaps, vegetation on moderate slopes</td>
</tr>
<tr>
<td></td>
<td>Samp52</td>
<td>Sharp ridges on the terrain</td>
</tr>
<tr>
<td></td>
<td>Samp53</td>
<td>Steep, terraced slopes</td>
</tr>
<tr>
<td></td>
<td>Samp54</td>
<td>Negative blunders, low resolution buildings</td>
</tr>
<tr>
<td></td>
<td>Samp61</td>
<td>Data gap and embankments</td>
</tr>
<tr>
<td></td>
<td>Samp71</td>
<td>Ramp, bridge and underpass</td>
</tr>
</tbody>
</table>

TABLE II
COMPARISONS BASED ON TOTAL ERROR (T.E) AND KAPPA (κ)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T.E(%) κ(%)</td>
<td>T.E(%) κ(%)</td>
<td>T.E(%) κ(%)</td>
</tr>
<tr>
<td>Samp1</td>
<td>13.36 72.42</td>
<td>10.76 78.48</td>
<td>8.64 82.40</td>
</tr>
<tr>
<td>Samp12</td>
<td>4.35 91.28</td>
<td>3.25 93.51</td>
<td>3.10 93.80</td>
</tr>
<tr>
<td>Samp21</td>
<td>0.89 97.38</td>
<td>4.25 86.34</td>
<td>1.88 94.43</td>
</tr>
<tr>
<td>Samp22</td>
<td>6.65 84.31</td>
<td>3.63 91.33</td>
<td>3.40 92.07</td>
</tr>
<tr>
<td>Samp23</td>
<td>11.21 77.52</td>
<td>4.00 91.97</td>
<td>6.48 87.02</td>
</tr>
<tr>
<td>Samp24</td>
<td>7.27 82.05</td>
<td>4.42 88.50</td>
<td>4.19 89.49</td>
</tr>
<tr>
<td>Samp31</td>
<td>2.63 94.72</td>
<td>4.78 90.43</td>
<td>2.48 95.00</td>
</tr>
<tr>
<td>Samp41</td>
<td>8.49 83.02</td>
<td>3.91 72.21</td>
<td>10.79 78.41</td>
</tr>
<tr>
<td>Samp51</td>
<td>2.13 94.91</td>
<td>1.62 96.15</td>
<td>2.03 93.07</td>
</tr>
<tr>
<td>Samp52</td>
<td>5.42 71.24</td>
<td>2.72 91.68</td>
<td>3.00 90.74</td>
</tr>
<tr>
<td>Samp53</td>
<td>5.23 50.32</td>
<td>8.91 39.13</td>
<td>7.41 47.24</td>
</tr>
<tr>
<td>Samp54</td>
<td>3.31 93.36</td>
<td>3.23 93.52</td>
<td>3.67 92.65</td>
</tr>
<tr>
<td>Samp61</td>
<td>0.99 85.30</td>
<td>2.08 74.52</td>
<td>2.02 75.38</td>
</tr>
<tr>
<td>Samp71</td>
<td>1.90 90.56</td>
<td>1.63 91.44</td>
<td>1.85 90.52</td>
</tr>
</tbody>
</table>

points that are erroneously classified as OBJ points, whereas Type II errors denote the OBJ points that are erroneously classified as BE points. In addition, the total error rate and the kappa value [21] are calculated for each sample in the data set and listed in Table II. The total error rate is found by dividing the sum of wrong detections by the total number of points in the sample. The Cohen’s kappa coefficient is a common statistical measure that evaluates the accuracy of the overall classification performance. The results of the proposed algorithm are compared with those of the progressive TIN densification (PTD) method by Axelsson [12], as it directly works on the original point cloud, and it is the best performing method among the original eight filtering algorithms tested on the ISPRS data set [7]. We also compare our algorithm with the simple morphological filter (SMRF) by Pingel et al. [7], which is one of the primary morphology-based methods proposed in recent years.

The k-NN topology in our algorithm is generated by selecting $k = 30$ on the 3-D geometry during the outlier removal stage, whereas, in the graph signal filtering stage, we select $k = 9$ based on the 2-D neighborhood distance. In the both stages, $\sigma$ parameter in the weight function (1) is determined as the mean neighbor distance.

Through the experiments, we argue the performance of our method by comparing with the other algorithms, based on some challenging features, such as data gaps; then, we show its strength on some rural samples. Beginning with the data gaps, the proposed method outperforms the competitors in Samp41 and Samp61, since it directly acts upon the original data points in contrast to the grid fitting algorithms. Another advantage of analyzing the original data is smaller vulnerability to the point density, which can be seen in the rural samples by the superior performance of the proposed method as well as the PTD method over the SMRF method. In some of the rural samples, such as Samp53 and Samp61, our algorithm performed better than the competitors, since the filtering procedure is concluded in a few iterations as there is no building in the scene. This consequently prevents the erosion of terrain discontinuities and keep the Type I error as small as possible.

The discontinuity in slope and elevation leads to nonground object labeling due to the design assumption of our algorithm. Hence, the terrain elements, such as terraced grounds, cliffs, sharp ridges, and hill cutoff edges, present delusive features as for many other ground filtering methods [22]. On the other hand, as long as the scene contains a continuous terrain, our algorithm performs fairly well by handling some challenging features, such as vegetation on steep terrain as in Samp51 or bridges as in Samp21. Bridges are not easy to detect in general, since a part of the object is smoothly connected to the ground [1]. During the experiments, a few filtering iterations
are usually sufficient to retrieve these thin objects before the erosion spreads on the ground connections. For example, the bridges in Samp71 and Samp22 are extracted successfully as nonground objects, although the underpass in Samp71 and the road continuing under the bridge in Samp22 raises the
Type I error result on these samples, since they create discontinuity on the ground.

Another key thing to remember is that the proposed method erodes the nonground objects, such as buildings, by beginning from their outline, yet it does not rely on the shape of their contours. Thus, it operates successfully on the courtyard on Samp41 and irregular shaped building on Samp31, which are counted as complex shapes.

A difficult scenario emerges on the urban samples where large buildings and discontinuous ground are present at the same time, such as the terraced terrain in Samp11 and Samp23. Large buildings require many iterations, which may cause wrong erosion to expand on terrain discontinuities. Another scene with buildings is apparent on Samp12, where our algorithm manages to classify the high nonground objects, the buildings, as well as some low objects, such as shrubs and fences, which are difficult to detect. On the other hand, this sample also contains some echelons on the ground between the buildings, which are detected as nonground as well, since they have about the same height with shrubs. Finally, on the samples Samp12, Samp31, and Samp54, the negative blunders and small pits on the terrain might increase the number of points around them to be detected as nonground in an iterative filtering procedure. We overcome the problem due to the negative blunders with the outlier removal step.

All things considered, the terrain discontinuity factor is the major issue on our filtering experiments. Nevertheless, this is an expected result as almost all the filtering algorithms are affected [3] and develop special precautions in the algorithm [2] or postprocessing steps [23] to overcome this problem. Likewise, in this letter, such terrain discontinuities require special attention, which will be the focus of our further studies.

IV. Conclusion

In this letter, we present a novel LiDAR data filtering algorithm which analyzes the 3-D aerial point cloud on its original domain by casting it as an elevation signal living on a weighted graph representation. Our framework starts with an outlier removal step, which eliminates the complications caused by the negative blunders and erroneous points. Then, we detect the object points on the scene through the graph signal filtering and thresholding steps. In order to remove the nonground objects completely, we adopt an iterative procedure consisting of an erosion step following the detection step. The graph signal filtering approach captures the nonground objects, such as bridges and vegetation, successfully. Then, it can reach good accuracy values on the rural samples, which demonstrates the potential of spectral graph techniques for the analysis of LiDAR data. Our further work will concentrate on improving the graph signal filtering framework to compensate the discontinuous terrain features by leveraging its capability of processing the original data points.

References