remote sensing

MDPI

Article

Semi-Automated Segmentation of Geometric Shapes from Point Clouds

Richard Honti *, Ján Erdélyi and Alojz Kopáčik

Department of Surveying, Faculty of Civil Engineering, Slovak University of Technology in Bratislava, 810 05 Bratislava, Slovakia

* Correspondence: richard.honti@stuba.sk

Abstract: Building information models (BIM) in the civil industry are very popular nowadays. The basic information of these models is the 3D geometric model of a building structure. The most applied methodology to model the existing buildings is by generating 3D geometric information from point clouds provided by laser scanners. The fundamental principle of this methodology is the recognition of structures shaped in basic geometric primitives, e.g., planes, spheres, and cylinders. The basic premise of the efficiency of this methodology is the automation of detection, since manual segmentation of a point cloud can be challenging, time-consuming, and, therefore, inefficient. This paper presents a novel algorithm for the automated segmentation of geometric shapes in point clouds without needing pre-segmentation. With the designed algorithm, structures formed in three types of basic geometrical primitive can be identified and segmented: planar (e.g., walls, floors, ceilings), spherical (e.g., laser scanner reference targets), and cylindrical (e.g., columns, pillars, piping). The RANSAC paradigm partially inspires the proposed algorithm; however, various modifications must be made. The algorithm was tested on several point clouds and was compared with the standard RANSAC algorithm; this part is described in the last section of the paper. One of the tests was performed on a double cylinder-shaped test object, the parameters (radius and height) of this object were available with high accuracy (0.1 mm), and the differences between the known and estimated parameters were below 0.5 mm in each case, indicating the correctness of the proposed algorithm. Also, a comparison with the standard RANSAC algorithm was performed, where the algorithm proposed showed better results than the standard RANSAC algorithm. The segmentation quality was, on average, increased from 50% to 100%.

Keywords: point cloud; automated detection; geometric shapes; plane; sphere; cylinder; point cloud segmentation

Citation: Honti, R.; Erdélyi, J.; Kopáčik, A. Semi-Automated Segmentation of Geometric Shapes from Point Clouds. *Remote Sens.* **2022**, *14*, 4591. https://doi.org/10.3390/rs14184591

Academic Editor: Krzysztof Stereńczak

Received: 15 July 2022
Accepted: 11 September 2022
Published: 14 September 2022

1. Introduction

Point clouds are becoming an increasingly common digital representation of real-world objects. They are the results of laser scanning or photogrammetry. With the increased availability of instruments needed for measurement, the popularity of point cloud usage is also growing. Point clouds can have an important role in creating high-quality 3D models of objects in a variety of areas, e.g., interior (exterior) design, building information modeling (BIM), urban information systems, documentation of objects [1], 3D cadaster, deformation analysis [2,3], etc. With the currently available technology, massive data sets (millions of points) can be collected relatively easily and in a short time. The next step in the information retrieval process is point cloud processing. The term processing often means initial adjustment (registration, filtration) and generation of 3D geometric information from point clouds.

The three most frequently occurring geometric shapes in in-built objects are planes, cylinders, and spheres. Since, in most cases, objects in the industrial environment consist of basic geometrical shapes, detecting and segmenting these shapes can be an essential step in

the data processing. The detected shapes can be used in a variety of tasks, e.g., simplified 3D model creation, BIM model generation (Scan-to-BIM), reverse engineering, terrestrial laser scanning (TLS) calibration, camera calibration, point cloud registration, simultaneous localization and mapping (SLAM) [4,5], etc.

Identification and segmentation of geometric primitives in point clouds has been investigated for a long time. The methods and algorithms that have been suggested can generally be divided into five categories [6,7]:

- **Edge-based methods** are based on detecting the boundaries of separate sections in a point cloud to obtain some segmented regions, for example [8,9]. These methods give us the ability for fast segmentation, but in most cases, they are susceptible to noise and uneven point cloud density, which is often the case [6,7].
- **Region-based methods** use neighborhood information to merge close points with identical properties to obtain segregated regions and consequently find dissimilarities between separate regions. A well-known approach from this category is the region-growing method [10], also applied in [11,12]. The region-growing method is based on the similarity between the neighboring points. Its first step is to select a seed point, and the regions grow from these points to adjacent points depending on a specified criterion. In most cases, some local surface smoothness index defines the requirements for point cloud segmentation. The result of this method is a set of segments, where each segment is a set of points that are considered as a part of the same smooth surface. An octree-based region-growing methodology was introduced in [11] by Vo et al. The region-based methods are generally less sensitive to noise than edge-based methods. In addition, they can be relatively simple and can be applied on large point clouds. Conversely, the disadvantage of these methods is that the result hardly depends on the choice of the seed point, local surface characteristics, and the choice of threshold values [6,7].
- **Attribute-based methods** (alternatively clustering-based methods) consist of two separate steps. The first step is the computation of the attribute, and in the second step, the point cloud is clustered based on the attributes computed, e.g., [13,14]. These methods usually are not based on a specific mathematical theory. The clustering-based methods can be divided into hierarchical (e.g., [15]) and non-hierarchical (e.g., [16]). Furthermore, attribute-based methods are often combined with other segmentation methods, such as region-based methods in [12]. The advantage of these methods is that they can also be employed for irregular objects, e.g., vegetation. Moreover, these approaches do not require seed points, unlike other methods like region-growing. Three of the most used algorithms from this category are *K*-means, mean shift, and fuzzy clustering [17].
- **Model-based methods** use geometric shapes (e.g., planes, spheres, cylinders, and cones) to organize points. Points that have the same mathematical representation are grouped as one segment. Two of the algorithms most widely used in this category are the random sample consensus (RANSAC) [18] and the Hough transform (HT) [19]. Various modifications of the original RANSAC algorithm can be found in [20–22]. The advantage of these methods is that they can be applied to noisy and complex point clouds. However, segmentation can be time-consuming in the case of large, rugged point clouds [6,7]. RANSAC is an iterative method for estimating the parameters of a mathematical model from an observed data set that contains outliers. The algorithm works by identifying the inliers in a data set and estimating the desired model using the data that do not contain outliers; therefore, the RANSAC paradigm extracts shapes from the point data and constructs the corresponding primitive forms based on the notion of minimal sets [18]. Authors Li et al. [23] made RANSAC the method of choice for fitting primitives in a point cloud in their approach called Globfit. Tran et al. [24] also focused on reliable estimation using RANSAC. The Hough transform is a well-known technique initially developed to extract straight lines; since then, it has been extended to extract parametric and nonparametric shapes. The main challenges of

the HT-based approaches are the memory requirements and the computation time needed. Moreover, it is also a sequential method that cannot detect multiple shapes simultaneously [25]. For example, Drost and Ilic [26] used the local Hough transform to detect cylinders, planes, and spheres in point clouds.

- **Graph-based methods** deal with point clouds in terms of a graph. For example, Strom et al. [27] extended a graph-based method to segment colored 3D laser data. Other approaches for segmentation-based on graph-based methods are presented in [28–31].

In recent years, numerous methods and approaches based on deep learning have been introduced for point cloud processing, e.g., 3D shape classification, 3D object detection and tracking, and 3D point cloud segmentation [32]. In general, the segmentation methods can be divided into four groups: projection-based, discretization-based, point-based, and hybrid. Some important approaches are presented in [33–35].

In some shape segmentation approaches, the point cloud of each structural element is first manually separated from the point cloud of the whole scanned structure. This step significantly reduces the processing efficiency and increases the time required.

This paper proposes an algorithm capable of automated detection and segmentation of several geometric shapes at once without the need for pre-segmentation. The segmentation process is effective in the case of complex point clouds with uneven density and a large number of outliers and noise. The proposed seed point selection technique and validation steps minimize the results' dependency on the seed point's choice and the local surface characteristics in the neighborhood of this point. The proposed algorithm combines the modified RANSAC algorithm with the region-growing method and the seed point selection technique, proposed based on local normal vector variation for each shape type. Three types of shapes can be segmented with the algorithm presented: planes, spheres, and cylinders.

The proposed approach for geometric shape segmentation from the point cloud is described in detail in the following section. After that, the results of testing the proposed method on several point clouds are illustrated.

2. Methodology

2.1. Proposed Approach for Point Cloud Segmentation

The following chapter proposes a robust algorithm for automated segmentation of geometric shapes (planes, spheres, cylinders) from point clouds. The proposed method can be applied to processing high-density point clouds. The algorithm process is shown in Figure 1. The input data for the algorithm is a point cloud. The next step is selecting the type of geometric shapes for segmentation and threshold values.

With the algorithm developed, it is possible to perform segmentation of only the picked shapes (e.g., only planes, only spheres, only cylinders) or their combination. The threshold type is the same for all types of geometric shapes, but their value must be selected separately for each type. The mentioned threshold parameters are the assumed maximal number of a given shape in the point cloud and threshold values for the distance- and normal-based filtering. The meaning of each parameter will be explained in the related chapters.

In the case of segmentation of planes, spheres, and cylinders at once, the algorithm process is illustrated by the flowchart in Figure 1. Before the segmentation procedure itself, some preprocessing steps are performed. First, the normal vectors at each point of the point cloud are calculated using small local planes, calculated from the 3D coordinates of the given point and the k-nearest neighbors. Orthogonal regression is used for plane estimation.

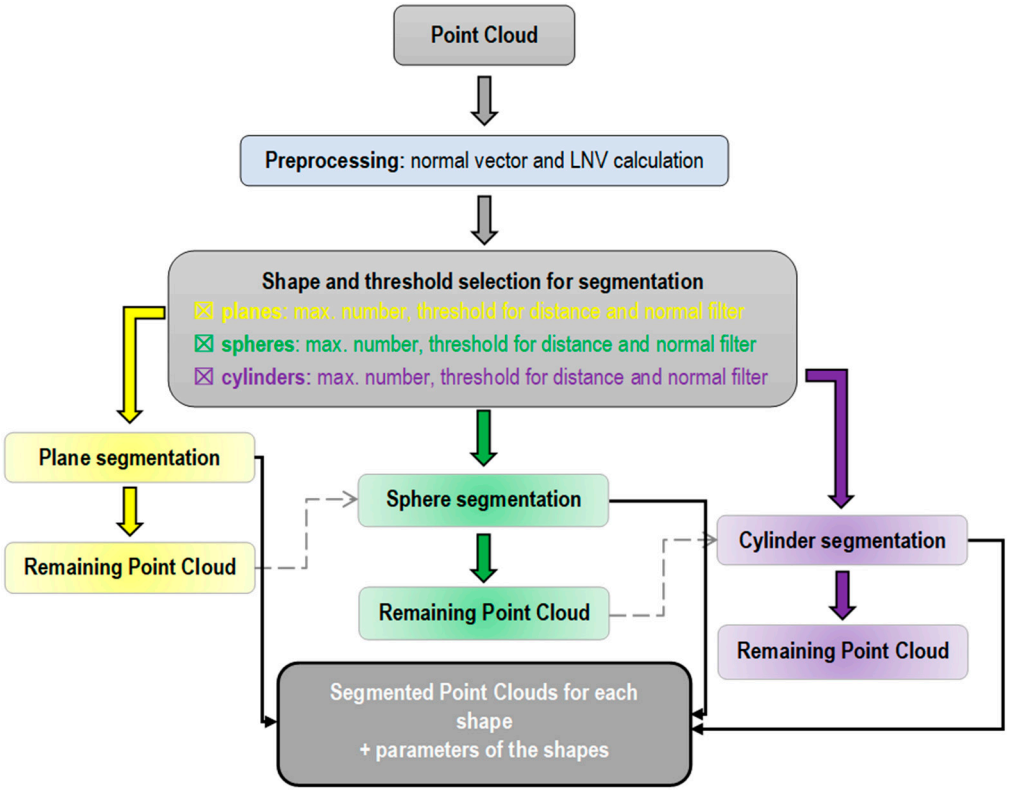

Figure 1. Flowchart of the algorithm proposed.

After the normal vectors are calculated, a new seed point selection technique is performed based on the local normal variation (LNV) value. The LNV value is estimated as the average value of the scalar products of the normal vectors from the *k*-nearest points based on Equation (1):

$$\boldsymbol{LNV}_i = 1 - \frac{1}{n}\sum_{i=1}^{n} abs(\boldsymbol{dotNorm}_i), \quad (1)$$

where $abs(\boldsymbol{dotNorm}_i)$ is the absolute value of the scalar product of the normal vector at a given point and the normal vectors at the neighboring points.

Based on the LNV values at each point, it is possible to approximately determine the points where the occurrence of a planar surface is assumed (or, on the contrary, where a curved surface is expected to occur). Thus, points on a planar surface have LNV values close to zero, while points on a curved surface have higher values of LNV. In Equation (1), the LNV is calculated as a unitless parameter, but for better understanding and imagination, it is expressed in degrees in the following sections.

The whole segmentation procedure is divided into three main stages based on the shape types and is described in the following subsections.

2.2. Plane Segmentation

Of the three shape types, the task of plane segmentation is the simplest, so the algorithm starts with this task. The plane segmentation algorithm combines a modified RANSAC algorithm and the region-growing method.

The plane segmentation starts with the selection of the seed points. The seed point candidates are determined based on the proposed seed point selection technique. The selected seed points are the points where the value of LNV is the points where the value of LNV is less than 1° (i.e., the orientation of the normal vector at the given point is approximately parallel to the normal vectors at the k-nearest neighbors). This seed point is used as a starting point for the plane estimation. The first plane is estimated using the

n_{est} nearest points. The number (n_{est}) of the nearest points depends on the local point density (LPD) at the selected seed point. In the proposed approach, this means selecting the points at a distance of 50 mm from the seed point. For the estimation of plane parameters, orthogonal regression is used, which minimizes the orthogonal distances to the estimated plane. The solution is based on the general equation of a plane (Equation (2)).

$$a \cdot X + b \cdot Y + c \cdot Z + d = 0, \tag{2}$$

where a, b, and c are the parameters of the normal vector of the plane; X, Y, and Z are the coordinates of a point lying on the plane; and d is the scalar product of the normal vector of the plane and the position vector of any point of the plane.

First, the elements of the best-fit regression plane's normal vector (a, b, and c) from the selected points are calculated using singular value decomposition (SVD). Next, the parameter d is calculated by substituting the coordinates of the center point and the elements of the normal vector into the rearranged form of the general equation of the plane (Equation (3)). These elements (a, b, c, and d) define the best-fit regression plane.

$$d = -(a \cdot X_0 + b \cdot Y_0 + c \cdot Z_0), \tag{3}$$

where X_0, Y_0, and Z_0 are the coordinates of the centroid point of the selected points for plane estimation.

Furthermore, the orthogonal distances of the points from the regression plane and the standard deviation of the plane estimation are calculated.

In the next step, the inliers for the given plane are identified (i.e., the points lying on the surface of the estimated plane) by testing the estimated regression plane against the nearest neighbors. Inlier selection is performed using two criteria:

- **distance-based criterion**: only the points that are closer to the estimated plane than the selected threshold values are considered as inliers;
- **normal-based criterion**: inliers are the points where the angle between the normal vector at a given point and between the normal vector of the regression plane is less than the threshold value.

The plane re-estimation and the inlier selection are performed iteratively, with a gradual increase in the number of the neighboring points tested. It is repeated until the number of points belonging to the plane stops increasing. Figure 2 illustrates the plane segmentation process, where the initial point cloud (top-left) is shown (the points are colored by the intensity of the reflected measuring signal), then the approx. 102 thousand nearest points (top-right with yellow color) to the seed point, followed by the approx. 1.6 million nearest neighbors (bottom-left with green color), and with red color, the inliers for the given plane are shown (bottom-right).

Since the plane estimation strongly depends on the seed point selection and its neighborhood, it was necessary to introduce several validation steps to eliminate incorrect estimations. These validation steps are based first on determining whether there are enough inliers at each iteration, i.e., after the second iteration, at least two times more inlier points than the number of points from the first estimation (n_{est}). It is expected that the number of inliers (points lying on the segmented surface) from the surroundings will increase gradually if the selected seed point lies on a planar surface. Then, it is determined whether the plane has sufficient point coverage. The local point density (LPD) value is calculated at each inlier point. In addition, the theoretical value of the LPD is calculated (if the plane has a uniform ideal coverage). The criterion is that at least 50% of the points need a higher LPD (with a certain tolerance) than the ideal LPD. This value (50%) was determined empirically based on testing the algorithm on several point clouds with various densities, complexities, and different levels of noise. This criterion is necessary in some cases, e.g., when processing a point cloud from an indoor environment of a building where there are several objects (e.g., furniture, PC accessories, etc.). In such cases, a plane can be estimated from some subsets of points that are lying on different objects (not lying on the

planar surface), i.e., the result of the estimation can be a plane (though this plane is not a real one), since these points are from a separate dense subset of points lying on a surface of any object. The mentioned cases are eliminated from the estimation by this criterion.

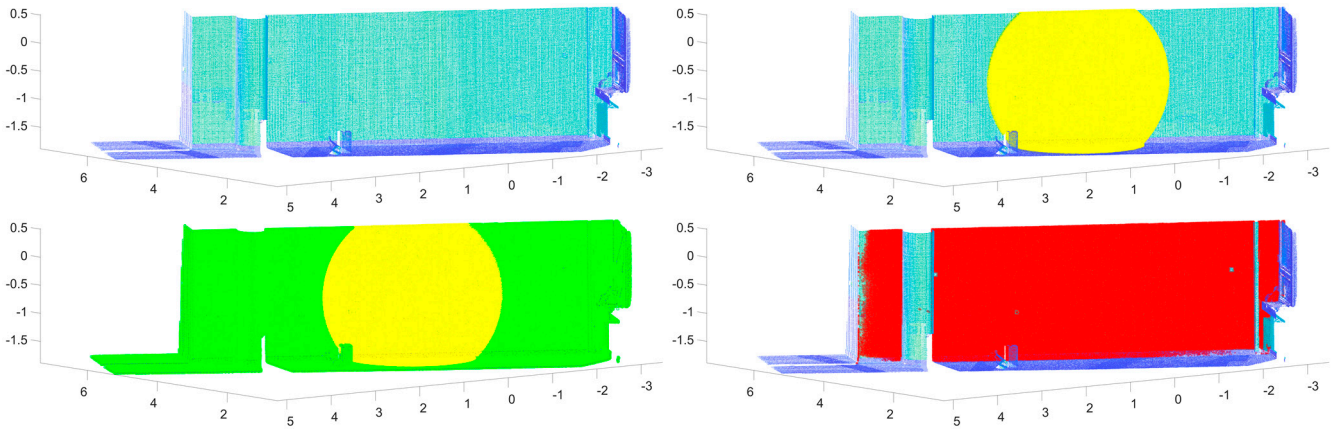

Figure 2. The plane segmentation process by the proposed approach: the initial point cloud (**top-left**), the approximately 102 thousand nearest points (**top-right**), the approximately 1,6 million nearest points (**bottom-left**), and the inliers for the given plane (**bottom-right**).

If any of the validation steps are unsuccessful, the algorithm skips to the new segmentation cycle with a new seed point for a new plane candidate.

In most cases, the number of planes in the point cloud is not known in advance. Therefore, a technique is proposed to stop the calculation. The algorithm automatically breaks the calculation after segmenting all the planes located in the point cloud. If 100 incorrect segmentation attempts (incorrect attempt means that no plane is found at the selected seed point) are made in a row (after several successfully segmented planes), the calculation is stopped. In practical applications of the described algorithm on complex point clouds with several shapes (several plane-, sphere- or cylinder-shaped objects), the number of individual geometric shapes is usually not known in advance. Due to the higher level of automation, in the case of our algorithm, it is not necessary to enter the number of these shapes precisely. However, the algorithm automatically stops after these 100 incorrect attempts. The number 100 was also determined empirically, based on testing, but it can be adjusted based on the size and the complexity of the point cloud. Otherwise, if the number of planes in the point cloud is known, this number can be selected at the beginning of the algorithm, and the algorithm segments the chosen number of planes.

The results of the plane segmentation are the segmented point clouds for each plane and the parameters of the planes, which are: the parameters of the normal vector of the plane (a, b, c), parameter d, number of inliers, the standard deviation of the plane estimation (calculated from the orthogonal distances of the inlier points from the best-fit regression plane). After this part of the algorithm, further processing is performed only on the remaining point cloud (i.e., the points belonging to the segmented planes are excluded from the initial point cloud). This step indeed contributes to increasing the efficiency of the algorithm.

2.3. Sphere Segmentation

The data input into the sphere segmentation part are the remaining point cloud safter the plane segmentation, if plane segmentation has been performed (otherwise, it is the input point cloud). The part of the algorithm which has the role of the sphere segmentation is also partially inspired by the RANSAC algorithm. The least-square spherical fit is used to calculate the sphere parameter.

The process of the algorithm is similar to the plane segmentation algorithm. The first step is the selection of seed points based on the LNV values at each point (seed point

candidates for spheres are the points where the LNV value is greater than 5°). This seed point selection technique significantly increases the efficiency of the algorithm. In cases of processing complex point clouds that contain several walls (planar surfaces, where the points have small local curvature—usually up to 5 degrees, because of undulation of the planar surface in some cases), with this step, the points lying on these surfaces are removed from the seed point candidates for sphere estimation. This technique is mainly for a rough removal of the points, where it is assumed that no sphere object can be found.

Then, the first approximate parameters of the sphere are calculated using the n_{est} number of nearest points to the selected seed point. The value of the n_{est} is calculated in the same way as in the case of planes. The estimation is based on a least-square spherical fit, which minimizes the perpendicular distances of the points from the sphere. The solution is based on the general equation of a sphere (Equation (4)).

$$(x - c_x)^2 + (y - c_y)^2 + (z - c_z)^2 = r^2, \tag{4}$$

where x, y, and z are the coordinates of the points on the surface of the sphere, c_x, c_y, and c_z are the coordinates of the center of the sphere, and r is the radius of the sphere.

By expanding Equation (4), we get the rearranged equation expressed in Equation (5):

$$x^2 + y^2 + z^2 = 2 \cdot x \cdot c_x + 2 \cdot y \cdot c_y + 2 \cdot z \cdot c_z + r^2 - c_x^2 - c_y^2 - c_z^2. \tag{5}$$

Next, Equation (5) is represented in consolidated terms; which serves as an overdetermined system suitable for spherical fit. In this way, the approximate parameters of the sphere are obtained.

The iterative fitting and extraction process then uses the sphere's approximate estimated parameters. Extraction, thus selecting the inliers for the estimated sphere candidate, is performed based on distance-based and normal-based filtration (similar to the plane segmentation part). In contrast to the plane algorithm, the extraction process is performed on the whole point cloud at once. The iterative re-estimation is performed until all the points of the detected sphere have been selected. The maximum number of iterations is set to 15. This number was determined empirically based on processing several point clouds with different complexity, noise, and number of spheres. However, this value is sufficiently oversized and was not reached in any of the mentioned tests due to several conditions to stop the calculation. It means that if no new point is added in 3 consecutive iterations, and there is no difference in the parameters of the sphere, the calculation is automatically stopped.

Moreover, several validation steps were proposed. The first was based on inlier number (whether there are enough inliers for the selected seed point—at least twice more inliers than the value of n_{est} after the fourth iteration). The next was based on the convergence of the sphere parameters. Therefore, the differences in the estimated parameters of the sphere are determined in two consecutive iterations (after the fifth iteration), and the convergence parameter gives the maximum allowed difference ($\varepsilon = 10^{-4}$). The last validation step ensures that the standard deviation of the spherical fit is not greater than the specified parameter. The standard deviation is estimated based on the orthogonal distances of the inliers from the sphere surface. If any of these validations are unsuccessful, the algorithm skips to a new segmentation cycle with a new seed point. The proposed technique to stop the calculation automatically is also included in this part (sphere segmentation part) of the algorithm.

The results are the segmented point clouds and the parameters for each shape (sphere parameters [c_x, c_y, c_z, r], inlier number, the standard deviation of the sphere fitting). The segmented clouds for each sphere are also excluded from further processing.

From experiments (processing of several point clouds containing sphere objects), it was found that coverage of only approximately 40–50% of the scanned sphere surface (e.g., when scanning only from a single position of the instrument) is sufficient for extraction of the sphere with the algorithm depicted. However, a certain LPD is needed, i.e., at

least 100 points are required to be homogeneously distributed on the scanned part of the sphere surface.

2.4. Cylinder Segmentation

The last part is for cylinder segmentation, which is the most complex. The algorithm developed for cylinder segmentation can be categorized into model-based methods and is partially based on the elements of the Hough transform. Similar to the sphere segmentation part, the input data is only the remaining point cloud if the segmentation of other geometric primitives has been performed.

The sequence of the processing steps is similar to the plane and sphere segmentation algorithms. The algorithm starts with selecting seed points based on the calculated LNV values (the seed point candidates are the points where the LNV is greater than 3°). Next, the first cylinder is estimated from the n_{est} number of closest neighbors to the seed point. The value of the n_{est} is calculated in the same way as in the case of planes and spheres. The estimated cylinder parameters are illustrated in Figure 3.

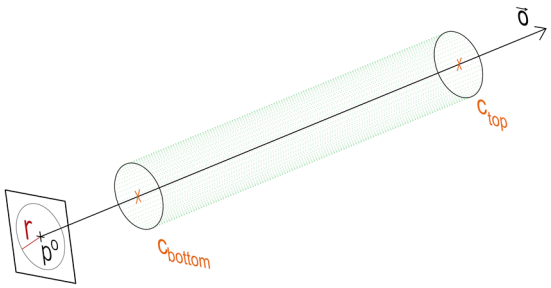

Figure 3. The parameters of the cylinder.

The cylinder estimation is divided into four main steps:

1. Computing the cylinder axis orientation ($\vec{o}$)—vector, perpendicular to the normal vectors in n_{est} closest neighbors.
2. Projection of these points onto a plane that is orthogonal to the cylinder axis ($\vec{o}$). If the selected points lie on the cylinder's surface, they are distributed in a circle.
3. Estimating the projected circle parameters (i.e., coordinates of the circle's center and radius). This estimation uses algebraic fitting, where algebraic distances are minimized [36]. The coordinates of the circle's center point are considered as the point $\boldsymbol{p}^o$, which lies on the cylinder axis. The radius of the estimated circle equals the radius of the cylinder shell.
4. Computation of the base centers of the cylinder (top and bottom base center points). First, the distances between the inliers and the center point of the cylinder axis $\boldsymbol{p}^o$ are calculated according to Formula (6). The maximal and minimal distances are then determined from the vector $\boldsymbol{t}$. These values are then substituted into the Formula (7) to calculate the coordinates of the top and bottom base centers.

$$\boldsymbol{t} = (\boldsymbol{p} - \boldsymbol{p}^o) \cdot \vec{\boldsymbol{o}}, \tag{6}$$

$$\boldsymbol{c}_{bottom} = \boldsymbol{p}^o + t_{min} \cdot \vec{\boldsymbol{o}}, \qquad \boldsymbol{c}_{top} = \boldsymbol{p}^o + t_{max} \cdot \vec{\boldsymbol{o}}, \tag{7}$$

where $\boldsymbol{p}$ is the vector containing the 3D coordinates of the inlier points, $\boldsymbol{c}_{bottom}$ and $\boldsymbol{c}_{top}$ are the coordinates of the center points of the top and bottom bases of the cylinder.

The estimation steps described above are applied iteratively for the inlier points. The set of points considered as inliers is updated at each iteration based on two criteria, mathematically formulated as follows:

$$(|\Delta \boldsymbol{dist}_i| < r \cdot t_d[\%]) \wedge (|\Delta \boldsymbol{norm}_i| < t_n), \tag{8}$$

where $\Delta \boldsymbol{dist}_i$ is the orthogonal distance between the selected point and the cylinder surface, $\Delta \boldsymbol{norm}_i$ is the angle between the normal vector of the selected point and the vector that is perpendicular to the cylinder axis $\vec{\boldsymbol{o}}$ in the selected point. The parameters t_d and t_n are the selected threshold values. The threshold parameter for the distance-based filter is chosen as a percentage of the radius of the cylinder, as the point cloud may contain several cylinders with different radii.

Based on the criteria in Equation (8), the inliers are automatically updated in every iteration, and the outliers are removed from the estimations. After every iteration, the cylinder parameters are re-estimated using all the inliers that meet the specified conditions. The inlier updating is performed on the whole point cloud at once, similar to the sphere segmentation part.

The iterative re-estimation is performed until all the points of the detected cylinder have been selected, and the maximum number of iterations is set to 15. In this case, several validation steps were also proposed. They are based on inlier numbers (similar to the planes and spheres) and parameter convergence (similar to the spheres). Furthermore, a novel validation step based on cylinder coverage was proposed. In this validation method, the cylinder shell is first transformed into a plane, then divided into a grid (the size of the grid is determined automatically based on the dimensions of the cylinder), and the number of points in each cell of the grid is determined. Next, a cell's ideal average density (in the case of an evenly covered cylinder surface with inliers) is computed. The value of the ideal density is estimated based on the known dimensions of the cylinder, the known size of the grid, and the known number of inliers. If at least 25% of the cells have ideal coverage, the cylinder is considered a reliable one.

The method of automatically stopping the calculations is also included in this part of the algorithm, as in the case of planes and spheres. The results are the segmented point clouds and the parameters (a point on the cylinder axis $\boldsymbol{p}^0$, the orientation of the cylinder axis $\vec{\boldsymbol{o}}$, the radius r and the height h of the cylinder shell) for each identified cylinder.

2.5. Development of a Standalone Application for Automation of the Segmentation Process

A standalone computational application (**PoCSegmentation**) was developed to simplify and automate the segmentation procedure. The application's graphical user interface (GUI) was designed in MATLAB® software. Also, the calculation takes place in the MATLAB® software environment, so the Matlab Runtime is required to run the application, which is freely available. The dialogue window of the application (Figure 4) consists of three major sections. The top section is for importing the point cloud. The point cloud can be imported in several file formats, which are as follows: **.txt*, **.xyz*, **.pts*. **.pcd*, **.ply*, **.mat*.

The middle part serves for the segmentation itself, where at the top the types of geometric shapes can be chosen, and the threshold parameters can be selected. The segmentation procedure is started by pressing the *Run Segmentation* button. At the bottom of this section, the application's progress is described, i.e., the individual processes executed step by step and the time required for its execution. The bottom part of the dialog window is used to export the segmentation results. The **PoCSegmentation** application offers various options to export the results. On the left, the segmented point clouds can be exported in several file formats (**.txt*, **.pts*, **.xyz*, **.pcd*, **.ply*, **.mat*). In addition, it is also possible to export the segmented point clouds into *.DXF (Drawing Exchange Format), which is a CAD data file format for vector graphics, and which can be imported into more than 25 applications from various software developers. The advantage of the application is that, in the DXF file, the individual segmented point clouds are divided into separate layers. Next, the parameters of the geometric shapes can be exported to an Excel file, and the remaining point cloud can also be exported to *.pts format.

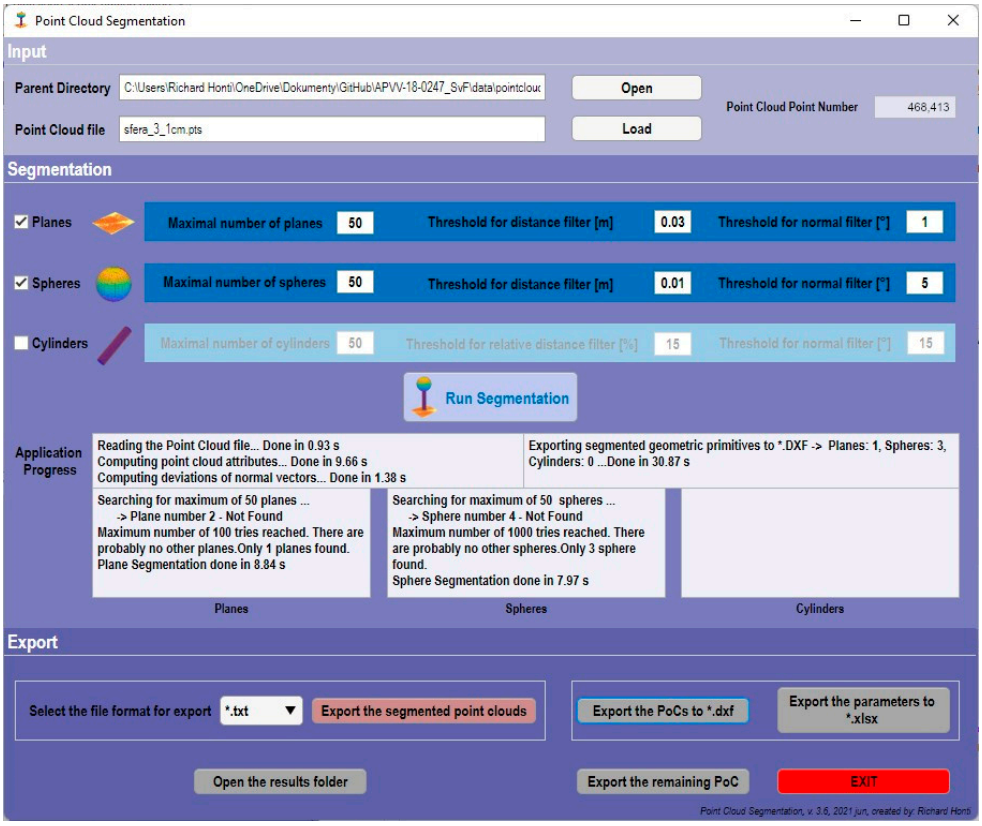

Figure 4. Dialog window of the **PoCSegmentation** application.

2.6. Testing of the Proposed Approach

Experimental testing of the **PoCSegmentation** application, implemented based on the algorithm proposed, was performed on several point clouds with various densities, complexities, and different levels of noise. For the first experiment, a point cloud from an industrial building (Point Cloud No. 1) (Figure 5) was used that contained 12 planar surfaces and 6 sphere objects. The mentioned point cloud was obtained from an online point cloud database accessible at [37].

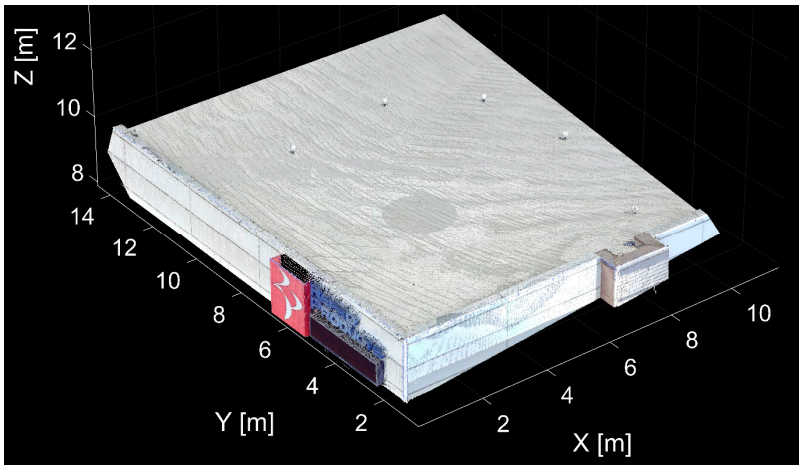

Figure 5. The initial point cloud of a part of an industrial building.

The initial point cloud contained approximately 815 thousand points. In this case, the threshold values were chosen as follows: the shapes for segmentation were planes and spheres, the threshold for the distance filter was 0.050 m for the planes and 0.020 m for the

spheres, and the threshold for the normal-based filter was 5° for the planes and 10° for the spheres. The developed application segmented 13 planes and six spheres (reference sphere targets).

The results of the segmentation process are shown in Figure 6. The left side of Figure 6 shows the segmented points of the individual planes differentiated by color, and on the right, the points of the segmented spheres are shown in red color. To verify the results of sphere segmentation, the differences among the known parameters (radius of the sphere targets are defined by the producer of the targets) and the estimated parameters from the application were calculated. The differences were below 1.2 mm in any case.

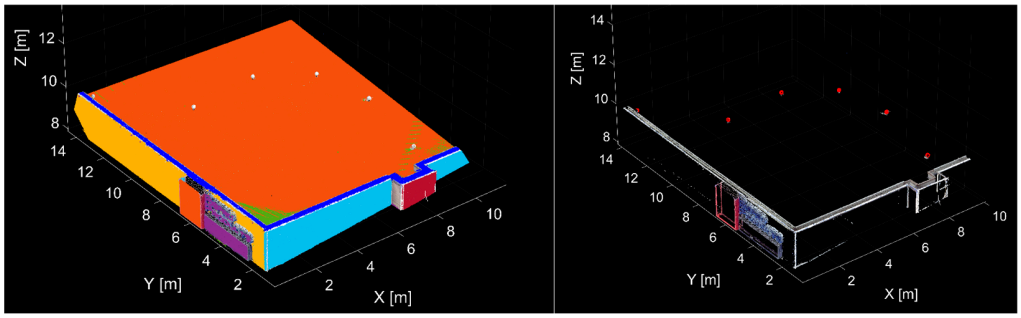

Figure 6. The segmented planes (**left**) and spheres (**right**).

The scanned object contained 12 planar and six spherical surfaces. Using the developed application, 13 planes and six spheres were automatically segmented. However, the difference in one plane is not caused by the imperfection of the algorithm but by the undulation and the bumpy surface of the roof of the building. The algorithm divided the roof into two parts while, based on the selected threshold values, some points of the roof did not belong to the plane shown in orange.

After segmenting the depicted planes and spheres, the remaining point cloud represented only 8% of the points from the initial point cloud. These points were mostly the edges of the individual planes and points on the mounting pads for the spherical targets, so the points did not belong to any plane or sphere.

The entire segmentation procedure (13 planes and six spheres) was executed in approximately 8 min on a PC with the following basic parameters: operating system—Windows 10 Pro, CPU—Intel Core i7 9700F Coffee Lake 4.7 GHz, RAM—32.0 GB DDR4, Motherboard—MSI B360 Mortar, Graphics—NVIDIA GeForce RTX 2070 SUPER 8 GB, SSD—WD Blue SN500 NVMe SSD 500 GB.

In the next test, the point cloud of a chosen room of the Pavol Országh Hviezdoslavov Theatre in Bratislava was used (Point Cloud No. 2), which contained more than 1.8 million points. Scanning was also performed with a Trimble TX5 3D laser scanner. The average point cloud density was 2 cm, and the accuracy in the spatial position of a single measured point was less than 5 mm. The initial point cloud from two views is shown in Figure 7.

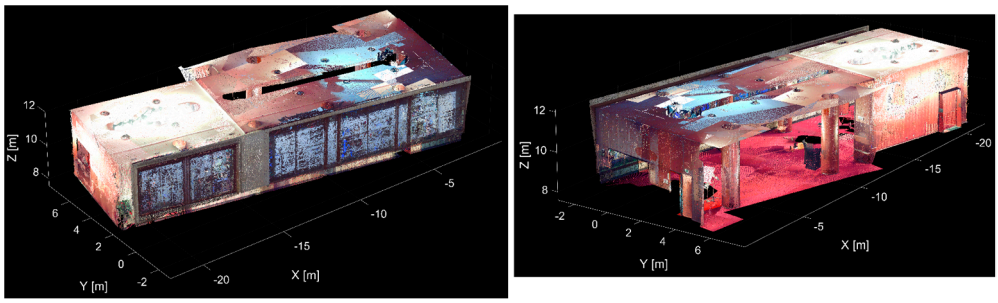

Figure 7. The initial point cloud of a room in the theatre.

The scanned room contained 22 planar surfaces (walls, floor, etc.) and six cylinder-shaped columns. The threshold values for segmentation were as follows: for the distance-based filter, 0.020 mm for the planes and 15% (of the estimated radius) for the cylinders; for the normal-based filter, 2° for the planes and 15° for the cylinders. The algorithm identified and segmented 22 planes and six cylinders from the point cloud. The result of the plane segmentation is shown in Figure 8, where the individual planes are differentiated by color.

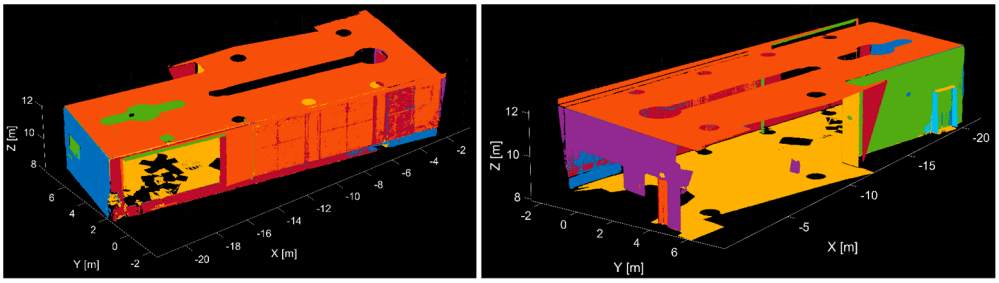

Figure 8. Result of the plane segmentation.

Figure 9 shows the remaining point cloud after the plane segmentation since, after the individual planes are segmented from the point cloud, they are removed from further processing. The noted point cloud represents only 34% of the initial point cloud, so in the next step of the algorithm, that is, cylinder segmentation, the processing is performed on a much smaller set of points.

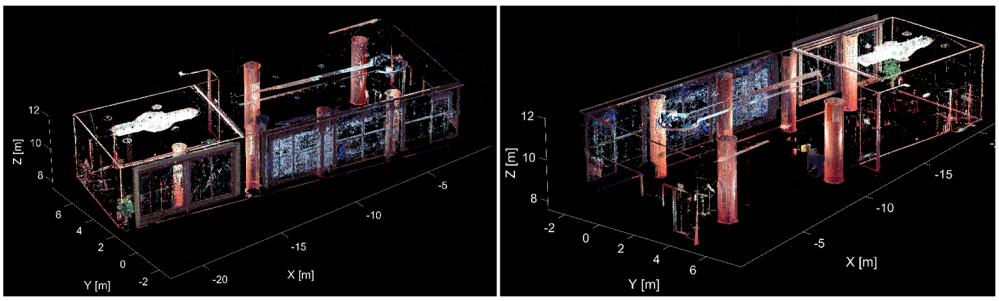

Figure 9. The remaining point cloud after the plane segmentation.

Figure 10 shows the remaining point cloud after the plane segmentation with the segmented cylinders, which are differentiated by color. With the mentioned threshold values, segmentation of 6 cylinders was performed.

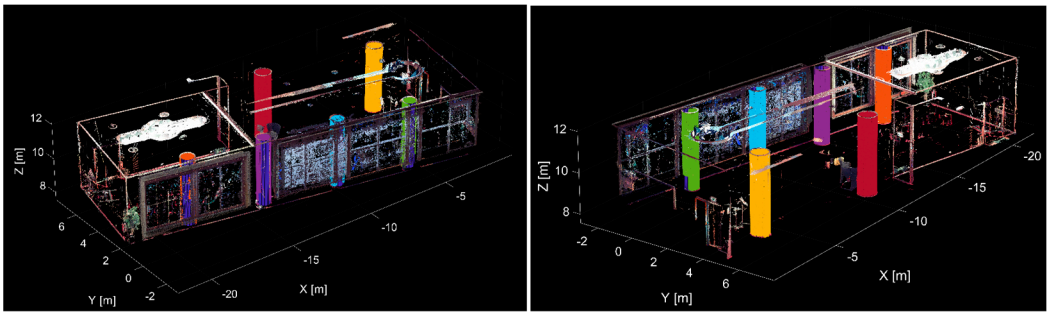

Figure 10. The remaining point cloud with the segmented cylinders.

To verify the results of cylinder segmentation, the known (real) parameters of individual cylinders were compared with the estimated parameters from processing the point cloud with the application developed. The cylindrical columns had a uniform radius (r_{real}) of approximately 0.400 m and a height (h_{real}) of approximately 3.900 m. The parameters

of each of the columns were measured by a measuring tape at various positions, and the final parameters were calculated as average values. The radiuses (r_{app}) and heights (h_{app}) obtained from the processing are shown in Table 1 with the calculated absolute deviations. The maximal deviation in radiuses was 9 mm and in height was 9 mm. In these deviations, the imperfection of the construction of these columns, the effect of the environmental conditions, the systematic errors of the instrument, the measurement error, and the processing errors (the standard deviation (*s* in Table 1) of the cylinder fitting to the segmented points was 10 to 15 mm at the mentioned threshold values) are also included.

Table 1. Real parameters and parameters from the application with the calculated absolute deviations for the cylinders.

Cylinder No.	Real Parameters		Parameters from the Application		Absolute Deviations		*s* [mm]
	r_{real} [m]	h_{real} [m]	r_{app} [m]	h_{app} [m]	Δr [mm]	Δh [mm]	
1	0.400	3.900	0.409	3.902	9	2	10
2			0.408	3.909	8	9	15
3			0.406	3.895	6	5	12
4			0.395	3.892	5	8	11
5			0.394	3.899	6	1	13
6			0.402	3.906	2	6	15

Figure 11 shows the remaining point cloud after segmenting 22 planes and six cylinders. These points are on the edges of the individual shapes and objects in the room (e.g., parts of the room, lightning, flowerpots with plants, chairs, tables, etc.). The remaining cloud represents only 15% of the initial point cloud.

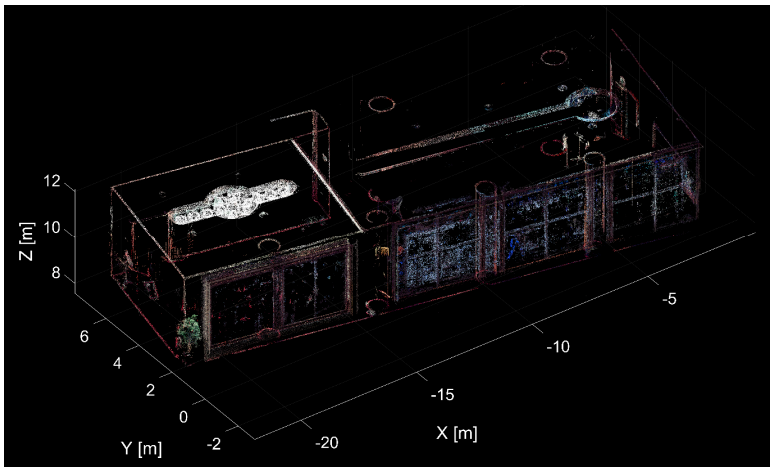

Figure 11. The remaining point cloud after plane and cylinder segmentation.

Using the proposed application, all the planes and cylinders that form the room's structural elements (walls, columns, etc.) were correctly and automatically detected and segmented from the point cloud.

The segmentation procedure (22 planes and 6 cylinders) was executed in approximately 11 min on the same PC, as in the first test case.

The last test was performed on a point cloud of the double-cylinder-shaped test object (Point Cloud No. 3). The measurement was performed with TLS Leica Scanstation 2 (average density 1 cm, the accuracy of a single measured point up to 2.5 mm, approximately 202 thousand points). The threshold values for segmentation were as follows: the threshold for the distance-based filter was 10 mm for the planes and 10% for the cylinders, and the thresholds for the normal-based filter were 0.5° for the planes and 5° for the cylinders.

Figure 12 shows the segmentation result; on the left side, the two segmented planes are shown, and on the right side, the two segmented cylinders are shown.

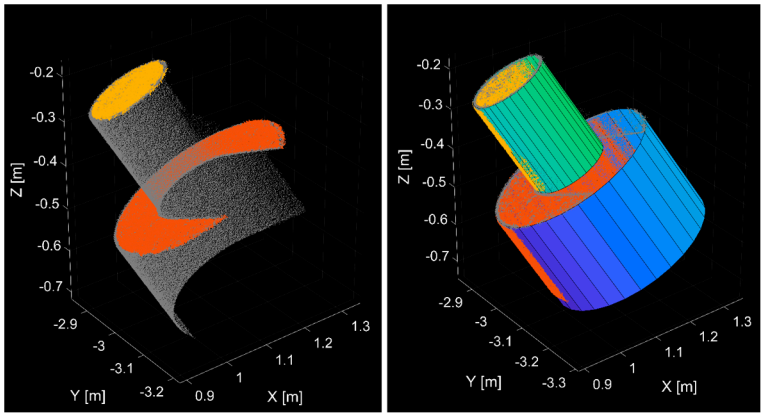

Figure 12. Result of the segmentation of the double-cylinder model.

In this case, the differences between the known geometric parameters (radius and height of the cylinders) and the estimated parameters from the processing were also calculated to verify the cylinder estimation (Table 2). The geometric parameters of the cylinder object were measured by a CMM (Coordinate Measuring Machine) measuring system with an accuracy of 0.1 mm. The radius differences were 0.2 mm for the larger cylinder and 0.1 mm for the smaller cylinder. The height differences were 0.3 mm for the larger cylinder and 0.5 mm for the smaller cylinder.

Table 2. Comparison of the real parameters and parameters from the application.

Cylinder	Real Parameters		Parameters from the Application		Absolute Deviations	
	r_{real} [m]	h_{real} [m]	r_{app} [m]	h_{app} [m]	Δr [mm]	Δh [mm]
Large	0.200	0.250	0.202	0.253	0.2	0.3
Small	0.090	0.250	0.091	0.255	0.1	0.5

The whole segmentation procedure (2 planes and 2 cylinders) was executed in less than 1 min on the same PC, as in the case of the first two tests.

3. Results and Discussion

3.1. Comparison with the Standard RANSAC Approach

For a demonstration of the potentiality and advantages of the proposed approach, a comparison with the standard RANSAC algorithm was performed on several datasets. The comparison is executed for each geometric shape type separately. The geometric shape segmentation process is performed 50 times for each dataset with the same parameters, and the comparison of the best results is described.

3.2. Plane Segmentation

First, plane segmentation was compared between the two mentioned algorithms. The thresholds adopted for the first point cloud (Figure 5, Point Cloud no. 1) were as follows: the maximum number of planes = 15, the distance threshold = 0.05 m, and the normal threshold = 5° (only for the proposed algorithm, as in the standard RANSAC algorithm, filtration based on normal vectors is not executed). The threshold values are specific to the density of the point cloud but were chosen to obtain the best results. The results are shown in Figure 13, where the segmented planes are differentiated by color.

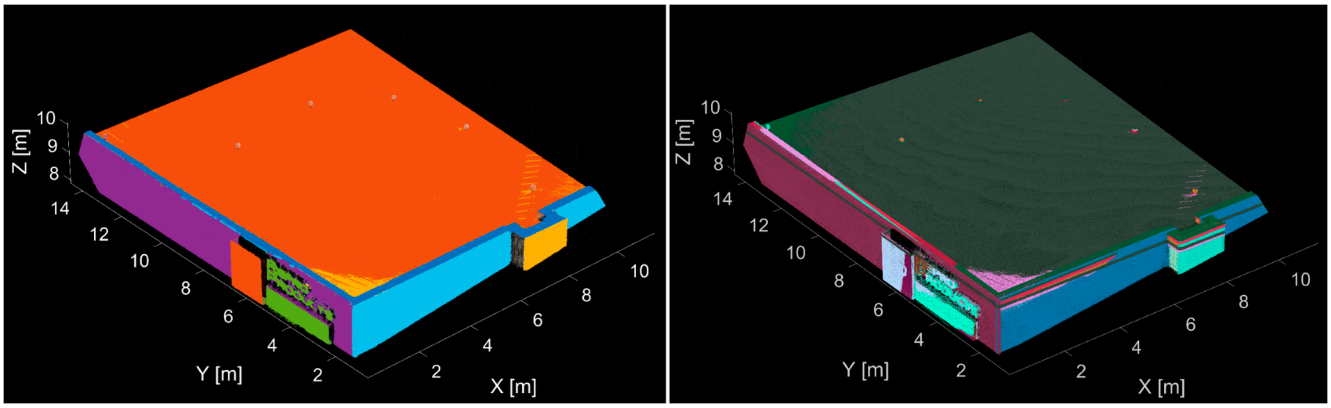

Figure 13. Result of the plane segmentation from the proposed algorithm (**left**) and the standard RANSAC algorithm (**right**).

With the proposed algorithm, 13 planes were segmented (Figure 13—left), which also shows the usability and the functionality of the proposed approach for stopping the calculations (described in Section 2.1) in case the number of planes in the point cloud is not known in advance. In this case, the maximum number of planes was set to 15, but the algorithm automatically stopped the calculations after 13 planes.

The right side of Figure 13 shows the plane segmentation result using the standard RANSAC algorithm. Segmentation of 15 planes was executed, although the point cloud consisted of 12 planar surfaces (since the technique to stop the calculations automatically is not included in the standard RANSAC algorithm).

The left side of Figure 14 shows the correctly (with some toleration) segmented planes (7) using the standard RANSAC algorithm. Although in these cases, some points are incorrectly assigned to the planes (e.g., in the case of the horizontal plane surface (roof) with green color, where some points on the edges are assigned from the nearby vertical plane surfaces). On the right side of Figure 14, the incorrectly segmented planes are shown, which are color differentiated.

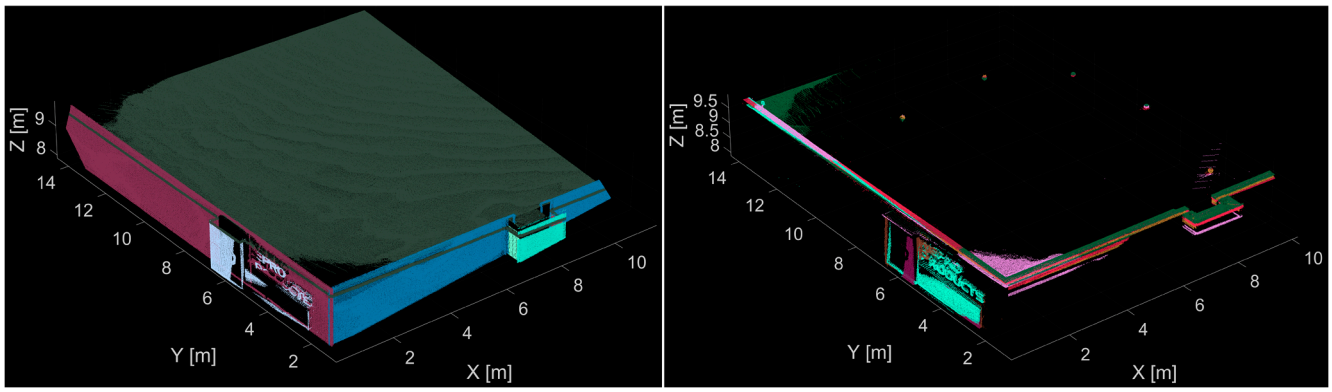

Figure 14. The plane segmentation result using the standard RANSAC algorithm with the correctly segmented planes (**left**), and the incorrectly segmented planes (**right**).

In addition to visual comparison, statistical analysis was performed, where a quality indicator of segmentation was calculated for both approaches. The quality indicator (Equation (9)) was calculated as a ratio of the number of correctly detected planes ($pl_{correct}$) to the incorrectly detected ($pl_{incorrect}$) and undetected planes ($pl_{undetect}$).

$$Q_{pl} = \frac{pl_{correct}}{pl_{correct} + pl_{incorrect} + pl_{undetect}} \quad (9)$$

Figures 13 and 14 show that the proposed algorithm shows better results than the standard RANSAC algorithm in the test described. The segmentation quality increased from 35% to 92% (Table 3).

Table 3. Characteristics of plane segmentation.

	Point Cloud No. 1		**Point Cloud No. 2**	
	Proposed Algorithm	**Standard RANSAC**	**Proposed Algorithm**	**Standard RANSAC**
$pl_{correct}$	12	7	22	12
$Pl_{incorrect}$	1	8	0	13
$pl_{undetect}$	0	5	0	10
time	**2 min 21 s**	**3 min 43 s**	**5 min 37 s**	**8 min 19 s**
$\bar{s}$	**8 mm**	**112 mm**	**5 mm**	**75 mm**
$\mathbf{Q_{pl}}$	**92%**	**35%**	**100%**	**34%**

The computational time needed for the entire segmentation process was reduced approximately by 30% in the case of the proposed algorithm. Table 3 also compares the standard deviation of plane fitting, which is calculated based on the orthogonal distances of the inlier points from the estimated best-fit regression plane. The average standard deviation ($\bar{s}$) is 8 mm with the proposed algorithm, which is 14 times less than in the case of the RANSAC algorithm.

The second comparison was made on the point cloud (Point Cloud No. 2), shown in Figure 7. The threshold values adopted for this point cloud were the following: maximum number of planes = 25, distance threshold 20 mm, and normal threshold 5°. The result of the plane segmentation with the proposed algorithm is shown in Figure 8. The result of the standard RANSAC algorithm is shown in Figure 15. The analysis of the results, with the quality indicator, is shown in Table 3.

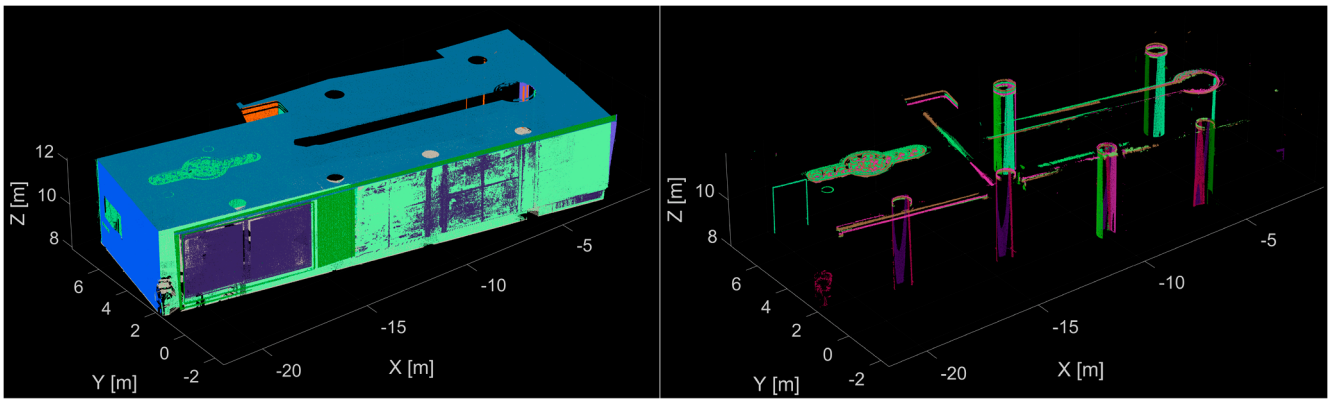

Figure 15. The plane segmentation result using the standard RANSAC algorithm in the case of a chosen room of the P.O. Hviezdoslavov Theater with the correctly segmented planes (**left**), and the incorrectly segmented planes (**right**).

Figure 15 shows the result of the plane segmentation using the standard RANSAC algorithm. On the left side of the figure, the correctly segmented planes are shown, also with some tolerances, as was in the previous case, shown in Figure 14. On the right side, the incorrectly segmented planes are shown, e.g., the parts of the cylinder-shaped columns, parts of the lightning, etc. (points that do not belong to a planar surface). In this case, the quality indicator of segmentation increased from 34% to 100% (Table 3). The standard deviation was 15 times less than the proposed algorithm, and the computational time was reduced approximately by 30%.

3.3. Sphere Segmentation

Next, the sphere segmentation part was compared to two point clouds, first on a point cloud (Point Cloud No. 4) with three sphere objects (reference targets) and a planar surface (Figure 16). The threshold values adopted for this point cloud were the following: maximum number of spheres = 6, distance threshold 20 mm, and normal threshold 5°.

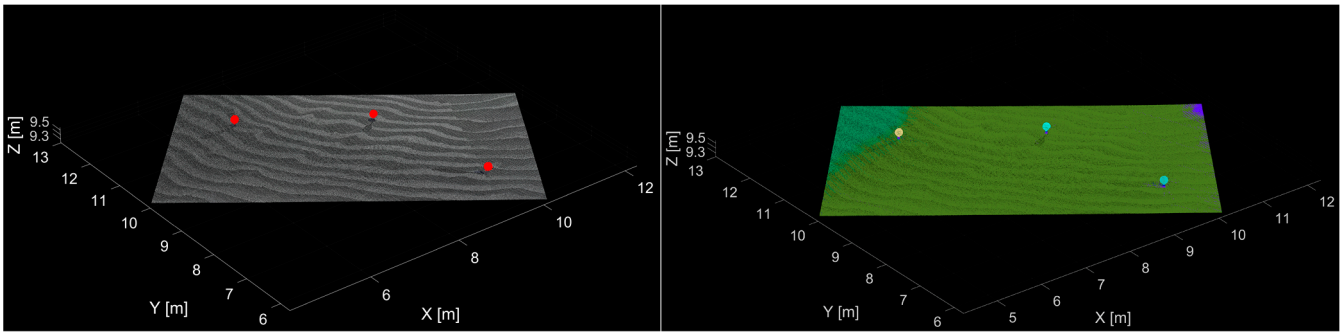

Figure 16. Comparison of the result of the sphere segmentation from the proposed algorithm (**left**) and the standard RANSAC algorithm (**right**).

The left side of Figure 16 shows the segmentation result from the proposed algorithm; with red color the segmented spheres are shown, and with grey, the initial point cloud. On the right side of the figure, the result from the standard RANSAC algorithm is shown; the segmented parts are color differentiated. With the algorithm proposed, the quality indicator increased from 50% to 100% since the standard RANSAC algorithm assigned the points of the planar surface situated in the point cloud to three separate sphere objects (apparently, these points are not lying on a surface of a sphere object). The statistical analysis is shown in Table 3. In this table, the average standard deviation ($\bar{s}$) of the sphere fitting is also compared (estimated only from the correctly detected spheres). The standard deviation is calculated based on the distances of the inlier points for the detected sphere from the estimated sphere surface. The standard deviation was 15 times lower than the proposed algorithm and the computational time was approximately equal.

The second comparison was performed on the point cloud (Point Cloud No. 1) of a part of an industrial building, shown in Figure 5. The threshold values adopted for this point cloud were the same as in the previous case. The statistical analysis is also shown in Table 4. In this case, the quality increased from 27% to 100% since the standard RANSAC algorithm resulted in several incorrectly detected spheres, and some of the sphere objects were undetected.

Table 4. Qualitative characteristics of sphere segmentation.

	Point Cloud No. 4		**Point Cloud No. 1**	
	Proposed Algorithm	**Standard RANSAC**	**Proposed Algorithm**	**Standard RANSAC**
$sph_{correct}$	3	3	6	3
$sph_{incorrect}$	0	3	0	5
$sph_{undetect}$	0	0	0	3
time	**1 min 17 s**	**1 min 35 s**	**5 min 29 s**	**5 min 15 s**
$\bar{s}$	**3 mm**	**46 mm**	**4 mm**	**58 mm**
Q	**100%**	**50%**	**100%**	**27%**

3.4. Cylinder Segmentation

Lastly, the cylinder segmentation part was compared based on two datasets. The first comparison was made to the point cloud of the selected room in the theatre building (Figure 7, Point Cloud no. 2). The threshold values adopted for this point cloud were

the following: maximum number of cylinders = 15, distance threshold 15%, and normal threshold 10°. The result from the proposed algorithm is shown in Figure 10, and Figure 17 shows the result from the standard RANSAC algorithm. The statistical analysis is shown in Table 5.

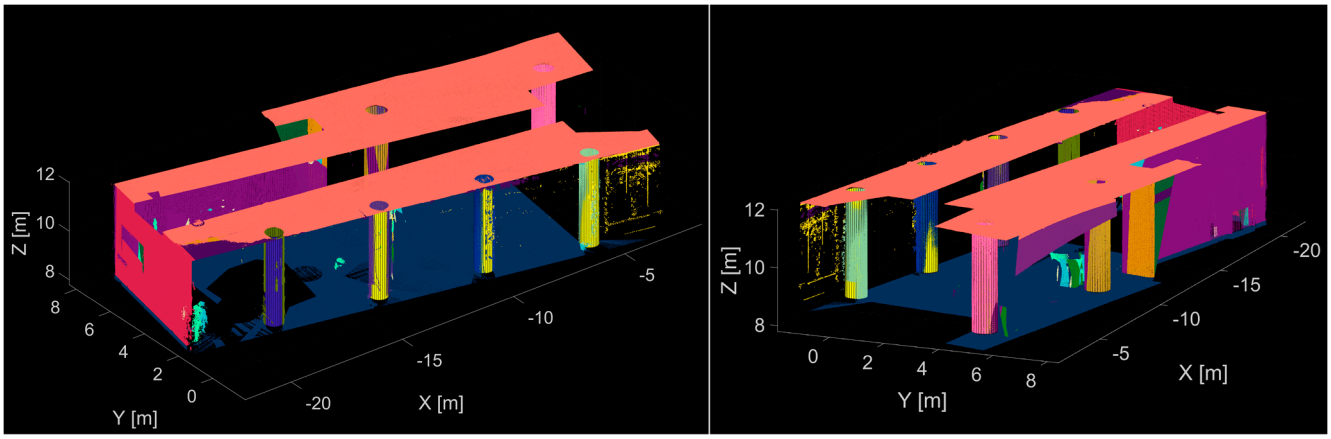

Figure 17. The cylinder segmentation results from the standard RANSAC algorithm.

Table 5. Qualitative characteristics of cylinder segmentation.

	Point Cloud No. 2		**Point Cloud No. 3**	
	Proposed Algorithm	**Standard RANSAC**	**Proposed Algorithm**	**Standard RANSAC**
$cyl_{correct}$	6	6	2	2
$cyl_{incorrect}$	0	7	0	3
$cyl_{undetect}$	0	0	0	0
time	**5 min 22 s**	**6 min 5 s**	**0 min 30 s**	**0 min 25 s**
$\bar{s}$	**3 mm**	**35 mm**	**2 mm**	**15 mm**
Q	**100%**	**46%**	**100%**	**40%**

The standard RANSAC algorithm resulted in six correctly and nine incorrectly segmented cylinders. However, the correctly segmented cylinders had a higher standard deviation of fitting since, in the case of the proposed algorithm, the outlier and noise removal from the fitting process is more thorough. As shown in Figure 17, the standard RANSAC algorithm segmented some planar surfaces (walls, floor, ceiling, doors, etc.) as cylinders. The segmentation quality increased to 100% from 46%, the standard deviation was reduced 11 times, and the computational time was reduced by 10% with the proposed approach.

The second comparison was made on the point cloud (Point Cloud No. 3) of the double-cylinder model (Figure 12). The threshold values were as follows: maximum number of cylinders = 5, distance threshold 10%, and normal threshold 5°. The statistical analysis of the results is shown in Table 5. The segmentation quality increased to 100% from 40%, the standard deviation was reduced 7 times, and computational time was the same as the proposed approach.

The proposed algorithm shows better results than the standard RANSAC algorithm based on the described experiments. On average, the segmentation quality calculated based on Equation (9) was less than 50% in the case of the standard RANSAC algorithm, and in the case of the proposed algorithm, it was mostly 100% (only in one case, in the case of the first plane comparison, it was 92% since the roof of the building was divided into 2 planar surfaces, due to its undulation, and so it was not caused by the imperfection of the proposed algorithm). The standard deviation of the fitting is ten times lower on average using the proposed algorithm since the outlier and noise removal process is more thorough,

and a normal-based filtration technique is added to the distance-based filtration. Moreover, several validation steps are executed to remove incorrectly detected shapes from the results. Furthermore, in the case of the proposed algorithm, there is no need to select the number of geometric shapes in the point cloud exactly since an approach was proposed to stop the calculations after the correctly segmented shapes. The next advantage of the proposed algorithm is the possibility to segment three shape types at once in a semi-automated way.

4. Conclusions

Data acquisition is almost fully automated with the currently available laser scanners. Measurement can be performed relatively easily and in a short time. However, manually processing the measured point clouds can be time-consuming and complicated. Therefore, a basic premise of the efficiency of using point clouds is a high degree of automation of the processing steps. For example, when creating a 3D model (or BIM) of an existing building, one of the basic steps is the identification and segmentation of the basic structural elements of the object. In most cases, these basic elements are formed in the shape of basic geometric primitives (e.g., walls—planes; columns or piping network—cylinders; etc.). Therefore, automation of identification and segmentation of sphere objects can be useful, for example, in the case of point cloud registration based on spherical targets, and it also simplifies the 3D model creation.

The paper describes the algorithm proposed for automated identification and segmentation of geometric shapes from point clouds with the requirement of selecting a minimal number of input parameters. The algorithm can detect end-segment subsets of points belonging to planes, spheres, and cylinders from complex, noisy, unstructured point clouds. Inlier detection is performed using distance-based and normal-based filtering. Additionally, several validation steps were proposed to eliminate incorrect estimations. The algorithm proposed was tested on several point clouds with various densities, complexity, and different levels of noise. Specifically, testing on three different point clouds was described, one containing planes and spheres and two other point clouds with planes and cylinders. In all cases, the proposed algorithm correctly identified the geometric shapes regardless of their size, number, or complexity. Moreover, one of the most significant advantages of the algorithm is that the results can be exported directly to DXF exchange format for further processing. Besides that, comparison between the proposed algorithm and the standard RANSAC algorithm was performed separately for the individual geometric shapes on several point clouds. On average, the segmentation quality was increased from 50% to 100% with the described algorithm.

A standalone application that enables semi-automation of the point cloud segmentation procedure is developed in MATLAB® software. However, for its execution, the Matlab Runtime is necessary. In the future, approaches for the identification and segmentation of free-form objects from point clouds will be proposed and programmed.

Author Contributions: Conceptualization, R.H. and J.E.; methodology, R.H., J.E. and A.K.; investigation, R.H.; resources, R.H.; writing—original draft preparation, R.H.; writing—review and editing, R.H. and J.E.; visualization, R.H.; project administration, J.E.; funding acquisition, A.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Slovak Research and Development Agency under the Contract no. APVV-18-0247.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Pavelka, K.; Matoušková, E.; Pavelka, K.; Pacina, J. Spatial 3D documentation of historical mining remnants in forested area in the Erzgebirge/Krušnohoří mining region UNESCO site. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2021**, *XLVI-M-1-2021*, 523–529. [CrossRef]
2. Štroner, M.; Urban, R.; Křemen, T.; Koska, B. Accurate Measurement of the Riverbed Model for Deformation Analysis using Laser Scanning Technology. *Geoinform. FCE CTU* **2018**, *17*, 81–92. [CrossRef]
3. Marendić, A.; Paar, R.; Tomić, H.; Roić, M.; Krkač, M. Deformation monitoring of Kostanjek landslide in Croatia using multiple sensor networks and UAV. In Proceedings of the INGEO 2017–7th International Conference on Engineering Surveying, Lisbon, Portugal, 18–20 October 2017.
4. Fan, W.; Shi, W.; Xiang, H.; Ding, K. A Novel Method for Plane Extraction from Low-Resolution Inhomogeneous Point Clouds and its Application to a Customized Low-Cost Mobile Mapping System. *Remote Sens.* **2019**, *11*, 2789. [CrossRef]
5. Xiao, J.; Zhang, J.; Zhang, J.; Zhang, H.; Hildre, H.P. Fast plane detection for SLAM from noisy range images in both structured and unstructured environments. In Proceedings of the IEEE International Conference on Mechatronics and Automation, Beijing, China, 7–10 August 2011; pp. 1768–1773. [CrossRef]
6. Nguyen, A.; Le, B. 3D Point Cloud Segmentation: A survey. In Proceedings of the 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM), Manila, Singapore, 12–15 November 2013; pp. 225–230, ISBN 978-1-4799-1201-8. [CrossRef]
7. Grilli, E.; Menna, F.; Remondino, F. A Review of Point Clouds Segmentation and Classification Algorithms. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2017**, *42W3*, 339–344. [CrossRef]
8. Bhanu, B.; Lee, S.; Ho, C.C.; Henderson, T. Range Data Processing: Representation of Surfaces by Edges. In Proceedings of the Eighth International Conference on Pattern Recognition, Paris, France, 27–31 October 1986; pp. 236–238.
9. Sappa, A.D.; Devy, M. Fast range image segmentation by an edge detection strategy. In Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling, Quebec City, QC, Canada, 28 May–1 June 2001; IEEE: New York, NY, USA; pp. 292–299.
10. Vosselman, G.; Gorte, B.G.H.; Sithole, G.; Rabbani, T. Recognising Structure in Laser Scanner Point Clouds. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2004**, *46 Pt 8/W2*, 33–38.
11. Vo, A.-V.; Truong-Hong, L.; Laefer, D.; Bertolotto, M. Octree-based region growing for point cloud segmentation. *ISPRS J. Photogramm. Remote Sens.* **2015**, *104*, 88–100. [CrossRef]
12. Yuan, H.; Sun, W.; Xiang, T. Line laser point cloud segmentation based on the combination of RANSAC and region growing. In Proceedings of the 2020 39th Chinese Control Conference (CCC), Shenyang, China, 27–29 July 2020; pp. 6324–6328. [CrossRef]
13. Biosca, J.M.; Lerma, J.L. Unsupervised robust planar segmentation of terrestrial laser scanner point clouds based on fuzzy clustering methods. *ISPRS J. Photogramm. Remote Sens.* **2008**, *63*, 84–98. [CrossRef]
14. Murtiyoso, A.; Grussenmeyer, P. Point cloud segmentation and semantic annotation aided by gis data for heritage complexes. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.* **2019**, *XLII-2/W9*, 523–528. [CrossRef]
15. Lu, X.; Yao, J.; Tu, J.; Li, K.; Li, L.; Liu, Y. Pairwise Linkage For Point Cloud Segmentation. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2016**, *3*, 201–208. [CrossRef]
16. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2017**, *521*, 436–444. [CrossRef]
17. Xie, Y.; Tian, J.; Zhu, X. Linking Points With Labels in 3D: A Review of Point Cloud Semantic Segmentation. *IEEE Geosci. Remote Sens. Mag.* **2020**, *8*, 38–59. [CrossRef]
18. Fischler, M.A.; Bolles, R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
19. Hough, P.V.C. Method and Means for Recognizing Complex Patterns. U.S. Patent 3069654, 18 December 1962.
20. Li, L.; Yang, F.; Zhu, H.; Li, D.; Li, Y.; Tang, L. An Improved RANSAC for 3D Point Cloud Plane Segmentation Based on Normal Distribution Transformation Cells. *Remote Sens.* **2017**, *9*, 433. [CrossRef]
21. Schnabel, R.; Wahl, R.; Klein, R. Efficient RANSAC for point-cloud shape detection. *Comput. Graph. Forum* **2007**, *26*, 214–226. [CrossRef]
22. Xu, B.; Chen, Z.; Zhu, Q.; Ge, X.; Huang, S.; Zhang, Y.; Liu, T.; Wu, D. Geometrical Segmentation of Multi-Shape Point Clouds Based on Adaptive Shape Prediction and Hybrid Voting RANSAC. *Remote Sens.* **2022**, *14*, 2024. [CrossRef]
23. Li, Y.; Wu, X.; Chrysathou, Y.; Sharf, A.; Cohe-Or, D.; Mitra, N. Globfit: Consistently fitting primitives by discovering global relations. *ACM Trans. Grap.* **2011**, *30*, 1–12. [CrossRef]
24. Tran, T.-T.; Cao, V.-T.; Laurendeau, D. Extraction of reliable primitives from unorganized point clouds. *3D Res.* **2015**, *6*, 44. [CrossRef]
25. Tran, T.-T.; Cao, V.-T.; Laurendeau, D. eSphere: Extracting spheres from unorganized point clouds. *Vis. Comput.* **2016**, *32*, 1205–1222. [CrossRef]
26. Drost, B.; Ilic, S. Local Hough Transform for 3D Primitive Detection. In Proceedings of the International Conference on 3D Vision 2015, Lyon, France, 19–22 October 2015; pp. 398–406. [CrossRef]
27. Strom, J.; Richardson, A.; Olson, E. Graph-based segmentation for colored 3D laser point clouds. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 October 2010; pp. 2131–2136. [CrossRef]

28. Cheng, M.; Hui, L.; Xie, J.; Yang, J. SSPC-Net: Semi-supervised Semantic 3D Point Cloud Segmentation Network. In Proceedings of the AAAI Conference on Artificial Intelligence, 2–9 February 2021; Volume 35, pp. 1140–1147.
29. Nurunnabi, A.; Belton, D.; West, G. Robust Segmentation in Laser Scanning 3D Point Cloud Data. In Proceedings of the International Conference on Digital Image Computing Techniques and Applications (DICTA), Fremantle, WA, Australia, 3–5 December 2012. [CrossRef]
30. Luo, N.; Yu, H.; Huo, Z.; Liu, J.; Wang, Q.; Xu, Y.; Gao, Y. KVGCN: A KNN Searching and VLAD Combined Graph Convolutional Network for Point Cloud Segmentation. *Remote Sens.* **2021**, *13*, 1003. [CrossRef]
31. Pierdicca, R.; Paolanti, M.; Matrone, F.; Martini, M.; Morbidoni, C.; Malinverni, E.S.; Frontoni, E.; Lingua, A.M. Point Cloud Semantic Segmentation Using a Deep Learning Framework for Cultural Heritage. *Remote Sens.* **2020**, *12*, 1005. [CrossRef]
32. Guo, Y.; Wang, H.; Hu, Q.; Liu, H.; Liu, L.; Bennamoun, M. Deep Learning for 3D Point Clouds: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 4338–4364. [CrossRef] [PubMed]
33. Charles, R.; Su, H.; Kaichun, M.; Guibas, L. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA; 2017; pp. 77–85. [CrossRef]
34. Charles, R.Q.; 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 (NIPS'17), Long Beach, CA, USA, 4–9 December 2017; Curran Associates Inc.: Red Hook, NY, USA, 2017; pp. 5105–5114.
35. Mi, Z.; Luo, Y.; Tao, W. SSRNet: Scalable 3D Surface Reconstruction Network. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 967–976. [CrossRef]
36. Pratt, V. Direct least-squares fitting of algebraic surfaces. *Newsl. ACM SIGGRAPH* **1987**, *21*, 145–152.
37. Point Cloud 3D Models, Sketchfab, [Online]. Available online: https://sketchfab.com/tags/point-cloud (accessed on 25 May 2019).