Unsupervised Weighed Graph-Based Terrain Filtering Using Terrestrial Laser Scanning Data

Y. Belkhouche

1Department of Computing Sciences, Texas A&M University-Corpus Christi, Corpus Christi, Texas, USA

Abstract—In this paper we establish a simple method for filtering terrestrial laser scanning data. The filtering problem is modeled as graph connected components problem. The proposed method uses weighted k-nearest neighbor graph(knn), it consists of four steps: construction of the knn graph, edge weigh assignment, edge thresholding and finding the graph connected components. In the first step, a knn graph is build using the 3D point cloud. In the second step, we introduce a new scheme for labeling the edges of knn graph. Thresholding is then used to remove edges with certain weights from the knn graph, this leads to disconnecting terrain points from object points. In the last step, a graph connected component algorithm is used to find the connected components and classify the points into either terrain points or object points. We validate our method using visual observation as well as a manually labeled datasets.

Keywords: Terrestrial laser scanning, Filtering, K-nearest neighbor, Graph connected components, object extraction, digital elevation model.

1. Introduction and literature review

Recent laser scanning instruments collect a very high density 3D point cloud describing the details of the surrounding environment. The collected data are used in many military and civilian applications. Visibility analysis [1], [2] plays an important role for battle field management, in this application the laser scanning data are used to determine what can be seen from a given location, this information is used later to determine strategic locations, positioning of troupes and provide situation awareness. Training soldiers in virtual environments [3] is another military application, where 3D city models are build using laser scanning data of the area of interest, this model is used as virtual environment to train soldiers. Urban planing [4], is another application where laser scanning data are analyzed to determine flood areas, emergency response routes, locations for placing communication antennas, and damage assessment after a disaster event. Infrastructures inspection [5], [6] such as roads, bridges, power lines, rail roads and tunnels are other applications that uses laser scanning data. Other applications that benefit greatly from using laser scanning data are 3D scene reconstruction/understanding, and autonomous robot/car navigation [7], [8], [9], in autonomous robot/car navigation, the 3D point cloud are used to build a 3D map of the environment. This map is used to determine a safe space for navigation by autonomous vehicles. It is important to mention that the 3D LiDAR data cannot be used directly in these applications, the data need to be preprocessed and mined in order to become useful. An important preprocessing task is to separate terrain points and object points, this process is known as filtering. After completing the filtering process, several products are generated using terrain points, among these products are digital elevation models (DEMs) which are used in wide range of applications. Occupancy grid maps used for autonomous vehicles navigation is another product generated using terrain points. On the other hand, object points are further processed and used to detect/classify objects into several categories such as walls, people, street sign, bicycles, cars, trees, ... etc, detecting these objects plays a very important role for 3D scene interpretation and understanding research (for indoors and outdoors scenes). Several terrain filtering(some times called digital elevation model (DEM) extraction) algorithms have been introduced in the past, however the vast majority of these methods are designed and implemented to filter airborne collected LiDAR data (the laser scanner is attached to an airplane). These algorithms do not incorporate the characteristics of terrestrial laser scanning data in the filtering process (terrestrial scanners can obtain laser returns from walls, these returns are missing in airborne collected data, on the other side, airborne scanners can obtain returns from buildings roofs, these returns are missing in the terrestrial collected data), these differences make the adopt these algorithms for filtering terrestrial LiDAR without major modifications a difficult task. Some of the most popular airborne LiDAR filtering methods were introduced in the flowing research papers [10], [11], [12], [13], [14], [15], [16]. Filtering algorithms have been classified into several categories. Segmentation/clustering based methods use a number of selected features to capture the smoothness of the terrain surface (these methods assume that the terrain surface is smooth) which is used to extract terrain points. Other segmentation-based methods use region growing algorithms, these methods require an initialization of the terrain area, the points that are similar (using a certain similarity measures) to the initialized area are added to the terrain points. Morphology-based methods use mathematical morphology operators (opening-closing and dilation-erosion) to remove non-terrain objects, leaving only terrain points. Irregular
The terrestrial scanning filtering problem

The filtering problem can be described as follows: Given a collected terrestrial scanning point cloud $P_{scan}$, the objective of the filtering process is to assign each 3D point in the dataset to either terrain point or object point, such that:

- $P_{Terrain}$: is the set of terrain points.
- $P_{Object}$: is the set of object points.
- $P_{Terrain} \cup P_{Object} = P_{scan}$
- $P_{Terrain} \cap P_{Object} = \emptyset$

These constraints mean that no point is considered as both terrain and object at the same time. Combining the terrain points and the object points give us the initial dataset. To illustrate the filtering process, consider the 3D point cloud shown in figure 1, these dataset is collected from an urban site, it consists of 79,000 points, and contains small and large buildings, several trees, bushes, and cars. Figure 1(a) shows the raw 3D data before filtering. Figure 1(b) shows an interpolated image of the of this area, in which 3D point cloud is converted into a grid representation, the height (z-value) of each point is converted into gray value (the darkest pixel value represents lower z-values, while the lightest pixel value represents high z-value). The filtered data is shown in figure 1(c), where the red points are terrain points and the blue points are object points.

There are many challenges that must be addressed when developing a filtering algorithms [32]. The most important challenges are the followings:

- Point spatial density: In a low density datasets, only few points are sampled from the surface. This means that many surfaces are not well represented, this leads to the failure of may filtering algorithms. The point spatial density may vary depending on the surface reflectivity (depending on the object material), occlusion, the distance from the surface to the laser instrument (the closer to the scanner the higher the spatial density), as well as the scanner orientation (the density of the areas in the front of the scanner is higher than those area scanned with an angle).

- Scene complexity: Urban scenes are usually complex scenes, they contain several objects with different characteristics. Some of these objects are man-made such are cars, traffic signs, buildings, ... etc. Other are natural objects such trees, bushes, ... etc. These objects vary in sizes, locations, and pose.

- Large size: Most laser scanning instruments collects up one million points per second, the collected data must processed in a very short time in order to be useful for real time applications such as autonomous robots/cars navigation. Real-time processing of LiDAR data is one of the most important challenges, it requires the selection of an appropriate data structure to represent the 3D point cloud as well as the design and implementation of simple and efficient algorithms to process these large datasets.

- Irregularity of LiDAR data: which make it difficult to use image processing algorithms directly on these data. LiDAR data can be interpolated into a gray level image, however, this process leads to losing important
Filtering algorithms may be biased toward terrain or toward object. Depending on the algorithm parameters, it could perform better on correctly identifying terrain points or object points. Some algorithms perform a very high accuracy extracting object points while others achieve better accuracy extracting terrain points.

3. Methodology

The proposed method consists of several steps. It starts by constructing the \( k^{\text{th}} \)-nearest neighbor (\( k^{\text{n}} \text{nn} \)) graph. In the second step, we introduce an edge weighting scheme, this scheme allows us to assign weights to each edge in the \( k^{\text{nn}} \) graph. The next step is thresholding, where we remove all edges smaller than a specified threshold from the \( k^{\text{nn}} \) graph leading to the creation of a new graph. A graph connected components algorithm is used to extract the connected components in the new graph. In the last step, each connected component is classified as either terrain or object. Figure 2 below shows the workflow of the proposed method.

3.1 Constructing and weighing the \( k^{\text{nn}} \) graph

A graph is represented as \( G(E, V) \) where \( V \) is the set of vertices and \( E \) is the set of edges connecting these vertices. The proposed filtering method uses the \( k^{\text{nn}} \) graph to represent the 3D scanning data. In this paper, we constructed the \( k^{\text{nn}} \) graph in 2D, in which the \( z \)-coordinate of each point is ignored (each point is projected onto the \( xy \)-plane). The initial \( k^{\text{nn}} \) graph is built using the \((x, y)\) coordinates of each point, the Euclidean distance as a similarity metric to decide which points are adjacent to each other. The nearest neighbor criterion described in equation (1) is used to find the first nearest neighbor.

\[
\text{NN}(p_i) = \{ p_j : \min_{p_j \in V} d(p_i, p_j) \}
\]

To build the \( k^{\text{nn}} \) graph, equation (1) is applied \( k \) times excluding the previously found nearest neighbor point. After constructing the \( k^{\text{nn}} \) graph, we assigned weights to the edges of this graph as follows:

\[
w_{ij} = e^{-|s_{ij}|}
\]

where \( w_{ij} \) is the weight of the edge connecting the points \( p_i \) and \( p_j \), \( s_{ij} \) is the slope between the points \( p_i \) and \( p_j \), it is computed as follows:

\[
s_{ij} = \frac{z_i - z_j}{\| p_i - p_j \|}
\]

where \( z_i \) and \( z_j \) are the \( z \)-values for the points \( p_i \) and \( p_j \) respectively. \( \| p_i - p_j \| \) is the distance between the point \( p_i \) and \( p_j \), it is computed using only the \( x \) and \( y \) coordinates. The weight function in equation (2) maps large slopes to small values. The weight values are in the interval \([0, 1]\). The edges with value close to 0 are most likely to be located in areas connecting terrain points to object points, while edges with weight values close to 1 lay in terrain areas (relatively flat areas).

To illustrate this weighting scheme, consider the example shown in figure 3. This example consists of four areas with different characteristics. Figure 3(a) shows an area containing three cars and other very small objects. Figure
3(b) shows a large building and some small objects on the side of the building. Figure 3(c) shows an area containing two trees near each other. Figure 3(d) shows a road segment.

For each area we visualize the weighed $knn$ graph using two colors: the red edges lay in the areas connecting terrain points to object points (these edges have a weight smaller than certain threshold, in this example these edges have weights $w_{ij} \leq 0.2$). The blue edges are found in relatively flat areas (terrain or flat building roofs).

In order to analyze the characteristics of the study areas, we used the weight histogram. Figure 4 below shows the weight histograms for the four areas in figure 3. Even though these histograms are for areas with different characteristics, there are some similarities between them. The left side of each histogram represent the edges with low weights, these edge are located in areas separating the terrain and object points (terrain discontinuity). The right side of the histogram represent edges laying in relatively flat areas areas (terrain, buildings with flat roofs), these edges gives an indication of the flatness of the surface. All weight histograms of areas containing object points, will have high values on the left side (figures 4(a), 4(b), 4(c)). Figure 4(d) shows the weight histogram for a road segment, notice that there are no edges connecting terrain to object points (this is the main reason that the left side of the histogram is empty). These histograms are very useful in the selection of the threshold used to eliminate the the edges connecting terrain points and object points.

### 3.2 Finding the connected components

In this paper, the filtering problem is modeled as graph connected components problem. Given a graph $G(V, E)$ a connected component of this graph is subgraph where there is at least one path between each two vertices belonging to that subgraph, and no path from any vertex of the subgraph to the any vertex outside that subgraph. The key idea behind the proposed method is to generate a graph containing several connected components, this is achieved by disconnecting object points from terrain points. In order to do that we need remove all edges connecting terrain and object components (these are the red edges shown in figure 3) from the $knn$ graph. These edges are the ones represented on the left side of the histograms in figure 4. Removing these edges from the $knn$ graph leads to creating new graph with several components. A threshold value is selected and all edges smaller than that value are removed from the initial $knn$ graph. After removing these edges, a connected components algorithm is applied to the new graph to find all connected components. To illustrate this concept, consider the area shown in figure 3(b), The connected components are shown using different colors (figure 5). There is one large component (points in blue color) representing the terrain points and may other small components representing object points.

### 3.3 Filtering

In this step, the connected components detected in the previous step are classified as either terrain components or object components. The largest component is classified as terrain, which represents the terrain points. The main reason for selecting the large component as terrain is mainly because there is no discontinuities in the points sampled from the terrain surface, and the change between neighboring
points is usually smooth, leading to one large connected component. All other components are classified as object components.

4. Experimental Results

To validate the efficiency of the proposed method, we conducted two experiments. In the first experiment, we used visual observation of the filtered area to validate the results. In the second experiment, we used a manually labeled dataset to assess the accuracy of our filtering method. The measures used to evaluate the filtering results are the total accuracy, the object accuracy, and the terrain accuracy. These measures are shown below:

<table>
<thead>
<tr>
<th>Table 1: Confusion-matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classified</td>
</tr>
<tr>
<td>Terrain</td>
</tr>
<tr>
<td>Reference</td>
</tr>
<tr>
<td>Object</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2: Accuracy measures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
</tr>
<tr>
<td>Total accuracy</td>
</tr>
<tr>
<td>Object accuracy</td>
</tr>
<tr>
<td>Terrain accuracy</td>
</tr>
</tbody>
</table>

Where:
- a: is the number of correctly classified terrain points.
- b: is the number of miss classified terrain points.
- c: is the number of miss classified object points.
- d: is the number of correctly classified object points.
- Object accuracy: the accuracy of correctly classified object points.
- Terrain accuracy: the accuracy of correctly classified terrain points.
- Total: is the total accuracy of correctly classified points.

The proposed method requires two parameters, the k value used to construct the knn graph, and threshold value used to remove certain edges from the knn graph. In order to gain an insight to how the k and the threshold values affect accuracy of our filtering method, we designed an experiment in which we changed the values of k and the threshold and calculate the accuracy. Figure 6 shows how the accuracy change when changing the parameters. This experiment demonstrate that the highest accuracy is obtained with higher k values and low threshold values.

| Fig. 6: The relationship between k, edge threshold and the accuracy. |

4.1 Experiment 1:

In this experiment we used a dataset collected using Velodyne laser scanner. The scanned area is an urban street scene, it contains building walls, cars, trees, power lines, power posts and may other objects. The area is 168 m by 168 m, it contains 3,200,000 points. The filtering results for this area is shown in figure 7. The red point are terrain points while the blue are object points.
4.2 Experiment 2:

In this experiment, we used the LiDAR dataset provided by [33], [34]. These datasets were collected in urban areas using Velodyne LiDAR scanner. Each dataset is a single scan collected form a rotating scanner, with no overlap between the scans. These datasets were manually labeled and contain several objects with different characteristics. Table 3 below summarizes the characteristics of each dataset.

Table 4 summarizes the result of our experiments. The table contains the k value and the threshold value used to filter each dataset. It also contains the total accuracy, the object accuracy, and the terrain accuracy. The table shows that on most datasets the total accuracy was more than a 95%.

5. Conclusion and future work

We established a simple and efficient method for filtering terrestrial laser scanning data. We modeled the filtering process as a graph connected components problem. We proposed a graph edge weighing scheme that allows the characterization of edges connecting terrain and object points. The removal of these edges led to disconnecting objects from the terrain surface. Experimental results demonstrate the effectiveness of our method. Ten manually labeled datasets were used, the proposed method achieved more than 95% accuracy on all the datasets used in the experiments. In future we will consider developing an automatic technique for selecting the value of k used for constructing kNN graph as well as selecting the threshold value used for disconnecting the graph.

References

Table 3: Characteristics of study areas

<table>
<thead>
<tr>
<th>#Points</th>
<th>Buildings</th>
<th>Trees</th>
<th>Fences</th>
<th>Humans</th>
<th>cars</th>
<th>street signs</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area 1</td>
<td>65554</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Area 2</td>
<td>65432</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Area 3</td>
<td>63363</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Area 4</td>
<td>63858</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Area 5</td>
<td>65398</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Area 6</td>
<td>66044</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Area 7</td>
<td>63783</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Area 8</td>
<td>66416</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Area 9</td>
<td>65023</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Area 10</td>
<td>67085</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 4: Experimental Results

<table>
<thead>
<tr>
<th>k</th>
<th>Threshold</th>
<th>Total Accuracy</th>
<th>Terrain Accuracy</th>
<th>Object Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area 1</td>
<td>51</td>
<td>0.07</td>
<td>97.38</td>
<td>96.49</td>
</tr>
<tr>
<td>Area 2</td>
<td>46</td>
<td>0.06</td>
<td>97.20</td>
<td>95.95</td>
</tr>
<tr>
<td>Area 3</td>
<td>70</td>
<td>0.03</td>
<td>91.34</td>
<td>93.77</td>
</tr>
<tr>
<td>Area 4</td>
<td>45</td>
<td>0.05</td>
<td>96.51</td>
<td>90.52</td>
</tr>
<tr>
<td>Area 5</td>
<td>45</td>
<td>0.08</td>
<td>96.63</td>
<td>95.64</td>
</tr>
<tr>
<td>Area 6</td>
<td>70</td>
<td>0.1</td>
<td>91.84</td>
<td>97.31</td>
</tr>
<tr>
<td>Area 7</td>
<td>40</td>
<td>0.06</td>
<td>96.46</td>
<td>94.71</td>
</tr>
<tr>
<td>Area 8</td>
<td>50</td>
<td>0.04</td>
<td>96.61</td>
<td>90.70</td>
</tr>
<tr>
<td>Area 9</td>
<td>45</td>
<td>0.06</td>
<td>94.46</td>
<td>91.75</td>
</tr>
<tr>
<td>Area 10</td>
<td>80</td>
<td>0.08</td>
<td>97.00</td>
<td>90.14</td>
</tr>
</tbody>
</table>