remote sensing

MDPI

Article

Automated Simulation Framework for Urban Wind Environments Based on Aerial Point Clouds and Deep Learning

Chujin Sun [1], Fan Zhang [2], Pengju Zhao [1], Xinyi Zhao [2], Yuli Huang [1] and Xinzheng Lu [1,*]

[1] Key Laboratory of Civil Engineering Safety and Durability of China Education Ministry, Department of Civil Engineering, Tsinghua University, Beijing 100084, China; scj18@mails.tsinghua.edu.cn (C.S.); zhaopj19@mails.tsinghua.edu.cn (P.Z.); yulihuang@gmail.com (Y.H.)
[2] Central Research Institute of Building and Construction Co., Ltd., MCC Group, Beijing 100088, China; zhangfan@cribc.com (F.Z.); zhaoxinyi@cribc.com (X.Z.)
* Correspondence: luxz@tsinghua.edu.cn

Abstract: Computational fluid dynamics (CFD) simulation is a core component of wind engineering assessment for urban planning and architecture. CFD simulations require clean and low-complexity models. Existing modeling methods rely on static data from geographic information systems along with manual efforts. They are extraordinarily time-consuming and have difficulties accurately incorporating the up-to-date information of a target area into the flow model. This paper proposes an automated simulation framework with superior modeling efficiency and accuracy. The framework adopts aerial point clouds and an integrated two-dimensional and three-dimensional (3D) deep learning technique, with four operational modules: data acquisition and preprocessing, point cloud segmentation based on deep learning, geometric 3D reconstruction, and CFD simulation. The advantages of the framework are demonstrated through a case study of a local area in Shenzhen, China.

Keywords: urban wind environment; computational fluid dynamics; aerial point clouds; deep learning

Citation: Sun, C.J.; Zhang, F.; Zhao, P.J.; Zhao, X.Y.; Huang, Y.L.; Lu, X.Z. Automated Simulation Framework for Urban Wind Environments Based on Aerial Point Clouds and Deep Learning. *Remote Sens.* **2021**, *13*, 2383. https://doi.org/10.3390/rs13122383

Academic Editor: Ahmed Mustafa

Received: 12 May 2021
Accepted: 14 June 2021
Published: 18 June 2021

1. Introduction

Wind environmental analysis is of the most interest in urban physics. It helps address challenges in climate, energy, health, and safety [1]. Many studies have conducted different forms of wind environmental analysis to study wind-induced disasters and their corresponding risk and economic impact [2,3]. Besides using historical data and empirical models [2,3], computational fluid dynamics (CFD) simulation emerges as an increasingly powerful tool for urban wind environment analysis. Existing studies also employed CFD to analyze wind-induced damages towards buildings, trees, and pedestrians [4–6].

CFD simulation is critically dependent on modeling, which is the first and most time-consuming step. For many cities, three-dimensional (3D) models that reflect the current environment are not available for urban wind environment analysis. The modeling of these cities relies on the static data from geographic information systems (GISs). The 3D model development based on the building perimeter has two problems: (1) The GIS data may be outdated or publicly unavailable. (2) The GIS data do not include the vegetation information that is critical to the simulation [6–8]. Critical projects often necessitate extra corrective efforts to reflects the current environment. The manual efforts can never meet urgent demands, such as the pollutant analysis for emergence hospital construction during COVID-19 [9].

Airborne remote sensing enables obtaining vast built environment information. It falls into active and passive categories. The former corresponds to synthetic-aperture radar and light detection and ranging (LiDAR), whereas the latter is associated with optical and thermal range [10]. Remote sensing is extensively employed in the land-use analysis [11,12], environmental change detection [13,14], and disaster management [15–17]. Remote sensing also helps obtain point clouds to reconstruct the 3D features of a built environment. There are two common methods to obtain point clouds for urban reconstruction: LiDAR and oblique photography. A LiDAR system actively emits laser pulses to a target and receives the reflected signals, and the point clouds are usually directly recorded by LiDAR devices and saved in LAS format. Oblique photography uses optical cameras to obtain overlapping images and then generates dense 3D point clouds based on multi-view images using structure from motion and multiple view stereo algorithms [18]. Both LiDAR devices and optical cameras can be installed on unmanned aerial vehicles (UAVs) to collect information about a built environment by aerial surveys. Because optical cameras cost less and generate denser point clouds than LiDAR devices [19], it is advantageous for characterizing a complex urban built environment comprising numerous buildings and trees.

CFD modeling requires assigning various physical parameters to geometric models based on their object classes. Therefore, it is essential to extract semantic and geometric information from remote sensing data. For semantic information, the traditional methods [20–23] require professional participation and are challenging to automate. Zhou [24] used a support vector machine (SVM) to perform semantic segmentation on LiDAR point clouds; however, the calculation of point-wise geometric features (i.e., the SVM input) had low efficiency and is based on a limited range of local features.

In recent years, deep learning networks have been extensively used in semantic segmentation. Deep learning uses multiple neural layers to encode the input data in order to obtain higher-level representations than conventional artificial neural networks and thus enhance the performance [25]. Common networks for 2D semantic segmentation include fully convolutional networks [26], U-Net [27], and DeepLab series [28–30]. When applied to 3D point clouds in urban wind environment analysis, 2D semantic segmentation is not competent. The prediction accuracy is low at the edges of objects because of the lack of height information. In addition, when point clouds within a certain range in the projection plane are classified as tree canopies, the underlying objects that are not supposed to be in the CFD model, such as tree trunks, may also be identified and included in the model. On the other hand, networks for 3D semantic segmentation include Voxnet [31], PointNet [32], and PointNet++ [33]. They have shown their power in understanding 3D geometries and the corresponding engineering applications [34,35]. Such kind of technique has also been employed for semantic segmentation of urban built environments [36] but faces some challenges. The required GPU memory increases exponentially with an increasing number of points. A practical simulation requires point-cloud downsampling that limits the local textural features and leads to inaccurate predictions. For point clouds generated from oblique photography images, occlusion produces low-quality parts, which results in compromised prediction associated with the errors in the height information. This study contributes by integrating the 2D and 3D semantic segmentation techniques and leveraging their advantages, to yield a more accurate classification result for the following modeling process.

There are various mature techniques for 3D reconstruction using point clouds, such as alpha shapes [37], ball pivoting [38], and Poisson surface reconstruction [39], which generate triangular mesh surfaces. The surfaces generated by these types of methods typically have flaws (e.g., holes and sharp corners), which cannot be directly used as the CFD models. Fixing these geometrical flaws entails huge human efforts. In addition, the number of polygons becomes extremely large as the detail level increases [40], which unnecessarily makes it challenging for the modeling operations such as fixing, importing, as well as observation and meshing in the CFD software. Some other methods utilize the

morphological characteristics of objects and establish clean and low-complexity models, which is more in line with the needs of CFD analysis than the methods generating mesh surfaces. Hågbo et al. [41] also showed that this kind of model had a minor impact on decisions made towards urban wind simulations compared to a highly detailed model.

The simulation of an urban wind environment mainly requires the reconstruction of buildings and tree canopies. Existing studies focused on the 3D reconstruction of buildings [19,24,42,43], as opposed to simulations. Three-dimensional reconstruction of tree canopies is mainly based on refined tree modeling [44–46], which is unable to meet the detail-level requirement for considering canopies in CFD analysis. In this work, multiple modeling techniques are utilized to form the geometric 3D reconstruction module in the proposed framework.

In brief, this study proposes an automated simulation framework for urban wind environments based on aerial point clouds and deep learning to address these limitations. By combining 2D and 3D deep learning techniques, the accuracy of semantic segmentation is significantly improved. Based on the existing studies on 3D reconstruction, both building models and canopy fluid volume models suitable for CFD simulation are established. The findings of this study provide automated modeling support for the CFD simulation of urban wind environments, which can further facilitate the analysis and decision-making towards wind-induced disasters.

2. Automated Simulation Framework for Urban Wind Environments

As shown in Figure 1, the proposed automated simulation framework for urban wind environments is composed of four modules: (1) data acquisition and preprocessing, (2) point cloud segmentation based on deep learning, (3) geometric 3D reconstruction, and (4) CFD modeling and simulation.

(1) Module 1: data acquisition and preprocessing

In this study, oblique photography is performed to obtain the data needed for modeling. Compared to LiDAR point clouds, the point clouds generated by oblique photography have (a) a lower acquisition cost, (b) a higher density, which is suitable for complex and dense objects in an urban built environment, (c) color information, which enhances the detection of canopies with a significant color feature, and (d) building façade information, which provides more complete object features for deep learning. The aerial trajectory and the camera parameters can be saved during the oblique photography process, and dense point clouds with RGB color information can be generated. Compared with modeling methods using GISs, this method can efficiently and conveniently collect the 3D up-to-date data of a target area and, thus, provide fundamental data support for modeling.

(2) Module 2: point cloud segmentation based on deep learning

Considering the impact of canopies, the CFD simulation of an urban wind environment requires models of the terrain, buildings, and canopy fluid volumes, which are assigned different physical parameters. To establish their respective models, the first step is to perform a semantic segmentation of the point clouds, which divides the point clouds into three parts: terrain, buildings, and tree canopies. A filter is applied to separate the point clouds of the terrain. Subsequently, the point clouds are segmented using deep learning, which remarkably reduces the work of feature engineering and enhances the capture of local features. The method combines the 2D network, DeepLabv3, and the 3D network, PointNet++. The point clouds are first rasterized into 2D images as the input of DeepLabv3, which subsequently predicts the probability vectors of pixel-by-pixel classification and maps them back to the points. Finally, the point clouds are sparsified and input into PointNet++ to obtain point-by-point classification results. Details of this module are elaborated in Section 3.

(3) Module 3: geometric 3D reconstruction

After obtaining the respective point clouds of the terrain, buildings, and tree canopies, it is necessary to establish clean and low-complexity models of the target area, which

are suitable for CFD simulation. For the terrain, Gaussian process regression is conducted to fit the elevation of the terrain point clouds, which completes the digital surface model of the area and establishes a 3D model. For buildings, the roof planes are extracted, and their boundaries are detected and refined to form 3D models. For tree canopies, the 2D boundary and height range of each cluster of canopies are determined, and the prisms of the canopy fluid volumes can be established. Finally, the above three parts are integrated, with the overlapping parts eliminated by a Boolean operation. Specific steps relating to this module are introduced in Section 4.

(4) Module 4: CFD modeling and simulation

The 3D models in STL format generated by Module 3 are directly imported into the Phoenics CFD simulation software [47]. Grids are generated using its automatic grid generation function. To perform a CFD simulation of an urban wind environment, the models of the terrain, buildings, and canopy fluid volumes are assigned different physical parameters by implementing an automated script.

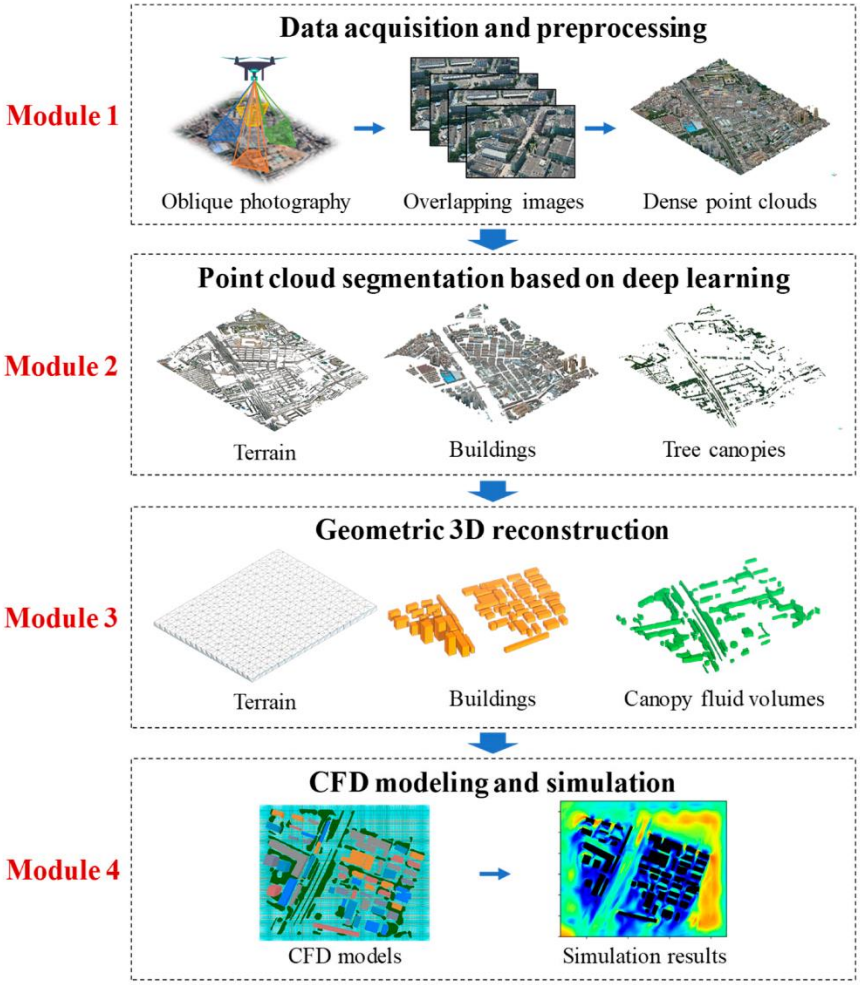

Figure 1. Proposed automated simulation framework for urban wind environments.

3. Point Clouds Separation Based on Deep Learning

3.1. Terrain Filtering

Both terrain and building roofs contain similar horizontal surfaces, which may cause unnecessary misunderstandings in 3D semantic segmentation. The cloth simulation filter (CSF) [48] is applied to the point clouds obtained by oblique photography to extract the terrain point clouds, using a plug-in in the open-source software CloudCompare [48,49].

3.2. Segmentation of Buildings and Tree Canopies

As mentioned in Section 1, point cloud segmentation of an urban environment using 2D networks may have limited prediction accuracy at the edges of the objects and cannot reflect 3D features. Although 3D networks overcome the above limitations, they may lose local texture information owing to the limitation of the GPU memory. Moreover, they are more sensitive to the errors in oblique photography point clouds. To overcome the shortcomings of a single 2D or 3D deep learning technique in the point cloud segmentation of an urban environment, both methods are combined to better utilize their respective advantages. The 2D image semantic segmentation network DeepLabv3 [30], which applies an atrous convolution to capture multiscale information, is used. A pre-trained ResNet18 is used as its backbone network. PointNet++ [33], with PointNet [32] as its backbone network, is used as the 3D point cloud semantic segmentation network. It is composed of set-abstraction layers for hierarchical feature extraction and feature-propagation layers for prediction. Using the coordinates and the point-by-point feature vectors as the input, PointNet++ can capture the local characteristics of the point clouds. As shown in Figure 2, the integrated method comprises four steps.

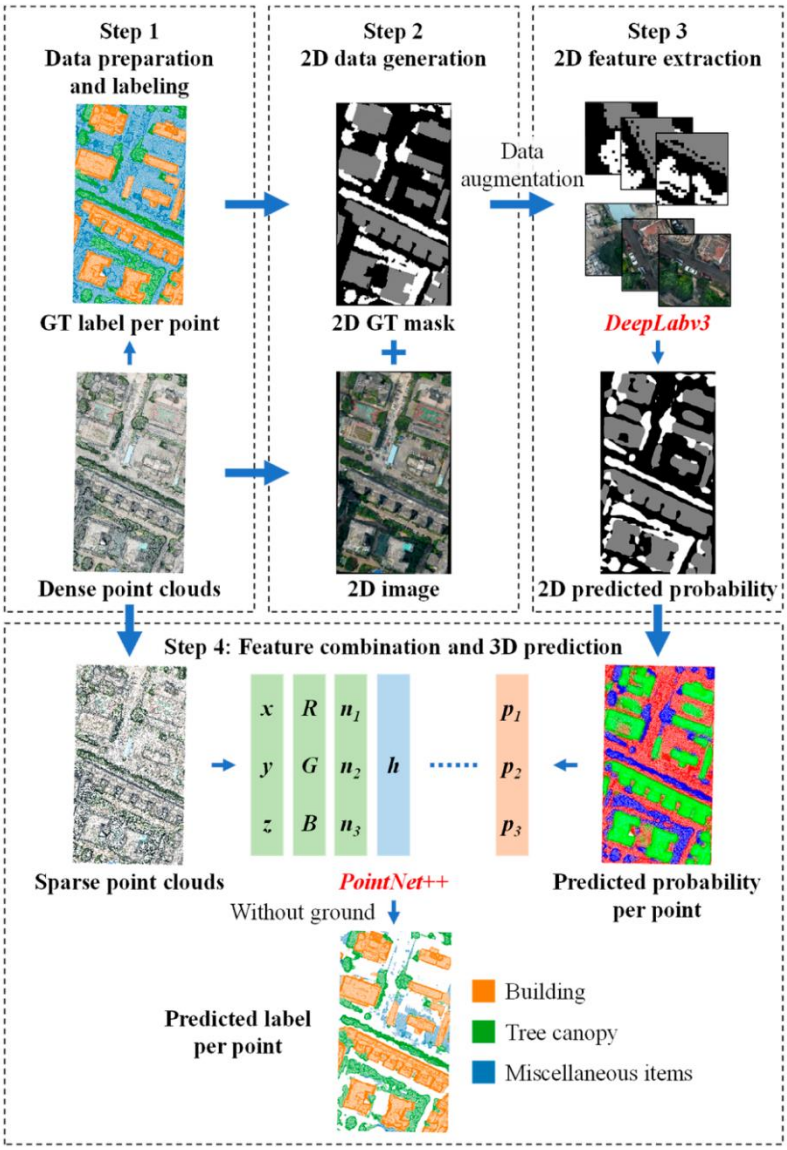

Figure 2. Point cloud semantic segmentation method combining 2D and 3D deep learning networks. (GT is the abbreviation for ground truth.).

Step 1: data preparation and labeling

In the first step, the dense point clouds are labeled manually, and the dataset is divided into training and test sets as the input of the deep learning networks.

Step 2: 2D data generation

The dense point clouds are projected onto a horizontal plane and rasterized into images. The point set of a tile, which is a point cloud segment of the entire model generated by the oblique photography processing software, is denoted as P. The rasterization process first creates a grid of dimensions $H \times W$ over the rectangular bounding area of the tile. Then, an image of the same dimensions $H \times W$ is created, and each pixel (m, n) in the mth row and the nth column corresponds to a grid cell $g_{(m,n)}$. Let $p \in P$ denote a point. The subset of P containing the points in $g_{(m,n)}$ is denoted as $P_{(m,n)} = \{p | p \in P, p \; in \; g_{(m,n)}\}$. The label set of the semantic classes is $L = \{l^{\text{building}}, l^{\text{tree}}, l^{\text{misc}}\}$, whose elements represent buildings, tree canopies, and miscellaneous items, respectively. The RGB color vector, $\boldsymbol{c}_{(m,n)}$, of each pixel (m, n) equals the color vector of the highest point in $g_{(m,n)}$, and it restores the aerial top view. $\boldsymbol{c}_{(m,n)}$ is calculated using Equation (1).

$$\boldsymbol{c}_{(m,n)} = \boldsymbol{c}_p, p = \underset{q \in P_{(m,n)}}{\operatorname{argmax}} z_q, \tag{1}$$

where $\boldsymbol{c}_p$ is the color vector of a point p, and z_q is the vertical coordinate of a point q. Similarly, the ground truth label is rasterized into images. The label of each pixel (m, n), $l_{(m,n)}$, is the label with the most occurrences in $g_{(m,n)}$, as expressed in Equation (2).

$$l_{(m,n)} = \underset{l \in L}{\operatorname{argmax}} \sum_{q \in P_{(m,n)}} \mathbb{I}[l_q = l], \tag{2}$$

where $\mathbb{I}(\cdot)$ is the counting function with a value of 0 or 1, and l_q is the label of a point q. If $g_{(m,n)}$ does not contain any point, $\boldsymbol{c}_{(m,n)}$ is calculated by linear interpolation, while $l_{(m,n)}$ is obtained by the nearest-neighbor interpolation. The data forms required for the 2D image semantic segmentation—2D images and ground truth masks—are thus available.

Step 3: 2D feature extraction

The number of 2D top view images generated by the rasterization is quite small because each tile corresponds to one image only, which is much less than the number of original aerial images directly obtained by a UAV. This amount of data cannot meet the training requirement of DeepLabv3. Therefore, the 2D image data need to be augmented. Random cropping, rotation, and flipping are conducted in this study. After training, DeepLabv3 can produce the probability vectors of the pixel-by-pixel classification, and the length of the vector equals the number of classes, namely $|L|$.

Step 4: feature combination and 3D prediction

The dense point clouds are randomly downsampled to a reasonable density based on the capacity of the computing device. The downsampled point clouds have spatial coordinates, RGB colors, and normal vectors. The location relationship between the points and the pixels is determined using the coordinates, and the predicted probability vectors of the pixel-by-pixel classification are mapped to each point. In addition, the relative height of a point is added as one of the features to reflect the vertical characteristics of the objects in the urban environment. The relative height, h_p, of a point p is calculated using Equation (3).

$$h_p = z_p - \min_{q \in P} z_q. \tag{3}$$

The combined feature is a 13-dimensional vector comprising spatial coordinates, RGB colors, normal vectors, 2D predicted probabilities, and a relative height. Only the non-terrain point clouds filtered by the CSF are retained for training and evaluation. PointNet++ yields the point-by-point classification results.

The proposed integrated method has the following three advantages:

(1) The 2D prediction of DeepLabv3 is combined with the 3D input features of PointNet++, which allows fully utilizing the advantages of the 2D data containing dense texture features and overcomes the shortcoming of the 3D network of losing the local characteristics when the point clouds are sparsified owing to device capacity.
(2) The z coordinates and the relative heights entailed in PointNet++ strengthen the importance of the vertical information and improve the accuracy at the edges of objects compared to that of the single 2D network.
(3) The input images of the 2D network are not oblique photos captured by a UAV but are the images rasterized from the projected point clouds. No extra efforts are needed to determine the mapping relationship between the oblique photos and the 3D point clouds. Labeling for training is required only once for the point clouds, which avoids the burden of labeling on 2D images.

4. Geometric 3D Reconstruction

4.1. Terrain Generation

The filtered terrain point clouds, described in Section 3.1, are used to establish a digital surface model. Regular rectangular meshes are applied by fitting the terrain point clouds with the Gaussian process regression model [50].

4.2. Building Reconstruction

As shown in Figure 3, the building reconstruction comprises three steps before establishing the geometric models.

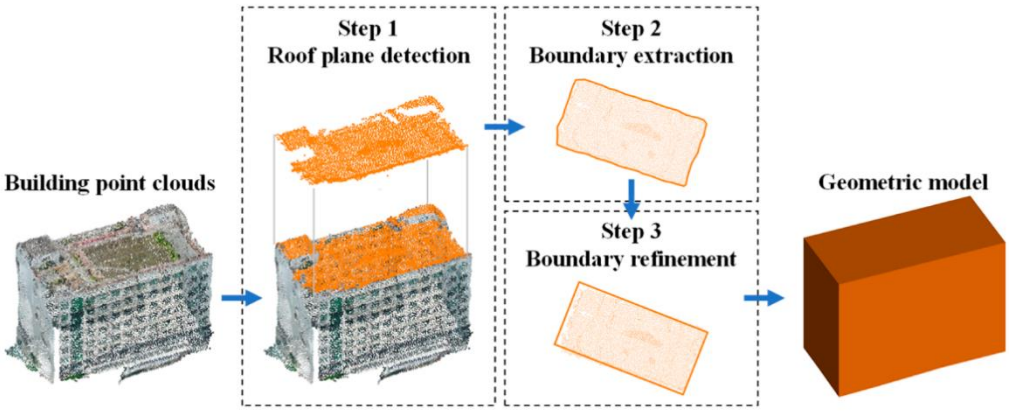

Figure 3. Workflow of building reconstruction.

Step 1: Roof plane detection

Wang et al. [43] used RANSAC to extract roof planes from airborne LiDAR point clouds. This method is adopted in this study as well. However, different from the technique of Wang et al. [43], in this study, (1) the building point clouds are already separated, as discussed in Section 3.2, which reduces the adverse impact of nonbuilding objects on the extraction of roof planes, (2) oblique photography point clouds are used, which contain the points of building façades; therefore, an angle threshold is set to filter façade planes. RANSAC is implemented using a plug-in of CloudCompare [49,51].

Step 2: Boundary extraction

Constructing 3D models requires the boundary lines of the roof planes. The 2D alpha shape algorithm [37] is adopted to extract the boundaries. To extract reasonable boundaries, the alpha value in the algorithm is adjusted based on the density of the point clouds.

Step 3: Boundary refinement

Owing to the irregularities of building facades and the errors caused by point cloud generation, the boundary geometries may present a zig-zag characteristic after the

boundaries are extracted. To create models maintaining general building geometrical and topological features for a CFD simulation, specific steps are taken as follows:

(1) To address the zig-zag problem, the method proposed by Poullis [52] is adopted, which detects principal directions and regularizes building boundaries based on Gaussian mixture models and energy minimization. The energy minimization problem is equivalent to a minimum cut problem and is solved using gco-v3.0 [53–56] in this study.

(2) The Ramer–Douglas–Peucker (RDP) algorithm [57,58] is used to sparsify the boundary points to retain the points that lie along straight lines.

(3) All boundary line segments in the target area are searched for the segment pairs whose two segments have a distance and an angle within certain thresholds. Subsequently, each segment pair is combined to make both segments in the pair collinear in the horizontal plane.

(4) The angle between each adjacent boundary line segment pair of a building is further revised. As shown in Figure 4, for the point set of a building boundary, B, and its sequential points p_{i-1}, p_i, and p_{i+1}, based on the threshold, the revision is as follows:

   (a) The extreme acute angles caused by outliers p_i are eliminated. The new point set of the building boundary, B^*, becomes

$$B^* = B - \{p_i\}. \tag{4}$$

   (b) p_i is moved along the median $\overline{p_i p_m}$ of the triangle to p_i^* when the angle is approximately a right angle. Coordinates $\boldsymbol{x}_i^*$ of p_i^* in B^* are revised as follows:

$$\boldsymbol{m} = \frac{\boldsymbol{x}_{i+1}+\boldsymbol{x}_{i-1}}{2} \text{ and} \tag{5}$$

$$\boldsymbol{x}_i^* = \boldsymbol{m} + \frac{1}{2}||\boldsymbol{x}_{i+1} - \boldsymbol{x}_{i-1}|| \frac{\boldsymbol{x}_i - \boldsymbol{m}}{||\boldsymbol{x}_i - \boldsymbol{m}||}. \tag{6}$$

   (c) The obtuse angles that are approximately 180° are eliminated to further smoothen the boundary. B^* is modified as in Equation (4).

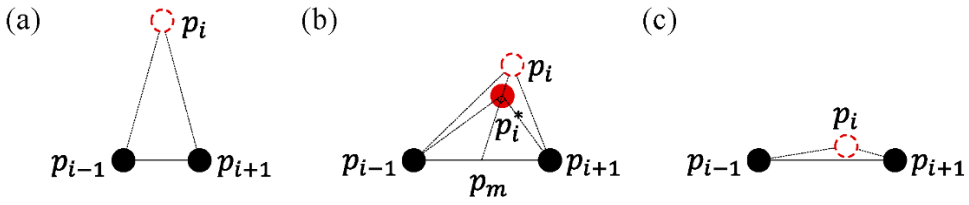

Figure 4. Angle revision of building boundaries for (**a**) extreme acute angles, (**b**) approximate right angles, and (c) obtuse angles approximate to 180°.

The 3D reconstruction of buildings is completed using the aforementioned procedure. It should be noted that existing studies have proposed various methods for building reconstruction, among which one feasible technique is adopted in this study. If a method can generate clean and low-complexity building models required for a CFD simulation, it can replace this part of the proposed framework.

4.3. Canopy Fluid Volumes

It is labor-intensive and unnecessary to model each tree in an urban environment for CFD simulations. Amorim et al. [7] grouped neighboring trees and manually modeled each group into a strip-like cuboid, whose height equaled the vertical distance between the average bottom and the average top of canopies in a local area. This can drastically improve the modeling and analysis efficiency and has a limited impact on accuracy [6,7].

In this study, the obtained canopy point clouds are used to build the models of prism-shaped canopy fluid volumes as described in Algorithm 1; specifically,

(1) The outliers with average distances to neighboring points remarkably larger than the average level in the entire area are removed.
(2) The point clouds need to be clustered into groups for modeling. Different clustering algorithms have been developed in existing studies [59–61]. For the grouping task based on the Euclidean distance, k-means-based algorithms require a pre-specified number of clusters and assume the clusters are convex. Thus, the DBSCAN algorithm [59] is adopted due to its robustness to outliers, explicit control over density via parameters, and variable cluster shapes. The *minPoints* and *eps* of DBSCAN are set to 1 and 3.0, respectively, in this study. The groups with a point number less than the threshold are ignored and removed.
(3) The 2D boundary of each canopy point group is extracted using the 2D alpha shape algorithm [37] and sparsified by the RDP algorithm [57,58].

Algorithm 1. Generation of boundaries of tree canopy volumes

Input: point clouds of tree canopies $P^{\text{tree}} = \{p_i | i = 1,2,\dots,k\}$
Output: set of boundaries of tree canopy volumes $\mathcal{B}^{\text{tree}}$
for $i \leftarrow 1$ **to** k **do**
    $N(p_i) \leftarrow \text{neighbors}(p_i, n)$ //Set of n-nearest neighbors of p_i
    $d(p_i) \leftarrow \frac{1}{|N(p_i)|}\sum_{q \in N(p_i)} \text{distance}(p_i, q)$ //Average distance in $N(p_i)$
end
$P^{\text{tree}} \leftarrow \{p | p \in P^{\text{tree}}, d(p) \le \frac{1}{k}\sum_{i=1}^{k} d(p_i) + \sigma_{\text{threshold}}\}$
$\mathcal{T} \leftarrow \text{DBSCAN}(P^{\text{tree}})$ //Set of clustered point clouds
$\mathcal{T} \leftarrow \{T | T \in \mathcal{T}, |T| \ge n_{\text{threshold}}\}$
foreach T **of** $\mathcal{T}$ **do**
    $B_T \leftarrow \text{AlphaShape2D}(T)$ //Boundary points of tree canopy T
    $B_T \leftarrow \text{RDP}(B_T)$
end
$\mathcal{B}^{\text{tree}} = \{B_T | T \in \mathcal{T}\}$

4.4. Postprocessing

The terrain, buildings, and canopy fluid volumes models are integrated. Owing to possible overlap, a Boolean operation is implemented in 3ds Max [62] via a script.

5. Case Study

5.1. Case Description

To validate the effectiveness of the proposed automated simulation framework for urban wind environments, a real city is chosen for a case study. Bao'an is an administrative district of Shenzhen, a city in southern China. Various types of buildings are densely built in the district, including residential, industrial, commercial, and other old buildings. In addition, as a typical city in southern China, the city has extensive vegetation, such as street trees. The densely distributed buildings and trees pose a significant challenge to the CFD modeling of its urban environment (Figure 5). By an aerial survey of a local part of Bao'an using a UAV, 3265 photos with a total of 78.4 × 10^3 trillion pixels are captured. Thirty tiles of point clouds having similar sizes are generated using ContextCapture [63] (Figure 6). Because oblique photography is easy to implement, the proposed method can better reflect the current environment of the target area than a GIS-based method. The data are labeled by professional students after using the CSF to filter the ground, as shown in Figure 7. The Thirty tiles are rasterized into images for 2D segmentation and feature extraction. The grid size is set to 0.1 m in this study, which approximates the density of the generated point clouds.

Figure 5. Baidu street view of Bao'an, Shenzhen.

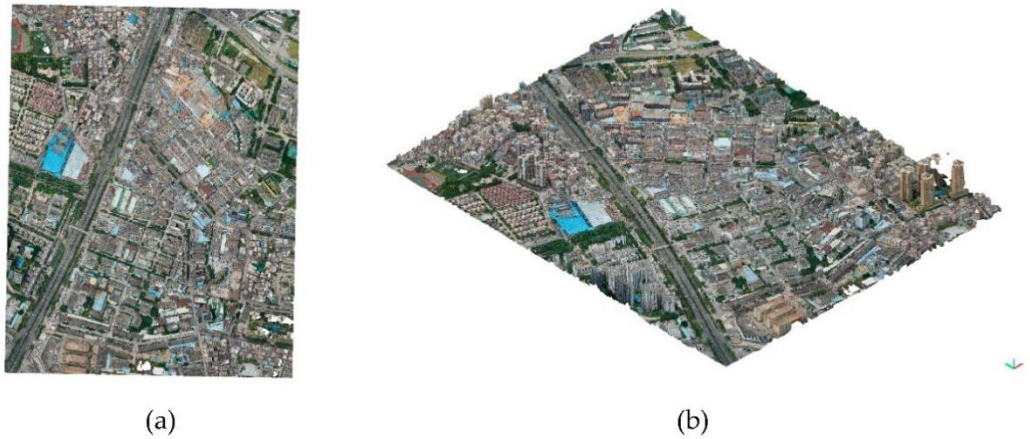

Figure 6. Point clouds generated from oblique photography: (**a**) top view (**b**) front isometric view.

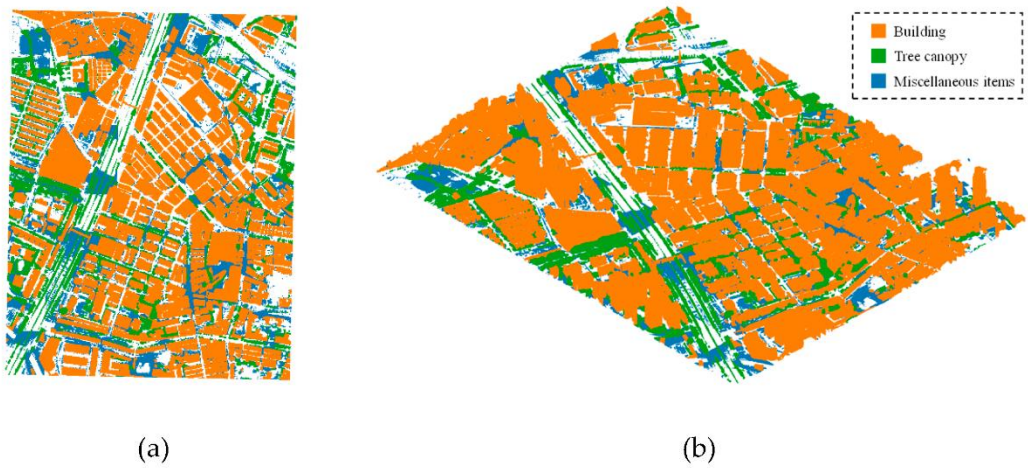

Figure 7. Point-by-point label without terrain: (**a**) top view (**b**) front isometric view.

5.2. Results and Analysis

5.2.1. Point Cloud Separation

After the terrain point clouds are filtered by the CSF, the 30 tiles of point clouds are divided in a 6:4 ratio into training and test sets, which contain 18 and 12 tiles, respectively. The prediction accuracies of classification of an SVM, a random forest (RF), DeepLabv3, PointNet++, and the proposed integrated method are compared. The SVM and the RF are implemented based on Scikit-learn [64], while the deep learning techniques are implemented based on PyTorch [65]. The SVM employs the features used by Zhou [24], which describe the regularity, horizontality, flatness, and normal vector distribution. Because Zhou [24] used LiDAR point clouds without color information, a greenness feature,

$S_{\text{greenness}}(p)$ [19], is added for the oblique photography point clouds, as expressed in Equation (7).

$$S_{\text{greenness}}(p) = \frac{1}{|N(p)|}\sum_{q\in N(p)}\frac{G_q}{R_q+G_q+B_q}, \quad (7)$$

where $N(p)$ is the neighborhood point set of a point p and has a range that is consistent with the features used by Zhou [24]. $[R_q, G_q, B_q]$ is the color vector of a point q. The RF uses the same features as the SVM, and classifiers containing 10, 50, and 100 trees are tested, which are denoted as RF-10, RF-50, and RF-100, respectively.

The backbone network of DeepLabv3 is a ResNet-18 pre-trained on ImageNet. DeepLabv3 applies Atrous Spatial Pyramid Pooling (ASPP) to resample features at different scales. The dilation rates of the kernels (also known as the atrous rates) of the paralleled atrous convolutions are set to (12, 24, 36) in this study. After the augmentation of the original training set through random rotation, flipping, and cropping, the input data of DeepLabv3 comprise 3000 images with a size of 256 × 256. The batch size is 30, and the Adam optimizer with a cosine learning rate initialized at 0.001 is adopted.

PointNet++ uses four set-abstraction layers with single-scale grouping and four feature-propagation layers in this study. The architecture is as follows:

SA(2048, 0.8, [32, 32, 64]) → *SA*(1024, 1.6, [64, 64, 128]) → *SA*(512, 3.2, [128, 128, 256]) → *SA*(256, 6.4, [256, 256, 512]) → *FP*(256, 256) → *FP*(256, 256) → *FP*(256, 128) → *FP*(128, 128, 128) → *FC*(128, 0.5) → *FC*(3)

SA(*K*, *r*, [l_1, …, l_d]) represents a set-abstraction layer with *K* feature points, a ball query radius *r*, and *d* fully connected layers with width l_i(i = 1, …, *d*); *FP*(l_1, …, l_d) denotes a feature-propagation layer with *d* fully connected layers; *FC*(*l*, r_{drop}) is a fully connected layer with width *l* and dropout ratio r_{drop}. The input data are downsampled to 50,000 points for each tile. The batch size is 6, and the Adam optimizer with a step learning rate initialized at 0.001 is adopted. Readers can refer to [30] and [33] for more details of the architectures of DeepLabv3 and PointNet++.

The prediction accuracy of the point cloud segmentation is measured in terms of the precision, recall, and F_1 score, which are calculated as follows:

$$precision = \frac{TP}{TP+FP}, \quad (8)$$

$$recall = \frac{TP}{TP+FN}, \text{ and} \quad (9)$$

$$F_1 = \frac{2\times precision\times recall}{precision+recall}, \quad (10)$$

where TP, FP, and FN denote true positive, false positive, and false negative, respectively. The F_1 score is a comprehensive equal-weight metric of precision and recall. Table 1 lists the average metrics achieved on the test set tiles. Taking one building with its surrounding environment as an example, the isometric and top views of the original point clouds, ground truth labels, and predicted labels of the five methods are shown in Figure 8.

It can be concluded that:

(1) The SVM fails to differentiate between buildings and tree canopies, leading to a misprediction of miscellaneous items (Figure 8c) and relatively low precision for buildings and canopies.
(2) The RF hardly improves its performance when the number of trees increases but has a slightly higher performance than the SVM; however, there are many outliers mixed in the true classes, which is disadvantageous for the subsequent modeling process.
(3) Because DeepLabv3 does not have height information, it has a low accuracy at the edges of objects and tends to predict the edge points as miscellaneous items (blue points at the edges of the buildings and the canopies in Figure 8e). This makes the

recall higher for miscellaneous items and significantly lower for buildings and canopies compared to their respective precision.

(4) Although PointNet++ has a satisfying result for buildings, the precision for canopies is low because the normal vector distribution of the canopy areas is irregular. As shown in Figure 8f, the canopy points on the right side of the building have a high probability of being predicted as building points. This may lead to unexpected building point clouds and incomplete canopy point clouds in the modeling step.

(5) The method proposed in this paper combining DeepLabv3 and PointNet++ improves the accuracy at the edges of objects as well as addresses the problems caused by the complexity of point cloud characteristics and the low generation quality due to occlusion. The accuracy for miscellaneous items is remarkably improved, and the precision and recall of buildings and canopies are balanced well, which can provide accurate point clouds for 3D modeling.

Table 1. Average metrics achieved by different methods on test set.

Classes	Building			Tree Canopy			Miscellaneous Items		
Metrics	**Precision**	**Recall**	**F_1**	**Precision**	**Recall**	**F_1**	**Precision**	**Recall**	**F_1**
SVM	0.88	0.96	0.92	0.72	0.81	0.76	0.00	0.00	0.00
RF-10	0.89	0.94	0.91	0.76	0.82	0.79	0.34	0.16	0.22
RF-50	0.89	0.95	0.92	0.76	0.84	0.80	0.42	0.12	0.19
RF-100	0.89	0.96	0.92	0.76	0.84	0.80	0.43	0.12	0.18
DeepLabv3	**0.97**	0.85	0.90	**0.90**	0.80	0.85	0.36	**0.79**	0.49
PointNet++	0.93	0.94	0.93	0.76	0.82	0.79	0.59	0.45	0.51
This work	0.96	**0.96**	**0.96**	0.86	**0.92**	**0.89**	**0.68**	0.62	**0.65**

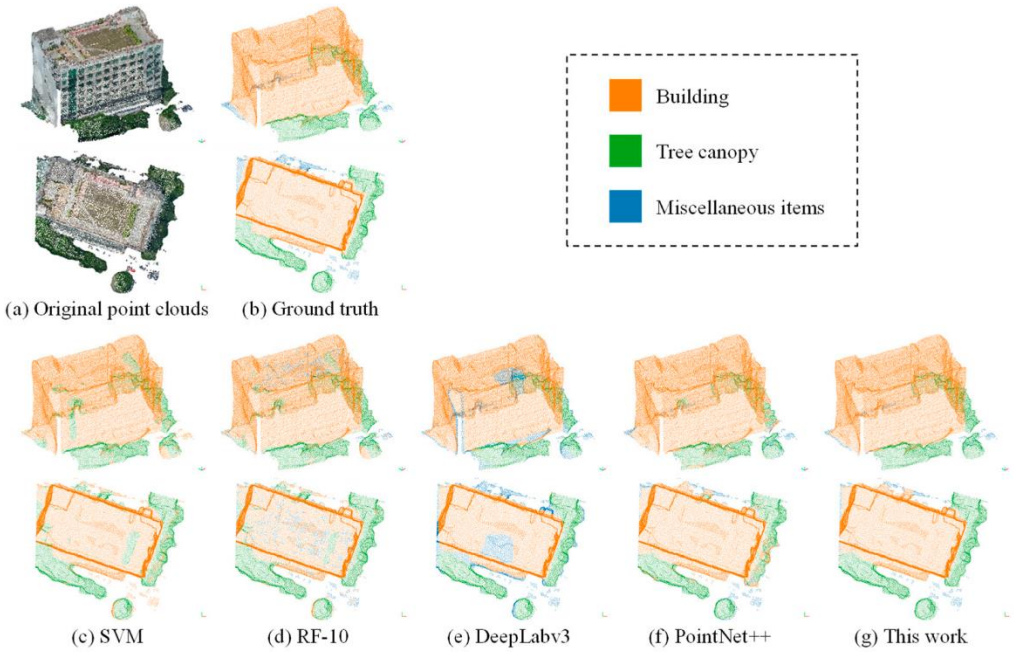

Figure 8. Original point clouds, ground truth labels, and predicted labels of five methods.

5.2.2. Three-Dimensional Reconstruction

To determine the effects of buildings and tree canopies on the flow field, a local area with densely distributed buildings and trees is selected as the case in the 3D reconstruction and CFD simulation. The selected area is approximately 650 m × 550 m and has 59 buildings in total, with a maximum height of 80 m. The method introduced in Section 4 is utilized to model the terrain, buildings, and canopy fluid volumes, as shown in Figure 9.

The models meet the requirement for CFD simulations and retain the geometric features of the main objects to a certain extent. The models are stored in STL format.

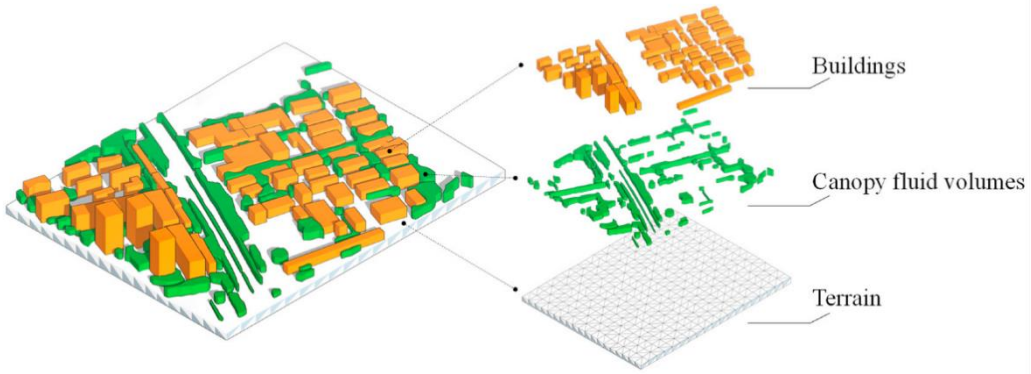

Figure 9. Three-dimensional models of terrain, buildings, and canopy fluid volumes of target area.

5.2.3. CFD Simulation

The Phoenics CFD simulation software is commonly used for the simulation of outdoor wind environments [66–68]; it is easy to use and computationally fast. Therefore, Phoenics is used in this study for the wind simulation of the target area. The generated STL files, described in Section 5.2.2, are directly imported as Phoenics geometric models, as shown in Figure 10a. The domain size is 2300 m × 2300 m × 500 m, which meets the requirements of the distance from domain boundaries to the target area under different wind directions [69]. Adaptive structural grids are generated using the automatic grid-creating function in Phoenics. The grid sizes for the building surfaces can reach approximately 1 m, which meets the requirement for a CFD simulation (Figure 10b).

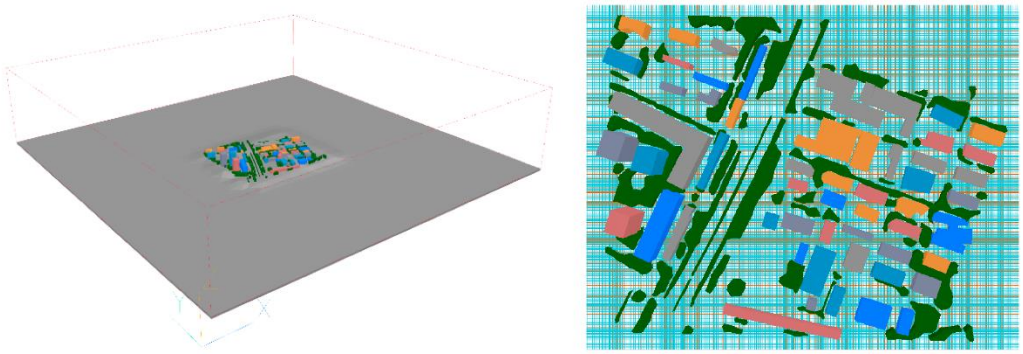

Figure 10. Computational models in Phoenics: (**a**) geometric models, (**b**) grids (red lines denote adaptive constrains, and blue represent grid lines).

Typical methods of urban wind environment analysis are adopted when setting the boundary conditions and model parameters in the CFD simulation. The details are provided in Appendix A.

Two scenarios are analyzed in this study.

(1) Annual dominant wind direction

Based on the Shenzhen Climate Bulletin 2019 [70], the annual dominant wind direction is north-northeast (NNE), with a frequency of 17% and an average wind speed of 2.1 m/s at a 10 m reference height. The wind speed, mean age of air, and wind pressure are analyzed under this scenario. The wind amplification factor (WAF) is defined as the local

air velocity divided by the wind-profile speed at the local height. The mean age of air represents the time that the air has taken to travel from the domain entry to the local point, and its relative value reflects the degree of air circulation. The height position of the following metrics refers to the height above the ground. Because the model contains a non-planar terrain, the simulation results are extracted in Phoenics using an automatic probe script.

Figure 11 shows the wind speed and the WAF at a pedestrian height of 1.5 m. The whole area is exposed to a wind speed of less than 5 m/s and a WAF of less than 2, which meets the requirements in the Chinese code [71] concerning outdoor walking and comfort. However, because of the dense buildings on the east side and a large L-shaped building on the west side, the wind speed in some internal streets is low and their mean age of air (Figure 12) is higher than that in the surrounding open area. This suggests a poor air circulation condition. Figure 13 shows the wind pressures at 1.5 m, 15 m, and 50 m. Most buildings have a wind pressure difference exceeding 0.5 Pa between their indoor and outdoor environments, which meets the requirements of natural ventilation. Except for the buildings facing streets and the top stories of the west-side tall buildings, the wind pressure difference between the windward and leeward sides meets the requirement of not exceeding 5 Pa [71].

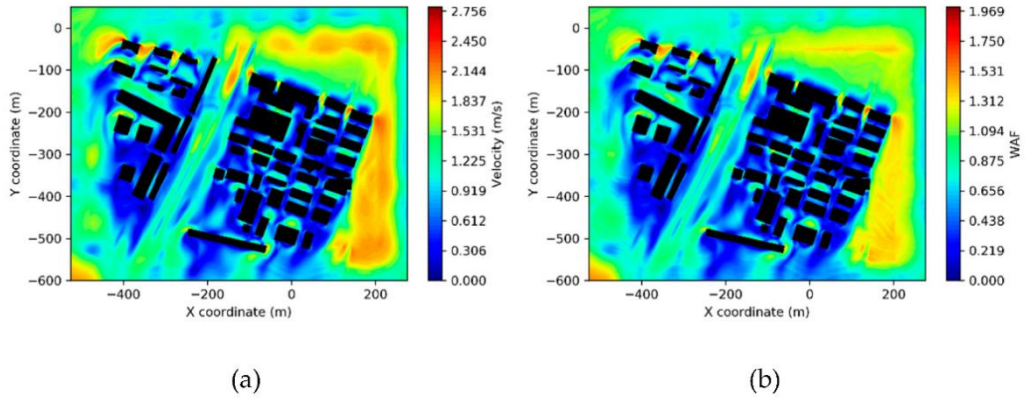

Figure 11. (**a**) Wind speed and (**b**) WAF at pedestrian height of 1.5 m (NNE 2.1 m/s).

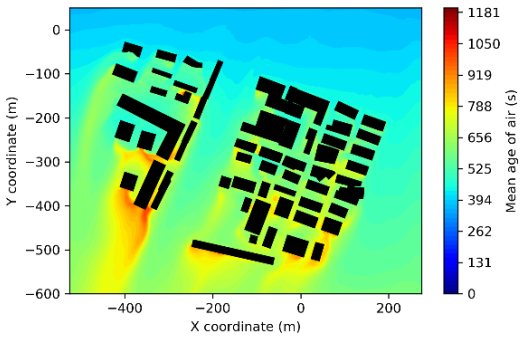

Figure 12. Mean age of air at pedestrian height of 1.5 m (NNE 2.1 m/s).

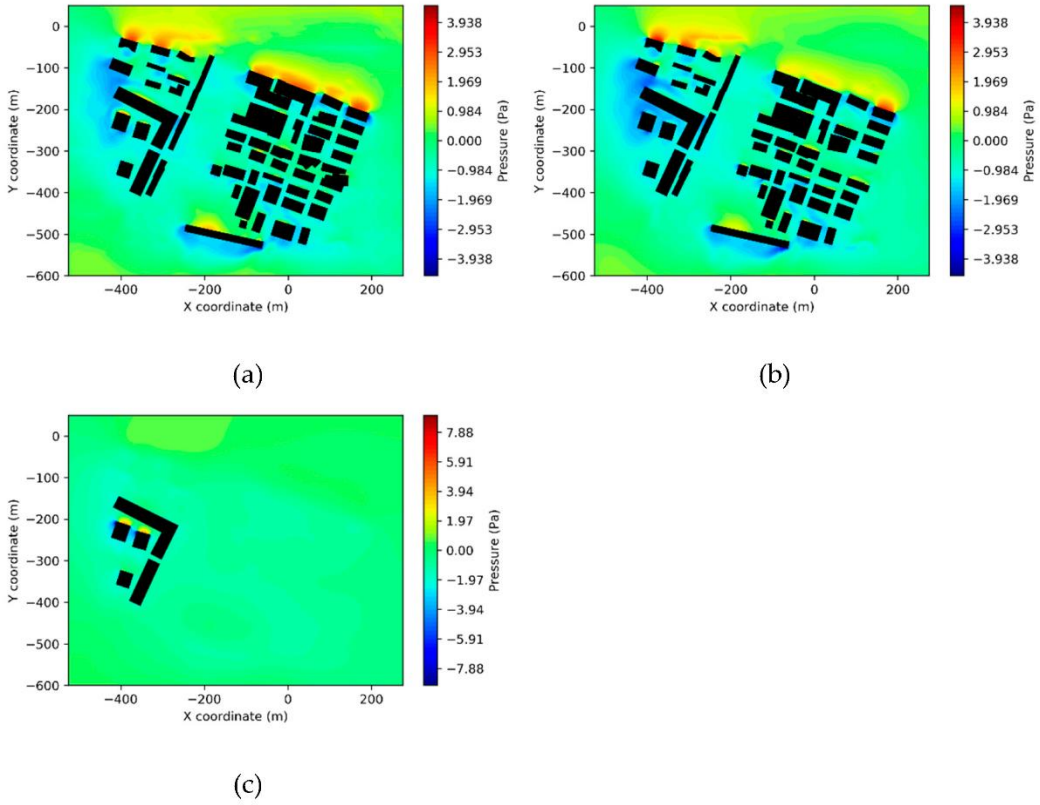

Figure 13. Wind pressure at different height levels: (**a**) 1.5 m, (**b**) 15 m, and (**c**) 50 m (NNE 2.1 m/s).

(2) Tropical storm

In 2019, Shenzhen experienced the tropical storm Wipha, which exhibited a wind speed of 14.1 m/s at a 10 m reference height [70]. The dominant wind direction was southeast (SE). The wind speed and pressure are analyzed under this scenario.

Figure 14 shows the wind speed and WAF at 1.5 m. Because the wind direction is close to the street direction inside the area, a significant wind amplification effect is observed near the street entrance, which makes pedestrian walking difficult. The wind pressures at 1.5 m, 15 m, and 50 m are shown in Figure 15. Under the tropical storm, some of the buildings facing streets experience a large pressure difference between the windward and leeward sides, which increases the risk of falling debris and thus requires attention.

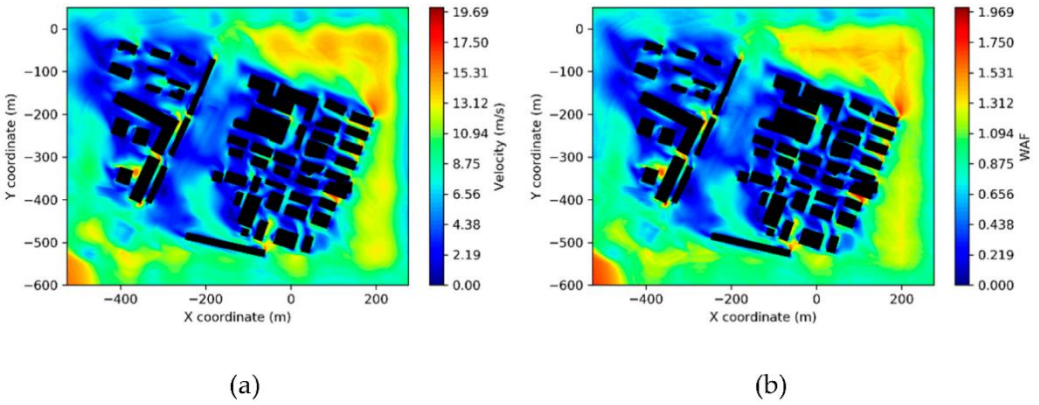

Figure 14. (**a**) Wind speed and (**b**) WAF at pedestrian height of 1.5 m (SE 14.1 m/s).

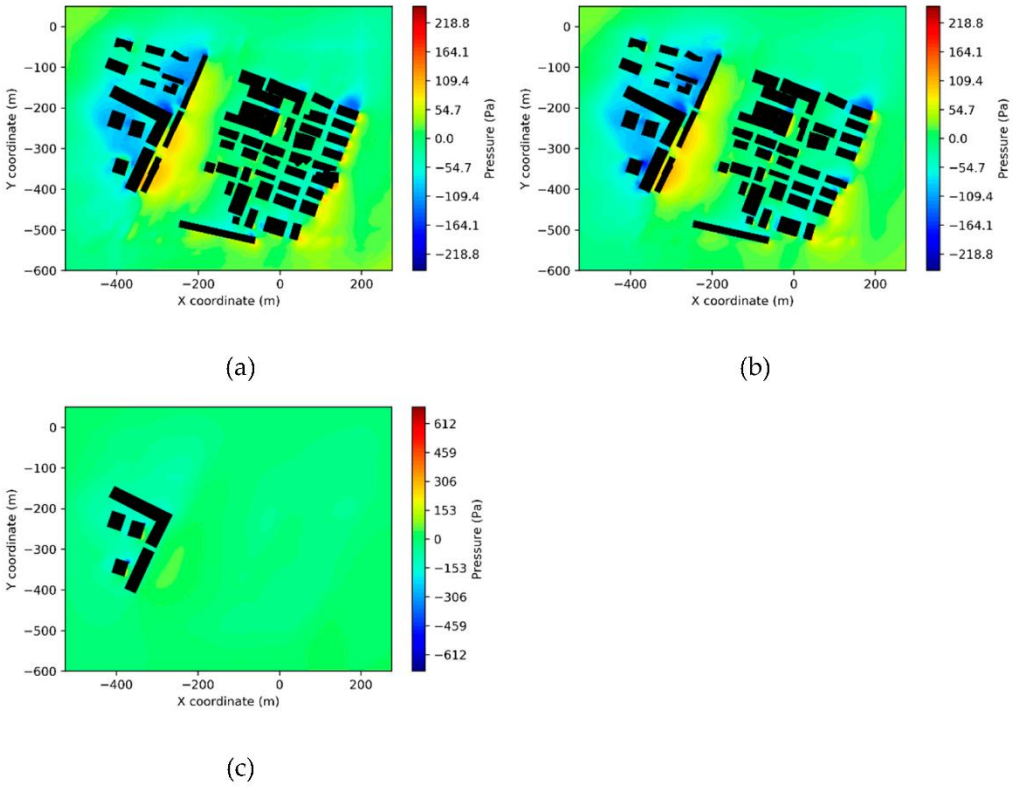

Figure 15. Wind pressures at different height levels: (**a**) 1.5 m, (**b**) 15 m, (**c**) 50 m (SE 14.1 m/s).

6. Discussion

6.1. Data Acquisition and Errors

The proposed framework adopts digital aerial photogrammetry as its data acquisition approach, considering that its cost is significantly lower than that of airborne laser scanning (ALS) using LiDAR [72]. That would enable a frequent implementation of the proposed method to consider changes in the CFD models. Nevertheless, noises exist in the point clouds generated by oblique photography due to reasons such as occlusion, parallax, texture loss, and lighting conditions [73]. Many studies have lain their emphasis on the issue of data noise [74,75]. However, common noise filtering algorithms for point clouds, such as statistical outlier removal and radius outlier removal, significantly rely on parameter settings. Unreasonable parameters would cause the point clouds to be sparse and thus lose geometric information. Since data noise is not the focus of this work, no noise filtering is applied before the semantic segmentation module to avoid an adverse effect on the local feature learning of the following classification models. Although the noise is not filtered in advance, the proposed method can be robust to outliers. That is because both the feature calculation required by SVM/RF and the ASPP(DeepLabv3)/SA(PointNet++) layer perform local feature extractions, which implicitly help identify the outliers. On the other hand, the building reconstruction first applies RANSAC to extract roof planes, equivalently performing a plane-wise outlier filtering. As for trees, the model of canopy fluid volumes is generated by the 2D alpha shape algorithm, making the detected boundary more sensitive to outliers than internal sparseness. Therefore, an outlier removal is performed in Algorithm 1. The noise of point clouds might also be brought by systematic errors. Gopalakrishnan et al. [76] used ALS data to address the vertical misregistration issue of the photogrammetry-based point clouds. This kind of method would be able to reduce the data noise, whose impact on the proposed method needs further study.

6.2. Efficiency

The segmentation experiment is carried out on a high-performance computer. The device is equipped with an Intel Xeon E5-2630 v3 @ 2.40GHz as CPU, an NVIDIA GeForce GTX1080 as GPU, and 64 GB RAM. The system is Ubuntu 20.04. The time cost of the proposed method mainly consists of two parts: the calculation time of DeepLabv3 and PointNet++. In the case study, the time cost is 1664 s during training, while being 147 s for the test. In comparison, the single DeepLabv3 method costs 344 s and 126 s for training and test respectively, while the figures for PointNet++ are 1316 s and 21 s. The SVM and RF methods spend a lot of time on point-wise feature calculation, which is 1377 s and 879 s for the training set and the test set, respectively. The training time for SVM is 221 s, while its test time is 0.1 s which can be ignored. As for RF, the training time is 191 s, 967 s, and 1939 s for RF with 10, 50, and 100 trees, respectively, while the test process costs 27 s, 124 s, and 261 s. It should be noted that the figures above are based on the same training and test set, and do not take into account the time for file I/O required by the implementation of the framework. In addition, the deep learning techniques leverage the GPU, while SVM and RF only use the CPU.

Although the proposed segmentation method slightly costs more time for an accuracy improvement, the time efficiency is competent in terms of the entire modeling framework. In the case study, using the point cloud segmentation and model reconstruction methods presented in this study, the CFD models are established by the well-trained model within minutes for a complex urban area including vegetation on a personal computer, saving hours of professional manual works on establishing the up to date as-built models. The CFD simulation considers complex buildings, vegetation, and their interaction effects. The proposed method significantly reduces the labor cost of CFD modeling of an urban wind environment, and professionals can focus more on the simulation and result analyses, which is surely beneficial to applications in wind disasters, such as fragility analysis, risk analysis, and decision making.

6.3. Geometric Quality

The geometric quality of the model is critical to an accurate CFD result. For the ground model, Gaussian process regression is used to reconstruct the digital surface model. If the accuracy needs to be improved, the sampling density on the surface can be increased, or methods such as Poisson surface reconstruction can be applied to build a high-precision surface. For building models, this work adopts common implementations in relevant regional simulation studies in terms of the level of detail [6,7]. Hågbo et al. [41] also showed that the adopted level of detail would have a limited impact on decision-making using wind environment analysis results. For canopy fluid volumes, Gu et al. [6] used the field measurement data to validate the model without vertical geometric changes, and the results showed that the simulation outcomes of the wind speed are within a standard deviation of the measurement. Thus, such a model is adopted in this work for simplicity. Normally, by increasing the level of details of the models, such as using building models established by the mesh surface reconstruction and canopy fluid models built hierarchically to consider vertical changes, the simulation accuracy would be enhanced. However, this would reduce modeling efficiency and result in higher requirements for the CFD meshing, which needs more numerical verifications.

7. Conclusions

CFD is an effective tool for the simulation of urban wind environments and can be employed to wind disaster analysis. A CFD simulation requires accurately and efficiently establishing clean and low-complexity models of a city, including terrain, buildings, and vegetation. This paper proposes an automated simulation framework for urban wind environments based on aerial point clouds and deep learning, and mainly focuses on the

modeling process. The practicality of the framework is validated by a case study on Bao'an, Shenzhen. The main conclusions are as follows:

(1) Compared with the traditional CFD modeling methods based on GISs, the automated method based on oblique photography point clouds can reflect the current environment of the target area and drastically reduce the labor cost.
(2) Compared to the point cloud semantic segmentation methods based on SVM, RF, or a single deep learning network, the proposed method combining 2D and 3D deep learning techniques achieves a higher accuracy, which provides more accurate classification results for the modeling process.
(3) The modeling method of the terrain, buildings, and canopy fluid volumes can retain general geometric characteristics of the objects while reducing the model complexity, which meets the requirements of CFD simulations.

Author Contributions: Conceptualization, X.L., F.Z. and C.S.; methodology, C.S.; software, C.S.; validation, P.Z.; data curation, X.Z.; writing—original draft preparation, C.S.; writing—review and editing, X.L. and Y.H.; funding acquisition, X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Central Research Institute of Building and Construction Co., Ltd., MCC Group, China (No. YYY2020Z047), and Tencent Foundation through XPLORER PRIZE.

Acknowledgments: The authors would like to appreciate Beijing PARATERA Tech Co., Ltd. for providing the computational support in this work, and also the anonymous reviewers for the constructive suggestions.

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

Appendix A

The inlet boundary is set as a velocity inlet and uses the power-law wind profile, as expressed in Equation (A1).

$$U(z) = U_{\mathrm{r}}\left(\frac{z}{z_{\mathrm{r}}}\right)^{\alpha}, \tag{A1}$$

where z_{r} and U_{r} represent the z coordinate and wind speed at the reference position, respectively; z_{r} is typically 10 m, and α is the power-law exponent, which is set as 0.22 for an urban area with dense buildings according to the Chinese code [8]. The surfaces of the terrain and buildings are set as nonslip boundaries and wall functions. Other domain boundaries are set as fixed-pressure boundaries. The Reynolds-averaged Navier–Stokes (RANS) model and the Chen–Kim k–ε turbulence model [77] are adopted for the computation.

The main impact of trees on the wind field is the reduction in the wind speed due to the drag forces and the additional turbulence levels produced by the canopies. These effects are simulated by adding a sink term to the momentum equation and source terms to the transport equations of k and ε. Equations (A2)–(A5) are the RANS equations of the incompressible flow used in the simulation. For the notations, the Einstein summation convention is used as follows:

$$\frac{\partial u_i}{\partial x_i} = 0, \tag{A2}$$

$$u_j\frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left[(\nu + \nu_t)\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}k\delta_{ij}\right] + S_d, \tag{A3}$$

$$\frac{\partial}{\partial x_i}(ku_i) = \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + \nu_t\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\frac{\partial u_i}{\partial x_j} - \varepsilon + S_k, \text{ and} \tag{A4}$$

$$\frac{\partial}{\partial x_i}(\varepsilon u_i) = \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\nu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] + C_{1\varepsilon}\frac{\varepsilon}{k}\nu_t\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\frac{\partial u_i}{\partial x_j} - C_{2\varepsilon}\frac{\varepsilon^2}{k} + S_\varepsilon^{CK} + S_\varepsilon, \quad \text{(A5)}$$

where u_i is the velocity (where i equals 1, 2, or 3), x_i denotes the spatial coordinates, ρ is the density; p is the pressure, δ_{ij} represents the Kronecker delta, ν is the viscosity, and ν_t is the eddy viscosity, which can be calculated based on the kinetic energy, k, and the turbulent energy dissipation rate, ε [78], as follows:

$$\nu_t = C_\mu \frac{k^2}{\varepsilon}. \quad \text{(A6)}$$

S_ε^{CK} is the source term introduced by Chen and Kim [77].

$$S_\varepsilon^{CK} = -\frac{C_{4\varepsilon}}{k}\left[\nu_t\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\frac{\partial u_i}{\partial x_j}\right]^2. \quad \text{(A7)}$$

S_d is the sink term associated with the drag forces of the tree canopies, and S_k and S_ε are the source terms introduced to account for the turbulent interaction between the airflow and the tree canopies.

$$S_d = -C_d\alpha|\boldsymbol{u}|u_i, \quad \text{(A8)}$$

$$S_k = C_d\alpha(\beta_p|\boldsymbol{u}|^3 - \beta_d|\boldsymbol{u}|k), \text{ and} \quad \text{(A9)}$$

$$S_\varepsilon = C_d\alpha\left(C_{5\varepsilon}\beta_p|\boldsymbol{u}|^3\frac{\varepsilon}{k} - C_{6\varepsilon}\beta_d|\boldsymbol{u}|\varepsilon\right), \quad \text{(A10)}$$

where α is the leaf area density (LAD), and $|\boldsymbol{u}|$ is the scalar value of the velocity vector. Various studies have proposed different parameter values to consider the influence of tree canopies [79–82]; the value suggested by Green [79] is adopted in this study. Because of the lack of detailed data on vegetation, the LAD is typically 4.0 and is assumed to be vertically invariant [83], and the drag coefficient, C_d, is set as 0.2 [7]. The constants used in the numerical equations are listed in Table A1.

Table A1. Constants used in numerical equations for modeling effects of trees on airflow.

C_μ	σ_k	σ_ε	$C_{1\varepsilon}$	$C_{2\varepsilon}$	$C_{4\varepsilon}$	$C_{5\varepsilon}$	$C_{6\varepsilon}$	β_p	β_d	C_d	α
0.09	0.75	1.15	1.15	1.9	0.25	1.50	1.50	1.00	4.00	0.20	4.00

References

1. Blocken, B. Computational Fluid Dynamics for Urban Physics: Importance, Scales, Possibilities, Limitations and Ten Tips and Tricks towards Accurate and Reliable Simulations. *Build. Environ.* **2015**, *91*, 219–245, doi:10.1016/j.buildenv.2015.02.015.
2. Zhang, Y.; Wei, K.; Shen, Z.; Bai, X.; Lu, X.; Soares, C.G. Economic Impact of Typhoon-Induced Wind Disasters on Port Operations: A Case Study of Ports in China. *Int. J. Disaster Risk Reduct.* **2020**, *50*, 101719, doi:10.1016/j.ijdrr.2020.101719.
3. Song, B.; Galasso, C.; Garciano, L. Wind-Uplift Fragility Analysis of Roof Sheathing for Cultural Heritage Assets in the Philippines. *Int. J. Disaster Risk Reduct.* **2020**, *51*, 101753, doi:10.1016/j.ijdrr.2020.101753.
4. Schulman, L.L.; DesAutels, C.G. Computational Fluid Dynamics Simulations to Predict Wind-Induced Damage to a Steel Building during Hurricane Katrina. In *Forensic Engineering 2012*; American Society of Civil Engineers: San Francisco, CA, USA, 2012; pp. 793–800; doi:10.1061/9780784412640.084.
5. Blocken, B.; Janssen, W.D.; van Hooff, T. CFD Simulation for Pedestrian Wind Comfort and Wind Safety in Urban Areas: General Decision Framework and Case Study for the Eindhoven University Campus. *Environ. Model. Softw.* **2012**, *30*, 15–34, doi:10.1016/j.envsoft.2011.11.009.
6. Gu, D.; Zhao, P.; Chen, W.; Huang, Y.; Lu, X. Near Real-Time Prediction of Wind-Induced Tree Damage at a City Scale: Simulation Framework and Case Study for Tsinghua University Campus. *Int. J. Disaster Risk Reduct.* **2021**, *53*, 102003, doi:10.1016/j.ijdrr.2020.102003.
7. Amorim, J.H.; Rodrigues, V.; Tavares, R.; Valente, J.; Borrego, C. CFD Modelling of the Aerodynamic Effect of Trees on Urban Air Pollution Dispersion. *Sci. Total. Environ.* **2013**, *461*, 541–551, doi:10.1016/J.SCITOTENV.2013.05.031.
8. Ministry of Housing and Urban-Rural Development of the People's Republic of China. *Standard for Green Performance Calculation of Civil. Buildings (JGJ/T 449-2018)*; China Architecture & Building Press: Beijing, China, 2018.
9. Gu, D.; Zheng, Z.; Zhao, P.; Xie, L.; Xu, Z.; Lu, X. High-Efficiency Simulation Framework to Analyze the Impact of Exhaust Air from COVID-19 Temporary Hospitals and Its Typical Applications. *Appl. Sci.* **2020**, *10*, 3949, doi:10.3390/app10113949.

10. Dalla Mura, M.; Prasad, S.; Pacifici, F.; Gamba, P.; Chanussot, J.; Benediktsson, J.A. Challenges and Opportunities of Multimodality and Data Fusion in Remote Sensing. *Proc. IEEE* **2015**, *103*, 1585–1601, doi:10.1109/JPROC.2015.2462751.
11. Jochem, W.C.; Bird, T.J.; Tatem, A.J. Identifying Residential Neighbourhood Types from Settlement Points in a Machine Learning Approach. *Comput. Environ. Urban. Syst.* **2018**, *69*, 104–113, doi:10.1016/j.compenvurbsys.2018.01.004.
12. Hecht, R.; Meinel, G.; Buchroithner, M. Automatic Identification of Building Types Based on Topographic Databases—A Comparison of Different Data Sources. *Int. J. Cartogr.* **2015**, *1*, 18–31, doi:10.1080/23729333.2015.1055644.
13. Shirowzhan, S.; Sepasgozar, S.M.E.; Li, H.; Trinder, J.; Tang, P. Comparative Analysis of Machine Learning and Point-Based Algorithms for Detecting 3D Changes in Buildings over Time Using Bi-Temporal Lidar Data. *Autom. Constr.* **2019**, *105*, 102841, doi:10.1016/j.autcon.2019.102841.
14. Chen, J.; Tang, P.; Rakstad, T.; Patrick, M.; Zhou, X. Augmenting a Deep-Learning Algorithm with Canal Inspection Knowledge for Reliable Water Leak Detection from Multispectral Satellite Images. *Adv. Eng. Inform.* **2020**, *46*, 101161, doi:10.1016/j.aei.2020.101161.
15. Cooner, A.J.; Shao, Y.; Campbell, J.B. Detection of Urban Damage Using Remote Sensing and Machine Learning Algorithms: Revisiting the 2010 Haiti Earthquake. *Remote. Sens.* **2016**, *8*, 868, doi:10.3390/rs8100868.
16. Gomez, C.; Purdie, H. UAV-Based Photogrammetry and Geocomputing for Hazards and Disaster Risk Monitoring—A Review. *Geoenviron. Disasters* **2016**, *3*, 23, doi:10.1186/s40677-016-0060-y.
17. Xiong, C.; Li, Q.; Lu, X. Automated Regional Seismic Damage Assessment of Buildings Using an Unmanned Aerial Vehicle and a Convolutional Neural Network. *Autom. Constr.* **2020**, *109*, 102994, doi:10.1016/j.autcon.2019.102994.
18. Zhou, Y.; Wang, L.; Love, P.E.D.; Ding, L.; Zhou, C. Three-Dimensional (3D) Reconstruction of Structures and Landscapes: A New Point-and-Line Fusion Method. *Adv. Eng. Inform.* **2019**, *42*, 100961, doi:10.1016/j.aei.2019.100961.
19. Sun, X.; Shen, S.; Hu, Z. Automatic Building Extraction from Oblique Aerial Images. In Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 4–8 December 2016; pp. 663–668, doi:10.1109/ICPR.2016.7899710.
20. Tomljenovic, I.; Tiede, D.; Blaschke, T. A Building ExSegmentation of Airborne Point Cloud Data for Autcanner Data Utilizing the Object Based Image Analysis Paradigm. *Int. J. Appl. Earth Obs. Geoinf.* **2016**, *52*, 137–148, doi:10.1016/j.jag.2016.06.007.
21. Gilani, S.A.N.; Awrangjeb, M.; Lu, G. Segmentation of Airborne Point Cloud Data for Automatic Building Roof Extraction. *GIScience Remote. Sens.* **2018**, *55*, 63–89, doi:10.1080/15481603.2017.1361509.
22. Verma, V.; Kumar, R.; Hsu, S. 3D Building Detection and Modeling from Aerial LIDAR Data. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, 17–22 June 2006; Volume 2, pp. 2213–2220, doi:10.1109/CVPR.2006.12.
23. Anuar, S.F.K.; Nasir, A.A.M.; Azri, S.; Ujang, U.; Majid, Z.; González Cuétara, M.; de Miguel Retortillo, G. 3D Geometric Extraction Using Segmentation for Asset Management. In *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*; Copernicus GmbH: Göttingen, Germany, 2020; Volume XLIV-4-W3-2020, pp. 61–69; doi:10.5194/isprs-archives-XLIV-4-W3-2020-61-2020.
24. Zhou, Q. 3D Urban Modeling from City-Scale Aerial LiDAR Data. Ph.D. Thesis, University of Southern California, Los Angeles, CA, USA, 2012.
25. Rida, I.; Almaadeed, N.; Almaadeed, S. Robust Gait Recognition: A Comprehensive Survey. *IET Biom.* **2019**, *8*, 14–28, doi:10.1049/iet-bmt.2018.5063.
26. Long, J.; Shelhamer, E.; Darrell, T. Fully Convolutional Networks for Semantic Segmentation. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3431–3440, doi:10.1109/CVPR.2015.7298965.
27. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015*; Springer International Publishing: Cham, 2015; pp. 234–241; doi:10.1007/978-3-319-24574-4_28.
28. Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. *arXiv* **2016**, *arXiv:1412.7062*.
29. Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 834–848, doi:10.1109/TPAMI.2017.2699184.
30. Chen, L.-C.; Papandreou, G.; Schroff, F.; Adam, H. Rethinking Atrous Convolution for Semantic Image Segmentation. *arXiv* **2017**, arXiv:1706.05587.
31. Maturana, D.; Scherer, S. VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; pp. 922–928, doi:10.1109/IROS.2015.7353481.
32. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; 77–85, doi:10.1109/CVPR.2017.16.
33. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2017; pp. 5105–5114.

34. Zhang, Y.; Huo, K.; Liu, Z.; Zang, Y.; Liu, Y.; Li, X.; Zhang, Q.; Wang, C. PGNet: A Part-Based Generative Network for 3D Object Reconstruction. *Knowl. Based Syst.* **2020**, *194*, 105574, doi:10.1016/j.knosys.2020.105574.
35. Kim, H.; Yoon, J.; Sim, S.-H. Automated Bridge Component Recognition from Point Clouds Using Deep Learning. *Struct. Control. Health Monit.* **2020**, *27*, e2591, doi:10.1002/stc.2591.
36. Lowphansirikul, C.; Kim, K.; Vinayaraj, P.; Tuarob, S. 3D Semantic Segmentation of Large-Scale Point-Clouds in Urban Areas Using Deep Learning. In Proceedings of the 2019 11th International Conference on Knowledge and Smart Technology (KST), Phuket, Thailand, 23–26 January 2019; pp. 238–243, doi:10.1109/KST.2019.8687813.
37. Edelsbrunner, H.; Kirkpatrick, D.; Seidel, R. On the Shape of a Set of Points in the Plane. *IEEE Trans. Inf. Theory* **1983**, *29*, 551–559, doi:10.1109/TIT.1983.1056714.
38. Bernardini, F.; Mittleman, J.; Rushmeier, H.; Silva, C.; Taubin, G. The Ball-Pivoting Algorithm for Surface Reconstruction. *IEEE Trans. Vis. Comput. Graph.* **1999**, *5*, 349–359, doi:10.1109/2945.817351.
39. Kazhdan, M.; Bolitho, M.; Hoppe, H. Poisson Surface Reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*; Eurographics Association: Goslar, Germany, 2006; pp. 61–70.
40. Xu, Z.; Wu, Y.; Lu, X.; Jin, X. Photo-Realistic Visualization of Seismic Dynamic Responses of Urban Building Clusters Based on Oblique Aerial Photography. *Adv. Eng. Inform.* **2020**, *43*, 101025, doi:10.1016/j.aei.2019.101025.
41. Hågbo, T.-O.; Giljarhus, K.E.T.; Hjertager, B.H. Influence of Geometry Acquisition Method on Pedestrian Wind Simulations. *arXiv* **2020**, arXiv:2010.12371.
42. Chen, L.; Teo, T.; Rau, J.; Liu, J.; Hsu, W. Building Reconstruction from LIDAR Data and Aerial Imagery. In Proceedings of the 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. (IGARSS '05), Seoul, Korea, 29–29 July 2005; Volume 4, pp. 2846–2849, doi:10.1109/IGARSS.2005.1525661.
43. Wang, X.; Chan, T.O.; Liu, K.; Pan, J.; Luo, M.; Li, W.; Wei, C. A Robust Segmentation Framework for Closely Packed Buildings from Airborne LiDAR Point Clouds. *Int. J. Remote. Sens.* **2020**, *41*, 5147–5165, doi:10.1080/01431161.2020.1727053.
44. Lu, X.; Guo, Q.; Li, W.; Flanagan, J. A Bottom-up Approach to Segment Individual Deciduous Trees Using Leaf-off Lidar Point Cloud Data. *ISPRS J. Photogramm. Remote. Sens.* **2014**, *94*, 1–12, doi:10.1016/j.isprsjprs.2014.03.014.
45. Lin, Y.; Jiang, M.; Yao, Y.; Zhang, L.; Lin, J. Use of UAV Oblique Imaging for the Detection of Individual Trees in Residential Environments. *Urban. For. Urban. Green.* **2015**, *14*, 404–412, doi:10.1016/j.ufug.2015.03.003.
46. Wang, Y.; Weinacker, H.; Koch, B. A Lidar Point Cloud Based Procedure for Vertical Canopy Structure Analysis and 3D Single Tree Modelling in Forest. *Sensors* **2008**, *8*, 3938–3951, doi:10.3390/s8063938.
47. Phoenics. Available online: http://www.cham.co.uk/phoenics.php (accessed on 5 January 2021).
48. Zhang, W.; Qi, J.; Wan, P.; Wang, H.; Xie, D.; Wang, X.; Yan, G. An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation. *Remote. Sens.* **2016**, *8*, 501, doi:10.3390/rs8060501.
49. CloudCompare: 3D Point Cloud and Mesh Processing Software, Open Source Project. Available online: https://www.danielgm.net/cc/ (accessed on 30 November 2020).
50. Vasudevan, S.; Ramos, F.; Nettleton, E.; Durrant-Whyte, H.; Blair, A. Gaussian Process Modeling of Large Scale Terrain. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 1047–1053, doi:10.1109/ROBOT.2009.5152677.
51. Schnabel, R.; Wahl, R.; Klein, R. Efficient RANSAC for Point-Cloud Shape Detection. *Comput. Graph. Forum* **2007**, *26*, 214–226, doi:10.1111/j.1467-8659.2007.01016.x.
52. Poullis, C. A Framework for Automatic Modeling from Point Cloud Data. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 2563–2575, doi:10.1109/TPAMI.2013.64.
53. Boykov, Y.; Veksler, O.; Zabih, R. Fast Approximate Energy Minimization via Graph Cuts. *IEEE Trans. Pattern Anal. Mach. Intell.* **2001**, *23*, 1222–1239, doi:10.1109/34.969114.
54. Kolmogorov, V.; Zabin, R. What Energy Functions Can Be Minimized via Graph Cuts? *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 147–159, doi:10.1109/TPAMI.2004.1262177.
55. Boykov, Y.; Kolmogorov, V. An Experimental Comparison of Min-Cut/Max- Flow Algorithms for Energy Minimization in Vision. *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 1124–1137, doi:10.1109/TPAMI.2004.60.
56. Delong, A.; Osokin, A.; Isack, H.N.; Boykov, Y. Fast Approximate Energy Minimization with Label Costs. *Int. J. Comput. Vis.* **2012**, *96*, 1–27, doi:10.1007/s11263-011-0437-z.
57. Ramer, U. An Iterative Procedure for the Polygonal Approximation of Plane Curves. *Comput. Graph. Image Process.* **1972**, *1*, 244–256, doi:10.1016/S0146-664X(72)80017-0.
58. Douglas, D.; Peucker, T. Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature. *Cartogr. Int. J. Geogr. Inf. Geovisualization* **1973**, *10*, 112–122, doi:10.3138/FM57-6770-U75U-7727.
59. Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*; AAAI Press: Portland, OR, USA, 1996; pp 226–231.
60. Pio, G.; Ceci, M.; Loglisci, C.; D'Elia, D.; Malerba, D. Hierarchical and Overlapping Co-Clustering of mRNA: MiRNA Interactions. In *Proceedings of the 20th European Conference on Artificial Intelligence*; IOS Press: Amsterdam, The Netherlands, 2012; pp 654–659.
61. Slonim, N.; Aharoni, E.; Crammer, K. Hartigan's K-Means versus Lloyd's K-Means: Is It Time for a Change? In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*; AAAI Press: Beijing, China, 2013; pp 1677–1684.

62. 3ds Max: 3D Modeling, Animation & Rendering Software. Available online: https://www.autodesk.com/products/3ds-max/overview (accessed on 5 January 2021).
63. Bentley Systems, Create 3D Models from Simple Photographs. Available online: https://www.bentley.com/en/products/brands/contextcapture (accessed on 30 November 2020).
64. Scikit-Learn: Machine Learning in Python. Available online: https://scikit-learn.org/stable/ (accessed on 7 January 2021).
65. PyTorch: An Open Source Machine Learning Framework that Accelerates the Path from Research Prototyping to Production Deployment. Available online: https://www.pytorch.org (accessed on 30 November 2020).
66. Guo, F.; Zhu, P.; Wang, S.; Duan, D.; Jin, Y. Improving Natural Ventilation Performance in a High-Density Urban District: A Building Morphology Method. *Proced. Eng.* **2017**, *205*, 952–958, doi:10.1016/j.proeng.2017.10.149.
67. Li, X.; Wang, J.; Eftekhari, M.; Qi, Q.; Jiang, D.; Song, Y.; Tian, P. Improvement Strategies Study for Outdoor Wind Environment in a University in Beijing Based on CFD Simulation. *Adv. Civ. Eng.* **2020**, *2020*, e8850254, doi:10.1155/2020/8850254.
68. Alhasan, W.; Yuning, C. Environmental Analysis of Nanjing Mosque Courtyard Layout Based on CFD Simulation Technology. *E3S Web Conf.* **2019**, *136*, 04040, doi:10.1051/e3sconf/201913604040.
69. Tominaga, Y.; Mochida, A.; Yoshie, R.; Kataoka, H.; Nozu, T.; Yoshikawa, M.; Shirasawa, T. AIJ Guidelines for Practical Applications of CFD to Pedestrian Wind Environment around Buildings. *J. Wind. Eng. Ind. Aerodyn.* **2008**, *96*, 1749–1761, doi:10.1016/j.jweia.2008.02.058.
70. Meteorological Bureau of Shenzhen Municipality. *Shenzhen Climate Bulletin 2019*; Meteorological Bureau of Shenzhen Municipality: Shenzhen, China, 2019.
71. Ministry of Housing and Urban-Rural Development of the People's Republic of China. *Assessment Standard for Green Building (GB/T 50378-2019)*; China Architecture & Building Press: Beijing, China, 2019.
72. White, J.C.; Wulder, M.A.; Vastaranta, M.; Coops, N.C.; Pitt, D.; Woods, M. The Utility of Image-Based Point Clouds for Forest Inventory: A Comparison with Airborne Laser Scanning. *Forests* **2013**, *4*, 518–536, doi:10.3390/f4030518.
73. Zhu, Q.; Li, S.; Hu, H.; Zhong, R.; Wu, B.; Xie, L. Multiple Point Clouds Data Fusion Method for 3D City Modeling. *Geomat. Inf. Sci. Wuhan Univ.* **2018**, *43*, 1962–1971. (In Chinese).
74. Despotović, I.; Jelača, V.; Vansteenkiste, E.; Philips, W. Noise-Robust Method for Image Segmentation. In *Advanced Concepts for Intelligent Vision Systems*; Lecture Notes in Computer Science; Blanc-Talon, J., Bone, D., Philips, W., Popescu, D., Scheunders, P., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 153–162; doi:10.1007/978-3-642-17688-3_16.
75. Corizzo, R.; Ceci, M.; Japkowicz, N. Anomaly Detection and Repair for Accurate Predictions in Geo-Distributed Big Data. *Big Data Res.* **2019**, *16*, 18–35, doi:10.1016/j.bdr.2019.04.001.
76. Gopalakrishnan, R.; Ali-Sisto, D.; Kukkonen, M.; Savolainen, P.; Packalen, P. Using ALS Data to Improve Co-Registration of Photogrammetry-Based Point Cloud Data in Urban Areas. *Remote. Sens.* **2020**, *12*, 1943, doi:10.3390/rs12121943.
77. Chen, Y.-S.; Kim, S.-W. Computation of Turbulent Flows Using an Extended K-Epsilon Turbulence Closure Model. *NASA STI/Recon Tech. Rep. N* **1987**, *88*, 11969.
78. Launder, B.E.; Spalding, D.B. The Numerical Computation of Turbulent Flows. *Comput. Methods Appl. Mech. Eng.* **1974**, *3*, 269–289, doi:10.1016/0045-7825(74)90029-2.
79. Green, R.S. Modelling Turbulent Air Flow in a Stand of Widely-Spaced Trees. *Phoenics J.* **1992**, *5*, 294–312.
80. Liu, J.; Chen, J.M.; Black, T.A.; Novak, M.D. E-ε Modelling of Turbulent Air Flow Downwind of a Model Forest Edge. *Bound. Layer Meteorol.* **1996**, *77*, 21–44, doi:10.1007/BF00121857.
81. Svensson, U.; Häggkvist, K. A Two-Equation Turbulence Model for Canopy Flows. *J. Wind. Eng. Ind. Aerodyn.* **1990**, *35*, 201–211, doi:10.1016/0167-6105(90)90216-Y.
82. Sanz, C. A Note on k-ε Modelling of Vegetation Canopy Air-Flows. *Bound. Layer Meteorol.* **2003**, *108*, 191–197, doi:10.1023/A:1023066012766.
83. Huang, J.; Cassiani, M.; Albertson, J.D. The Effects of Vegetation Density on Coherent Turbulent Structures within the Canopy Sublayer: A Large-Eddy Simulation Study. *Bound. Layer Meteorol.* **2009**, *133*, 253–275, doi:10.1007/s10546-009-9423-1.