remote sensing

MDPI

Article

An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation

Wuming Zhang [1], Jianbo Qi [1,*], Peng Wan [1], Hongtao Wang [2], Donghui Xie [1], Xiaoyan Wang [1] and Guangjian Yan [1]

[1] State Key Laboratory of Remote Sensing Science, Beijing Key Laboratory of Environmental Remote Sensing and Digital City, School of Geography, Beijing Normal University, Beijing 100875, China; wumingz@bnu.edu.cn (W.Z.); wanpeng@mail.bnu.edu.cn (P.W.); xiedonghui@bnu.edu.cn (D.X.); xinxin1594@aliyun.com (X.W.); gjyan@bnu.edu.cn (G.Y.)

[2] School of Surveying and Land Information Engineering, Henan Polytechnic University, Jiaozuo 454003, China; wht_31@hpu.edu.cn

* Correspondence: qijb@mail.bnu.edu.cn; Tel.: +86-10-5880-9246

Academic Editors: Jie Shan, Juha Hyyppä, Lars T. Waser and Prasad S. Thenkabail
Received: 13 March 2016; Accepted: 3 June 2016; Published: 15 June 2016

Abstract: Separating point clouds into ground and non-ground measurements is an essential step to generate digital terrain models (DTMs) from airborne LiDAR (light detection and ranging) data. However, most filtering algorithms need to carefully set up a number of complicated parameters to achieve high accuracy. In this paper, we present a new filtering method which only needs a few easy-to-set integer and Boolean parameters. Within the proposed approach, a LiDAR point cloud is inverted, and then a rigid cloth is used to cover the inverted surface. By analyzing the interactions between the cloth nodes and the corresponding LiDAR points, the locations of the cloth nodes can be determined to generate an approximation of the ground surface. Finally, the ground points can be extracted from the LiDAR point cloud by comparing the original LiDAR points and the generated surface. Benchmark datasets provided by ISPRS (International Society for Photogrammetry and Remote Sensing) working Group III/3 are used to validate the proposed filtering method, and the experimental results yield an average total error of 4.58%, which is comparable with most of the state-of-the-art filtering algorithms. The proposed easy-to-use filtering method may help the users without much experience to use LiDAR data and related technology in their own applications more easily.

Keywords: LiDAR point cloud; ground filtering algorithm; cloth simulation

1. Introduction

High-resolution digital terrain models (DTMs) are critical for flood simulation, landslide monitoring, road design, land-cover classification, and forest management [1]. Light detection and ranging (LiDAR) technology, which is an efficient way to collect three-dimensional point clouds over a large area, has been widely used to produce DTMs. To generate DTMs, ground and non-ground measurements have to be separated from the LiDAR point clouds, which is a filtering process. Consequently, various types of filtering algorithms have been proposed to automatically extract ground points from LiDAR point clouds. However, developing an automatic and easy-to-use filtering algorithm that is universally applicable for various landscapes is still a challenge.

Many ground filtering algorithms have been proposed during previous decades, and these filtering methods can be mainly categorized as slope-based methods, mathematical morphology-based methods, and surface-based methods. The common assumption of slope-based algorithms is that the change in the slope of terrain is usually gradual in a neighborhood, while the change in slope between buildings or trees and the ground is very large. Based on this assumption,

Vosselman [2] developed a slope-based filtering algorithm by comparing slopes between a LiDAR point and its neighbors. To improve the calculation efficiency, Shan and Aparajithan [3] calculated the slopes between neighbor points along a scan line in a specified direction, which was extended to multidirectional scan lines by Meng *et al.* [4]. Acquiring an optimal slope threshold that can be applied to terrain with different topographic features is difficult with these methods. To overcome this limitation, various automatic threshold definitions have been studied, e.g., adaptive filters [5,6] and dual-direction filters [7]. Nonetheless, their results suggested that slope-based algorithms were not guaranteed to function well in complex terrain, as the filtering accuracy decreased with increasingly steeper slopes [8].

Another type of filtering method uses mathematical morphology to remove non-ground LiDAR points. Selecting an optimal window size is critical for these filtering methods [9]. A small window size can efficiently filter out small objects but preserve larger buildings in ground points. On the other hand, a large window size tends to smooth terrain details such as mountain peaks, ridges and cliffs. To solve this problem, Zhang *et al.* [10] developed a progressive morphological filter to remove non-ground measurements by comparing the elevation differences of original and morphologically opened surfaces with increasing window sizes. However, poor ground extraction results may occur because the terrain slope is assumed to be a constant value in the whole processing area. To overcome this constant slope constraint, Chen *et al.* [11] extended this algorithm by defining a set of tunable parameters to describe the local terrain topography. Other improved algorithms that are based on mathematical morphology can be found in [12–15]. The advantage of mathematical morphology-based methods is that they are conceptually simple and can be easily implemented. The accuracy of morphological based approach is also relatively good. However, additional priori knowledge of the study area is usually required to define a suitable window size because local operators are used [16].

Previous algorithms separated ground and non-ground measurements by removing non-ground points from LiDAR datasets. In contrast to these algorithms, surface-based methods gradually approximate the ground surface by iteratively selecting ground measurements from the original dataset, and the core of this type of filtering method is to create a surface that approximates the bare earth. Axelsson [17] proposed an adaptive triangulated irregular network (TIN) filtering algorithm that gradually densified a sparse TIN that was generated from the selected seed points. In this algorithm, two important threshold parameters need to be carefully set: one is the distance of a candidate point to the TIN facet, and the other is the angle between the TIN facet and the line that connects the candidate point with the facet's closest vertex. These parameters are constant values for the entire study area in the adaptive TIN filter, which makes it difficult to detect ground points around break lines and steep terrain. Recently, Zhang and Lin [18] improved this algorithm by embedding smoothness-constrained segmentation to handle surfaces with discontinuities. Another typical surface-based filtering algorithm was developed by Kraus and Pfeifer [19], who used a weighted linear least-squares interpolation to identify ground points from LiDAR data. This algorithm was first used to remove tree measurements and generate DTMs in forest areas and was subsequently extended to process LiDAR points in urban areas by incorporating a hierarchical approach [20]. By using this filtering algorithm, ground measurements can be successfully detected on flat terrain, but the filtering results are less reliable on terrain with steep slopes and large variability. To cope with this problem, multi-resolution hierarchical filtering methods have been proposed to identify ground LiDAR points based on point residuals from a thin plate spline-interpolated surface [16,21,22]. Recently, Hui *et al.* [23] proposed an improved filtering algorithm which combines the traditional morphological filtering algorithm and multi-level interpolation filtering algorithm. It can achieve promising results in both of urban areas and rural areas.

Another special surface-based filtering algorithm was proposed by Elmqvist [24], who employed active shape models to approximate the ground surface. In this algorithm, an energy function is designed as a weighted combination of internal forces from the shape of the contour and external

forces from the LiDAR point clouds. Minimizing this energy function determines the ground surface, which behaves like a membrane that sticks to the lowest points. This algorithm is a new idea to model ground surface from LiDAR data, but the optimization only achieve an global optimum solution, which does not guarantee to get all the local optimums. Thus, some local details may be ignored. Meanwhile, this algorithm performs relatively poorly in complex areas as reported in [25]. They may also fail to effectively model terrain with steep slopes and large variability because they are based on the assumption that the terrain is a smooth surface. Furthermore, another challenge of these methods is how to increase the efficiency when the accuracy is fixed [18].

The use of the aforementioned filtering algorithms has proven to be successful, but the performance of these algorithms changes according to the topographic features of the area, and the filtering results are usually unreliable in complex cityscapes and very steep areas. In addition, the implementation of these filtering methods requires a number of suitable parameters to achieve satisfactory results, which are difficult to determine because the optimal filter parameters vary from landscape to landscape, so these filtering methods are not easy to use by users without much experience. To cope with these problems, this paper proposes a novel filtering algorithm which is capable of approximating the ground surface with a few parameters. Different from other algorithms, the proposed method filters the ground points by simulating a physical process that an virtual cloth drops down to an inverted (upside-down) point cloud. Compared to existing filtering algorithms, the proposed filtering method has some advantages: (1) few parameters are used in the proposed algorithm, and these parameters are easy to understand and set; (2) the proposed algorithm can be applied to various landscapes without determining elaborate filtering parameters; and (3) this method works on raw LiDAR data.

The remainder of this paper is organized as follows. A new ground filtering algorithm is proposed in Section 2. Section 3 presents the experimental results, and the proposed algorithm is discussed in Section 4. Finally, Section 5 concludes this paper.

2. Method

Our method is based on the simulation of a simple physical process. Imagine a piece of cloth is placed above a terrain, and then this cloth drops because of gravity. Assuming that the cloth is soft enough to stick to the surface, the final shape of the cloth is the DSM (digital surface model). However, if the terrain is firstly turned upside down and the cloth is defined with rigidness, then the final shape of the cloth is the DTM. To simulate this physical process, we employ a technique that is called cloth simulation [26]. Based on this technique, we developed our cloth simulation filtering (CSF) algorithm to extract ground points from LiDAR points. The overview of the proposed algorithm is illustrated in Figure 1. First, the original point cloud is turned upside down, and then a cloth drops to the inverted surface from above. By analyzing the interactions between the nodes of the cloth and the corresponding LiDAR points, the final shape of the cloth can be determined and used as a base to classify the original points into ground and non-ground parts.

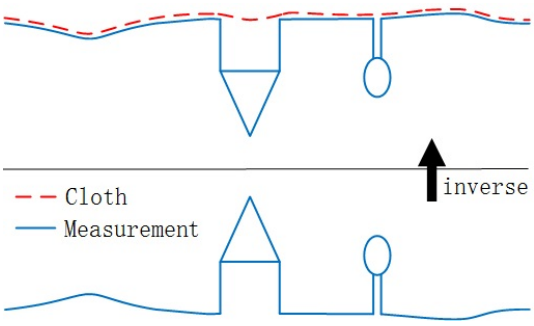

Figure 1. Overview of the cloth simulation algorithm.

2.1. Fundamental of the Cloth Simulation

Cloth simulation is a term of 3D computer graphics. It is also called cloth modeling, which is used for simulating cloth within a computer program. During cloth simulation, the cloth can be modeled as a grid that consists of particles with mass and interconnections, called a Mass-Spring Model [27]. Figure 2 shows the structure of the grid model. A particle on the node of the grid has no size but is assigned with a constant mass. The positions of the particles in three-dimensional space determine the position and shape of the cloth. In this model, the interconnection between particles is modeled as a "virtual spring", which connects two particles and obeys Hooke's law. To fully describe the characteristics of the cloth, three types of springs have been defined: shear spring, traction spring and flexion spring. A detailed description about the functions of these different springs can be found in [27].

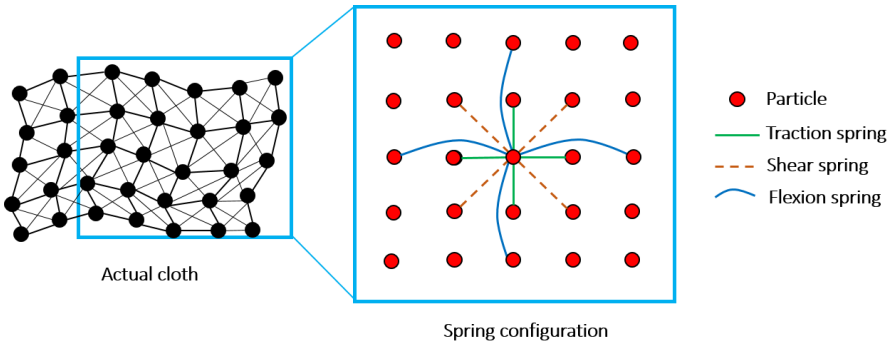

Figure 2. Schematic illustration of mass-spring model. Each circle indicates a particle and each line represents a spring.

To simulate the shape of the cloth at a specific time, the positions of all of the particles in the 3D space are computed. The position and velocity of a particle are determined by the forces that act upon it. According to Newton's second law, the relationship between position and forces is determined by Equation (1):

$$m\frac{\partial X(t)}{\partial t^2} = F_{ext}(X,t) + F_{int}(X,t) \quad (1)$$

where X means the position of a particle at time t; $F_{ext}(X,t)$ stands for the external force, which consists of gravity and collision forces that are produced by obstacles when a particle meets some objects in the direction of its movement; and $F_{int}(X,t)$ stands for the internal forces (produced by interconnections) of a particle at position X and time t. Because both the internal and external forces vary with time t, Equation (1) is usually solved by a numerical integration (e.g., Euler method) in the conventional implementation of cloth simulation.

2.2. Modification of the Cloth Simulation

When applying the cloth simulation to LiDAR point filtering, a number of modifications have been made to make this algorithm adaptable to point cloud filtering. First, the movement of a particle is constrained to be in vertical direction, so the collision detection can be implemented by comparing the height values of the particle and the terrain (e.g., when the position of a particle is below or equal to the terrain, the particle intersects with the terrain). Second, when a particle reaches the "right position", *i.e.*, the ground, this particle is set as unmovable. Third, the forces are divided into two discrete steps to achieve simplicity and relatively high performance. Usually, the position of a particle is determined by the net force of the external and internal forces. In this modified cloth simulation, we first compute the displacement of a particle from gravity (the particle is set as unmovable when it reaches the ground, so the collision force can be omitted) and then modify the position of this particle according to the internal forces. This process is illustrated in Figure 3.

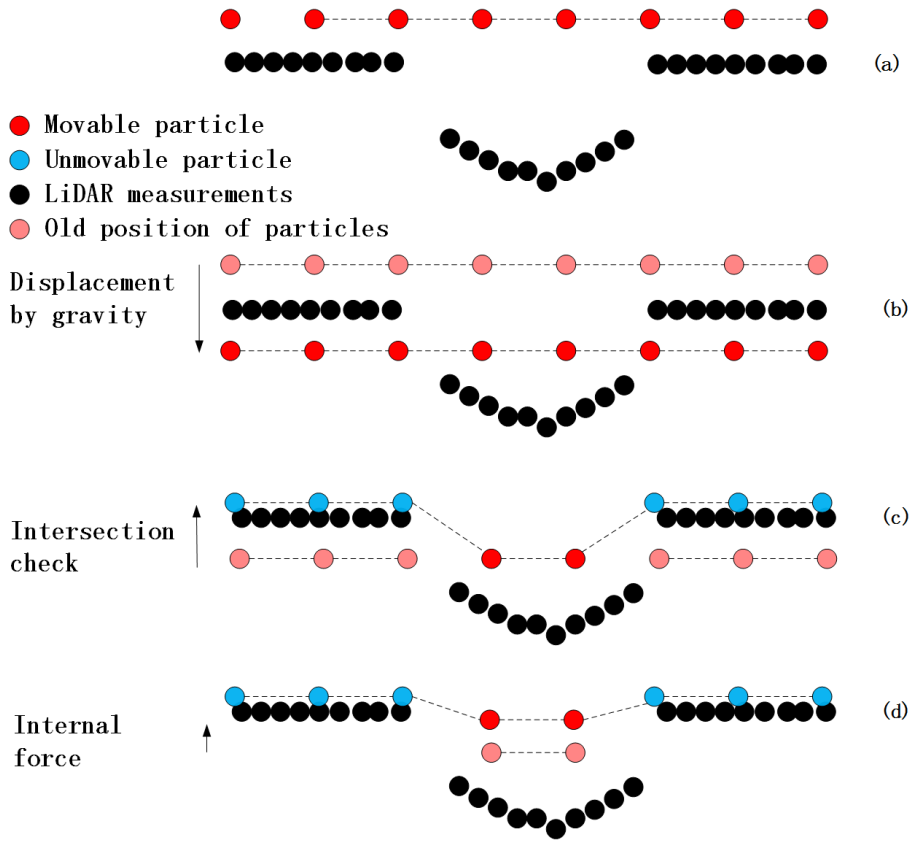

Figure 3. Main Steps in CSF: (**a**) Initial state. A cloth is place above the inverted LiDAR measurements; (**b**) The displacement of each particle is calculated under the influence of gravity. Thus, some particles may appear under the ground measurements; (**c**) Intersection check. For those who are under the ground, they are moved on the ground and set as unmovable; (**d**) Considering internal forces. The movable particles are moved according to forces produced by neighbour particles.

2.3. Implementation of CSF

As described above, the forces that act on a particle are considered as two discrete steps. This modification was inspired by [28]. First, we calculate the displacement of each particle only from gravity, *i.e.*, solve Equation (1) with internal forces equal to zero. Then, the explicit integration form of this equation is

$$X(t + \Delta t) = 2X(t) - X(t - \Delta t) + \frac{G}{m}\Delta t^2 \tag{2}$$

where m is the mass of the particle (usually, m is set to 1) and Δt is the time step. This equation is very simple to solve. Given the time step and initial position, the current position can be calculated directly because G is a constant.

To constrain the displacement of particles in the void areas of the inverted surface, we consider the internal forces at the second step after the particles have been moved by gravity. Because of internal forces, particles will try to stay in the grid and return to the initial position. Instead of considering neighbors of each particle one by one, we simply traverse all the springs. For each spring, we compare the height difference between the two particles which form this spring. Thus, the 2-dimensional (2-D) problem are abstracted as a one-dimensional (1-D) problem, which is illustrated in Figure 4.

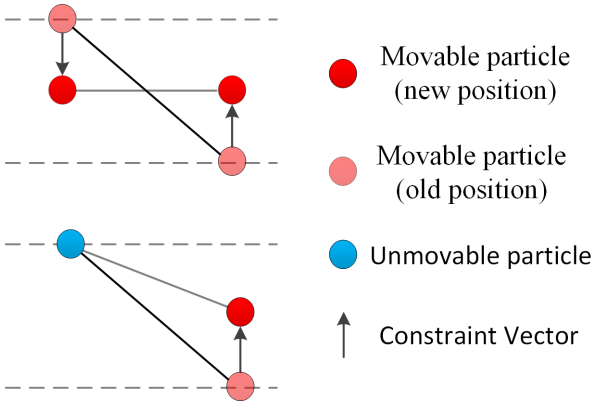

Figure 4. Constraint between particles.

As we have restricted the movement directions of the particles, two particles with different height values will try to move to the same horizon plane (cloth grid is horizontally placed at the beginning). If both connected particles are movable, we move them by the same amount in the opposite direction. If one of them is unmovable, then the other will be moved. Otherwise, if these two particles have the same height value, neither of them will be moved. Thus, the displacement (vector) of each particle can be calculated by the following equation:

$$\overrightarrow{d} = \frac{1}{2} b(\overrightarrow{p_i} - \overrightarrow{p_0}) \cdot \overrightarrow{n} \tag{3}$$

where $\overrightarrow{d}$ represents the displacement vector of a particle; b equals to 1 when the particle is movable, otherwise it equals to 0. $\overrightarrow{p_0}$ is the position of current particle that is ready to be moved. $\overrightarrow{p_i}$ is the position neighboring particle that connects with p_0; and $\overrightarrow{n}$ is a normalized vector that points to vertical direction, $\overrightarrow{n} = (0,0,1)^T$. This movement process can be repeated; we set a parameter rigidness (RI) to represent the repeated times. This parameterization process is shown in Figure 5. If RI is set to 1, the movable particle is just moved only once, and the displacement is half of the vertical distance (VD) between the two particles. If the RI is set to 2, the movable particle will be moved twice, the total displacement is 3/4VD. Finally, if RI is set to 3, the movable particle will be moved three times and the total displacement is 7/8VD. The value of 3 is enough to produce a very hard cloth. Thus, we constrain the rigidness to values of 1, 2 and 3. The larger the rigidness is, the more rigidly the cloth will behave.

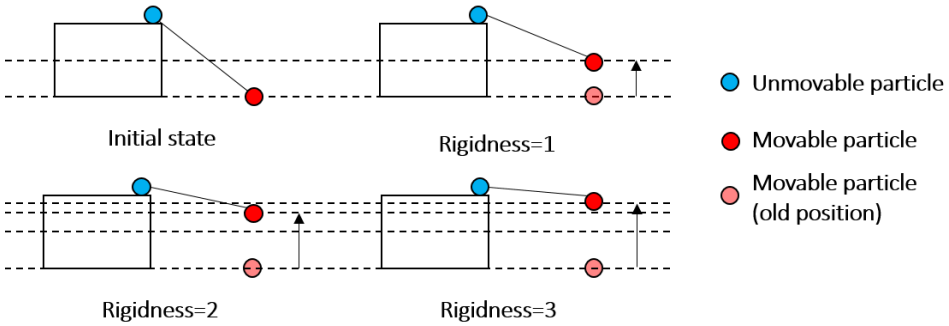

Figure 5. Parameterization of rigidness.

The main implementation procedures of CSF are described as follows. First, we project the cloth particles and LiDAR points into the same horizontal plane and then find a nearest LiDAR point (named corresponding point, CP) for each cloth particle in this 2D plane. An intersection height value (IHV) is defined to record the height value (before projection) of CP. This value represents the lowest

position that a particle can reach (*i.e.*, if the particle reaches the lowest position that is defined by this value, it cannot move forward anymore). During each iteration, we compare the current height value (CHV) of a particle with IHV; if CHV is equal or lower than IHV, we move the particle back to the position of IHV and make the particle unmovable.

An approximation of the real terrain is obtained after the simulation, and then the distances between the original LiDAR points and simulated particles are calculated by using a cloud-to-cloud distance computation algorithm [29]. LiDAR points with distances that are less than a threshold h_{cc} are classified as BE (bare earth), while the remaining points are OBJ (objects).

The procedure of the proposed filtering algorithm is presented as follows:

1. Automatic or manual outliers handling using some third party software (such as cloudcompare).
2. Inverting the original LiDAR point cloud.
3. Initiating cloth grid. Determining number of particles according to the user defined grid resolution (GR). The initial position of cloth is usually set above the highest point.
4. Projecting all the LiDAR points and grid particles to a horizontal plane and finding the CP for each grid particle in this plane. Then recording the IHV.
5. For each grid particle, calculating the position affected by gravity if this particle is movable, and comparing the height of this cloth particle with IHV. If the height of particle is equal to or less than IHV, then this particle is placed at the height of IHV and is set as "unmovable".
6. For each grid particle, calculating the displacement of each particle affected by internal forces.
7. Repeating (5)–(6). The simulation process will terminate when the maximum height variation (M_HV) of all particles is small enough or when it exceeds the maximum iteration number which is specified by the user.
8. Computing the cloud to cloud distance between the grid particles and LiDAR point cloud.
9. Differentiating ground from non-ground points. For each LiDAR points, if the distance to the simulated particles is smaller than h_{cc}, this point is classified as BE, otherwise it is classified as OBJ.

2.4. Post-Processing

For steep slopes, this algorithm may yield relatively large errors because the simulated cloth is above the steep slopes and does not fit with the ground measurements very well due to the internal constraints among particles, which is illustrated in Figure 6. Some ground measurements around steep slopes are mistakenly classified as OBJ. This problem can be solved by a post-processing method that smoothes the margins of steep slopes. This post-processing method finds an unmovable particle in the four adjacent neighborhoods of each movable particle and compares the height values of CPs. If the height difference is within a threshold (h_{cp}), the movable particle is moved to the ground and set as unmovable. For example, for point D in Figure 6, we find that point A is the unmovable particle from the four adjacent neighbors of D. Then, we compare the height values between C and B (the CPs for D and A, respectively). If the height difference is less than h_{cp}, then this candidate point D is moved to C and is set as unmovable. We repeat this procedure until all the movable particles are properly handled (either set as unmovable or kept movable).

To implement post-processing, all the movable particles should be traversed, if we scan the cloth grid row by row, the results may be affected by this particular scan direction. Thus, we first build up sets of strongly connected components (SCCs) and each SCC contains a set of connected movable particles. In a SCC, it usually contains two kinds of particles, those particles that have at least one unmovable neighbor are marked as type M1, and the others are marked as M2 (see Figure 7). Using M1 as initial seeds, we perform the breath-first traversal for the SCC, with which movable particles are handled one by one from 1 to 18 in Figure 7. This process guarantees that the post-processing are performed from edge to center, regardless of scan direction.

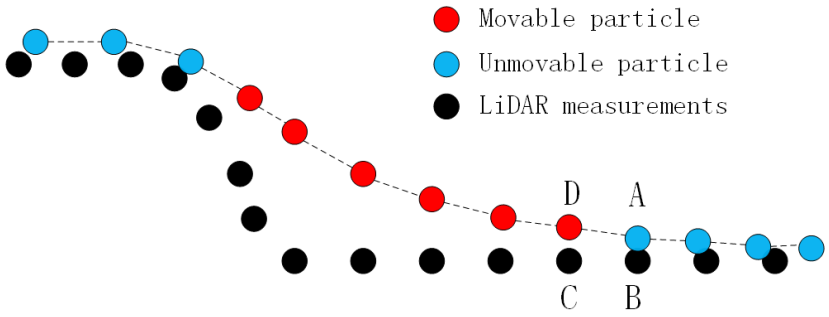

Figure 6. Post-processing of the steep slope.

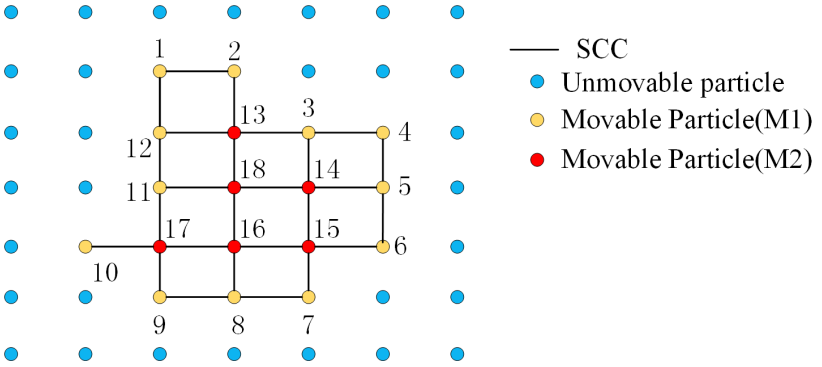

Figure 7. Illustration of strongly connected component (SCC). Movable particles are handled from 1 to 18.

2.5. Parameters

CSF mainly consists of four user-defined parameters: grid resolution (GR), which represents the horizontal distance between two neighboring particles; time step (dT), which controls the displacement of particles from gravity during each iteration; rigidness (RI), which controls the rigidness of the cloth; and an optional parameter steep slope fit factor (ST), which indicates whether the post-processing of handling steep slopes is required or not.

In addition to these user-defined parameters, two threshold parameters have been used in this algorithm to aid the identification of ground points. The first is a distance threshold (h_{cc}) that governs the final classification of the LiDAR points as BE and OBJ based on the distances to the cloth grid. This parameter is set as a fixed value of 0.5 m. Another threshold parameter is the height difference (h_{cp}), which is used during post-processing to determine whether a movable particle should be moved to the ground or not. This parameter is set to 0.3 m for all of the datasets.

3. Experiment and Results

3.1. Validation of the Filtering

This method was first tested by datasets that were provided by the International Society for Photogrammetry and Remote Sensing (ISPRS) Working Group III/3 to quantitatively test the performance of different filters and identify directions for future research [30]. In these datasets, fifteen samples with different characteristics were selected to test the performance of the proposed CSF algorithm, which are shown in Table 1. The reference datasets were generated by manually filtering the LiDAR datasets, and each point in the samples was classified as BE or OBJ.

Table 1. Characteristics of all samples [15].

Environment	Site	Sample	Features
Urban	1	11	Mixture of vegetation and buildings on hillside
		12	Buildings on hillside
	2	21	Large buildings and bridge
		22	Irregularly shaped buildings
		23	Large, irregularly shaped buildings
		24	Steep slopes
	3	31	Complex buildings
	4	41	Data gaps
		42	Railway station with trains
Rural	5	51	Mixture of vegetation and buildings on hillside
		52	Buildings on hillside
		53	Large buildings and bridge
		54	Irregularly shaped buildings
	6	61	Large, irregularly shaped buildings
	7	71	Steep slopes

According to the implementation of CSF, when the original LiDAR point cloud is turned upside down, the objects above ground will appear below the ground measurements. Then, the complexity of object measurements (such as rooftops) seldom influences the simulation process. Based on this feature, we visually classified the samples into different groups according to the properties of the topography. These properties indicate the existence of steep slopes or terraced slopes. If the terrain is very flat and has no steep or terraced slopes, RI is set to a relatively large value ($RI = 3$), and no post-processing is needed ($ST = false$). If steep slopes exist (e.g., river bank, ditch, and terrace), a medium soft cloth ($RI = 2$) and post-processing ($ST = true$) are needed. When handling very steep slopes, we need post-processing ($ST = true$) and a very soft cloth ($RI = 1$). Thus, the fifteen samples are classified into three groups, each sharing the same set of parameters, which are illustrated in Table 2.

Table 2. Parameters for each group of samples (dT = 0.65, GR = 0.5).

Group	Feature	Parameters	Samples
I	Flat terrain or gentle slope, no steep slopes	RI = 3 ST = false	21, 31, 42, 51, 54
II	With steep or terraced slopes (e.g., river bank, ditch, terrace)	RI = 2 ST = true	11, 12, 22, 23, 24, 41
III	High and steep slopes (e.g., pit, cliff)	RI = 1 ST = true	52, 53, 61, 71

The main parameters that control the results and vary with the scene type were RI and ST, which reduce the complexity and improve the usability of this algorithm. For dT and GR, we set them as fixed values of 0.65 and 0.5, respectively. These two values are universally applicable to all of the reference datasets according to our tests. The influence of these two parameters on the results is discussed in Section 4.2.

To evaluate the performance of this algorithm, the type I (T.I), type II (T.II) and total errors (T.E.) for all the fifteen samples were calculated. The type I error is the number of BE points that are incorrectly classified as OBJ divided by the true number of BE points; the type II error is the number of OBJ points that that incorrectly classified as BE points divided by the true number of OBJ points, and

the total error is the number of mistakenly classified points divided by the total points. Besides, the Cohen's Kappa coefficient [31], which measures the overall agreement between two judges [15,21], is also calculated in this study. The calculations of T.I, T.II, T.E. and Kappa coefficient can refer to Hu *et al.* [32].

The errors and the Kappa coefficients are shown in Table 3. The results show that CSF has relatively good results for samp21, samp31, samp42, samp51 and samp54 respecting total error and kappa coefficient. All these samples belong to group I, which indicates that the most suitable types for our method is urban areas. For group II and group III, the total errors are relatively large (especially for samp11) compared to group I, which shows that CSF performs relatively poorly in complex regions similar to other filtering algorithms [21]. Overall, our method is not sensitive to the type and distribution of object above ground because the LiDAR point cloud is inverted and the shape of the terrain mostly determines the filtering accuracy. However, in high relief areas with very steep slopes (e.g., pit, cliff) and low rise buildings, our method perform worst, because when a soft cloth fit with the terrain, it may also reach the rooftops of low rise buildings.

Table 3. Errors and Kappa coefficients for all samples.

Samples	T.I(%)	T.II(%)	T.E.(%)	Kappa(%)
samp11	7.23	18.44	12.01	75.17
samp12	1.15	4.9	2.97	94.04
samp21	3.89	1.78	3.42	90.47
samp22	1.29	25.9	8.94	77.72
samp23	3.52	6.21	4.79	90.38
samp24	1.03	7.73	2.87	92.68
samp31	0.96	2.38	1.61	96.75
samp41	1.48	8.78	5.14	89.73
samp42	3.28	0.87	1.58	96.18
samp51	2.67	4.57	3.08	91.13
samp52	1.01	28.79	3.93	77.05
samp53	3.85	37.08	5.2	46.86
samp54	3.79	2.64	3.18	93.61
samp61	0.87	18.94	1.49	78.1
samp71	1.61	37.85	5.71	68.03

Figure 8 shows the original dataset, reference DTM, produced DTM and the space distribution of Type I and Type II errors of some representative samples (samp31, samp11 and samp51). Compared with the reference DTM, the produced DTM has successfully preserved the main terrain shape and also the microtopography, especially in mine field (Figure 8k). It can also be seen that the error points mainly exist on the edges of objects for samp11 and samp31, in which some object measurements with low height values may be classified as BE and some ground measurements with relatively high height values may also be classified as OBJ. samp11 has a large group of error points (type II) that almost classifies a whole building as ground measurements, it occurs because the building is located on a slope and the roof is nearly connected to the ground, causing the building to be treated as ground in the post-processing step. For samp53 (the same as samp52 and samp61), the very soft cloth (group III) makes some small vegetation points mistakenly classified as BE. Besides, the total number of OBJ points is much smaller compared to BE points. These jointly cause a large

Type II error. However, if a harder cloth is used it may yield the opposite error (BE points around the steep slopes may be identified as OBJ). Thus, adjusting the parameters is necessary to balance type I and type II errors. If a large number of low objects exists above the ground, the cloth should be harder (RI should be set larger), which will guarantee that fewer object measurements are mistakenly classified as ground objects. Among the samples which have very poor Type II error, samp22 and samp71 are caused by the incorrectly identification of bridges, which is classified as BE under CSF. The details will be discussed in Section 4.4.

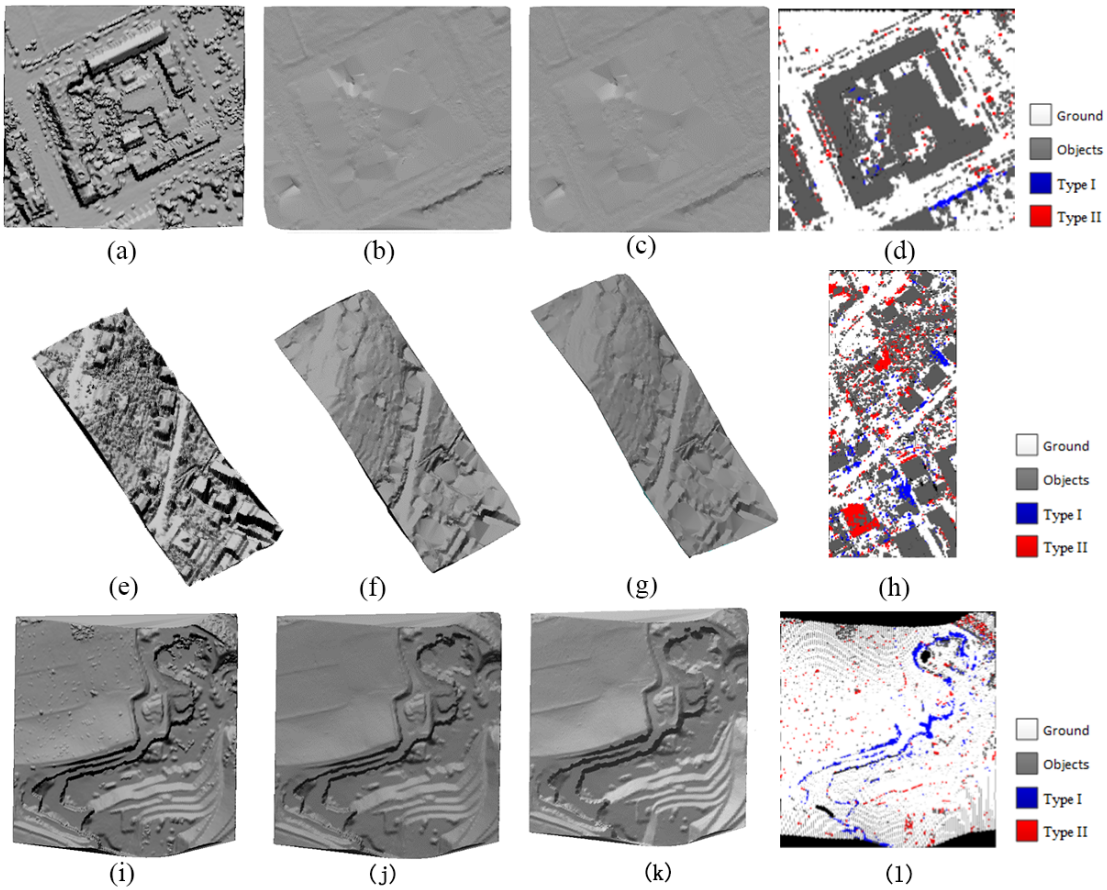

Figure 8. Results of each group (choose samp31, samp11, and samp53 as representatives): (**the first column**) are original datasets; (**the second column**) are the DTMs that are generated from the reference data of samp31, samp11, and samp53; (**the third column**) are the DTMs that are produced from the CSF algorithm; (**the last column**) are the spatial distributions of the type I and type II errors.

To quantitatively analyze the accuracy of the CSF algorithm, we compared the total error and kappa coefficient with some existing top algorithms. The total errors and Kappa coefficients of these algorithms and our algorithm are shown in Tables 4 and 5. On the whole, the accuracy of our method is close to some top filtering algorithms, except for samples 22 and 71. The total errors for sample 22 and 71 are relatively high, which are also mainly caused by the bridge.

Table 4. Total error compared to other reported algorithms (%).

Samples	Axelsson (1999)	Elmqvist (2000)	Pfeifer (2001)	Mongus (2012)	Li (2013)	Chen (2013)	Pingel (2013)	Zhang (2013)	Hu (2014)	Mongus (2014)	Hui (2016)	CSF
samp11	10.76	22.4	17.35	11.01	12.85	13.01	8.28	18.49	8.31	7.5	13.34	12.01
samp12	3.25	8.18	4.5	5.17	3.74	3.38	2.92	5.92	2.58	2.55	3.5	2.97
samp21	4.25	8.53	2.57	1.98	2.55	1.34	1.1	4.95	0.95	1.23	2.21	3.42
samp22	3.63	8.93	6.71	6.56	4.06	4.67	3.35	14.18	3.23	2.83	5.41	8.94
samp23	4.00	12.28	8.22	5.83	6.16	5.24	4.61	12.06	4.42	4.34	5.11	4.79
samp24	4.42	13.83	8.64	7.98	5.67	6.29	3.52	20.26	3.80	3.58	7.47	2.87
samp31	4.78	5.34	1.8	3.34	2.47	1.11	0.91	2.32	0.90	0.97	1.33	1.61
samp41	13.91	8.76	10.75	3.71	6.71	5.58	5.91	20.44	5.91	3.18	10.6	5.14
samp42	1.62	3.68	2.64	5.72	3.06	1.72	1.48	3.94	0.73	1.35	1.92	1.58
samp51	2.72	21.31	3.71	2.59	3.92	1.64	1.43	5.31	2.04	2.73	4.88	3.08
samp52	3.07	57.95	19.64	7.11	15.43	4.18	3.82	12.98	2.52	3.11	6.56	3.93
samp53	8.91	48.45	12.6	8.52	11.71	7.29	2.43	5.58	2.74	2.19	7.47	5.2
samp54	3.23	21.26	5.47	6.73	3.93	3.09	2.27	6.4	2.35	2.16	4.16	3.18
samp61	2.08	35.87	6.91	4.85	5.81	1.81	0.86	16.13	0.84	0.96	2.33	1.49
samp71	1.63	34.22	8.85	3.14	4.58	1.33	1.65	10.44	1.50	2.49	3.73	5.71
Avg.	4.82	20.73	8.02	5.62	6.18	4.11	2.97	10.63	2.85	2.74	5.33	4.39
Std.	3.44	15.92	5.09	2.39	3.84	3.06	2.00	6.01	2.03	1.64	3.23	2.76

Table 5. Kappa coefficient compared to other reported algorithms (%).

Samples	Axelsson (1999)	Elmqvist (2000)	Pfeifer (2001)	Chen (2013)	Pingel (2013)	Hu (2014)	Hui (2016)	CSF
samp11	78.48	56.68	66.09	74.12	83.12	82.97	72.92	75.17
samp12	93.51	83.66	91	93.23	94.15	94.83	93.00	94.04
samp21	86.34	77.4	92.51	96.1	96.77	97.23	93.35	90.47
samp22	91.33	80.3	84.68	89.03	92.21	92.04	87.58	77.72
samp23	91.97	75.59	83.59	89.49	90.73	91.14	89.74	90.38
samp24	88.5	54.13	78.43	84.53	91.13	90.39	81.93	92.68
samp31	90.43	89.31	96.37	97.76	98.17	98.19	97.33	96.75
samp41	72.21	82.46	78.51	88.83	88.18	88.18	78.78	89.73
samp42	96.15	90.86	93.67	95.81	96.48	98.25	95.38	96.18
samp51	91.68	52.74	89.61	95.17	95.76	93.9	85.06	91.13
samp52	83.63	9.36	41.02	78.91	81.04	86.24	69.51	77.05
samp53	39.13	7.05	30.83	46.69	68.12	66.43	41.84	46.86
samp54	93.52	55.88	88.93	93.9	95.44	95.28	91.63	93.61
samp61	74.52	10.31	47.09	77.36	87.22	86.76	67.82	78.1
samp71	91.44	26.26	75.27	93.19	91.81	92.59	79.86	68.03
Avg.	84.19	56.8	75.84	86.27	90.02	90.29	81.72	83.86
Std.	13.9	29.18	19.87	12.72	7.58	7.74	13.95	13.12

3.2. Testing with Dense Point Cloud

The datasets from ISPRS were obtained many years ago, as the development of LiDAR technology, the density of collected point cloud are continuously arising. Thus, we tested the performance of CSF with datasets that have more dense points with average point distance equal to 0.6 m–0.8 m, the information of these datasets are shown in Table 6. By comparing with the true ground obtained by standard industrial semi-automatic software, the accuracy of CSF is evaluated. The results are shown in Table 7. It can be seen that CSF has achieved a relatively high accuracy for all the datasets from urban to rural areas. However, A large T.I error also has been noted for dataset 3, this is because ground measurements is very sparse in this area. For dataset 4, the error mainly occurs around steep slopes, since this area contains large number of steep slopes.

Table 6. Characteristics of testing datasets.

Dataset	Type	Point Number	Scope	Features
1	Urban	1559933	1 km × 1 km	Flat terrain, large and dense buildings, high vegetation coverage
2	Urban	1522256	1 km × 1 km	Flat terrain with dense bungalow areas
3	Rural	2093506	2 km × 1 km	dense vegetation coverage
4	Rural	1418228	0.5 km × 0.5 km	Large number of steep slopes

Table 7. Accuracy evaluation with true ground measurements.

Dataset	T.I(%)	T.II(%)	T.E.(%)
1	0.72	13.36	6.84
2	5.29	9.29	7.84
3	36.09	1.84	5.49
4	8.57	22.61	14.09

To analyse the details of CSF, we presented the generated DTM and the cross section of each dataset. Since CSF will first invert the original point cloud, thus large buildings will produce large holes. However, when applying a relative hard cloth, this hole can be crossed. Then large buildings can be removed (see Figure 9). If post-processing is enabled ($ST = true$), the cloth can fit the ground very well. Through this way, microtopography (e.g., low-lying areas) in urban areas can be preserved (Figure 10). In mountain areas, CSF performs relatively poorly, especially in dense vegetation areas where ground measurements are usually sparse. If the cloth is too soft, many object measurements may be mistakenly classified as BE. Otherwise, ground measurements may be classified as OBJ due to the hilly topography (see Figure 11). For areas with large number of steep slopes, cloth should be more soft and post-process is also needed. Figure 12 shows a typical area with large number of steep slopes, it can be seen that the main skeleton of terrain has been preserved well. However, some small houses may be missed (red circle in Figure 12c).

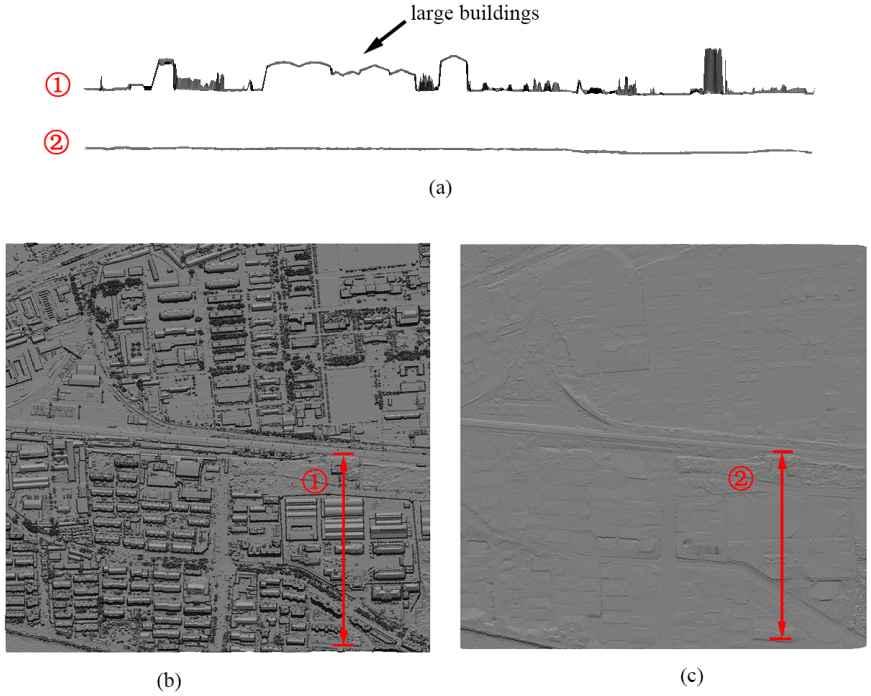

Figure 9. Removal of large buildings in urban area: (**a**) Cross sections from (**b**) and (**c**); (**b**) Dataset 1; (**c**) Produced DTM. In this dataset, it contains a number of connected large low buildings (see the cross section), when the cloth is relatively hard, it will not drop into this large hole, then these buildings can be removed.

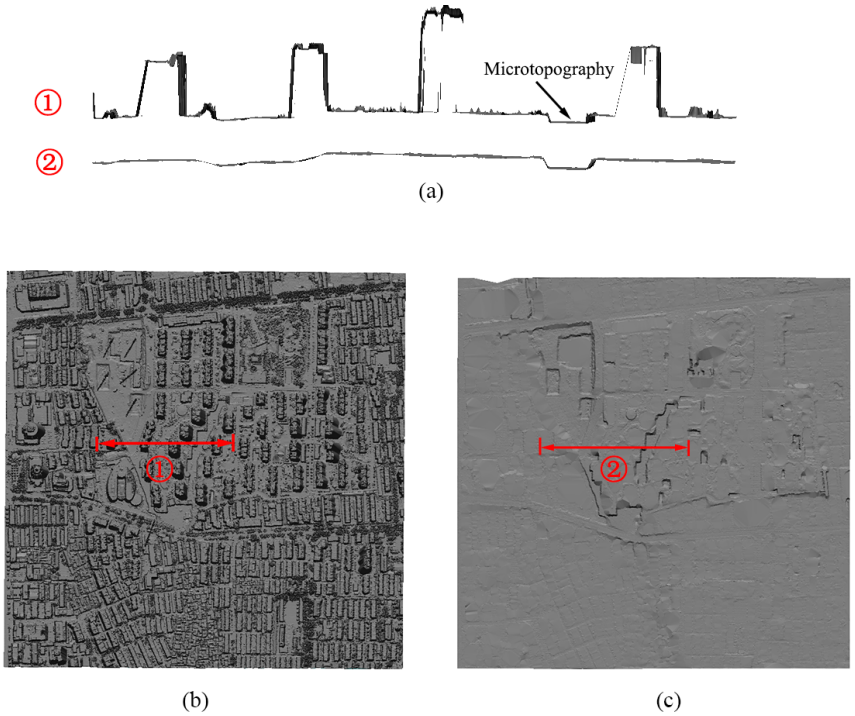

Figure 10. Preservation of microtopography: (**a**) Cross sections from (**b**) and (**c**); (**b**) Dataset 2; (**c**) Produced DTM. When post-processing is enabled, the cloth can stick to the surface more closely, some small steep slopes can be preserved.

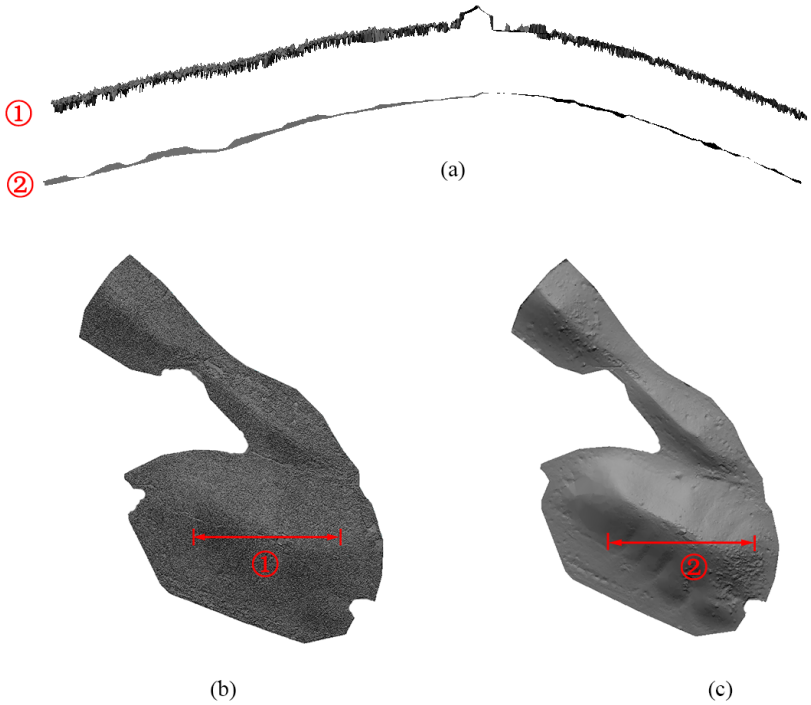

Figure 11. Sparse ground measurements: (**a**) Cross sections from (**b**) and (**c**); (**b**) Dataset 3; (**c**) Produced DTM. In some hilly topography areas, some parts of the cloth may not sticks to the ground well, this will cause classification errors (BE may be treated as OBJ).

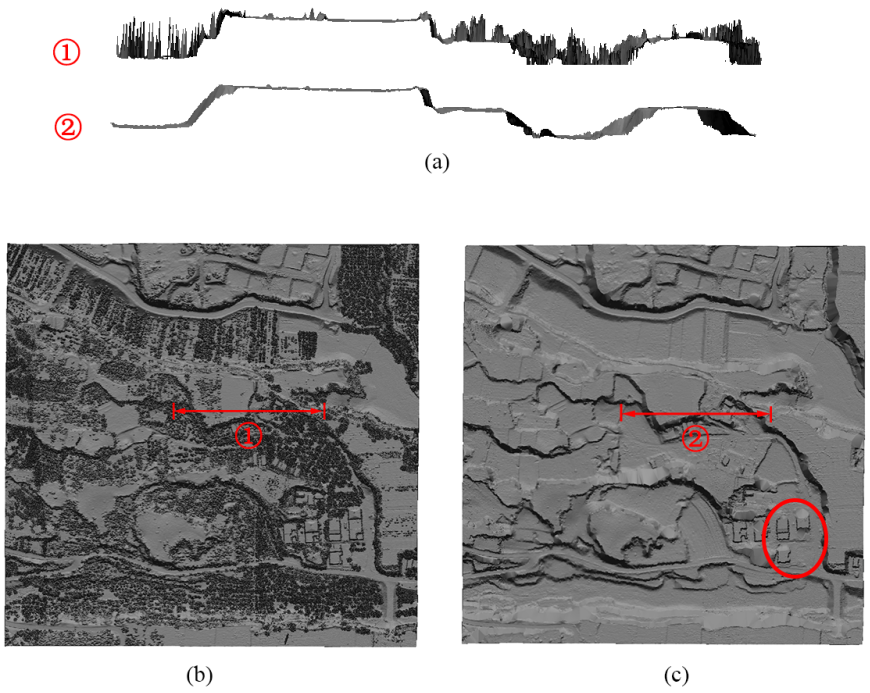

Figure 12. Preservation of steep slopes: (**a**) Cross sections from (**b**) and (**c**); (**b**) Dataset 4; (**c**) Produced DTM. In this dataset, the main objects are vegetation and contains large number of ground measurements, the cloth can be very soft to maximumly fit the terrain shape with less consideration of T.II error. Companied with post-processing, large steep slopes can be preserved well.

4. Discussion

4.1. Accuracy

Among all the reported algorithms, the Axelsson's algorithm had been implemented in a commercial software package called Terrasolid [33] and the Pfeifer's algorithm was implemented into a commercially available software package called SCOP++ from the German company Inpho GmbH [34]. The overall performance of our algorithm also showed high accuracy and stability, as both the mean total error (4.39) and standard deviation (2.76) of all the samples are relatively low compared to all of the other algorithms. This result is inspirational and demonstrates that our algorithm can be adapted to various environments and achieves relatively high accuracy.

4.2. Parameter Setting

Usually, we only modify RI and ST for different groups of samples and set dT and GR as fixed values. These universally applicable parameters (dT and GR) were determined by a number of tests. Theoretically, smaller dT value would make behavior of simulated cloth more like a true cloth, but it dramatically increases the computing time. To quantitatively evaluate the influence of dT, we tested all of the samples with different time steps (from 0.4 to 1.5 with steps of 0.05; 0.4 was chosen because the value would take too much time to compute when it was smaller than 0.4). The total errors of each group and the mean total error of all the samples, which depend on the time step, are illustrated in Figure 13. This figure indicates that the total error increases after an initial decline for all groups, and all of them achieve the lowest total error around the 0.65 time step. When dT is small, the displacement of particle in each step is also small. As we have set a maximum iteration number of 500, if dT is too small, the cloth may not reach the LiDAR measurements or fit with the terrain well after simulation process. Thus, very small dT may produce large error. On the other hand, When the dT is too large, the simulated cloth may stick to rooftops, this also increases the total error. Thus, 0.65 was chosen as the value of dT because it produces relatively good results for all of the samples and it can be applied to many situations without adjustments.

The grid resolution (GR) parameter in the simulation process has strong relationship with simulation time because it determines how many cloth particles are created for a specific dataset. Figure 14 shows the total errors at different GR values. It can be seen that the accuracies of group I and group III are relatively stable than group II because group II usually have complicated terrain shape and buildings (e.g., areas with terraced slopes and low rise buildings). However, almost all samples get the highest accuracy around 0.5, which was then been used as a fixed value.

Except the parameters above, there are two threshold parameters: h_{cc} and h_{cp}. h_{cc} governs the final classification which separates LiDAR measurements into BE or OBJ. Most particles will stick to ground after simulation. And OBJ measurements (e.g., buildings and trees) are usually taller than 0.5 m. Thus, we set h_{cc} as 0.5, which is also a fixed value. The influences of h_{cc} on total errors are illustrated in Figure 15. It shows that this value has limited impact on total errors. As for the parameter h_{cp}, it is used in the post-processing to decide whether a movable particle should be moved to ground according to its neighbors. We simply set this parameter to 0.3 m, which indicates the height difference between two adjacent ground measurements is usually less than 0.3 m on a flat terrain. Since this parameter is only used when post-process is enabled, and it also only influence the movable particles over steep slopes, the influence is also limited.

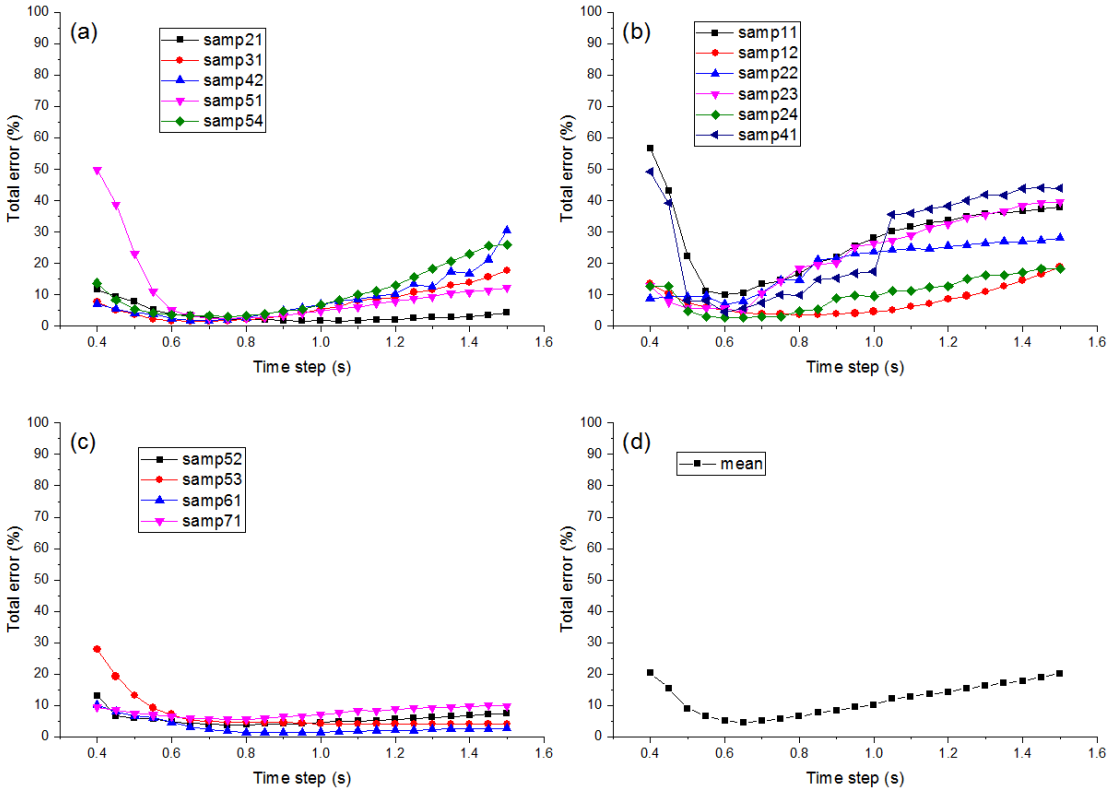

Figure 13. Total errors for each time step: Group I (**a**); Group II (**b**); Group III (**c**); Mean (**d**).

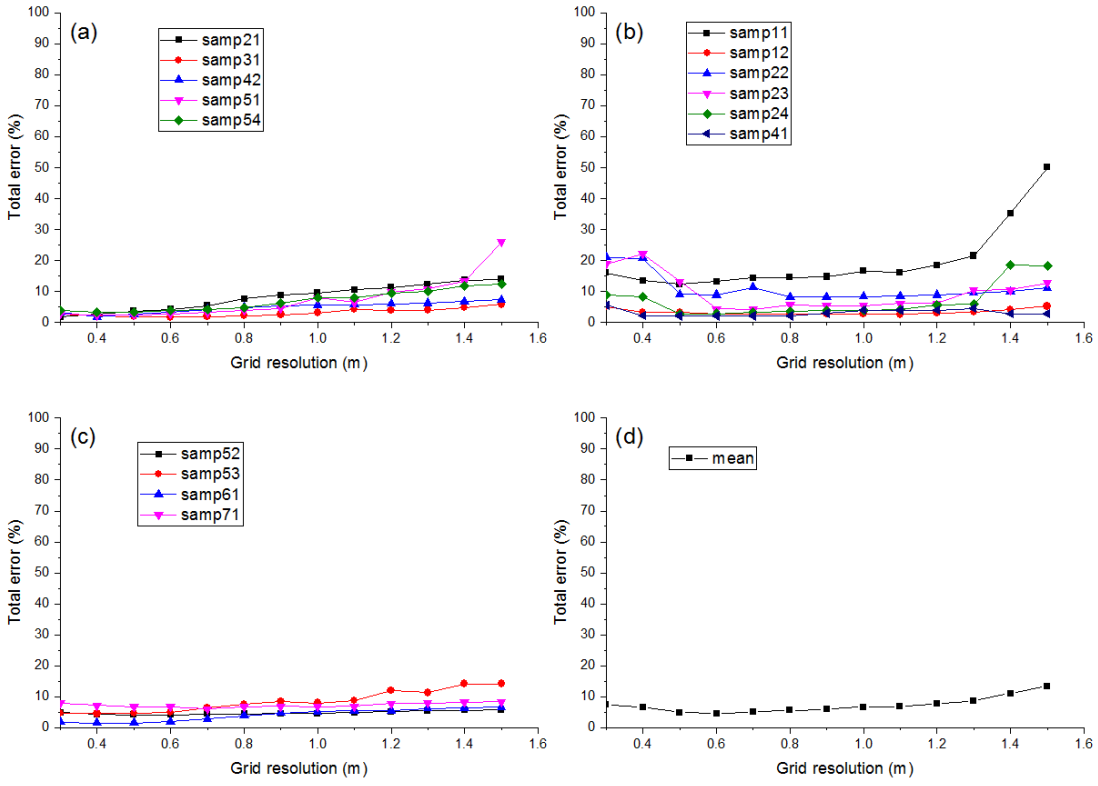

Figure 14. Total errors for each grid resolution: Group I (**a**); Group II (**b**); Group III (**c**); Mean (**d**).

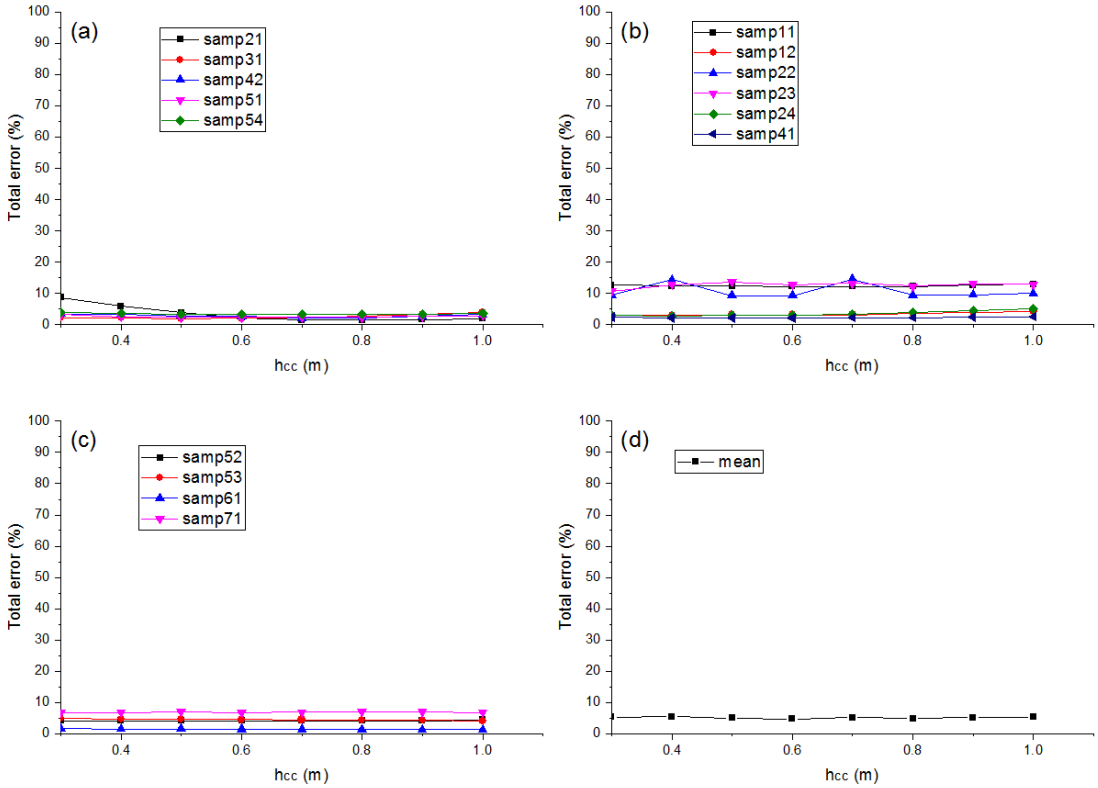

Figure 15. The influences of h_{cc} on total errors: Group I (**a**); Group II (**b**); Group III (**c**); Mean (**d**).

During the simulation, CSF will terminate when the M_HV is less than a threshold or the maximum iterations reach a user specified value. Usually, this user-defined value is set to 500. However in most cases, CSF will end according to the former criterial. Figure 16 shows the trend of M_HV and A_HV (average height variation) of a typical scene. It can be seen that both M_HV and A_HV decreased to a very low value around 100 iterations. Actually, M_HV is 0.0033 when iteration number equal to 150. That means the CSF will give a satisfied result with a relatively low iteration times.

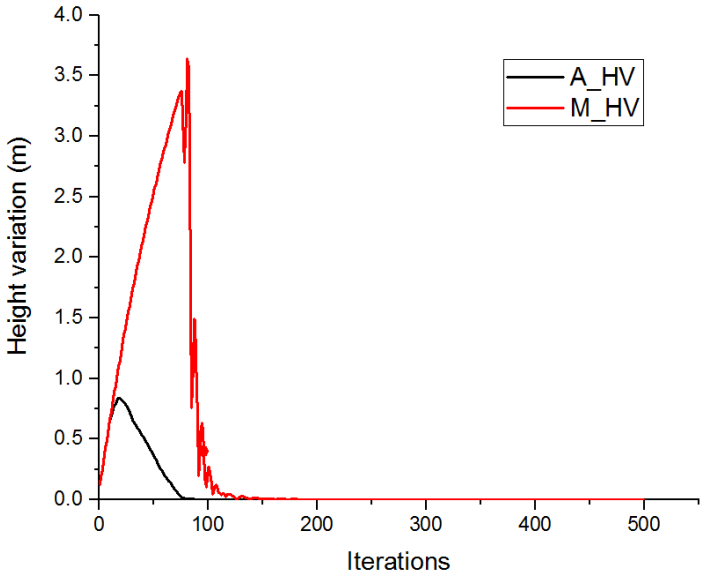

Figure 16. Maximum height variation (M_HV) and average height variation (A_HV).

From the discussion above, the parameter settings in the CSF algorithm are relatively simple and intuitive. Only two parameters (RI and ST) must be determined through visual judgment by the user. A rough estimation is enough to determine these parameter values. Usually, RI can be set to 1, 2 or 3 according to the features of the terrain, they are applied to areas with high steep slopes, terraced slopes and gentle slopes, respectively. ST is set to "true" or "false". "true" means that there exists steep slopes and post-processing is needed. "false" means post-processing is not needed (See details in Section 4.3).

4.3. Steep Slopes

For group II and group III, which contain many steep or terraced slopes in the scene, the slope is an important factor that influences the accuracy of the CSF algorithm. The simulated cloth will lie over the slope but usually cannot stick to the ground perfectly; at the edge of the slope, some distances will appear between the cloth and the ground. If this distance exceeds h_{cc}, the ground measurements will be mistakenly classified as OBJ. A direct method to mitigate this problem is to set the rigidness to a lower value, but some low objects may be classified as BE as a result. To balance these two types of errors, a post-processing method for the margin area is proposed in this study, as mentioned in Section 2.3. Thus, we can use a relatively hard cloth and post-processing to remove lower objects and correctly handle steep slope areas. A typical result is shown in Figure 17, in which the cloth properly sticks to the terrain near the steep margin after the post-processing procedure.

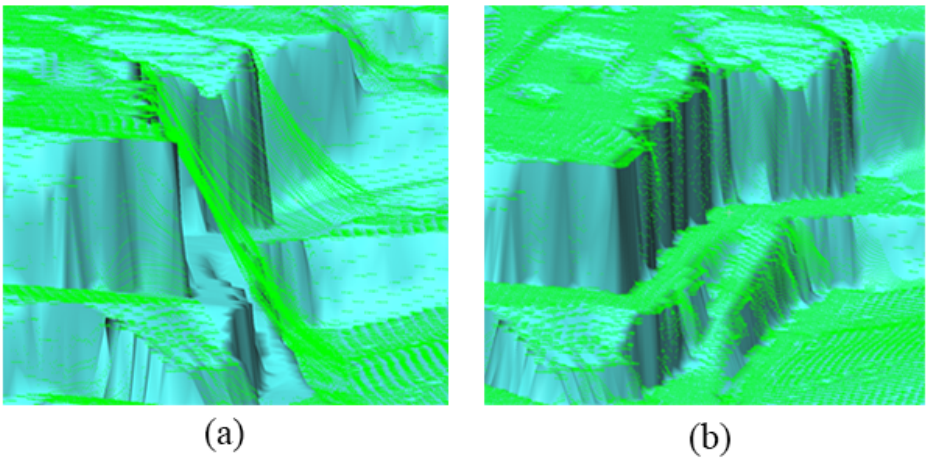

Figure 17. Simulated cloth over an area with steep slopes: (**a**) simulated cloth before post-processing; (**b**) simulated cloth after post-processing.

4.4. Bridge

When handling steep slopes, an exception is made for bridges, which are defined as OBJ in the ISPRS reference datasets, but the adjacent road is treated as BE. Usually, this scene will show a gentle slope along the direction of the road but will show an abrupt elevation change in the direction of the river (Figure 18). During the post-processing procedure, movable particles over the bridge may be set as unmovable particles and stick to the bridge because the height difference between two CPs is very small along the direction of the road. Thus, the bridge will usually be classified as BE through post-processing in our algorithm.

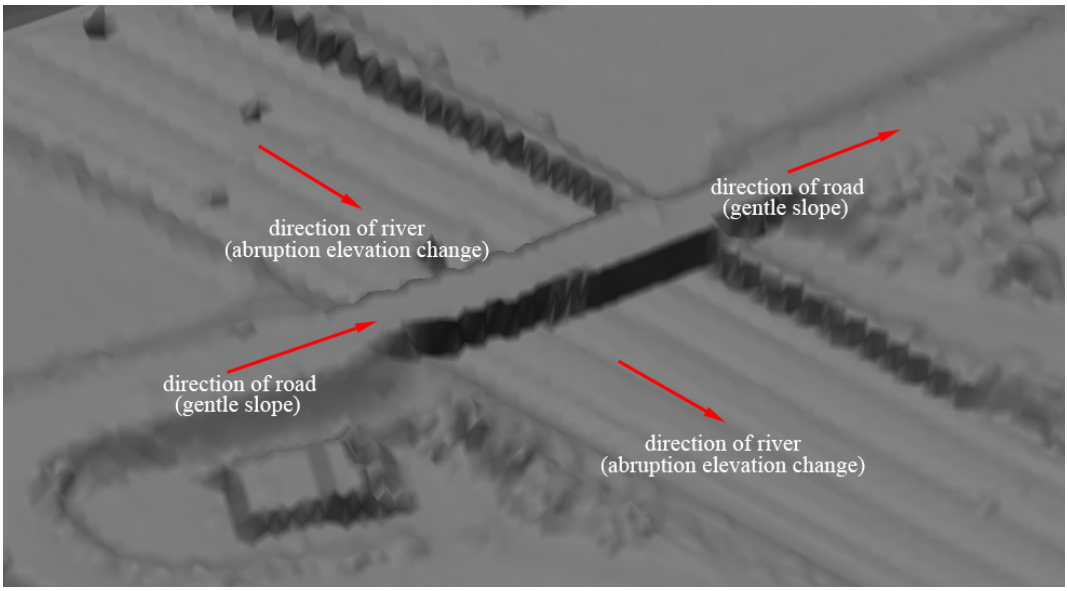

Figure 18. Illustration of a bridge: height variation along road is much less than that in direction of river.

4.5. Outlier Processing

As described in Section 2.2, the cloth particles will stop moving as soon as they reach the IHV; if some outliers exist under the ground measurements, then some particles will be obscured, and the cloth is usually propped up by these outliers. This phenomenon can increase the errors around the outliers because of the rigidness of the cloth. Thus, the outliers should be removed by some statistical filters before applying the CSF algorithm to the point cloud. Among the fifteen samples that were used in this study, sample 41 is an exception that has a large area of outliers [11] under the ground, which cannot be easily removed by a statistical approach. If the outliers could be removed either automatically or manually before applying the CSF algorithm, the results can be satisfactory. The total error before removing the outliers was 5.14, which reduced to 1.63 after removing the outliers.

5. Conclusions

This research proposes a novel filtering method named CSF based on a physical process. It utilizes the nature of cloth and modifies the physical process of cloth simulation to adapt to point cloud filtering. Compared to conventional filtering algorithms, the parameters are less numerous and are easy to set. Regardless of the complexity of ground objects, the samples were divided into three categories according to the shape of the terrain. Few parameters are needed, and these parameters hardly changed among the three sample categories; only an integer parameter rigidness and a Boolean parameter ST are required to be set by the user. These three groups of parameters exhibit relatively high accuracies for all fifteen samples of the ISPRS benchmark datasets. Another benefit of the CSF algorithm is that the simulated cloth can be directly treated as the final generated DTM for some circumstances, which avoids the interpolation of ground points, and can also recover areas of missing data. Moreover, we have released our software to the public [35], and we will also release our source code to the research community. We hope that the proposed novel physical process simulation-based ground filtering algorithm could help promote the scientific, government, and the public's use of LiDAR data and technology to the applications of flood simulation, landslide monitoring, road design, land-cover classification, and forest management.

However, the CSF algorithm has limitations as well. Because we have modified the physical processes of particle movement into two discrete steps, the particles may stick to roofs and some OBJ points may be mistakenly classified as BE when dealing with very large low buildings. This process

usually produces some isolated points at the centers of roofs, an noise filtering can help to mitigate this problem. Additionally, the CSF algorithm cannot distinguish objects that are connected to the ground (e.g., bridge). In the future, we will try to use the geometry information of LiDAR points or combine optical images (such as multispectral images) to clearly distinguish bridges from roads.

Acknowledgments: This work was supported by the National Basic Research Program of China (973 Program) Grant No. 2013CB733402 and the CAS Key Laboratory of Lunar and Deep Space Exploration through Grant No. YQSYS-HT-140630-1. This work was also supported by the National Natural Science Foundation of China Grant No. 41171265, 41331171 and 40801131, and was partially supported by China Scholarship Council (CSC). Special thanks to LiDAR 2015 conference (http://LiDAR2015.org/) for providing the testing data, and Daniel, the maintainer of Cloudcompare (http://www.cloudcompare.org/), who gave us much valuable advice and helped to optimize the source code.

Author Contributions: Wuming Zhang had the original idea that uses cloth simulation to filter LiDAR data. Jianbo Qi refined and implemented this idea, and wrote this manualscript. Peng Wan helps to refine some parameters settings, and developed cloudcompare plugin. Hongtao Wang summarized some existing filtering algorithms and give much useful advice to promote this method. Donghui Xie and Xiaoyan Wang helped to process some LiDAR data, and tested this method with them. Guangjian Yan helped to review this article and give much advice.

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

References

1. Kobler, A.; Pfeifer, N.; Ogrinc, P.; Todorovski, L.; Oštir, K.; Džeroski, S. Repetitive interpolation: A robust algorithm for DTM generation from Aerial Laser Scanner Data in forested terrain. *Remote Sens. Environ.* **2007**, *108*, 9–23.
2. Vosselman, G. Slope based filtering of laser altimetry data. *Int. Arch. Photogramm. Remote Sens.* **2000**, *33*, 935–942.
3. Shan, J.; Aparajithan, S. Urban DEM generation from raw lidar data. *Photogramm. Eng. Remote Sens.* **2005**, *71*, 217–226.
4. Meng, X.; Wang, L.; Silván-Cárdenas, J.L.; Currit, N. A multi-directional ground filtering algorithm for airborne LIDAR. *ISPRS J. Photogramm. Remote Sens.* **2009**, *64*, 117–124.
5. Sithole, G. Filtering of laser altimetry data using a slope adaptive filter. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2001**, *34*, 203–210.
6. Susaki, J. Adaptive slope filtering of airborne LiDAR data in urban areas for digital terrain model (DTM) generation. *Remote Sens.* **2012**, *4*, 1804–1819.
7. Wang, C.K.; Tseng, Y.H. Dem generation from airborne LiDAR data by an adaptive dual-directional slope filter. In Proceedings of the ISPRS Commission VII Mid-Term Symposium 100 Years ISPRS—Advancing Remote Sensing Science, Vienna, Austria, 5–7 July 2010.
8. Liu, X. Airborne LiDAR for DEM generation: Some critical issues. *Progress Phys. Geogr.* **2008**, *32*, 31–49.
9. Sithole, G.; Vosselman, G. Filtering of airborne laser scanner data based on segmented point clouds. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2005**, *36*, W19.
10. Zhang, K.; Chen, S.C.; Whitman, D.; Shyu, M.L.; Yan, J.; Zhang, C. A progressive morphological filter for removing nonground measurements from airborne LIDAR data. *IEEE Trans. Geosci. Remote Sens.* **2003**, *41*, 872–882.
11. Chen, Q.; Gong, P.; Baldocchi, D.; Xie, G. Filtering airborne laser scanning data with morphological methods. *Photogramm. Eng. Remote Sens.* **2007**, *73*, 175–185.
12. Li, Y. Filtering Airborne LIDAR Data by AN Improved Morphological Method Based on Multi-Gradient Analysis. *ISPRS Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2013**, *XL-1/W1*, 191–194.
13. Li, Y.; Yong, B.; Wu, H.; An, R.; Xu, H. An Improved Top-Hat Filter with Sloped Brim for Extracting Ground Points from Airborne Lidar Point Clouds. *Remote Sens.* **2014**, *6*, 12885–12908.
14. Mongus, D.; Lukač, N.; Žalik, B. Ground and building extraction from LiDAR data based on differential morphological profiles and locally fitted surfaces. *ISPRS J. Photogramm. Remote Sens.* **2014**, *93*, 145–156.
15. Pingel, T.J.; Clarke, K.C.; McBride, W.A. An improved simple morphological filter for the terrain classification of airborne LIDAR data. *ISPRS J. Photogramm. Remote Sens.* **2013**, *77*, 21–30.

16. Mongus, D.; Žalik, B. Parameter-free ground filtering of LiDAR data for automatic DTM generation. *ISPRS J. Photogramm. Remote Sens.* **2012**, *67*, 1–12.
17. Axelsson, P. DEM generation from laser scanner data using adaptive TIN models. *Int. Arch. Photogramm. Remote Sens.* **2000**, *33*, 111–118.
18. Zhang, J.; Lin, X. Filtering airborne LiDAR data by embedding smoothness-constrained segmentation in progressive TIN densification. *ISPRS J. Photogramm. Remote Sens.* **2013**, *81*, 44–59.
19. Kraus, K.; Pfeifer, N. Determination of terrain models in wooded areas with airborne laser scanner data. *ISPRS J. Photogramm. Remote Sens.* **1998**, *53*, 193–203.
20. Pfeifer, N.; Reiter, T.; Briese, C.; Rieger, W. Interpolation of high quality ground models from laser scanner data in forested areas. *Int. Arch. Photogramm. Remote Sens.* **1999**, *32*, 31–36.
21. Chen, C.; Li, Y.; Li, W.; Dai, H. A multiresolution hierarchical classification algorithm for filtering airborne LiDAR data. *ISPRS J. Photogramm. Remote Sens.* **2013**, *82*, 1–9.
22. Su, W.; Sun, Z.; Zhong, R.; Huang, J.; Li, M.; Zhu, J.; Zhang, K.; Wu, H.; Zhu, D. A new hierarchical moving curve-fitting algorithm for filtering lidar data for automatic DTM generation. *Int. J. Remote Sens.* **2015**, *36*, 3616–3635.
23. Hui, Z.; Hu, Y.; Yevenyo, Y.; Yu, X. An Improved Morphological Algorithm for Filtering Airborne LiDAR Point Cloud Based on Multi-Level Kriging Interpolation. *Remote Sens.* **2016**, *8*, 35.
24. Elmqvist, M. Automatic Ground Modelling Using Laser Radar Data. Master's Thesis, Linköping University, Linköping, Sweden, 2000.
25. Guan, H.; Li, J.; Yu, Y.; Zhong, L.; Ji, Z. DEM generation from lidar data in wooded mountain areas by cross-section-plane analysis. *Int. J. Remote Sens.* **2014**, *35*, 927–948.
26. Weil, J. The synthesis of cloth objects. *ACM Siggraph Comput. Graph.* **1986**, *20*, 49–54.
27. Provot, X. *Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behaviour*; Graphics Interface; Canadian Information Processing Society: Mississauga, ON, Canada, 1995.
28. Mosegaards Cloth Simulation Coding Tutorial. Available online: http://cg.alexandra.dk/?p=147 (accessed on 8 June 2016).
29. Girardeau-Montaut, D. *Cloud Compare-Open Source Project*; OpenSource Project: Grenoble, France, 2011.
30. Sithole, G.; Vosselman, G. Experimental comparison of filter algorithms for bare-Earth extraction from airborne laser scanning point clouds. *ISPRS J. Photogramm. Remote Sens.* **2004**, *59*, 85–101.
31. Cohen, J. A coefficient of agreement for nominal scales. *Educ. Psychol. Measur.* **1960**, *20*, 37–46.
32. Hu, H.; Ding, Y.; Zhu, Q.; Wu, B.; Lin, H.; Du, Z.; Zhang, Y.; Zhang, Y. An adaptive surface filter for airborne laser scanning point clouds by means of regularization and bending energy. *ISPRS J. Photogramm. Remote Sens.* **2014**, *92*, 98–111.
33. Terrasolid, Ltd. *TerraScan User's Guide*; Terrasolid, Ltd.: Helsinki, Finland, 2010.
34. Pfeifer, N.; Stadler, P.; Briese, C. Derivation of digital terrain models in the SCOP++ environment. In Proceedings of the OEEPE Workshop on Airborne Laserscanning and Interferometric SAR for Detailed Digital Terrain Models, Stockholm, Sweden, 1–3 March 2001; Volume 3612.
35. CSF Software and Introduction. Available online: http://ramm.bnu.edu.cn/researchers/wumingzhang/english/default_contributions.htm (accessed on 8 June 2016).