sensors

MDPI

Article

# Automatic Indoor as-Built Building Information Models Generation by Using Low-Cost RGB-D Sensors

**Yaxin Li [1,2,*], Wenbin Li [2], Shengjun Tang [3], Walid Darwish [4,5], Yuling Hu [2] and Wu Chen [1,2]**

1 Shenzhen Research Institute, The Hong Kong Polytechnic University, Shenzhen 518057, China; wu.chen@polyu.edu.hk
2 Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, Hong Kong, China; wb.li@polyu.edu.hk (W.L.); lindsayuling.hu@polyu.edu.hk (Y.H.)
3 Guangdong Key Laboratory of Urban Informatics & Shenzhen Key Laboratory of Spatial Smart Sensing and Services & Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ) & Research Institute for Smart Cities, School of Architecture and Urban Planning, Shenzhen University, Shenzhen 518052, China; shengjuntang@szu.edu.cn
4 Department of Electronic and Informatics, Faculty of Engineering, Vrije Universiteit Brussel, 1050 Brussels, Belgium; dawalid@etrovub.be
5 Geomatics Engineering Lab, Civil Engineering Department, Faculty of Engineering, Cairo University, Cairo 12613, Egypt
* Correspondence: yaxin.pu.li@connect.polyu.hk

Received: 10 December 2019; Accepted: 2 January 2020; Published: 4 January 2020


**Abstract:** To generate indoor as-built building information models (AB BIMs) automatically and economically is a great technological challenge. Many approaches have been developed to address this problem in recent years, but it is far from being settled, particularly for the point cloud segmentation and the extraction of the relationship among different elements due to the complicated indoor environment. This is even more difficult for the low-quality point cloud generated by low-cost scanning equipment. This paper proposes an automatic as-built BIMs generation framework that transforms the noisy 3D point cloud produced by a low-cost RGB-D sensor (about 708 USD for data collection equipment, 379 USD for the Structure sensor and 329 USD for iPad) to the as-built BIMs, without any manual intervention. The experiment results show that the proposed method has competitive robustness and accuracy, compared to the high-quality Terrestrial Lidar System (TLS), with the element extraction accuracy of 100%, mean dimension reconstruction accuracy of 98.6% and mean area reconstruction accuracy of 93.6%. Also, the proposed framework makes the BIM generation workflows more efficient in both data collection and data processing. In the experiments, the time consumption of data collection for a typical room, with an area of 45–67 $m^2$, is reduced to 4–6 min with an RGB-D sensor from 50–60 min with TLS. The processing time to generate BIM models is about half minutes automatically, from around 10 min with a conventional semi-manual method.

**Keywords:** as-built BIMs; automatic; RGB-D sensors

## 1. Introduction

Building information models (BIMs), including as-designed BIMs (AD BIMs) and as-built BIMs (AB BIMs), are the digital representations for the whole life cycle management from design, construction to demolition [1], which are widely used in the areas of construction management [2], computer games [3], indoor navigation [4], and emergency response [5]. Different from traditional 3D models contain spatial information only, the BIMs have more information (e.g., material, functions, and topological

information) and have digital representations of related information, which makes the modification and maintenance operation more practical and efficient. AD BIMs generation is the process in the design stage of facilities and becoming more and more common, while AB BIMs generation is to create models for existing facilities, which is more challenge, especially for the indoor environment considering disturbances from numerous occlusion and complex non-planar elements [6]. Furthermore, most of the existing facilities do not have the BIM document, or the BIM document is out of date due to constant renovation [7].

The standard process of generating indoor BIMs for existing buildings includes three steps [1,6,8]. The first step collects a 3D point cloud with spatial and/or color information. For the traditional surveying methods, pieces of the range-based equipment, such as the range finder, handheld 3D scanner [9], or the terrestrial laser scanner (TLS) [10], are widely used to capture 3D point clouds of facilities. Meanwhile, 2D image-based sensing technologies, such as RGB cameras, can also be used for the 3D point cloud generation with the help of structure from motion (SfM) [11,12] or simultaneous localization and mapping (SLAM) technologies [13–15]. The second step is the semantization of the 3D point cloud. That semantic information includes not only the spatial information, such as positions, dimensions (areas and volume), and attribute information (material and functions) of the individual construction components, but also the relationship to each other, such as dependency, topology, or joints. In the final step, well-trained and professional modelers create AB BIMs manually, referring to the semantic information. However, there are many problems with this conventional approach. Firstly, TLS is very expensive and ponderous which makes it not efficient and economical to collect the 3D point cloud, especially for the indoor environment [1]. The RGB camera is cheaper and easier to use in the indoor environment, but the 3D point cloud extraction from camera data requires a heavy computation cost, and time-consuming processing, and extra scale control. Also, the dense 3D point cloud is not always available with an RGB camera, especially in indoor environments with low lighting or sparsely textured areas [16]. Recently, some handheld 3D scanners (e.g., Zeb one and Zeb Revo), with the advantages of relatively low-cost and portable when compared with LTS, are widely used in the 3D modeling field. However, the price of them still up to several thousand dollars, which is not affordable for some commercial users. Besides, most of them require an external battery which is very heavy and not friendly for the data collection operation. Secondly, because the manual semantic processes are labor-intensive and skill-demanding, the productivity of those methods cannot keep up with the growing requirement for BIMs in the architectural/engineering/construction and facility management industry (ACE/FM).

In the past few years, researches have paid great efforts trying to find feasible solutions to reduce the cost and to enhance the efficiency of the indoor AB BIMs generation. To reduce the cost of the hardware, some researchers [16–19] proposed to apply the RGB-D technology to the indoor 3D reconstruction, as RGB-D sensors have the advantages of low cost, excellent portability and provision of synchronized color images together with dense depth maps, compared with traditional TLS and RGB camera. However, the 3D semantic information extraction from the point cloud from RGB-D sensors becomes more difficult, as the depth measurement quality of RGB-D sensors is much lower than that from TLS.

Meanwhile, some works were focused on replacing the manual process of 3D point cloud semantization with "automatic" or "semiautomatic" approaches, which can be divided into two groups: data/model-driven method [1] and deep learning-based methods.

The data-driven methods extract the semantic building information based on the features, shapes, materials, and statistics of raw measurement data, whereas model-driven methods customarily utilize predefined information or knowledge. Most of the current methods were developed by integrating both of them. Okorn et al. [20] proposed a method to extract the floor and ceiling data by using a histogram of height data with the assumption that the coordinate system of point cloud always being consistent with the real world in the height direction. The remaining points are projected onto a 2D ground plane, which is estimated from the floor data, and the line segments corresponding

to walls are extracted based on the Hough transform. However, the output of this method is only the floor plans, and other elements, such as the window or door, are unavailable. Also, the line segmentation would be problematic when the dataset with too many noises. In the method developed by Wang and Cho [21], the surfaces of the elements are estimated firstly by using the region growing algorithm, and the building geometries are then extracted from unorganized point clouds by collecting the intersection lines created by two extended surfaces in vertical and horizontal. This method is practical for high-quality point clouds collected by TLS, but it would be difficult to apply to the case when the quality of point clouds is low, i.e., the region growing algorithms are useless when the wall plane is narrow, or data have strong noise. Furthermore, some methods reconstruct the properties of doors and windows as well as the relationship between them and walls by detecting the openings. Previtali et al. [22] integrated the openings detection and a shape-driver algorithm to generate the BIM model with the windows and doors at the assumption of Manhattan-World geometry. This method provides a potential solution for the relationship reconstruction of sub-element (i.e., doors and windows) and walls, but the opening detection-based algorithms require the doors/windows should being opening during data collection. Macher et al. [6] proposed a semi-automatic approach for indoor 3D reconstruction of the entire building to reduce the time consumption and errors caused by the manual operation. However, this approach also detected the window and door based on openings. Moreover, this approach may be impractical to handle the low-quality point cloud because it was designed for high-quality point cloud collected by LTS. This similar problem is existing in [23,24].

The development of deep-learning technology opens new avenues for automatic AB BIMs generation but still in its infancy. Dai et al. [25] developed one indoor scene reconstruction system called ScanNet. This system firstly produces a coarse semantic segmentation result by using deep-learning and then refines the result manually in a WebGL application. The output only includes limited attribute information rather than full digital spatial and relationship information. Another work (named IM2CAD [26]) uses only one single photo as the input and output the CAD model of the scene by using a fully convolutional network (FCN) and faster region-based convolutional neural networks (FRCN) integrated architecture. This method is innovative for the information semantization but unpractical for automatic AB BIMs generation because the process from image to CAD model would miss important details of the sense (such as small elements, and boundaries of objects), and the BIM requires a 3D model for the whole environment rather than a single scene.

Although recent development makes the generation of indoor AB BIMs more efficient and convenient, there are still many problems need to be overcome: (1) current approaches are designed for high-quality point cloud and are impotent for the low-quality dataset from low-cost sensors, i.e., RGB-D sensors; (2) traditional method is inefficient and restricted because additional manual intervention is required in semantization and digitization; (3) the extraction of digital spatial and relationship information still not been addressed, especially for the low-quality point cloud dataset.

In this paper, we aim at establishing an automatic and efficient indoor AB BIMs generation framework with low-cost RGB-D sensors to address issues about the extraction of spatial and topological information and the digitalization related information to BIM format document. Inspired by the success of deep-learning technology in the semantic segmentation areas, we modify the fully convolutional network (FCN) to extract the attribute information of the elements (e.g., wall, ceiling, floor, windows, and door) from low-quality RGB-D data (color and depth images) [27]. Furthermore, a new method to extract digital spatial information and their relationships from the low-quality point cloud is proposed through its error distribution and some empirical knowledge of the indoor environment. Finally, we implement the workflows from raw color and depth sequences in commercial software to generate BIMs automatically.

The organization of the paper is as follows: Section 2 describes the workflow as well as the detail algorithms of the system. Section 3 outlines the result of the experiment. Section 4 includes the conclusions and discussions on the future development of the system.

## 2. Automatic BIM Generation Framework

This section describes the workflows, as well as the detail algorithms of our automatic BIM generation method. As shown in Figure 1, the whole framework can be divided into three stages.

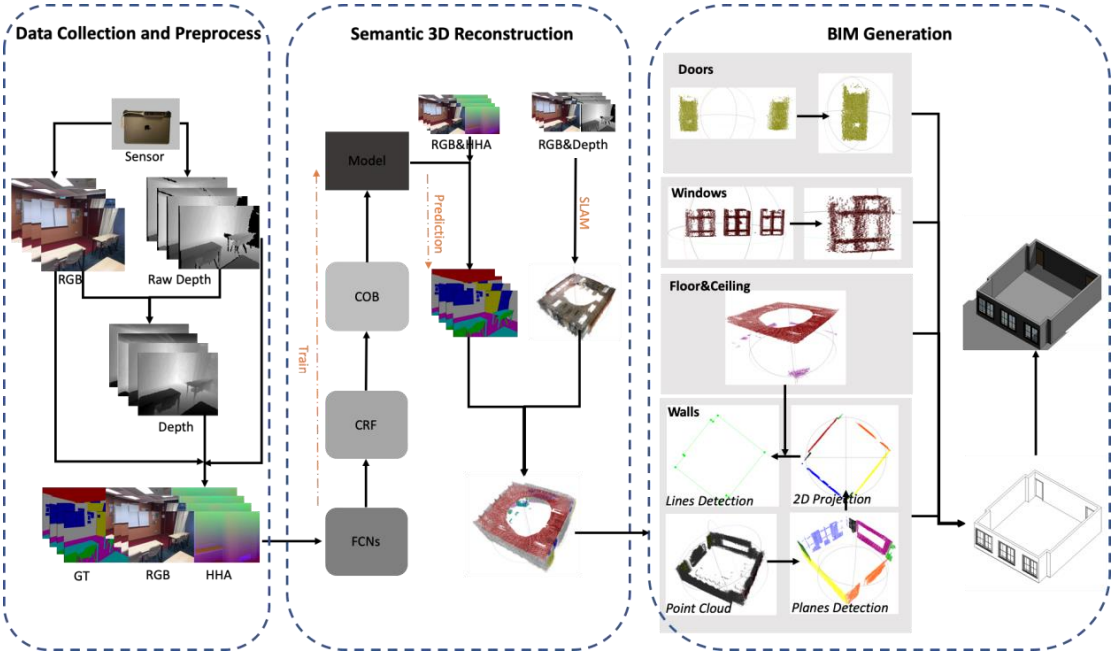



**Figure 1.** Three stages of automatic building information models (BIM) generation by using a low-cost RGB-D sensor.

The first stage is to collect datasets and to transfer the dataset to the structure accepted by neural networks. The raw RGB images and depth images are collected by an RGB-D sensor. Then, we apply the calibration method proposed in [28] to reduce the systemic error of the RGB-D sensor. A depth information encoding method proposed by Gupta et al. [29], named HHA, is used to encode the depth information as the neural networks are weak to process the unstructured geometric information duo to the fixed grid kernel structure. Finally, the RGB images, HHA images together with the ground truth labels, are packed together as the input of the training procedure of semantic 3D reconstruction stage.

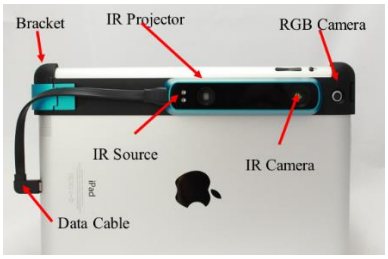
The second stage is to get the semantic 3D reconstruction result of the environment. First of all, a neural network is established for the 2D image semantic segmentation in the training procedure. This neural network is trained first with the image pairs and ground truth labels and is used to predict the semantic label of each pixel in the prediction procedure with the input of RGB images and HHA images. Then, with the input of RGB images and depth images, the 3D reconstruction is done by using the simultaneous localization and mapping (SLAM) method [30] with the output of the textured point cloud. Finally, the point cloud with semantic labels is generated by integrating the 2D semantic segmentation result and 3D reconstruction product.

The final stage is to extract the digital spatial and relationship information for the BIM generation. In this stage, the semantic point clouds are divided into different parts based on the labeled information provided in stage 2. The planes of floors and ceilings are estimated first by using the random sample consensus (RANSAC) algorithm [31]. As mentioned in Section 1, the point clouds collected by low-cost RGB-D sensors are much noisy, which makes it challenging to extract planes with conventional plane extraction methods. We propose a new "add-remove" method to overcome this problem to improve the quality of the extracted wall planes. Then, the wall planes are projected to the floor plane to get the 2D map of the walls. The start points and the endpoints of all lines are recovered using the line fitting algorithm. At the same time, the door and window point cloud are segregated into several parts

which only have one element by using local descriptors [32]. The position of the elements and their relationship to walls are recovered by calculating the closest distances. Meanwhile, the sizes of the elements are calculated based on the single frame point cloud rather than all point clouds to reduce the effects of measurement noise. Finally, the BIM model is automatically generated by integrating all the segmentations with one plug-in program under the Revit platform.

### *2.1. Data Collection and Preprocessing*

The hardware used to collect data in this paper is one type of RGB-D sensors named Structure sensor, which can be fitted to an iPad, an iPhone, or other mobile instruments (Figure 2). The system outputs 640 × 480 aligned RGB and depth images with the frequency up to 30 frames per second. The benefit of semantic segmentation with the depth information has been demonstrated by many studies [33–35]. However, the depth image from low-cost RGB-D sensors contains notable systemic errors. The calibration method [28] is first applied to improve depth measurement accuracy. The compare between the raw depth values and calibrated depth values for a wall is shown in Figure 3, and it is clearly shown the improvement of the calibrated depth as the wall should be a straight line. Additionally, as shown in Figure 4b, the occlusion or clutter, caused by the unpredictable variation of scene illumination, reflection of objects, or the out-of-range issues, makes the use of depth information more challenging. In this paper, the algorithm proposed by Levin et al. [36] is adopted to fill the occlusion region (as shown in Figure 4c), considering the filled depth is the input of the depth encoding algorithm used in the following steps. Finally, as shown in Figure 5, the depth information is encoded as HHA (horizontal disparity, height above ground, and the angle with gravity) format because the neural networks are weak to process the unstructured geometric information duo to the fixed grid kernel structure [29].




| Dimensions | 29mm x 28mm x 119.2mm |
|---|---|
| Weight | 95g |
| Max. Recommended Range | 3.5m+ |
| Min. Recommended Range | 40cm |
| Resolution | VGA (640 x 480)<br>QVGA (320 x 240) |
| Framerate | 30/60 frames per second |
| Field of View | Horizontal: 58 degrees<br>Vertical: 45 degrees |

**Figure 2.** The main elements of hardware used in this paper, which includes one RGB-D camera and one iPad [37]. The total cost of the equipment is about 708 USD, 379 USD for the Structure sensor, and 329 USD for iPad.

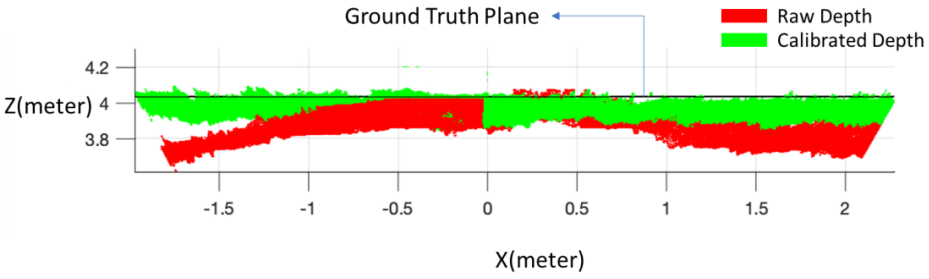



**Figure 3.** The comparison between the raw depth and calibrated depth.

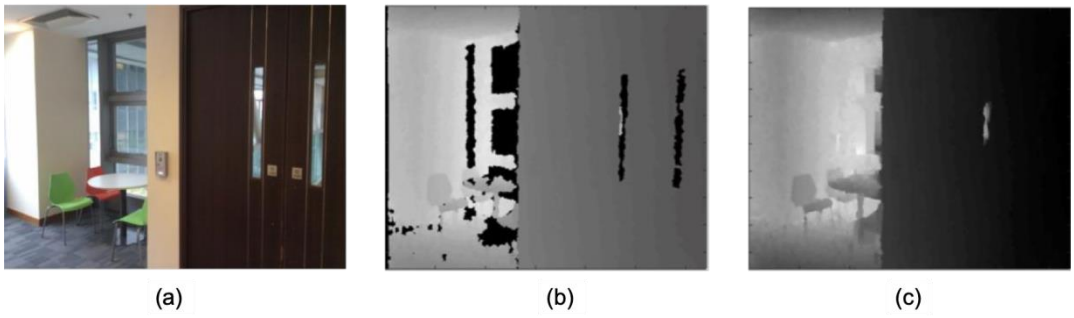



**Figure 4.** The example of (**a**) RGB image, (**b**) raw depth, and (**c**) filled depth.

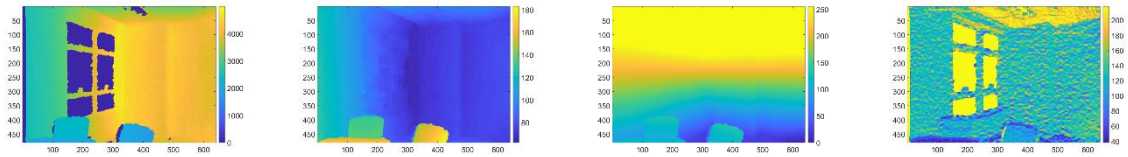



**Figure 5.** Examples of depth encoding. (**a**) Raw depth image, (**b**) the horizontal disparity, (**c**) the height above ground, (**d**) the angle with gravity.

### *2.2. Semantic 3D Reconstruction*

The semantic 3D reconstruction stage is divided into two parts. The first part is the semantic segmentation in 2D images based on the RGB information and depth information. The second part is the integration of 2D semantic segmentation results with RGB-D SLAM mapping based on the relationship between pixel-level semantic labels and point clouds.

For the pixel-level semantic segmentation task, the FCN [27] is accepted as the standard approach based on deep-learning, which is the first end-to-end neural network adopting the arbitrary size input and output pixel-level dense with the corresponding size [27,38,39]. However, the semantic segmentation result from FCN is not elaborate enough even with the combination of high layer coarse information and low layer excellent information. The conditional random field (CRF) [40] is one of the widely used algorithms to refine the semantic segmentation result of RGB images, but it only uses the boundary information from the color image. Meanwhile, convolutional oriented boundaries (COB) is used to extract boundary information based on the depth information [41]. Considering that the RGB-D sensors provide aligned color and depth images, we added a CRF layer and a COB layer at the end of raw two-branch FCN architecture to refine the semantic segmentation result of the neural network. The neural network used is still one end-to-end architecture with the image pairs as input and pixel-level semantic segmentation as output. In this paper, the label classes we used include door, window, floor, ceiling, and wall.

Then, we generate the semantic 3D point clouds by integrating the textured point clouds from RGB-D SLAM mapping and the 2D semantic segmentation from the neural network. First of all, the RGB-D SLAM method used by Endres et al. [42] is employed to generate the 3D point clouds from the RGB and depth information. This graph-based SLAM system calculates geometric relationships of adjacent frames based on RANSAC and iterative closest point (ICP). For each frame, the textured point cloud in the local coordinate system is generated based on the color and depth information. The textured 3D point cloud is generated by combining the point clouds of different frames with transformation matrixes from the SLAM system. Secondly, with the depth information and pixel-level semantic segmentation result, the semantic point cloud for each frame is outputted. Finally, similar to the conventional 3D reconstruction method, the semantic 3D point clouds are generated by integrating the semantic point cloud of each frame and the transformational information from the SLAM system. In practice, the overlap of adjacent frames and the incorrect 2D semantic segmentation result make

label information of the 3D location ambiguous. Our system accepts to fuse the semantic segmentation results with overlap by using a Bayesian update. In practice, by using the transformation information provided by SLAM, the segmentation results can be aligned into the same coordinate system, and the overlap information is used to update the label probability distribution. The recursive update function is shown in Equation (1).

$$P(X = L_i | C_{1,\ldots\ldots,m}) = \frac{P(X = L_i | C_{1,\ldots\ldots,m-1}) P(X = L_i | C_m)}{K}, \quad (1)$$

where, $L_i$ is the prediction result of the label; $C_k$ is the semantic point cloud for $m^{th}$ frame; $K$ is one constant to normalize the distribution.

### *2.3. The Transformation from Semantic 3D Reconstruction to BIM Format 3D Model*

In this stage, the semantic 3D point cloud would be separated into different parts (e.g., floor, ceiling, wall, door, and window) based on the labeled information firstly. Then, the properties of the elements, as well as the bilateral relationship with each other, are extracted. Finally, the BIM format 3D models are generated based on the obtained information using the Revit platform. As shown in Figure 6, the point cloud generated from low-cost RGB-D sensors is very noisy even after the systemic error calibration, which makes most elements extraction methods not suitable for this situation. We develop one new element extraction algorithm based on the feature of the dataset as well as the empirical knowledge of the indoor environment.

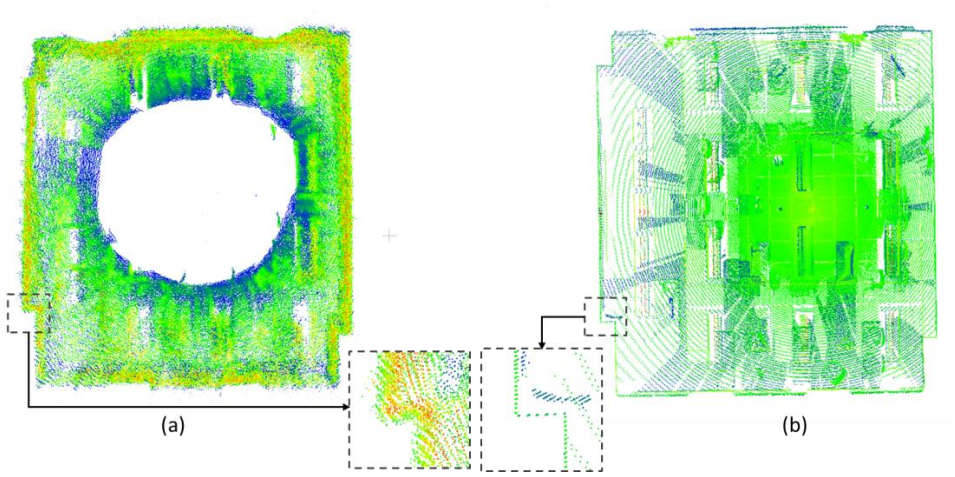



**Figure 6.** (**a**) The point cloud generated by low-cost RGB-D sensor, the empty area in the center area is caused by the data collection trajectory, (**b**) is the point cloud generated by terrestrial laser scanner (TLS).

#### 2.3.1. Wall Boundary Extraction

In the wall boundary extraction procedure, the wall point cloud is extracted based on the semantic label firstly, as Figure 7b shows. This point cloud always has a large number of noisy points (Figure 7b red circle areas) due to the measurement errors and aligned error of color and depth images. In this study, we remove those sparse outliers based on the distance distribution of point to neighbors in the input point cloud. The compare between the raw point cloud and the filtered result is shown in Figure 7b,c.

The first step for the wall boundary extraction is wall plane detection. As Figure 7d shows, point cloud in a different color is points for different wall planes. Traditionally, the plane extraction for a dense point cloud is based on the iteration of the RANSAC plane fitting method. For iteration $i$, the algorithm detects a plane $p_i$ from the input point cloud $\{P_i\}$. Those points, whose distance to the plane $p_i$ is less than the threshold $d_0$, are treated as inlier points and will be removed from $\{P_i\}$.

The remaining point cloud is $\{P_{i+1}\}$ which is the input of the next iteration $i+1$. This process repeats until there are not enough points in the remaining point cloud $\{P_{i+1}\}$. This method is effective for the point cloud collected by TLS because it nearly has no overlap for the points and always in high quality. However, as Figure 8a–c shows, this method can cause an over-detection problem when the input is the low-quality point cloud collected by a low-cost RGB-D sensor. The reason for the over-detection is the value of $d_0$ is too small to remove all the points of plane $p_i$. As Figure 8c shows, some noisy points belonging to the plane $p_i$ are not removed, which leads to another detected plane close to plane $p_i$. The parameter $d_0$ is used to number the inlier points for the RANSAC-based plane fitting method. In this paper, we still use $d_0$ for the plane fitting but use another parameter $T_{d1}$ to remove the points from $\{P_i\}$. This method can significantly reduce the influence the over-detection as Figure 8c–e shows.

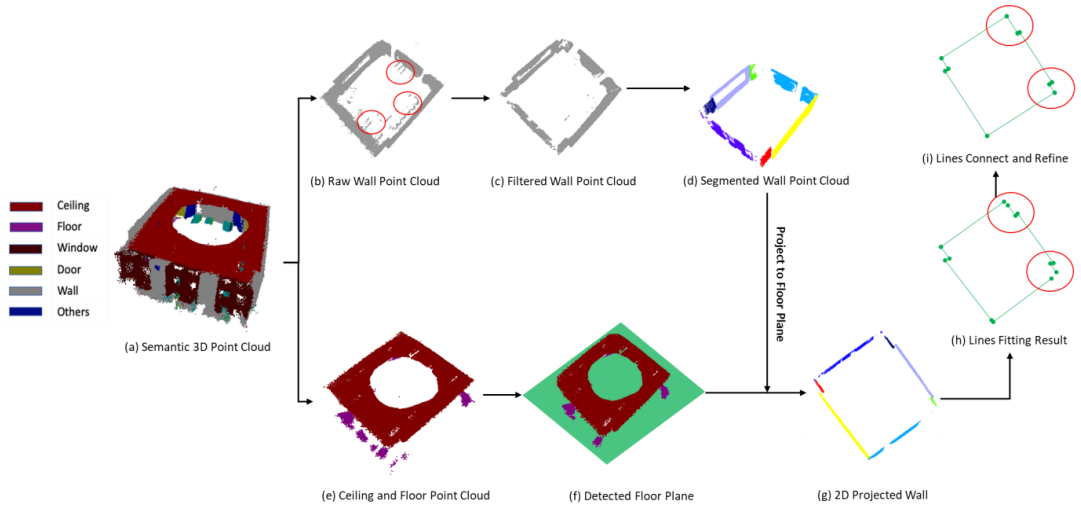



**Figure 7.** The workflow of wall boundary extraction.

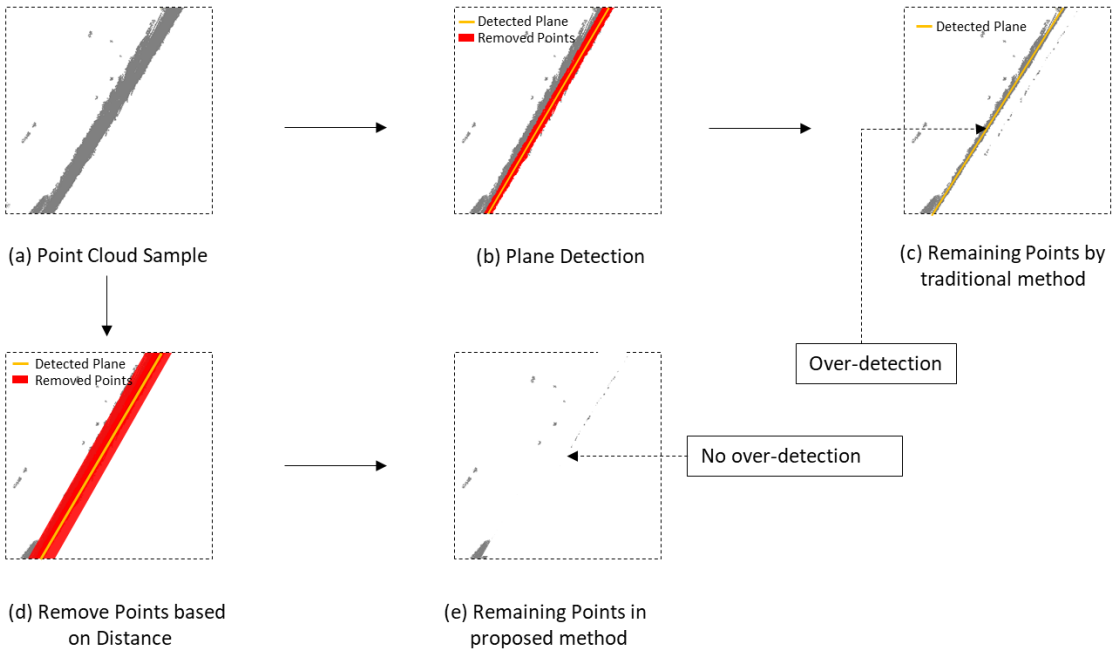



**Figure 8.** Description of the over-detection problem and the solution used in this paper.

Another problem is the over-removing of the point cloud. We remove the points close to the detected wall plane to overcome the over-detection problem. However, some of those points, which are at the joint area of walls, are useful for the detection of another plane. As Figure 9 shows, the gray point is the input point cloud, the green line is the wall plane detected by the plane fitting algorithm, and the yellow area is the region of removed points. In this case, the red rectangular area is another wall with a small size that will not be detected because the points have been removed. To address this problem, we project the detected wall plane to the floor plane to get the line of the wall in 2D mapping. The points around the endpoints of the line, the blue circle areas in Figure 9 are reserved to overcome the over-removing problem.

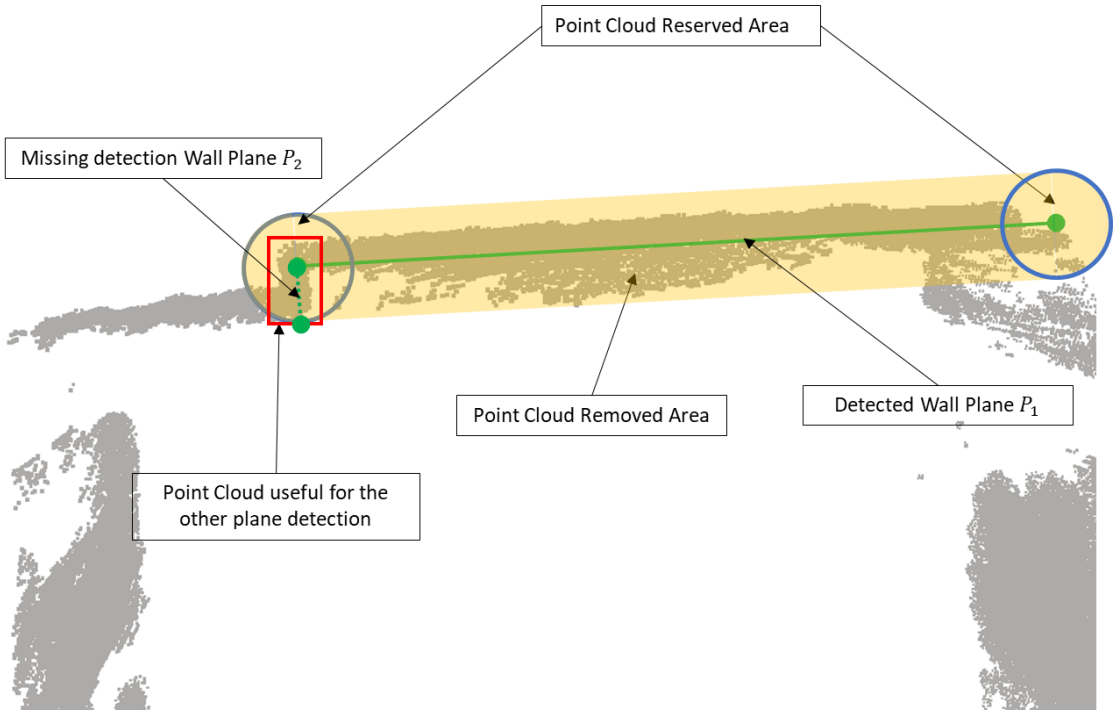



**Figure 9.** Description of the over removing problem as well as the proposed solution.

The detail implementation of the algorithm is presented in Algorithm 1. Where, $F(.)$ is the plane fitting function based on RANSAC algorithm, $C(.)$ is the function to count the point number of the point cloud, $D_1(.)$ is the function to calculate the distance between the points and plane, $D_1(.)$ is the function to calculate the minimum distance between the points and line in floor plane, $Dis(.)$ is the function to calculate the distance between the planes, $Ang(.)$ is the function to calculate the angle between the planes, and $Proj(.)$ is the function to project plane to the floor plane. Additionally, Figure 10 presents an example to extract 2D wall lines from the raw point cloud. Figure 10a is the input point cloud of the walls. Figure 10b,c is the iteration of the plane detection and points removing with the method mentioned above. The iteration will terminate when the number of remaining points is less than the threshold. Then, we can get the segmented wall planes, which are set to different colors as Figure 10d. Finally, the 2D wall lines are extracted by projecting the wall planes to the floor plane and the result is showed in Figure 10e.

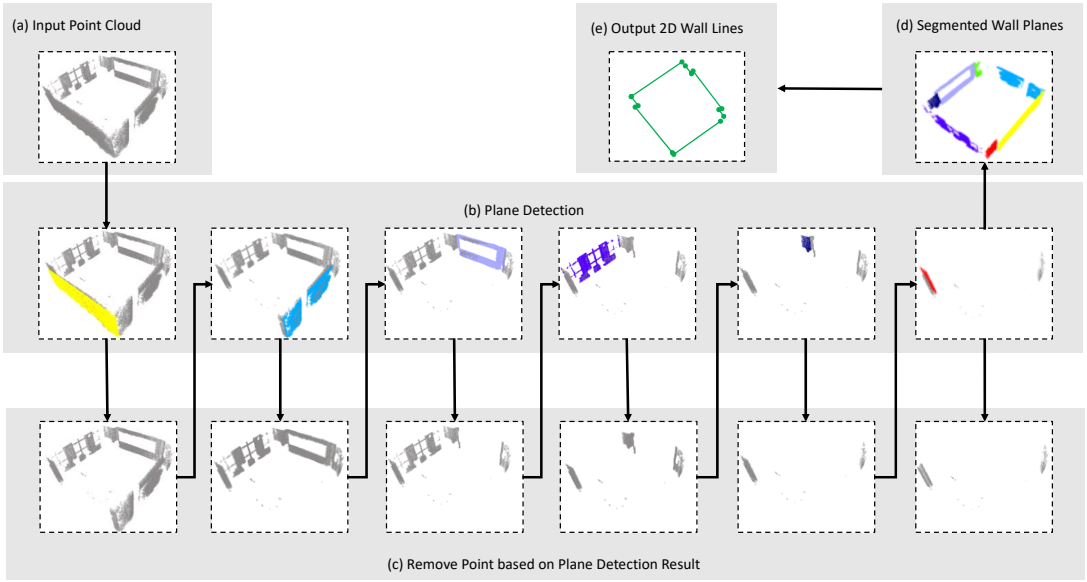



**Figure 10.** One detail example of 2D wall lines extraction.

As shown in Figure 7h, some extracted lines are not connected because the point cloud collected not covers the whole environment, or there are still some deviations between the true value and detected results. To overcome this problem, we propose a 2D wall line connect and refine algorithm based on the vertical distances of the lines and the normal angles of the corresponding detected wall planes. Also, this method is designed with the assumption that the walls in the applied environment are perpendicular and straight. First of all, one reference line $L_0$ are random chose from all the 2D wall lines $\{L\}$, and two vertexes of $L_0$ are presented as the start point $Point_s$ and the endpoint $Point_e$. Moreover, all the distance between the endpoint of $L_0$ and the vertexes of the other lines are calculated. The line whose vertex with the minimum distance is treated as the adjacent line present as $L_1$, and this vertex is treated as the start point of $L_1$ and another vertex is the endpoint. Then, with line $L_1$ as the reference line, this procedure will repeat until the found start point to be the start point of $L_0$. In practice, as Figure 11 shows, the isolation of the lines can be divided into two groups, the first group is caused by the missing of the edges and the second group is caused by the adjacent lines are too short to get the intersection point. The classification of these two groups is based on the angle between the normal vectors of corresponding detected wall planes. For example, line $L_i$ with the plane normal vector $N_i$ and adjacent line $L_{i+1}$ with the plane normal vector $N_{i+1}$, the angle between $N_i$ and $N_{i+1}$ could be calculated and presented as $\varphi_i^{i+1}$. As Figure 11 shows, if the value of $\varphi_i^{i+1}$ is smaller than the threshold $\varphi_0$, this connection relationship is initialized as the first group. The central point of the line, whose vertexes is the endpoint of $L_i$ and start point of $L_{i+1}$, are calculated. The end point of $L_i$ is adjusted to the foot of perpendicular through the central point to $L_i$. Similarly, the start point of $L_{i+1}$ is adjusted to the foot of perpendicular trough the central point to $L_{i+1}$. Otherwise, if the value of $\varphi_i^{i+1}$ is larger than the threshold $\varphi_0$, the connection relationship would be initialized as the second group. In this group, $L_i$ and $L_{i+1}$ are connected by replace the end point of $L_i$ and the start point of $L_{i+1}$ with the extended intersection point of two lines. Here, $\varphi_0$ is equal to 45 degrees. Finally, with the height of the wall calculated from a distance between the floor plane and ceiling plane, the space boundary could be obtained. The detail implementation of the algorithm is presented in Algorithm 2, where, $A(.)$ is the function to calculate the angle between lines, and $D_v(.)$ is the function to calculate the close vertexes distance between lines.

**Algorithm 1** Wall Boundary extraction

**Input**: Lebeled wall point cloud:$\{P_0\}$
Distance threshold to remove the point cloud: $T_{d1}$
Distance threshold to filter the plane: $T_{d2}$
Angle threshold to filter the plane: $T_a$
Distance threshold to reserve the point cloud: $T_{d3}$
Percentage threshold to end the loop: $T_p$

**Output**: 2D Wall Lines in floor plane: $\mathbb{L}$

**initialize:** remaining point cloud:$\{P_r\} = \{P_0\}$
**while** $C(\{P_r\}) < C\big(\{P_0\} * T_p\big)$ **do**
    Plane candidate: $p_c = F(\{P_r\})$
    **if** $\mathbb{L}$ is empty **then**
        add $Proj(p_c)$ to $\mathbb{L}$
        Distance between points and plane: $\mathbb{DP} = D_1(\{P_r\})$
        Distance between points and line in floor plane: $\mathbb{DF} = D_2(Proj(p_c), \{P_r\})$
        **for** $i = 1,\ i \leq C(\{P_r\}),\ i++$ **do**
            **if** $\mathbb{DP}_i < T_{d1}$ and $\mathbb{DF}_i < T_{d3}$ **then**
                add $\{P_r\}_i$ to ready to remove set $\{P_m\}$
            **end if**
        **end for**
    **else**
        **for** $j = 1,\ j \leq length(\mathbb{L}),\ j++$ **do**
            Distance between planes: $d_p = Dis\big(p_c, \mathbb{L}_j\big)$
            Angle between planes: $a_p = Ang\big(p_c, \mathbb{L}_j\big)$
            **if** $d_p < T_{d2}$ and $a_p < T_a$ **then**
                Continue
            **end if**
        **end for**
        **if** $j == length(\mathbb{L})$ **then**
            add $Proj(p_c)$ to $\mathbb{L}$
            Distance between points and plane: $\mathbb{DP} = D_1(\{P_r\})$
            Distance between points and line in floor plane: $\mathbb{DF} = D_2(Proj(p_c), \{P_r\})$
            **for** $i = 1,\ i \leq C(\{P_r\}),\ i++$ **do**
                **if** $\mathbb{DP}_i < T_{d1}$ and $\mathbb{DF}_i < T_{d3}$ **then**
                    add $\{P_r\}_i$ to ready to remove set $\{P_m\}$
                **end if**
            **end for**
        **end if**
    **end if**
    remove point from $\{P_r\}$: $\{P_r\} = \{P_r\} - \{P_m\}$
**end while**
**Return**: $\mathbb{L}$

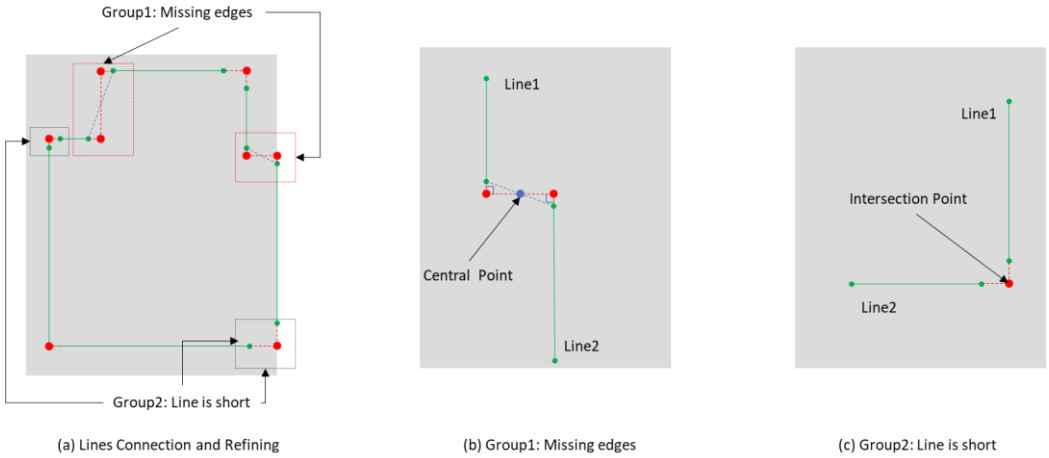



**Figure 11.** 2D wall lines connection and refining.

**Algorithm 2:** 2D Wall Lines Connection and Refining

**Input**: 2D Wall Lines in floor plane: $\mathbb{L}$
Angle threshold: $\varphi_0$

**Output**: Connection vectors of 2D wall lines: $L$

**initialize**: Random Select one line $L_0$ from $\mathbb{L}$
Remove $L_0$ from $\mathbb{L}$, and add $L_0$ to $L$
**while** $\mathbb{L}$ is not empty **do**
    **for** $i = 1,\ i \leq\ length(\mathbb{L}),\ i++$ **do**
        For distance set $\mathbb{D}:\ \mathbb{D}_i = D_v(L_0,\ \mathbb{L}_i)$
    **end for**
    Find line candidate $L_c$ referring to the minimum value in $\mathbb{D}$
    Remove $L_c$ from $\mathbb{L}$, and add $L_c$ to $L$
**end while**
**for** $j = 1,\ j \leq length(L),\ j++$ **do**
    Calculate the angle between adjacent lines: $Ang = A\left(L_j, L_{j+1}\right)$
    **if** $Ang <\ \varphi_0$ **then**
        Extend lines to obtain intersection point
        Update the vertexes of $L_j, L_{j+1}$
    **else**
        Add one new line between $L_j, L_{j+1}$
    **end if**
**end for**
**Return:** $L$

### 2.3.2. Door and Window Extraction

The basic information of the BIM elements, such as door and windows, normally includes the position, size, and relationship with the other relative elements. In this section, we try to estimate the position and the size of doors and windows and reconstruct the relationship to the wall.

As Figure 12a,b shows, the input point cloud is segmented into several clusters based on the local descriptors [32] to make sure each cluster has only one interesting element. For each cluster, as Figure 12c shows, the point cloud is projected onto the floor plane for the 2D line fitting process. The position of the central point of the optimally fitted lines is treated as the $X, Y$ coordinates of corresponding element. Considering the bottom of doors always aligns with the floor plane, we assign the $Z$ value of door position as zero. For the window, the $Z$ value of the position is calculated by projecting the point cloud to the wall plane to get the optimal fitted line. The height of the central

point of the line is treated as the $Z$ value of the window position. In this paper, as Figure 12d shows, the width and the height of the elements are estimated from the image frame rather than the global 3D point cloud because the point cloud noise makes the extraction of boundary difficult.

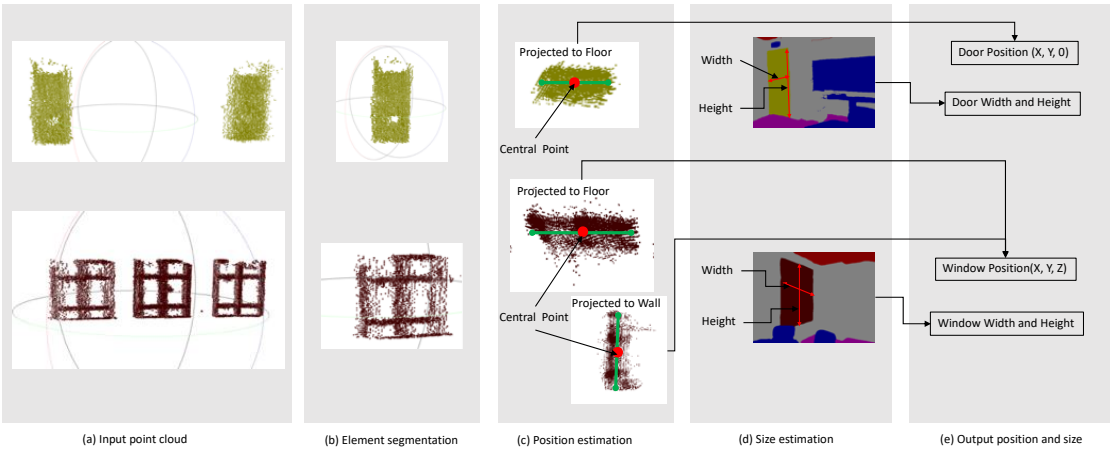



**Figure 12.** Position and size extraction of door and windows.

In the relationship reconstruction, we use the constraint that the door or the window always on the wall. Firstly, for each element, a door or window, the distances $\{Dis\}$ between its position and wall planes $\{P\}$, as well as the angle $\{A\}$ between the element plane and wall planes, are calculated. Then, $\{P\}$ would be filtered with the condition that the value of $\{A\}$ less than the threshold $\theta_0$, and the filtered result is $\left\{P^f\right\}$. Moreover, the element is subject to the wall which is in $\left\{P^f\right\}$ and have minimum value in $\{Dis\}$. Here, the value of $\theta_0$ is set to 30 degrees.

### 2.3.3. BIM Format 3D Model Generation Based on Geometry Information

Finally, we developed a plug-in based on the Revit 2018 for the BIM format 3D model generation. As Figure 13 shows, the input is the information extracted in the last stage, and the BIM format models could be generated automatically. The first line stored a sequence of points with the format $(X, Y)$, which represent the corner coordinate of the wall boundary in 2D projected mapping. The second line is the height of the wall calculated from the height difference between the floor plane and ceiling plane. The third line stores the information of doors with the position coordinate $(X, Y)$, height, and width. The last line stores the information of the windows, which includes the position coordinate $(X, Y, Z)$ as well as the height and width of the corresponding window element.

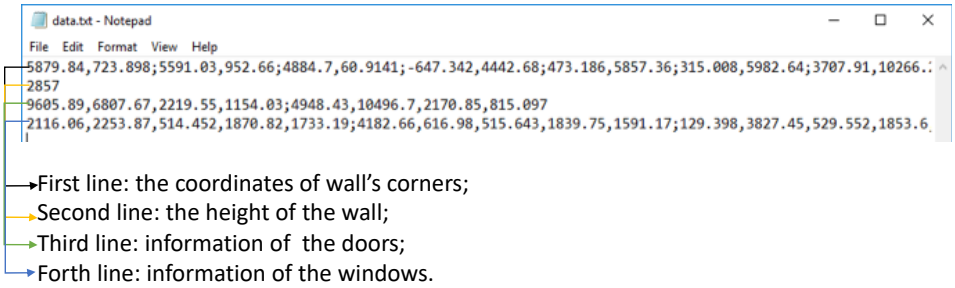



**Figure 13.** The example of the output geometry information of elements.

## 3. Experimental Tests and Discussion

In this section, three experiments have been done in three different classrooms at block Z of the Hong Kong Polytechnic University to test the performance of our proposed method. The operator collects the raw color and depth images by holding the hardware (showed in Figure 2) in hand and

walks around the room at a given route to cover as much area as possible. The metric used to validate the performance of the algorithm includes the accuracy of element detection, the length measurement accuracy of the room dimension, and the area measurement accuracy of the main reconstructed elements. The actual values of the measurement dimension are collected manually by a range finder, and the actual values of the measurement area are calculated manually with correct dimensions. Moreover, the efficiency of the proposed method is evaluated by being compared with the TLS based method and range finder based method in the time consumption and manual load. The values of the parameters used in the test are listed in Table 1.

**Table 1.** Values of parameters used in the test.

| $d_0$ (m) | $T_{d1}$ (m) | $T_{d2}$ (m) | $T_{d3}$ (m) | $T_a$ (degree) | $T_p$ | $\varphi_0$ (degree) |
|---|---|---|---|---|---|---|
| 0.1 | 0.25 | 0.25 | 0.25 | 5 | 0.05 | 45 |

Figure 14 shows the detailed processes of the three experiments. In these three rooms, each room has one ceiling and one floor. Meanwhile, the room for the first experiment has two windows, two doors, and eight different walls, the second room has three windows, two doors, and eight walls, and the last room which is more complex, has three windows, two doors, and ten different walls. Figure 15 presents the BIM format 3D models generated by the proposed method as well as the ground truth manually generated by the skillful modeler.


**Figure 14.** The detail processes from the raw point clouds to BIM format models.

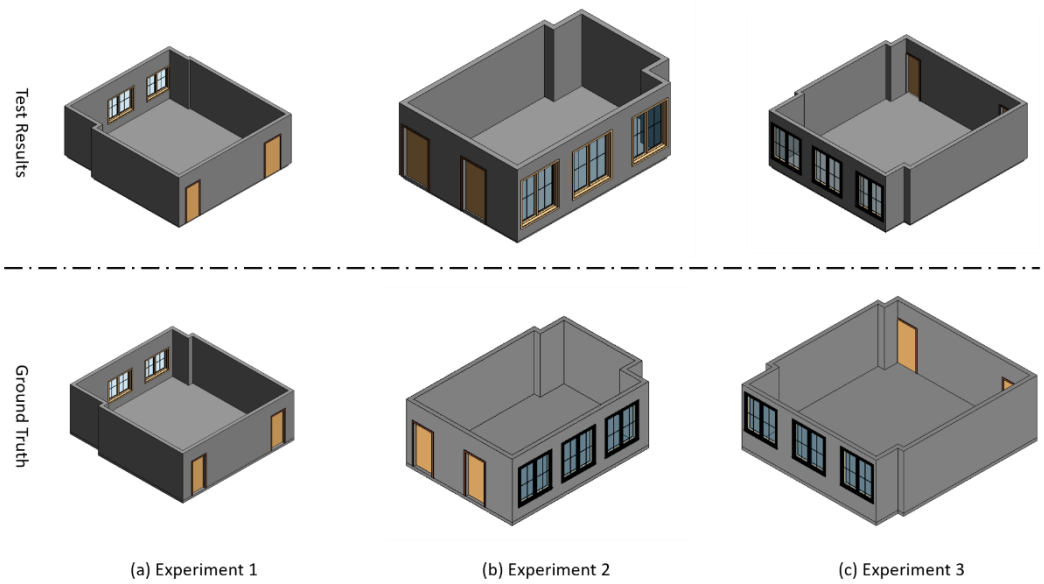



**Figure 15.** The compare of BIM format 3D models between the proposed method and ground truth.

Firstly, we validate the element extraction accuracy of the proposed, and the results are shown in Table 2. The result indicates that our proposed method extracts all the element objects, even for some small dimension walls, as shown in the red cycle area in Figure 14e.

**Table 2.** The element extraction results of the proposed method.

| Experiment ID | Elements | True Number | Extracted Number | Accuracy (%) |
|---|---|---|---|---|
| 1 | Windows | 2 | 2 | 100 |
| | Doors | 2 | 2 | 100 |
| | Walls | 8 | 8 | 100 |
| | Ceiling | 1 | 1 | 100 |
| | Floor | 1 | 1 | 100 |
| 2 | Windows | 3 | 3 | 100 |
| | Doors | 2 | 2 | 100 |
| | Walls | 8 | 8 | 100 |
| | Ceiling | 1 | 1 | 100 |
| | Floor | 1 | 1 | 100 |
| 3 | Windows | 3 | 3 | 100 |
| | Doors | 2 | 2 | 100 |
| | Walls | 10 | 10 | 100 |
| | Ceiling | 1 | 1 | 100 |
| | Floor | 1 | 1 | 100 |

Secondly, we compare the measured dimensions of rooms with the actual values measured by the range finder. Figure 16 shows the detail of the compare results. The first row is the measurement dimensions from our proposed method, and the second row is the true value. Also, quantitative analysis is shown in Table 3. For each room, we measure the width and length of the room, which determines the size of the room as well as the other two dimensions for evaluation. As Table 3 shows, average accuracy for three experiments is 98.6%%, 98.4%, 98.6% respectively with the maximal error for 214 mm and minimal error for 20 mm. The semantic segmentation based on deep learning can effectively extract the classes of elements in each frame, which makes the recognition more robust when compared to traditional methods.

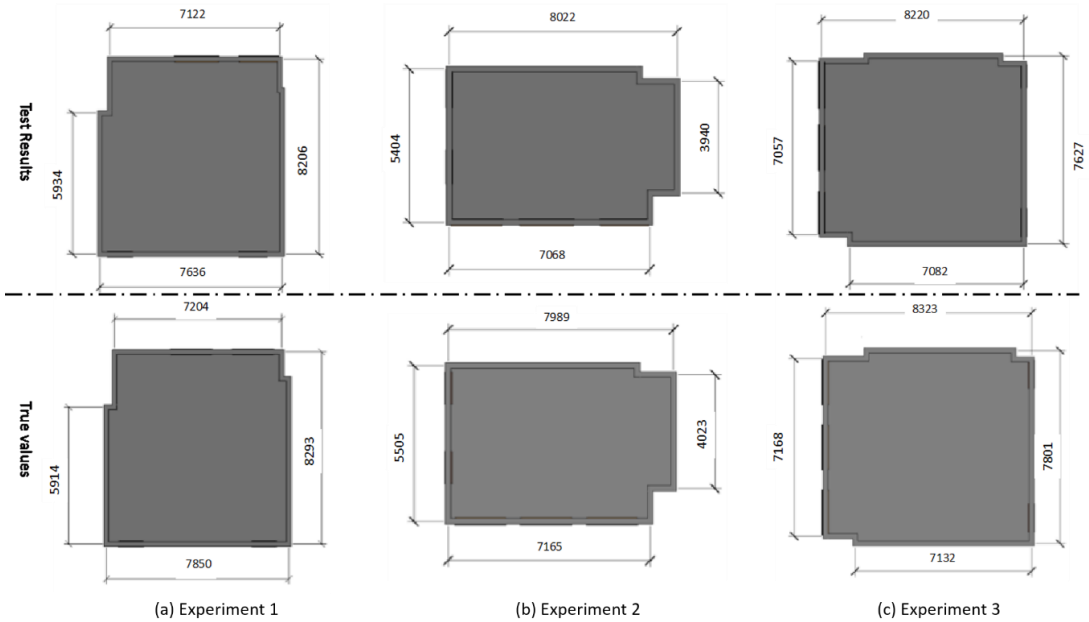



**Figure 16.** The compare between measurement dimensions of the room and the actual values collected by range finder.

**Table 3.** Quantitative analysis of measured room dimensions.

| Experiment ID | Wall ID | True Length (mm) | Detected Length (mm) | Error (mm) | Accuracy (%) | Average Accuracy (%) |
|---|---|---|---|---|---|---|
| 1 | Wall1 | 7240 | 7122 | 118 | 98.4 | 98.6 |
| | Wall2 | 5914 | 5934 | −20 | 99.7 | |
| | Wall3 | 7850 | 7636 | 214 | 97.3 | |
| | Wall4 | 8293 | 8206 | 87 | 99.0 | |
| 2 | Wall1 | 7989 | 8022 | −33 | 99.6 | 98.4 |
| | Wall2 | 5505 | 5404 | 101 | 98.2 | |
| | Wall3 | 7165 | 7068 | 97 | 98.6 | |
| | Wall4 | 4023 | 3904 | 119 | 97.0 | |
| 3 | Wall1 | 8323 | 8220 | 103 | 98.8 | 98.6 |
| | Wall2 | 7168 | 7057 | 111 | 98.5 | |
| | Wall3 | 7132 | 7082 | 50 | 99.3 | |
| | Wall4 | 7801 | 7627 | 174 | 97.8 | |

Thirdly, considering the area is one of the most important attributions of the BIM elements, we compare the area measurement of extracted elements with the actual values, and the results are shown in Table 4. The average area measurement accuracy for three experiments is better than 91.9%, with the best 96.5% for experiment three. Meanwhile, for all three experiments, the area measurements of the walls, ceiling, and floor are in high accuracy better than 92.2%. The measured accuracies of doors and windows range from 74.7% to 96.9%, which is not as good as the other elements. The reason is that the true areas of the windows and doors are minimal, which makes the accuracy more sensitive to measurement errors.

**Table 4.** Quantitative analysis of measured element areas.

| Experiment ID | Elements | True Area ($m^2$) | Detected Area ($m^2$) | Error ($m^2$) | Accuracy (%) | Average Accuracy (%) |
|---|---|---|---|---|---|---|
| 1 | Walls | 88.44 | 86.18 | 2.26 | 97.4 | 92.4 |
| | Ceiling | 67.2 | 64.8 | 2.4 | 96.4 | |
| | Floor | 67.2 | 64.8 | 2.4 | 96.4 | |
| | Doors | 4.2 | 4.07 | 0.13 | 96.9 | |
| | Windows | 6.12 | 4.57 | 1.55 | 74.7 | |
| 2 | Walls | 67.77 | 65.88 | 1.89 | 97.2 | 91.9 |
| | Ceiling | 45.35 | 41.79 | 3.56 | 92.2 | |
| | Floor | 45.35 | 41.79 | 3.56 | 92.2 | |
| | Doors | 4.2 | 4.8 | −0.6 | 85.7 | |
| | Windows | 9.18 | 9.87 | −0.69 | 96.8 | |
| 3 | Walls | 83.55 | 81.65 | 1.9 | 97.7 | 96.5 |
| | Ceiling | 67.03 | 64.82 | 2.21 | 96.7 | |
| | Floor | 67.03 | 64.82 | 2.21 | 96.7 | |
| | Doors | 4.2 | 3.97 | 0.23 | 94.5 | |
| | Windows | 9.18 | 9.47 | −0.29 | 96.8 | |

Fourth, we test the performance of the proposed method in the "narrow" wall extraction. In this test, we treated the wall of which the length less than three meters as a "narrow" wall. There are four, four, and six narrow walls, respectively, for three experiments. As shown in Figure 17, all the narrow walls, length ranges from 319 mm to 2554 mm, are detected by the proposed algorithm. This is due to that the algorithm developed in this paper overcomes the influence of the over-detection problem significantly and prevents the removal of the point cloud of narrow walls. Also, Table 5 shows the quantitative analysis of measured narrow walls. Apart from some very narrow walls (length less than 400 mm) and individual cases, the accuracy of the most measurements is better than 80%, with the average measured accuracy 75.3%, 81.3%, and 80.5% for three experiments. There are two reasons for the narrow wall measured results are not as good as the measured room sizes. The first one is that the length of some walls is too small, which make the accuracy is sensitive to the measurement error. The second one is that the accumulation error of the SLAM system will cause a significant error at the end of the frame sequences. The closure error between the start frames and end frames makes the extraction of narrow walls more challenge especially when the point cloud is in low-quality.

**Table 5.** Quantitative analysis of measured "narrow" walls.

| Experiment ID | Narrow Wall ID | True Length (mm) | Detected Length (mm) | Error (mm) | Accuracy (%) | Average Accuracy (%) |
|---|---|---|---|---|---|---|
| 1 | Wall1 | 380 | 433 | −53 | 86.1 | 75.3 |
| | Wall2 | 320 | 200 | 120 | 62.5 | |
| | Wall3 | 2554 | 2223 | 331 | 87.0 | |
| | Wall4 | 1054 | 1406 | −352 | 65.5 | |
| 2 | Wall1 | 319 | 458 | −139 | 56.4 | 81.3 |
| | Wall2 | 824 | 961 | −140 | 83.0 | |
| | Wall3 | 1163 | 1031 | 132 | 88.7 | |
| | Wall4 | 1284 | 1320 | −36 | 97.2 | |
| 3 | Wall1 | 332 | 202 | 130 | 60.8 | 80.5 |
| | Wall2 | 1655 | 1805 | −150 | 90.9 | |
| | Wall3 | 720 | 951 | −231 | 67.9 | |
| | Wall4 | 358 | 322 | 36 | 89.9 | |
| | Wall5 | 1191 | 1138 | 53 | 95.5 | |
| | Wall6 | 301 | 368 | −67 | 77.7 | |

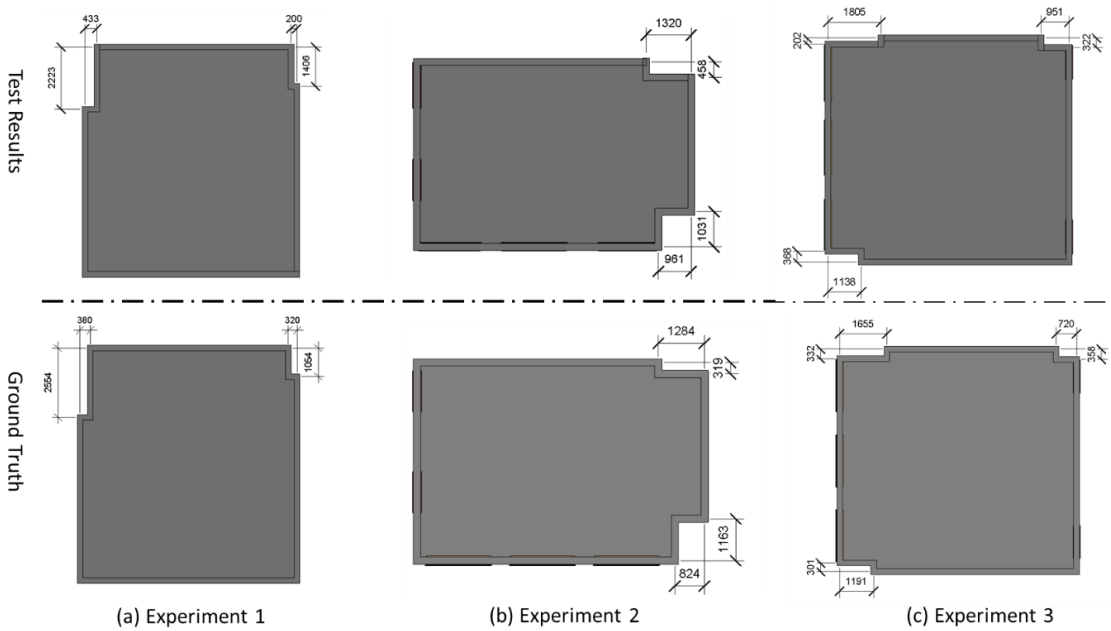



**Figure 17.** The compare between measurement length of the "narrow" walls and the actual values collected by range finder.

Finally, we compare the time consumption of our proposed framework with the conventional TLS based method and manual surveying method. In manual surveying method, the operator should measure the dimensions of all the required elements such as the height of the room, the size of the doors, the length of the walls et al., and create the BIM format model based on collected information without the colored point cloud or 3D mesh output. The LTS used in the test is Leica ScanStation 2, which is interoperable with Lecia System 1200. Table 6 shows the information about the dataset collected by the LTS and structure sensor for three experiments. Also, a downsample operation with factor five is applied to the raw point cloud generated by the structure sensor because the frame overlapping makes the raw point cloud enormous. As Table 7 shows, using TLS to collect the dataset always costs more time and requires more manual load because of that, the setup of the equipment and the scanning process are typically time-consuming. The manual surveying would cost less time for data collection when compared to TLS based method, but the data processing work would cost more time without the point cloud as the reference. Our proposed method costs much less time than those two methods, not only in data collection but also in the data processing. In the data collection, the handheld RGB-D sensor we used does not require as much preparation work as TLS and does not need to record the measurement manually like a manual surveying method. Also, the data collection in our framework is handled by only one operator. In data processing, our method is genuinely automatic without any manual intervention and the processing time for all three cases is around 30s. With the improvements in those two aspects, the whole workflow is accelerated from about 200 min for TLS to about 17 min with our method.

**Table 6.** Information about dataset generated by TLS and the Structure sensor.

| | TLS | | Structure Sensor | | |
|---|---|---|---|---|---|
| **Experiment ID** | **Point Number** | **Station Number** | **Frame Number** | **Raw Point Number** | **Sampled Point Number** |
| **1** | 576,627 | 1 | 61 | 2,939,060 | 587,812 |
| **2** | 373,510 | 1 | 37 | 1,713,590 | 342,718 |
| **3** | 571,825 | 1 | 66 | 2,892,520 | 578,504 |

**Table 7.** Efficiency evaluation of the proposed framework.

| | TLS | | Range Finder | | Our Proposed | |
|---|---|---|---|---|---|---|
| **Experiment ID** | **Collection** | **Processing** | **Collection** | **Processing** | **Collection** | **Processing** |
| 1 | ~3600 s | ~600 s | ~360 s | ~900 s | ~300 s | 34.2 s |
| 2 | ~3000 s | ~480 s | ~300 s | ~600 s | ~240 s | 26.7 s |
| 3 | ~3600 s | ~720 s | ~420 s | ~1020 s | ~360 s | 36.5 s |
| **Total** | 200 min | | 60 min | | 16.7 min | |
| **Output Point Cloud** | Yes | | No | | Yes | |
| **Automatically** | No | | No | | Yes | |
| **Manual Load** | Much | | Median | | Little | |

## 4. Conclusions

In this paper, we proposed an automatic and efficient indoor AB BIMs generation framework by using low-cost RGB-D sensors. Firstly, we calibrate the low accuracy RGB-D sensor to increase the measurement accuracy and operation range. Then, a deep-learning-based method is used for the semantic 3D reconstruction of the indoor environment with the color and depth images pairs as the input. This method is more effective and economical when compared to traditional manual methods for segmentation. Also, we design a new procedure to transform unstructured 3D point cloud to BIM format 3D model with a low-cost RGB-D sensor.

The experiment results indicate that this method is robust and has acceptable accuracy to handle the noisy 3D point cloud with the average accuracy of 98.6% for dimension reconstruction, average accuracy of 93.6% for area reconstruction and average accuracy about 80% for the narrow wall dimension reconstruction. The time consumption is reduced from 120 min to 16.7 min for the three experiments when compared to the traditional manual TLS based method. In detail, the time requirement of data collection is reduced from 170 min to 15 min, and the time requirement of data processing is reduced from 30 min to 1.7 min. Thus, the framework proposed in this paper, which using the low-cost and portable RGB-D sensor to replace the costly TLS to collect the 3D indoor dataset, provides a potential solution for the AB BIMs generation. The next step of this research will address the issues about the extraction of the attribute (e.g., material and functions) information of the individual construction components as well as the extraction of more complex construction components (e.g., furniture and appliance). Also, we will apply the proposed framework to other sensors, such as Structure Mark II and Microsoft Azure Kinect DK, to get a better result.

**Author Contributions:** Conceptualization, Y.L.; methodology, Y.L., W.L. and S.T.; software, Y.L., W.L., W.D. and S.T.; validation, Y.L., W.L. and Y.H.; formal analysis, Y.L. and W.L.; data curation, Y.L. and Y.H.; writing—original draft preparation, Y.L.; writing—review and editing, W.C. and Y.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** The research was substantially funded by Shenzhen Science and Technology Innovation Commission (Project No. JCYJ20170818104822282), Hong Kong Research Grants Council (RGC) Competitive Earmarked Research Grant (Project No: 152223/18E), and research fund from the Research Institute of Sustainable Urban Development, The Hong Kong Polytechnic University.

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

## References

1. Volk, R.; Stengel, J.; Schultmann, F. Building Information Modeling (BIM) for existing buildings—Literature review and future needs. *Autom. Constr.* **2014**, *38*, 109–127. [CrossRef]
2. Hartmann, T.; Van Meerveld, H.; Vossebeld, N.; Adriaanse, A. Aligning building information model tools and construction management methods. *Autom. Constr.* **2012**, *22*, 605–613. [CrossRef]
3. Yan, W.; Culp, C.; Graf, R. Integrating BIM and gaming for real-time interactive architectural visualization. *Autom. Constr.* **2011**, *20*, 446–458. [CrossRef]

4. Becker, T.; Nagel, C.; Kolbe, T.H. A multilayered space-event model for navigation in indoor spaces. In *3D Geo-information Sciences*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 61–77.
5. Li, N.; Becerik-Gerber, B.; Krishnamachari, B.; Soibelman, L. A BIM centered indoor localization algorithm to support building fire emergency response operations. *Autom. Constr.* **2014**, *42*, 78–89. [CrossRef]
6. Macher, H.; Landes, T.; Grussenmeyer, P.J.A.S. From point clouds to building information models: 3D semi-automatic reconstruction of indoors of existing buildings. *Appl. Sci.* **2017**, *7*, 1030. [CrossRef]
7. Xue, F.; Lu, W.; Chen, K.; Zetkulic, A. From Semantic Segmentation to Semantic Registration: Derivative-Free Optimization–Based Approach for Automatic Generation of Semantically Rich As-Built Building Information Models from 3D Point Clouds. *J. Comput. Civil Eng.* **2019**, *33*, 04019024. [CrossRef]
8. Hong, S.; Jung, J.; Kim, S.; Cho, H.; Lee, J.; Heo, J. Semi-automated approach to indoor mapping for 3D as-built building information modeling. *Comput. Environ. Urban. Syst.* **2015**, *51*, 34–46. [CrossRef]
9. Thomson, C.; Apostolopoulos, G.; Backes, D.; Boehm, J. Mobile laser scanning for indoor modelling. *ISPRS Ann. Photogramm. Remote. Sens. Spat. Inf. Sci.* **2013**, *5*, W2. [CrossRef]
10. Mill, T.; Alt, A.; Liias, R. Combined 3D building surveying techniques–terrestrial laser scanning (TLS) and total station surveying for BIM data management purposes. *J. Civ. Manag.* **2013**, *19*, S23–S32. [CrossRef]
11. Ullman, S. The interpretation of structure from motion. Proceedings of the Royal Society of London. *Ser. B Biol. Sci.* **1979**, *203*, 405–426.
12. Turner, D.; Lucieer, A.; Watson, C. An automated technique for generating georectified mosaics from ultra-high resolution unmanned aerial vehicle (UAV) imagery, based on structure from motion (SfM) point clouds. *Remote. Sens.* **2012**, *4*, 1392–1410. [CrossRef]
13. Mur-Artal, R.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
14. Engel, J.; Schöps, T.; Cremers, D. LSD-SLAM: Large-scale direct monocular SLAM. In Proceedings of the 13th European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 834–849.
15. Mur-Artal, R.; Tardós, J.D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
16. Henry, P.; Krainin, M.; Herbst, E.; Ren, X.; Fox, D. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *Int. J. Robot. Res.* **2012**, *31*, 647–663. [CrossRef]
17. Alexiadis, D.S.; Kordelas, G.; Apostolakis, K.C.; Agapito, J.D.; Vegas, J.; Izquierdo, E.; Daras, P. Reconstruction for 3D immersive virtual environments. In Proceedings of the 13th International Workshop on Image Analysis for Multimedia Interactive Services, Dublin, Ireland, 23–25 May 2012; pp. 1–4.
18. Choi, S.; Zhou, Q.-Y.; Koltun, V. Robust reconstruction of indoor scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5556–5565.
19. Wang, J.; Zhang, C.; Zhu, W.; Zhang, Z.; Xiong, Z.; Chou, P.A. 3D scene reconstruction by multiple structured-light based commodity depth cameras. In Proceedings of the 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Kyoto, Japan, 25–30 March 2012; pp. 5429–5432.
20. Okorn, B.; Xiong, X.; Akinci, B.; Huber, D. Toward automated modeling of floor plans. In Proceedings of the symposium on 3D data processing, visualization and transmission, Thessaloniki, Greece, 9 September 2004.
21. Wang, C.; Cho, Y.K.; Kim, C.J.A.i.C. Automatic BIM component extraction from point clouds of existing buildings for sustainability applications. *Autom. Constr.* **2015**, *56*, 1–13. [CrossRef]
22. Previtali, M.; Barazzetti, L.; Brumana, R.; Scaioni, M. Towards automatic indoor reconstruction of cluttered building rooms from point clouds. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2014**, *2*, 281–288. [CrossRef]
23. Valero, E.; Adán, A.; Cerrada, C. Automatic method for building indoor boundary models from dense point clouds collected by laser scanners. *Sensors* **2012**, *12*, 16099–16115. [CrossRef]
24. Ikehata, S.; Yang, H.; Furukawa, Y. Structured indoor modeling. In Proceedings of the IEEE International Conference on Computer Vision, Honolulu, HI, USA, 21–26 July 2017; pp. 1323–1331.
25. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5828–5839.
26. Izadinia, H.; Shan, Q.; Seitz, S.M. Im2cad. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5134–5143.

27. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
28. Darwish, W.; Tang, S.; Li, W.; Chen, W.J.S. A new calibration method for commercial RGB-D sensors. *Sensors* **2017**, *17*, 1204. [CrossRef]
29. Gupta, S.; Girshick, R.; Arbeláez, P.; Malik, J. Learning rich features from RGB-D images for object detection and segmentation. In Proceedings of the 13th European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 345–360.
30. Kerl, C.; Sturm, J.; Cremers, D. Dense visual SLAM for RGB-D cameras. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 2100–2106.
31. 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]
32. Rusu, R.B. Semantic 3d object maps for everyday manipulation in human living environments. *KI-Künstliche Intell.* **2010**, *24*, 345–348. [CrossRef]
33. Silberman, N.; Hoiem, D.; Kohli, P.; Fergus, R. Indoor segmentation and support inference from rgbd images. In Proceedings of the 12th European Conference on Computer Version, Florence, Italy, 7–13 Ocotober 2012; pp. 746–760.
34. Xiao, J.; Owens, A.; Torralba, A. Sun3d: A database of big spaces reconstructed using sfm and object labels. In Proceedings of the IEEE International Conference on Computer Vision, Sydney, Australia, 3 January 2016; pp. 1625–1632.
35. Song, S.; Lichtenberg, S.P.; Xiao, J. Sun rgb-d: A rgb-d scene understanding benchmark suite. In Proceedings of the IEEE conference on computer vision and pattern recognition, Boston, MA, USA, 7–12 June 2015; pp. 567–576.
36. Levin, A.; Lischinski, D.; Weiss, Y. Colorization using optimization. *ACM Trans. Graph. (tog)* **2004**, *23*, 689–694. [CrossRef]
37. Occipital. Structure Sensor. Available online: https://structure.io/structure-sensor (accessed on 5 December 2019).
38. Wang, P.; Chen, P.; Yuan, Y.; Liu, D.; Huang, Z.; Hou, X.; Cottrell, G. Understanding convolution for semantic segmentation. In Proceedings of the 2018 IEEE winter conference on applications of computer vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 1451–1460.
39. Li, Y.; Qi, H.; Dai, J.; Ji, X.; Wei, Y. Fully convolutional instance-aware semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2359–2367.
40. Zheng, S.; Jayasumana, S.; Romera-Paredes, B.; Vineet, V.; Su, Z.; Du, D.; Huang, C.; Torr, P.H. Conditional random fields as recurrent neural networks. In Proceedings of the IEEE international conference on computer vision, Santiago, Chile, 13–16 December 2015; pp. 1529–1537.
41. Maninis, K.-K.; Pont-Tuset, J.; Arbeláez, P.; Van Gool, L. Convolutional oriented boundaries: From image segmentation to high-level tasks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 819–833. [CrossRef] [PubMed]
42. Endres, F.; Hess, J.; Sturm, J.; Cremers, D.; Burgard, W. 3-D mapping with an RGB-D camera. *IEEE Trans. Robot.* **2014**, *30*, 177–187. [CrossRef]