sensors

Article

Robust Spacecraft Component Detection in Point Clouds

Quanmao Wei [1,2], Zhiguo Jiang [1,2] and Haopeng Zhang [1,2,*]

1 Image Processing Center, School of Astronautics, Beihang University, Beijing 100191, China; weiqm@buaa.edu.cn (Q.W.); jiangzg@buaa.edu.cn (Z.J.)
2 Beijing Key Laboratory of Digital Media, Beijing 100191, China
* Correspondence: zhanghaopeng@buaa.edu.cn; Tel.: +86-10-8233-8061

Received: 10 January 2018; Accepted: 13 March 2018; Published: 21 March 2018

Abstract: Automatic component detection of spacecraft can assist in on-orbit operation and space situational awareness. Spacecraft are generally composed of solar panels and cuboidal or cylindrical modules. These components can be simply represented by geometric primitives like plane, cuboid and cylinder. Based on this prior, we propose a robust automatic detection scheme to automatically detect such basic components of spacecraft in three-dimensional (3D) point clouds. In the proposed scheme, cylinders are first detected in the iteration of the energy-based geometric model fitting and cylinder parameter estimation. Then, planes are detected by Hough transform and further described as bounded patches with their minimum bounding rectangles. Finally, the cuboids are detected with pair-wise geometry relations from the detected patches. After successive detection of cylinders, planar patches and cuboids, a mid-level geometry representation of the spacecraft can be delivered. We tested the proposed component detection scheme on spacecraft 3D point clouds synthesized by computer-aided design (CAD) models and those recovered by image-based reconstruction, respectively. Experimental results illustrate that the proposed scheme can detect the basic geometric components effectively and has fine robustness against noise and point distribution density.

Keywords: geometric primitive; component detection; spacecraft; 3D point clouds

1. Introduction

Automatic tracking of space objects, recognizing them and estimating their relative poses are the main tasks in space exploitation and space-based space surveillance [1–5]. Recent advances in three-dimensional (3D) data acquisition have resulted in a broad availability of 3D data [6]. The 3D models of space objects have obvious benefits in space applications such as pose estimation, autonomous rendezvous and docking, on-orbit servicing and active debris removal [7–10], especially for non-cooperative spacecraft without cooperative markers. Comparing to 2D images, the 3D data are free of perspective projection and can reveal the structural information of objects in 3D space, such as shape, dimension, position and orientation. Unstructured point cloud data are a popular kind of 3D data, which can be generated by directly scanning with 3D sensors or image-based reconstruction techniques such as simultaneous localization and mapping (SLAM) [11,12] and structure from motion (SFM) [13,14]. However, to use the point cloud data for higher level tasks, such as on-orbit operation and situational awareness, further processing is still required in order to extract meaningful abstract information of the recorded object.

Recognition of patterns in an image is a rather easy task for a human; however, it is really hard for a computer. Such a fact facilitates studies in the fields of image processing and computer vision. The same case occurs in understanding 3D point clouds. For most of the 3D data processing and analyzing approaches, detection of geometric primitives is a crucial procedure [15–17]. For example, to generate 3D models of city buildings with LiDAR data, plane detection is generally employed for rooftop segmentation [18,19]. While to model installations in industrial sites where pipes are frequently encountered, cylinders are usually detected [20–22]. Hough transform (HT) [23], random sample consensus (RANSAC) [24,25] and surface growing [19] are frequently used for 3D plane and cylinder detection in most of these approaches. Plane detection by HT is a natural extension of the 2D Hough transform for line detection in images. Due to directly parameterizing five freedoms for a cylinder could lead to large memory and computational consumptions in HT, cylinder detection is usually divided into detection of cylinder axes and estimation of radius and position [20]. Besides, model fitting is an enabling alternative, where the shapes to be detected are treated as undetermined models [26,27] and finally detected along with supports by model fitting. The work in [26] formulates geometric multi-model fitting as an optimal labeling problem, in which the global energy function can be minimized by an extension version [28] of the α-expansion algorithm [29]. Then, multi-models are fitted in the iteration of the proposed expanding and re-estimating labels (PEARL). Such an energy-based multi-model fitting method is successfully used for plane detection in [27].

Most of these 3D data processing frameworks are interested in the modeling of city buildings and industrial installations, while little effort is made for other targets, such as spacecraft. As for the 3D data of spacecraft, feature points and local descriptors or global descriptors are often used for applications such as pose estimation and tracking, while further structure analysis is rarely involved, which is however crucial for future on-orbit operation and situational awareness, especially for non-cooperative space targets. Spacecraft mainly consist of solar panels, cuboid or cylinder modules; thus, they can also be regarded as a combination of geometric primitives like planes, cuboids or cylinders. Based on this simplification, a structural representation with higher level abstractions can be generated via the detection of such geometric primitives. Moreover, most of the 3D data processing approaches deal with laser scanning data, metric information and controllable quality, which are usually available; while this paper focuses on 3D data of spacecraft recovered by reconstruction, since it has more potential than laser scanning for the sake of hardware complexity, size and power requirement. Generally, such recovered data are often a scaled reconstruction rather than a metric one, and these data might also have different densities and noise levels due to variant texture quality. Therefore, special attention must be paid to scale variation, distribution variation and severe noise and outliers.

To address the automatic component detection for spacecraft, a robust detection scheme is developed in this paper, where components, such as solar panels, cuboid and cylinder modules, are detected as geometric primitives:

(1) Cylinders are first detected by PEARL [26], where initial cylinder proposals are generated in RANSAC, and energy-based multi-cylinder fitting and parameter estimation via least square fitting are then iteratively executed to detect the desired cylinders.
(2) Planes are detected using Hough transform and finally represented as bounded patches with their minimum bounding rectangles (MBR).
(3) Cuboids are recognized with pair-wise geometry relations from the detected patches at last.

A concise and abstract mid-level geometry representation of the spacecraft can be finally delivered. The performance of the proposed scheme is tested with synthesized point cloud data of eight spacecraft and the more challenging reconstructed point cloud data of 10 spacecraft. This paper is a further improvement of previous work [30]. Thus, we use the results of [30] as a baseline for comparison. Results on synthesized and reconstructed data are both promising. The contributions of this work are two-fold.

(1) The cuboid is a common geometry primitive, which can however not be parameterized directly. Detection and recognition of cuboids are still open problems. In this paper, we propose a new method to recognize cuboids from rectangle patches using geometry relation criteria to infer opposite and adjacent cuboid faces. Moreover, robust estimation methods for cuboid orientation and dimension are also proposed.

(2) We propose an entire automatic spacecraft component detection scheme, which can provide concise and abstract geometric representations of the space objects from unstructured 3D point cloud data. To make the scheme robust, we use some special procedures and improvements, such as the use of adaptive dimensional unit ε, utilization of PEARL for robust cylinder detection and improvement on MBR estimation.

The rest of this paper is organized as follows. Section 2 gives a detailed description of the proposed component detection scheme. Experimental results are presented in Section 3. Section 4 provides the conclusion.

2. The Proposed Scheme

To automatically detect components of spacecraft in 3D point clouds, this paper develops a robust scheme, where cylinders, planar patches and cuboids can be successively detected. The procedure of the proposed scheme is shown in Figure 1. The scheme consists of four modules, namely preprocessing (Figure 1a), detection of cylinders (Figure 1b), planar patches (Figure 1c) and cuboids (Figure 1d). The preprocessing module is aimed at removing outliers and giving a relative dimensional unit ε, and the detection modules are designed to detect the corresponding geometric primitives.

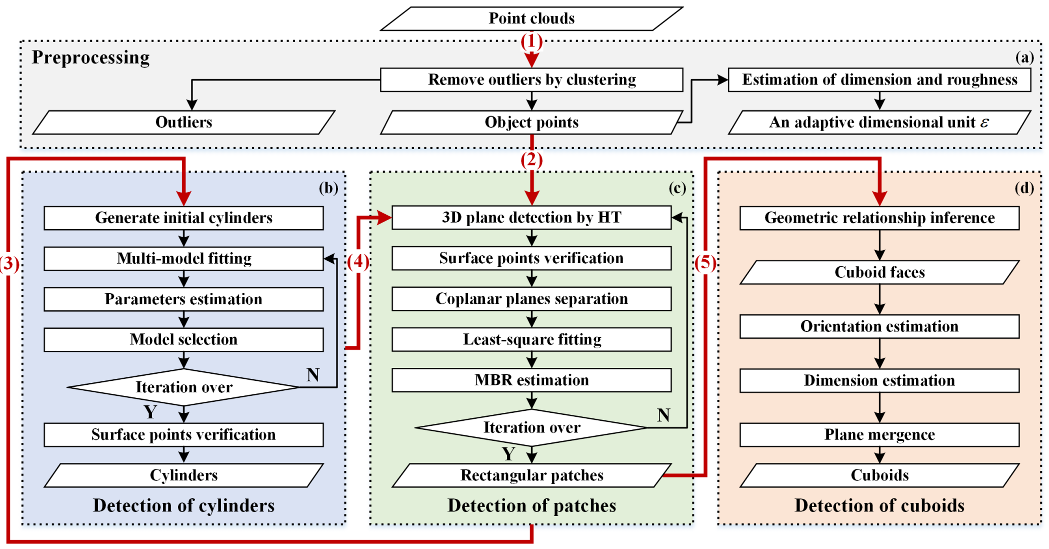

Figure 1. Procedure of the proposed spacecraft component detection scheme. The scheme consists of four modules: (**a**) preprocessing; (**b**) detection of cylinders; (**c**) detection of planar patches and (**d**) detection of cuboids. The scheme is implemented in the order from (1) to (5). Note that the flow lines starting with the borders of module blocks indicate that the remaining points are taken as input. HT, Hough transform; MBR, minimum bounding rectangles.

The scheme is implemented with five steps:

(1) First, point cloud preprocessing is conducted to remove outliers. Meanwhile, an adjective dimensional unit ε is also estimated, which is useful in the subsequent detection steps.
(2) Since local areas of the cylinder surface can be regarded as planes in approximation and a plane can be also treated as a local area of a cylinder with a large radius, the existence of plane points and cylinder points will influence the detection of each other. Thus, to avoid the influence of plane points on cylinder detections, prominent patches (i.e., the top 10 planes with the most points in our cases) are removed in advance before cylinder detection via the patches detection module. Note that the actual patch detection is performed after cylinder detection, i.e., the next Steps (3) and (4).

(3) Cylinders are detected by the cylinder detection module in the fashion of PEARL, where initial cylinder proposals are first generated in RANSAC, then energy-based multi-cylinder fitting and parameter estimation via least square fitting are iteratively executed to detect the desired cylinder primitives. The surface points of these cylinder primitives are verified at last. Note that plane detection results of Step (2) will be revoked after primitives detection and before point verification.

(4) Planar patches are detected by the patches detection module, where the Hough transform is mainly employed. MBRs of these patches are estimated for a further representation.

(5) Cuboids are recognized by the cuboid detection module from the patches results of Step (4). Geometric relation criteria are first proposed to infer opposite and adjacent cuboid faces, through which patches belonging to the same cuboid could be detected. Then, the cuboid orientation and dimension are robustly estimated. Plane mergence is performed at last to append the missed cuboid faces.

Details of all four modules are explained respectively in the following subsections.

2.1. Preprocessing

2.1.1. Removal of Outliers

Point cloud data generated by laser scanning or image-based reconstruction usually contain outliers that drift far away from the object. Statistical analysis with the k nearest neighbor (KNN) can effectively classify outliers that are randomly distributed. However, outliers could be also structured where outliers are in the form of high-density continuous clusters separating from the object points, especially for point cloud data obtained by image-based reconstruction, as shown in Figure 2. To remove such outliers, the input point clouds are clustered into groups by region growing in 3D space, and the largest group with the most points is selected as the point set of the object, while other groups are regarded as outliers. Results of outliers' removal are displayed in the second column of Figure 2b, where the removed outliers are colored in green.

2.1.2. Estimation of Dimensional Unit ε

To handle the scale variation in different point cloud data, an adaptive dimensional unit ε is estimated for usage in the subsequent detection procedures. Two reference lengths l_D and l_R are used to estimate ε. l_D is the minimum length of the edges of the minimum enclosing box of the 3D point data $\mathcal{P}$. l_R is a length that measures the surface roughness of $\mathcal{P}$. For each point $p \in \mathcal{P}$, a local plane π_p can be estimated by least square fitting with its k nearest neighbors $\mathcal{N}_k(p)$. l_R is then defined as:

$$l_R = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} a\mu_p + b\sigma_p, \tag{1}$$

where $|\mathcal{P}|$ is the size of $\mathcal{P}$, μ_p and σ_p are the mean and standard deviation of the distances from $\mathcal{N}_k(p)$ to π_p and a and b are two coefficients, which are set as $a = 1$ and $b = 3$ in practice. Then, the dimensional unit ε is defined as:

$$\varepsilon = \min(\beta l_D, \max(\alpha l_D, l_R)), 0 < \alpha < \beta < 1, \tag{2}$$

where α and β are two small factors with $\alpha = 1$ and $\beta = 0.03$ respectively in this paper. The definition of dimensional unit ε considers both point cloud dimensions and surface roughness. Dimension length l_D enables ε to handle the scale variation, while roughness length l_R makes ε robust to point position noise. Meanwhile, the upper and lower bounds determined by α and β achieve a trade-off between precision and computational consumption.

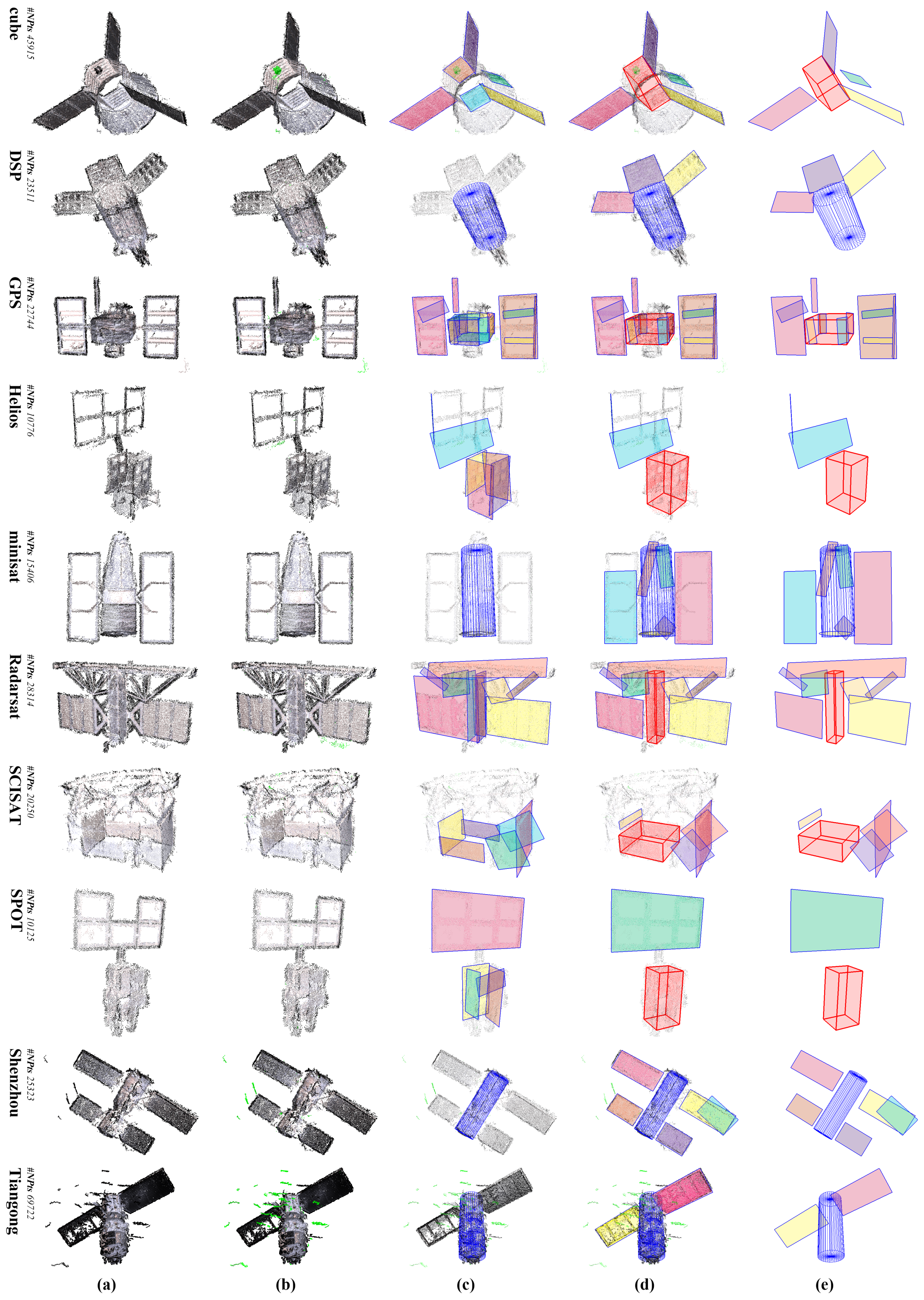

Figure 2. Detection results of the reconstructed point cloud data. From left to right: (**a**) the origin input point clouds; (**b**) results of outliers removal, where the identified outliers are colored in green; (**c**) results of cylinder detection or patch detection; the cylinders are rendered in blue, and patches are rendered in different colors; (**d**) final results; the detected cuboids are rendered as red boxes; (**e**) a clear view of the detected components. The number of points (#NPts) is noted above model names.

2.2. Detection of Cylinders

Multiple cylinder proposals are initially generated in a RANSAC paradigm, and the desired cylinder primitives are then detected in iterations of energy-based multi-model fitting [26] and cylinder parameter estimation [31]. Surface points of the detected cylinders are finally verified by distance proximity and orientation proximity, and points that do not belong to cylinders will be taken as the input of the subsequent path detection.

2.2.1. Generation of Initial Cylinders

To estimate a cylinder, only two points with the normal are needed. Given cylinder surface points p at $\mathbf{c}_p$ with normal $\mathbf{n}_p$ and points q at $\mathbf{c}_q$ with normal $\mathbf{n}_q$, the axis of the cylinder can be found as the common perpendicular line between $\mathbf{n}_p$ and $\mathbf{n}_q$ with direction $\mathbf{n}_p \times \mathbf{n}_q$. Meanwhile, the radius of the cylinder is estimated as the average distance of $\mathbf{c}_p$ and $\mathbf{c}_q$ to the axis. The initial cylinders are generated with random sampled point pairs by RANSAC. Notice that obviously improper cylinder estimations are discarded, such as the cylinder estimated in the cases:

- the distances from $\mathbf{c}_p$ and $\mathbf{c}_q$ to the axis are inconsistent;
- the center of the cylinder is out of the minimum enclosing box of the point clouds;
- the radius of the cylinder exceeds the minimum dimension of the minimum enclosing box;
- the inlier percentage is less than preset threshold T_{cyl}.

2.2.2. Energy-Based Multi-Cylinder Fitting

Given a point p in point cloud $\mathcal{P}$, $f_p \in \mathcal{L}$ is the label assigned to p, i.e., p is classified as a point of the cylinder associated with label f_p. Then, multiple cylinders are detected by minimizing energy $\mathrm{E}(\mathcal{L})$ of labeling $\mathcal{L} = \{f_p \mid p \in \mathcal{P}\}$ for all points as:

$$\mathrm{E}(\mathcal{L}) = \sum_{p \in \mathcal{P}} \mathrm{D}_p(f_p) + \sum_{(p,q) \in \mathcal{N}} \mathrm{V}_{pq}(f_p, f_q) + \eta |L_{\mathcal{L}}|. \tag{3}$$

Energy $\mathrm{E}(\mathcal{L})$ consists of three terms, i.e., data term $\mathrm{D}_p(f_p)$, smooth term $\mathrm{V}_{pq}(f_p, f_q)$ and label cost $\eta|L_{\mathbf{L}}|$. Their definitions are as follows:

- Data term $\mathrm{D}_p(f_p)$ is a geometric error energy, which measures the disagreement between p and the corresponding cylinder model $\mathrm{Cyl}(f_p)$. It is defined as the sum of distance proximity and orientation proximity, i.e.,

$$\mathrm{D}_p(f_p) = \|p - \mathrm{Cyl}(f_p)\| = \frac{\mathrm{D}_{dist}(f_p, \mathrm{Cyl}(f_p))}{\varepsilon} + \frac{\mathrm{D}_{ang}(f_p, \mathrm{Cyl}(f_p))}{d_\Delta}, \tag{4}$$

  where D_{dist} is the distance variance of p to the surface of $\mathrm{Cyl}(f_p)$, D_{ang} is the angle variance of the normal of p. D_{dist} and D_{ang} are separately discretized with resolution ε and d_Δ, and d_Δ is set to 15° in our experiments.
- Smooth term $\mathrm{V}_{pq}(f_p, f_q)$ is a discontinuity preserving smoothness error energy, which constrains the labeling consistency among the neighboring points. In our scheme, the neighborhood system $\mathcal{N}$ employs the KNN. $\mathrm{V}_{pq}(f_p, f_q)$ is described by the Potts model [29] as:

$$\mathrm{V}_{pq}(f_p, f_q) = \lambda \omega_{pq} \delta(f_p \neq f_q), \tag{5}$$

  where λ is a weight coefficient and $\delta(\cdot)$ is an indicator, which takes one if its argument is true, and otherwise zero. ω_{pq} is the coefficient to penalize the discontinuity of neighboring points, with the definition as:

$$\omega_{pq} = \exp\left(-\frac{(\|p - q\|/\varepsilon)^2}{\gamma^2}\right), \tag{6}$$

  where γ is a constant coefficient.

- Label cost $\eta|L_{\mathcal{L}}|$ is the label energy to prevent over-fitting, where $L_{\mathcal{L}}$ is the set of distinct labels assigned to data points by labeling $\mathcal{L}$, $|L_{\mathcal{L}}|$ is the cardinality of $L_{\mathcal{L}}$ and η is a coefficient.

To handle the outliers, a special outlier model is introduced with label notation $\emptyset$. Points assigned this label are considered as outliers, and the data term for $\emptyset$ is a constant $e_{\emptyset}$, i.e., $\|p - \emptyset\| = e_{\emptyset}$. Then, the energy function for multi-cylinder detection can be expressed as:

$$\mathrm{E}(\mathcal{L}) = \sum_{p \in \mathcal{P}} \|p - \mathrm{Cyl}(f_p)\| + \lambda \sum_{(p,q) \in \mathcal{N}} \omega_{pq} \delta(f_p \neq f_q) + \eta |L_{\mathcal{L}}|. \tag{7}$$

Minimization of this energy function can be effectively solved by the extended graph-based α-expansion method [28].

2.2.3. Iteration of Model Fitting and Estimation

The energy-based multi-model fitting will deliver a labeling result, where points in $\mathcal{P}$ are classified into different cylinders or outliers. The cylinders are then refined via least square fitting (LSF) [31] with their support (inliers). The error function of LSF is the total squared distance error of the input point data to the surface of the cylinder, and parameter estimation is performed by searching a large number of directions and finding the optimal radius and location. In the model selection step in Figure 1, cylinders with a low inlier percentage are discarded, and those close to each other are merged. We update the initial models of energy-based model fitting with these refined cylinders and iteratively perform the fitting and estimation steps to get a better set of cylinders. As the iteration could rapidly converge to a stable result within just a few times, the max iteration time is set to three in our experiments.

2.3. Detection of Planar Patches

2.3.1. Plane Detection

In our detection scheme, planes in 3D space are iteratively detected by the 3D Hough transform method. A 3D plane Π can be formulated as $s_x x + s_y y + s_z z + s_d = 0$ in Cartesian coordinates. To uniquely define the plane, one parameter among s_x, s_y and s_z is often fixed in advance, e.g., set s_x to one. In this case, planes in specific orientations will result in rather small (perpendicular to the X-axis) or large (parallel to the X-axis) values of s_y and/or s_z. Namely, both large range and small accuracy are required for s_y and s_z, which is obviously memory and time consuming. To handle this problem, our scheme uses three separate Hough transformations in one detection. For each Hough transformation, one parameter among s_x, s_y and s_z is fixed to 1, and the other two are discretized in the range $[-1, 1]$. The desired plane is finally determined by the most voted point among all three Hough transformations. To accelerate the computation, these three Hough transformations are performed in parallel. Meanwhile, the voting is constrained with the normals, i.e., only planes nearly perpendicular to the normal of the space point are voted for, and this also accelerates the computation greatly. In this paper, s_x, s_y and s_z are discretized in resolution 0.01, and s_d is discretized with the dimensional unit ε estimated in the preprocessing stage.

Given a detected plane, its surface points are then verified by distance proximity and orientation proximity. However, these points do not necessarily belong to the same plane. It may consist of multiple coplanar planes and the parts of the component surfaces that intersect this plane. To distinguish these points, a distance-proximity-based region-growing approach is utilized. The largest group with most points is kept, while the other points are put into the next iteration of plane detection.

2.3.2. MBR Extraction

Since the planes are actually bounded planar patches rather than infinitely extended ones, the representation of the detected plane primitive is further improved to a patch by finding its minimum bounding rectangle. Notice that the plane for minimum bounding rectangle calculation is refined by least square fitting, and all these surface points are projected to the fitted plane. There may be outlier points making wrong bounding rectangle, and such outliers usually locate at the bound of the wrong rectangle and could result in a significant increase of the area of the bounding rectangle. The outlier bounding points are detected by local density and removed during the MBR extraction. The specific details of the MBR extraction are explained in Algorithm 1.

Algorithm 1: Robust extraction of minimum bounding rectangle.

```
Data: Projected points $\mathcal{P}_\pi \in \mathbb{R}^3$ on plane $\pi$.
Result: Minimum bounding rectangle $MBRect(\mathbf{o}, \mathbf{u}, \mathbf{v})$.
begin
    $MBRect \leftarrow \emptyset$;
    $S_{min} \leftarrow \infty$;
    $(\mathbf{o}, \mathbf{u}_0, \mathbf{v}_0) \leftarrow$ an arbitrary orthogonal coordinates on $\pi$;
    for $\theta \leftarrow 0$ to 90 do
        $(\mathbf{o}, \mathbf{u}_\theta, \mathbf{v}_\theta) \leftarrow$ rotate $(\mathbf{o}, \mathbf{u}_0, \mathbf{v}_0)$ $\theta$ degrees;
        foreach $\mathbf{k} \in \{\mathbf{u}_\theta, \mathbf{v}_\theta\}$ do
            $\mathcal{P}_\mathbf{k} = \{p_i^\mathbf{k} \in \mathbb{R}\} \leftarrow$ project $\mathcal{P}_\pi$ to $\mathbf{k}$;
            Sort $\mathcal{P}_\mathbf{k}$ in ascending order by distance to origin $\mathbf{o}$;
            repeat
                /* $\tau$ is a threshold and set $\sqrt{2}$ in our cases.   */
                while $\frac{\tau}{|p_0^\mathbf{k} - p_1^\mathbf{k}|} < \frac{|\mathcal{P}_\mathbf{k}|}{|p_{end}^\mathbf{k} - p_0^\mathbf{k}|}$ do
                    Remove $p_0^\mathbf{k}$;
                while $\frac{\tau}{|p_{end}^\mathbf{k} - p_{end-1}^\mathbf{k}|} < \frac{|\mathcal{P}_\mathbf{k}|}{|p_{end}^\mathbf{k} - p_0^\mathbf{k}|}$ do
                    Remove $p_{end}^\mathbf{k}$;
            until $|\mathcal{P}_\mathbf{k}| \leq 2$ or no point is removed;
            $L^\mathbf{k} \leftarrow (p_0^\mathbf{k}, p_{end}^\mathbf{k})$;
        $S_\theta \leftarrow |L_2^{\mathbf{u}_\theta} - L_1^{\mathbf{u}_\theta}| \times |L_2^{\mathbf{v}_\theta} - L_1^{\mathbf{v}_\theta}|$;
        if $S_\theta < S_{min}$ then
            $MBRect \leftarrow Rect(\mathbf{o} + L_1^{\mathbf{u}_\theta}\mathbf{u}_\theta + L_1^{\mathbf{v}_\theta}\mathbf{v}_\theta, |L_2^{\mathbf{u}_\theta} - L_1^{\mathbf{u}_\theta}|\mathbf{u}_\theta, |L_2^{\mathbf{v}_\theta} - L_1^{\mathbf{v}_\theta}|\mathbf{v}_\theta)$;
            $S_{min} \leftarrow S_\theta$;
```

Multiple patches are detected by performing plane detection and MBR extraction iteratively, until no point remains or the inlier percentage of the current detected patch is lower than a preset threshold T_{ple}. Note that in our proposed scheme, the circle patch is considered as a special rectangle patch in plane detection and distinguished by calculating its length-to-width ratio and point distribution in its inscribed circle.

2.4. Detection of Cuboids

2.4.1. Geometry Relation Criteria

Faces of the cuboid main bodies, if they exist, may also be detected as patches in the plane detection stage. Therefore, a geometry relationship inferring approach is proposed to distinguish these patches. As for a cuboid, there are two kinds of pair-wise geometry relations among its six faces, the opposite faces and the adjacent faces. Our scheme exploits the following criteria to identify the relations.

- Criterion for two opposite faces: (i) their plane normals are parallel, and their edges are respectively parallel, as well; (ii) after projecting each patch to the other and calculating the ratio of intersection over union (IoU), the average IoU should be greater than a preset threshold T_{IoU}.
- Criterion for two adjacent faces: (i) their plane normals are perpendicular, and their edges are respectively parallel or perpendicular; (ii) some edge of one patch is adjacent to that of the other with similar lengths.

For criteria of both opposite and adjacent faces, Condition (i) constrains the orientation relations so that patch pairs with improper relative normal directions (Figure 3a) or edge directions (Figure 3b) can be filtered; while Condition (ii) constrains both the relative location and dimension so that patch pairs with a dramatic position deviation (Figure 3c) or unmatched dimensions (Figure 3d) can be filtered. It should be noted that the parallel and perpendicular relations are determined by a given maximal angle error T_θ in this paper.

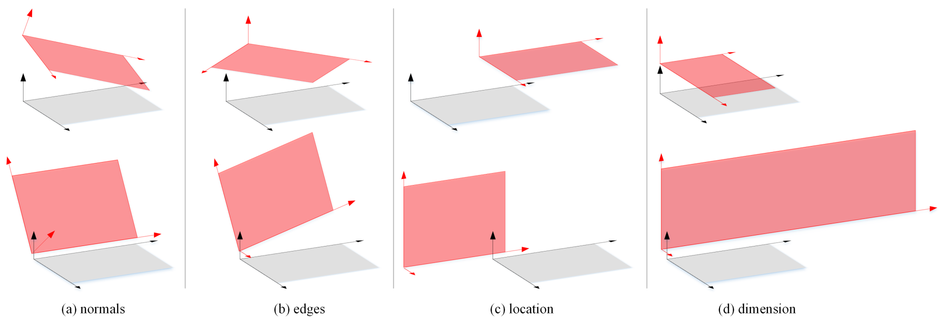

Figure 3. Criteria for opposite faces (the top row) and adjacent faces (the bottom row). The gray patch is a reference patch, and the red ones are patches that do not satisfy the (**a**) normal direction; (**b**) edge directions; (**c**) relative location or (**d**) dimension requirements.

Through these geometry relation criteria, we can generate multiple cuboid face patch groups, among which each pair of patches satisfies the opposite or adjacent criterion. Meanwhile, these groups must agree with the cuboid configure, i.e., one cuboid face patch could have only one opposite patch and four adjacent patches at most, and the adjacent patches should be located at different directions. Face patch groups containing more than three faces or at least one pair of opposite faces is used to estimate the cuboids.

2.4.2. Orientation Estimation

A face patch group of a cuboid C can be defined as $\mathcal{S}_c\{P_i\}$. A face patch P_i is defined as a combination of vectors, i.e., $P_i = \{\mathbf{o}_i, \mathbf{n}_i, \mathbf{u}_i, \mathbf{v}_i\}$, where $\mathbf{n}_i$ is the normalized normal, $\mathbf{o}_i$ is one reference vertex of P_i and $\mathbf{u}_i$ and $\mathbf{v}_i$ are the corresponding edges with lengths $|\mathbf{u}_i|$ and $|\mathbf{v}_i|$. All these elements are 3×1 vectors. The orientation of cuboid C is first estimated with $\mathcal{S}_c$. For each face direction $\mathbf{k} \in \{\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z\}$ of cuboid C, where $\mathbf{n}_x$, $\mathbf{n}_y$ and $\mathbf{n}_z$ are the 3×1 normals of three orthogonal faces of C, $\mathcal{S}_c$ can be divided into two groups, i.e., perpendicular patches $\mathcal{S}_\perp^{\mathbf{k}}\{P_i \mid P_i \in \mathcal{S}_c, P_i \perp \mathbf{k}\}$ and parallel patches $\mathcal{S}_\parallel^{\mathbf{k}}\{P_i \mid P_i \in \mathcal{S}_c, P_i \parallel \mathbf{k}\}$. Then, $\mathbf{k}$ can be estimated as:

$$\hat{\mathbf{k}} = \begin{cases} \dfrac{\sum\limits_{P_i \in \mathcal{S}_\perp^{\mathbf{k}}} \mathrm{W}(\mathbf{n}_i)\mathbf{n}_i}{\sum\limits_{P_i \in \mathcal{S}_\perp^{\mathbf{k}}} \mathrm{W}(\mathbf{n}_i)} & \mathcal{S}_\perp^{\mathbf{k}} \neq \emptyset, \\ \dfrac{\sum\limits_{P_i \in \mathcal{S}_\parallel^{\mathbf{k}}} \mathrm{W}(\mathbf{u}_i)\delta(\mathbf{k}\perp\mathbf{u}_i)\frac{\mathbf{u}_i}{|\mathbf{u}_i|} + \mathrm{W}(\mathbf{v}_i)\delta(\mathbf{k}\perp\mathbf{v}_i)\frac{\mathbf{v}_i}{|\mathbf{v}_i|}}{\sum\limits_{P_i \in \mathcal{S}_\parallel^{\mathbf{k}}} \mathrm{W}(\mathbf{u}_i)\delta(\mathbf{k}\perp\mathbf{u}_i) + \mathrm{W}(\mathbf{v}_i)\delta(\mathbf{k}\perp\mathbf{v}_i)} & \text{otherwise.} \end{cases} \quad (8)$$

In Equation (8), $\delta(\mathbf{p} \perp \mathbf{q})$ is an indicator, and it takes one if $\mathbf{p}$ and $\mathbf{q}$ are parallel, and otherwise zero. $\mathrm{W}(\mathbf{n}_i)$ is a coefficient defined as the reciprocal of fitting error of the plane. $\mathrm{W}(\mathbf{u}_i)$ and $\mathrm{W}(\mathbf{v}_i)$ are confidence coefficients that indicate the uniformity of the points distributing along $\mathbf{u}_i$ and $\mathbf{v}_i$. Given an edge $\mathbf{q} \in \{\mathbf{u}, \mathbf{v}\}$ of a patch P, points $\mathcal{P}_P$ belonging to P are first projected to $\mathbf{q}$, and then, the distribution histogram of the projected points can be computed within N_{Bin} ($N_{Bin} = 10$ in our experiments) bins. Thus, $\mathrm{W}(\mathbf{q})$ can be defined as:

$$\mathrm{W}(\mathbf{q}) = \frac{\sum_{i=1}^{N_{Bin}} -\frac{n_i}{|\mathcal{P}_P|} \ln \frac{n_i}{|\mathcal{P}_P|}}{\ln N_{Bin}}, \tag{9}$$

where n_i is the number of points falling in the i-th bin and $|\mathcal{P}_P|$ is the number of points in $\mathcal{P}_P$. The numerator represents the distribution entropy and can reach the maximum value (i.e., the denominator) when the histogram is uniformly distributed.

During orientation estimation of cuboid C, patch normals are preferred to be utilized as they are generally more accurate and credible than edges. In addition, to guarantee the orthogonality of the estimated face normals, the diagonal matrix of singular value decomposition (SVD) of the direction matrix $[\hat{\mathbf{n}}_x, \hat{\mathbf{n}}_y, \hat{\mathbf{n}}_z]$ is forced to be an identity.

2.4.3. Dimension Estimation

The dimensions of cuboid C are estimated after orientation estimation. For each direction $\mathbf{k} \in \{\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z\}$, vertexes of all the patches in $\mathcal{S}_c$ are projected to $\mathbf{k}$, and the projected points would distribute in two clusters, $\mathcal{P}^1_{\mathbf{k}}\{p_i \in \mathbb{R}\}$ and $\mathcal{P}^2_{\mathbf{k}}\{p_i \in \mathbb{R}\}$. $\mathcal{P}^1_{\mathbf{k}}$ represents that group that is closer to the origin $\mathbf{o}$, and $\mathcal{P}^2_{\mathbf{k}}$ represents the farther one. Therefore, the dimension along $\mathbf{k}$ is defined as $l_{\mathbf{k}} = abs(p^2_{\mathbf{k}} - p^1_{\mathbf{k}})$, and $p^n_{\mathbf{k}}$ is the clustering center of $\mathcal{P}^n_{\mathbf{k}}$, which can be computed as:

$$p^n_{\mathbf{k}} = \begin{cases} \mathrm{mean}(\mathcal{P}^n_{\perp\mathbf{k}}) & \mathcal{P}^n_{\perp\mathbf{k}} \neq \emptyset, \\ \mathrm{median}(\mathcal{P}^n_{\mathbf{k}}) & \text{otherwise.} \end{cases}, n = 1, 2. \tag{10}$$

$\mathcal{P}^n_{\perp\mathbf{k}} \subset \mathcal{P}^n_{\mathbf{k}}$ consists of project points of patch $P^n_{\perp\mathbf{k}} \in \mathcal{S}^{\mathbf{k}}_{\perp}$. The cuboid is finally represented as $C = (\mathbf{o} + p^1_x\mathbf{n}_x + p^1_y\mathbf{n}_y + p^1_z\mathbf{n}_z, l_x\mathbf{n}_x, l_y\mathbf{n}_y, l_z\mathbf{n}_z)$.

2.4.4. Plane Mergence

The cuboid face patches may be incorrectly detected as a larger patch along with other patches. For example, the front and back faces of the cuboid main body in spacecraft Radarsat are detected as bigger patches along with the points of its antenna, as shown in Figure 4c. To correctly pick out the cuboid face patches, patches that are close to faces of cuboid C will be partly merged into C, and the MBRs of the remaining parts of such patches will be updated.

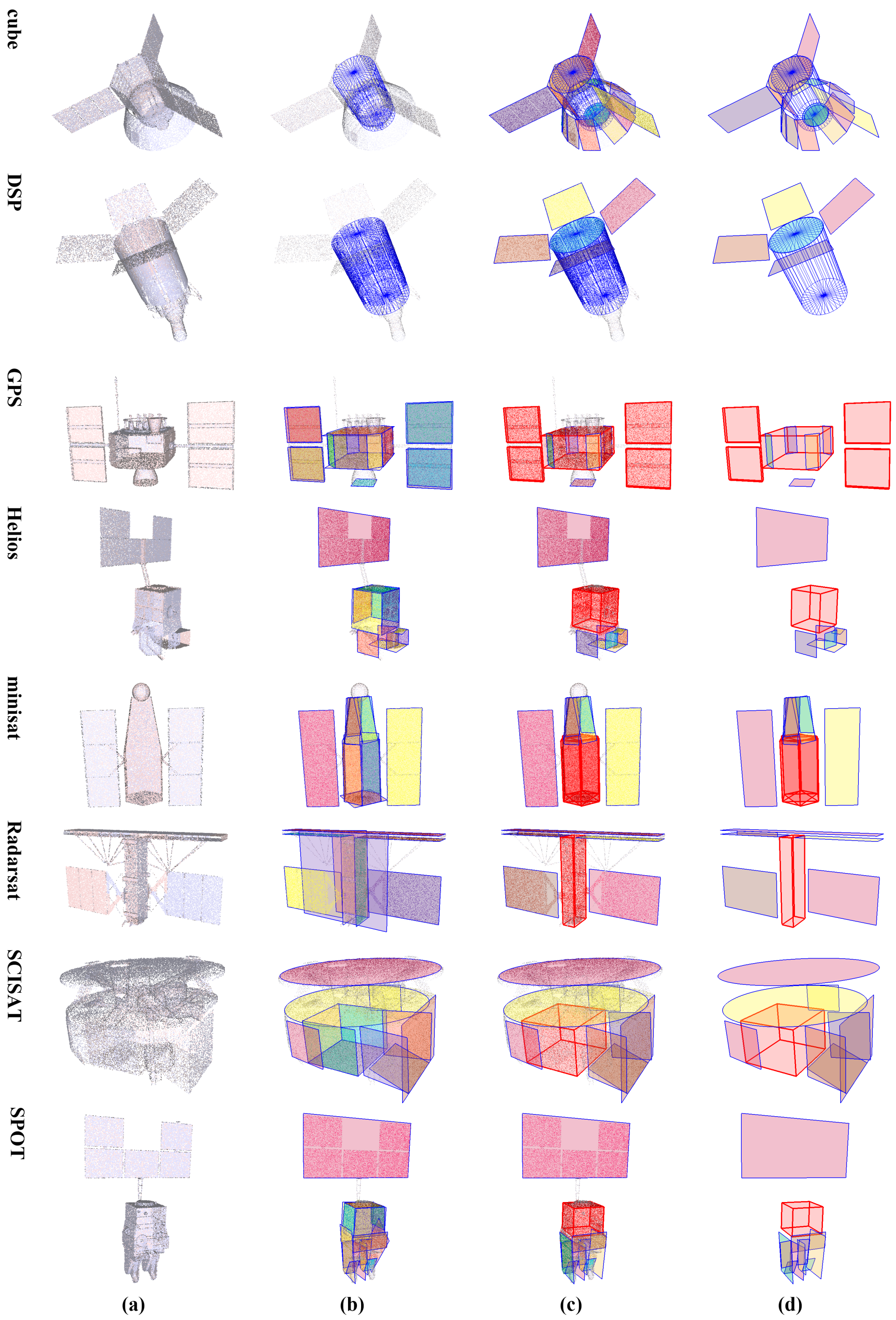

Figure 4. Detection results of the synthesized point cloud data *50K_00U_00D*. From left to right: (**a**) the origin input point clouds; (**b**) results of cylinder detection or patch detection after detection of cylinders (the cylinders are rendered in blue, and patches are rendered in different colors.); (**c**) final results (the detected cuboids are rendered as red boxes.); (**d**) a clear view of the detected components.

3. Experiments

3.1. Data Collection and Parameter Configure

Both synthesized spacecraft point cloud data and those recovered by image-based reconstruction are tested in our experiments.

- Synthesized point clouds: The synthesized point clouds are generated by uniformly sampling the surfaces of 8 spacecraft computer-aided design (CAD) models in the BUAA space object image dataset (BUAA-SID) [4]. Origin CAD mesh models of the 8 spacecraft are shown in Figure 5. To test our scheme and evaluate its robustness, synthesized data with different point distribution densities, position noise and direction noise are generated. To add position noise, a deviation of $k_u n$ along a random direction is added to every point. n is a standard Gaussian noise, and k_u is a distance deviation coefficient, which takes $1 \times (0.01 \times l_D)$, $2 \times (0.01 \times l_D)$ or $4 \times (0.01 \times l_D)$ (l_D is the dimension length described in Section 2.1). To add direction noise, a rotation of a random direction with angle $k_d n$ is applied to the normal of every point. k_d is a direction deviation coefficient, which takes $5°$, $10°$ or $15°$. We use {01U, 02U, 04U} and {05D, 10D, 15D} to denote different levels of position and direction noise, respectively.

  Synthesized point cloud of each spacecraft containing 50Knoise-free points are used for basic testing. For robustness analysis on density, point clouds with 20K, 10K and 05K noise-free samples are used. For robustness analysis on position noise, point clouds with 50K samples and distance noise levels of 01U, 02U and 04U are used. For robustness analysis on direction noise, point clouds with 50K samples and direction noise levels of 05D, 10D and 15D are used. In addition, point clouds with 20K, 10K and 05K samples, distance noise level of 04U and direction noise level of 15D are tested for integrated analysis. For clarity, each synthesized point cloud is referred to as ***K_**U_**D*, where ***K*, ***U* and ***D*denote the sample number, distance noise level and direction noise level, respectively. For example, a point cloud with 50K samples, distance noise level of 04U and direction noise level of 15D is referred to as *50K_04U_15D*.
- Reconstructed point clouds: the reconstructed point cloud data are recovered from simulated images sequences of the 8 spacecraft models via the image-based reconstruction method [14]. Besides, reconstruction results of real scaled models of spacecraft Shenzhou and Tiangong [14] are also included reconstructed point cloud data. Reconstructed point clouds of these 10 models are shown in the first column in Figure 2.

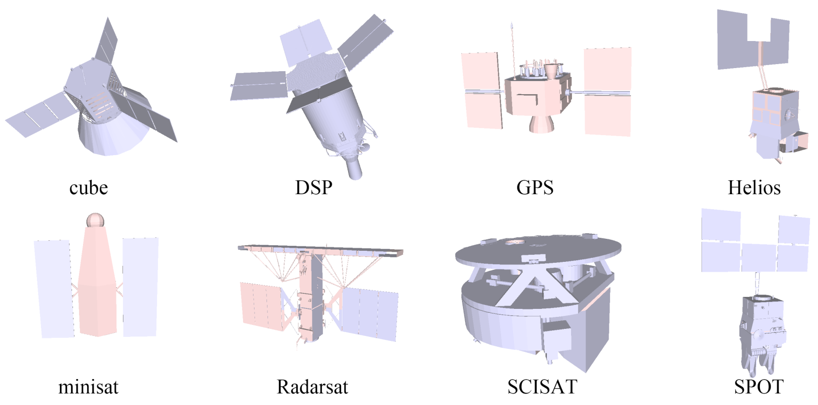

Figure 5. Origin CAD mesh models of 8 spacecraft. Synthesized point clouds are generated by sampling surfaces of these models.

Several preset thresholds are used in our scheme, including inlier rate threshold for patch detection T_{ple} and cylinder detection T_{cyl}, thresholds used for cuboid detection in geometry relation criteria T_{IoU}

and T_θ, distance proximity threshold T_{dist} and orientation proximity threshold T_{ang} used for surface point verification of both planes and cylinders. Coefficients used in the energy function for cylinder fitting are adaptively set as $e_\varnothing = \lambda = \gamma = 2 \times T_{dist}$, $\eta = T_{cyl} \times |\mathcal{P}|$. Definitions of these coefficients can be found in Section 2.2. Other fixed coefficients not mentioned here are given where they appear in Section 2. Since these parameters are robust, we used the same threshold configures for most experiments as listed in Table 1. An exception T_{ang} takes 15° for synthesized data ***K_**U_00D* and ***K_**U_05D* and 30° to handle heavier direction noise for ***K_**U_15D* and the reconstructed data.

Table 1. Thresholds used in our detection scheme. IoU, intersection over union.

Notation	Definition	Value
T_{ple}	Inlier rate threshold for planes in patch detection.	0.05
T_{cyl}	Inlier rate threshold for cylinders in cylinder detection.	0.1
T_{IoU}	IoU threshold for opposite faces in patch-wise geometry relation criteria.	0.6
T_θ	Angle deviation for parallel and perpendicular patches.	15°
T_{dist}	Distance proximity threshold for surface point verification.	$2 \times \varepsilon$
T_{ang}	Orientation proximity threshold for surface point verification.	$15^\circ \sim 30^\circ$

3.2. Results on Synthesized Point Clouds

Spacecraft component detection results of synthesized point cloud data *50K_00U_00D* are shown in Figure 4. It can be seen that the major components of these spacecraft are precisely detected by our scheme. Since arbitrary prisms (a cuboid is a special quadrangular prism) and cones are currently not covered in our scheme, they will be detected as other proximate geometric primitives. The hexagonal prism in spacecraft cube is detected as a cylinder, while the hexagonal prism in spacecraft minisat is detected as three cuboids, which are supported by only a pair of opposite faces. Besides, the cone in spacecraft cube is detected as patches. Such results may not be accurate, but still quite appropriate. Moreover, although the front and back faces of the cuboid main body in spacecraft Radarsat are detected as bigger patches along with some points of its antenna, these cuboid faces are effectively picked out in the patch mergence step, and the overall result is quite fine. The effectiveness of our detection scheme is thus demonstrated.

3.3. Robustness Analysis

To further evaluate the robustness of our scheme, we perform experiments on synthesized point cloud data with different factors, including point distribution density, position noise and direction noise. Note that the position noise can be directly seen from the point clouds, while the direction noise can be observed only when normals are rendered, as shown in Figure 6.

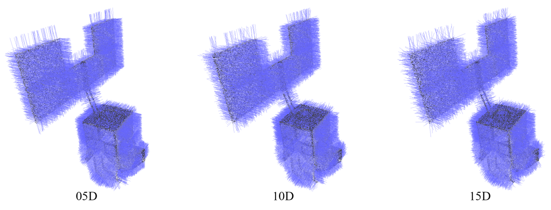

Figure 6. Point clouds with normals rendered. Point clouds of spacecraft Helios with different direction noise levels.

Results of robustness analysis with a single factor are illustrated in Figure 7. It should be noticed that only selected results of spacecraft DSP and Helios are displayed in Figure 7, and the results of all 8 spacecraft with intermediate detection results can be found in our Supplementary Material.

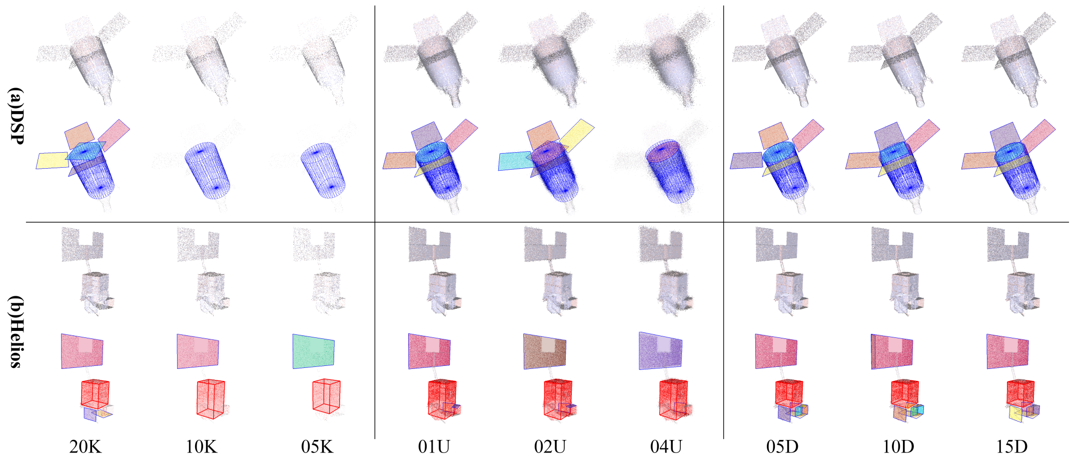

Figure 7. Results of robustness analysis with a single factor. The first grid row is the detection results of spacecraft (**a**) DSP, and the second grid row is the results of (**b**) Helios. From left to right, the grid columns are point clouds with different sample numbers (***K_00U_00D*), different distance noises (*50K_**U_00D*) and different direction noises (*50K_00U_**D*). The top row in each grid displays the origin input point clouds, and the bottom row displays the final results. The cylinders are rendered in blue; patches are rendered in different colors; and cuboids are rendered as red boxes.

The first grid column in Figure 7 demonstrates the result of point cloud data with different sample numbers. The inlier rate would get lower as the density gets sparser, thus leading to missed detection of small components; for example, the solar panels of spacecraft DSP and below the structure of Helios in point clouds *10K_00U_00D*, *05K_00U_00D* and *50K_04U_00D*. Nevertheless, the major components are always effectively detected. The second and third grid columns in Figure 7 are the results of points cloud data with different distance noise and direction noise, where only solar panels of spacecraft DSP in point clouds *50K_04U_00D* are missed. In general, the proposed scheme copes with the density variation and has fine robustness to noise.

An integrated analysis with multiple factors is further conducted, in which the proposed scheme is tested on point clouds with a noise level of 4U and 15D (***K_04U_15D*). The results in Figure 8 show that the major components are effectively detected and the overall results are still quite fine. As stated above, our proposed scheme can robustly achieve spacecraft component detection.

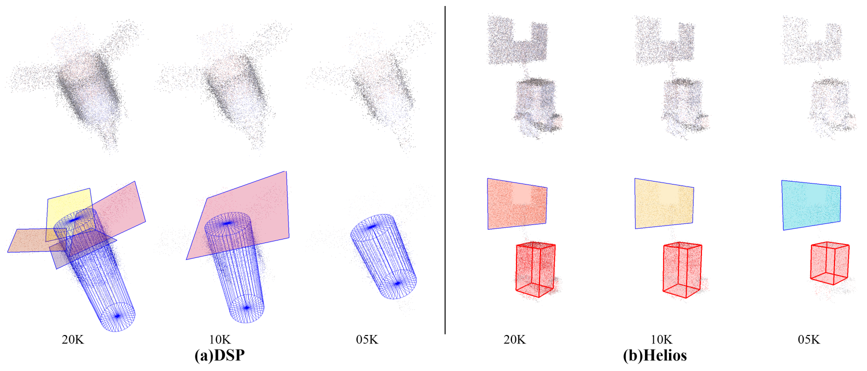

Figure 8. Results of robustness analysis with integrated factors. The left grid column is the detection results of spacecraft (**a**) DSP, and the right grid column is the results of (**b**) Helios. The top row in each grid displays the origin input point clouds, and the bottom row displays the final results. From left to right in each grid are point clouds with different sample numbers (***K_04U_15D*). The cylinders are rendered in blue; cuboids are rendered as red boxes; and patches are rendered in different colors.

3.4. Results on Reconstructed Point Clouds

Reconstructed point clouds are more challenging, because: (a) these point clouds contain severe noise and numerous outliers; (b) many parts of the surface failed to be recovered due to their lack of textures; and (c) the points are not uniformly distributed as synthesized data either. However, as shown in Figure 2, detection results of these reconstructed point clouds are gratifying. Components in the spacecraft models are effectively detected. Two faces of hexagonal prisms in a cube are detected, from which a cuboid is generated. The hexagonal prism in minisat is detected as a whole cylinder. Such situations are quite acceptable since we focus on detecting geometric primitives like plane, cuboid and cylinder. In addition, similar to Figure 4, the front face of the main body in Radarsat is again detected as a whole patch along with its left solar panel, but it is correctly handled by patch mergence.

Figure 9 shows the plane detection results of [25] and our scheme on the reconstructed point cloud data. In [25], the 3D points are stored as an octree, and planes are detected by RANSAC. Note that [25] focuses only on the detection of planes; outliers are not specially concerned, and no further bounded patch model is generated either. As shown in Figure 9, the result points of [25] (the left ones) are rendered with different colors according to their verification results. For convenient comparison with [25], results of our scheme just after the patch detection step are also given in Figure 9 (the right ones), which are rendered with the same pattern as the results of [25]. We can see that the noise of these point cloud data causes many undesired planes in the results of [25], such as the solar panel areas of spacecraft GPS, Helios and minisat. Moreover, the division nature of octree in [25] could also lead to additional planes; for example, the blue plane of metric SCISAT as emphasized in Figure 9; while the planes generated by our scheme are more compact and clean. The reason mainly lies in the employment of adaptive dimensional unit ε, which makes our scheme robust to both scale variation and noise.

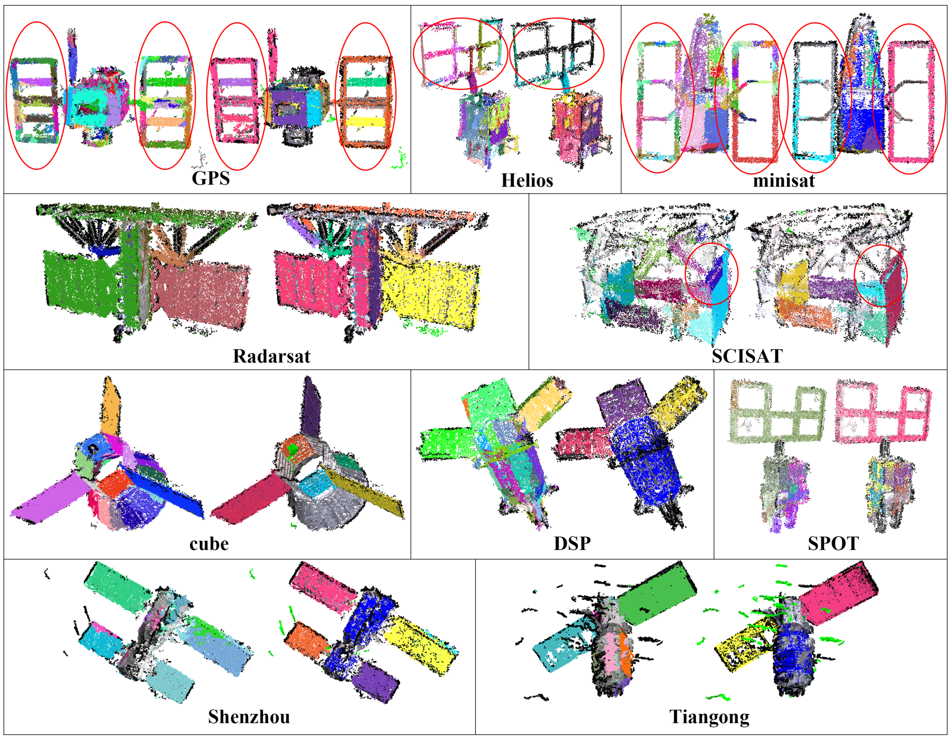

Figure 9. Comparison of plane detection results of [25] (the left ones) and our scheme (the right ones). In [25], no further bounded patch model is generated, and points are rendered in different colors to indicate different planes. For convenience of comparison with [25], the results of ours are rendered in the same pattern. The main differences are marked with red circles.

Figure 10 gives the comparison of the component detection results of [30] (the left ones) and our scheme (the right ones). Since this paper is a further improvement of [30], the approach in [30] is similar to that of this paper, except that: (a) the cylinder is detected with a novel axial-symmetry-based cylinder detection method lastly, rather than at beginning; (b) the MBR estimation approach is not improved; and (c) the patch mergence step is not introduced either. Through the comparison of Figure 10, we can see that: (a) the cylinders are more precisely detected due to utilization of more robust cylinder detection, for instance the cylinder main body of DSP; (b) contours of the detected patches are also better fitted due to improvement on MBR estimation, for instance the solar panels of GPS and Tiangong; and (c) the cuboids are better extracted due to the patches mergence steps, for instance the cuboid main body of Radarsat.

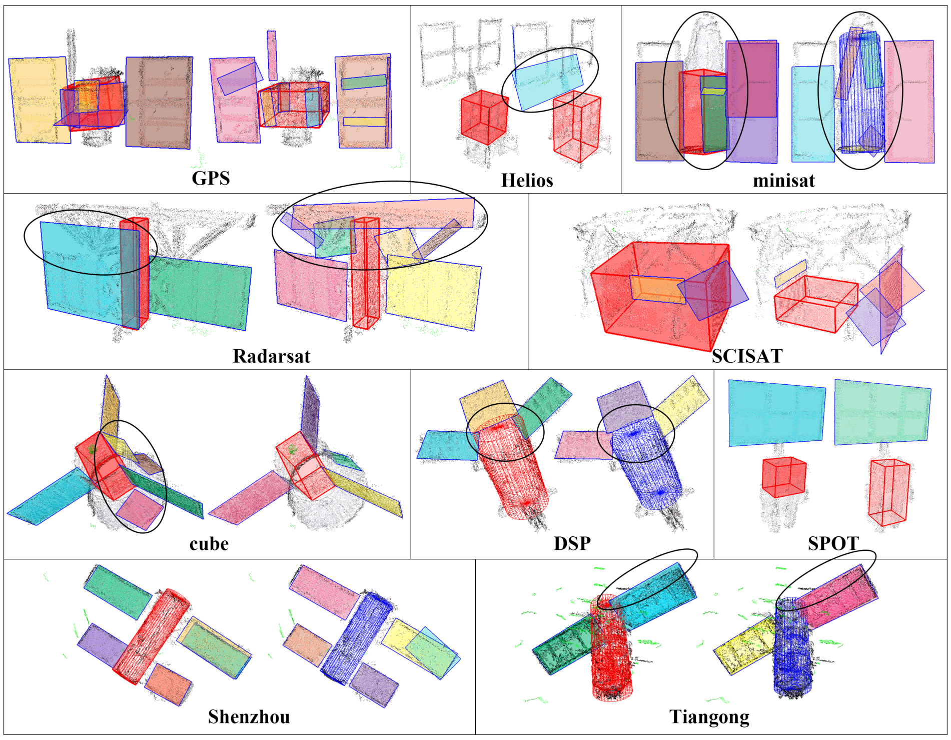

Figure 10. Comparison of component detection results of [30] (the left ones) and our scheme (the right ones). The identified outliers are colored in green; patches are rendered in different colors; the detected cuboids are rendered as red boxes; and the detected cylinders are rendered as red or blue wire-frame.

To quantitatively evaluate the accuracy of the generated models, the evaluation in [32,33] is utilized, which was originally designed for accuracy evaluation of multi-view reconstruction. Let the reconstructed point clouds be the reference point clouds $\mathcal{R}$; we first convert the generated geometric model into point clouds $\mathcal{M}$ by uniform sampling, where the sample number is equal to the point number of $\mathcal{R}$. Then, the precision measurement $P(d)$ for distance threshold d is defined as:

$$P(d) = \frac{100}{|\mathcal{M}|} \sum_{\mathbf{m} \in \mathcal{M}} [e_{\mathrm{m} \to \mathcal{R}} < d] \tag{11}$$

where $e_{\mathrm{m} \to \mathcal{R}} = \min_{\mathrm{r} \in \mathcal{R}} |\mathrm{m} - \mathrm{r}|$ is the distance of point $\mathbf{m} \in \mathcal{M}$ to R, and $[\cdot]$ is the Iverson bracket. Similarly, the completeness measurement $C(d)$ for distance threshold d is defined as:

$$C(d) = \frac{100}{|\mathcal{R}|} \sum_{\mathbf{r} \in \mathcal{R}} [e_{\mathrm{r} \to \mathcal{M}} < d] \tag{12}$$

where $e_{\mathrm{r}\to\mathcal{M}} = \min_{\mathrm{m}\in\mathcal{M}} |\mathrm{r}-\mathrm{m}|$ is the distance of point $\mathbf{r} \in \mathcal{R}$ to $\mathcal{M}$. In addition, the F-score $F(d)$ is computed to combine the precision and completeness measurements, which is defined as the harmonic mean of $P(d)$ and $C(d)$, i.e.,

$$F(d) = \frac{2P(d)C(d)}{P(d)+C(d)}. \tag{13}$$

In this paper, we use the measurements $P(d)$, $C(d)$ and $F(d)$ to evaluate the accuracy of the generated models, and the distance threshold d is set to $d = 2\varepsilon$. $C(d)$ is determined by the precision (location, orientation and dimension) of the generated model and detection capability. Poor and unsuccessful detections would lead to large $e_{\mathrm{r}\to\mathcal{R}}$ and low $C(d)$; while $P(d)$ is mainly determined by the detection precision and influenced by the completeness of the input point cloud. Missed parts of the input point cloud, such as hollows and sparse areas, would lead to large $e_{\mathrm{m}\to\mathcal{R}}$, thus resulting in low $P(d)$.

The evaluation of detection results of both [30] and ours is listed in Table 2. Our scheme outperforms [30] on both measurements $P(d)$ and $C(d)$ for most spacecrafts, except cube, Helios, minisat and Radarsat. As for spacecraft minisat, the hexagonal prism is detected as planes and cuboid by [30], while our scheme treats it as planes and cylinder. Since cuboid fits the hexagonal prism better than cylinder, method in [30] obtains higher scores on both $C(d)$ and $F(d)$. As for Helios and Radarsat, additional planes, where the point clouds are sparse, are only detected by our scheme, and it causes the lower $P(d)$. The additional detected planes are however valid and contribute to higher completeness of measurements $C(d)$ and $F(d)$, which means that our detection results are more complete. Similarly, method in [30] detects two more planes on cube than our scheme and thus achieves a higher $C(d)$. However, these two planes are not sparse; therefore, the precision measurement $P(d)$ of [30] on cube is not lower than ours.

In summary, our proposed scheme outperforms [30] on both regularization and accuracy in most cases, which demonstrates the effectiveness and accuracy of our scheme.

Table 2. Accuracy evaluation of the generated geometric models.

Objects	**Method in [30]**			**Our Scheme**		
	$P(d=2\varepsilon)$	$C(d=2\varepsilon)$	$F(d=2\varepsilon)$	$P(d=2\varepsilon)$	$C(d=2\varepsilon)$	$F(d=2\varepsilon)$
cube	**72.0**	**71.5**	**71.7**	70.5	64.2	67.2
DSP	80.9	73.6	77.1	**89.8**	**77.0**	**82.9**
GPS	46.3	50.1	48.2	**64.4**	**71.6**	**67.8**
Helios	**61.8**	36.5	45.9	48.7	**50.7**	**49.7**
minisat	**33.6**	**48.1**	**39.6**	28.6	39.0	33.0
Radarsat	**80.0**	60.9	69.1	77.3	**71.3**	**74.2**
SCISAT	27.8	42.3	33.6	**51.0**	**42.6**	**46.4**
SPOT	37.1	37.9	37.5	**39.5**	**41.5**	**40.5**
Shenzhou	81.3	80.3	80.8	**87.3**	**85.7**	**86.5**
Tiangong	86.0	84.2	85.1	**95.4**	**86.5**	**90.7**

4. Conclusions

In this paper, we have proposed a novel component detection scheme to automatically detect the basic components of spacecraft. The basic components, including solar panels, cuboid and cylinder modules, are treated as geometric primitives such as planar patches, cuboids or cylinders. We revised the detection procedure of previous work [30], utilized a more robust cylinder detection, improved MBR estimation, and introduced a patch mergence step. The performance of the proposed scheme was evaluated on synthesized spacecraft point cloud data and more challenging reconstructed point cloud data. Experimental results demonstrate the effectiveness, robustness and accuracy of our spacecraft component detection scheme. In future work, cones, arbitrary prisms and general polygons can be further considered in spacecraft component detection to make it more comprehensive.

Supplementary Materials: Detailed component detection results of robustness analysis are available in supplementary: www.mdpi.com/1424-8220/18/4/933/s1, including results for different point distribution density (20K, 10K and 05K), position noise (01U, 02U and 04U) and direction noise (05D, 10D and 15D) of all the 8 spacecraft (cube, DSP, GPS, Helios, minisat, Radarsat, SCISAT and SPOT), along with the intermediate detection results.

Acknowledgments: This work was supported in part by the National Natural Science Foundation of China (Grant Nos. 61501009, 61771031 and 61371134), the National Key Research and Development Program of China (2016YFB0501300, 2016YFB0501302) and the Aerospace Science and Technology Innovation Fund of CASC (China Aerospace Science and Technology Corporation).

Author Contributions: Q.W., Z.J. and H.Z. conceived of and designed the experiments. Q.W. performed the experiments. Q.W. and H.Z. wrote the paper.

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

References

1. Cefola, P.J.; Alfriend, K.T. Sixth US/Russian Space Surveillance Workshop. In Proceedings of the AIAA/AAS Astrodynamics Specialist Conference and Exhibit, Keystone, CO, USA, 21–24 August 2006; American Institute of Aeronautics and Astronautics: Reston, VA, USA, 2006; pp. 1677–1693.
2. Meng, G.; Jiang, Z.; Liu, Z.; Zhang, H.; Zhao, D. Full-viewpoint 3D Space Object Recognition Based on Kernel Locality Preserving Projections. *Chin. J. Aeronaut.* **2010**, *23*, 563–572.
3. Zhang, H.; Jiang, Z.; Elgammal, A. Vision-based pose estimation for cooperative space objects. *Acta Astronaut.* **2013**, *91*, 115–122.
4. Zhang, H.; Jiang, Z. Multi-view space object recognition and pose estimation based on kernel regression. *Chin. J. Aeronaut.* **2014**, *27*, 1233–1241.
5. Zhang, H.; Jiang, Z.; Elgammal, A. Satellite recognition and pose estimation using Homeomorphic Manifold Analysis. *IEEE Trans. Aerosp. Electron. Syst.* **2015**, *51*, 785–792.
6. Grilli, E.; Menna, F.; Remondino, F. A Review of Point Clouds Segmentation and Classification Algorithms. *ISPRS-Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2017**, *42*, 339–344.
7. Ruel, S.; Luu, T.; Berube, A. Space shuttle testing of the TriDAR 3D rendezvous and docking sensor. *J. Field Robot.* **2012**, *29*, 535–553.
8. Benninghoff, H.; Boge, T.; Rems, F. Autonomous navigation for on-orbit servicing. *KI-Künstliche Intell.* **2014**, *28*, 77–83.
9. Zhang, X.; Zhang, H.; Wei, Q.; Jiang, Z. Pose Estimation of Space Objects Based on Hybrid Feature Matching of Contour Points. In *Advances in Image and Graphics Technologies, Proceedings of the 11th Chinese Conference, IGTA 2016, Beijing, China, 8–9 July 2016*; Springer: Singapore, 2016; pp. 184–191.
10. Opromolla, R.; Fasano, G.; Rufino, G.; Grassi, M. Pose Estimation for Spacecraft Relative Navigation Using Model-Based Algorithms. *IEEE Trans. Aerosp. Electron. Syst.* **2017**, *53*, 431–447.
11. Tan, W.; Liu, H.; Dong, Z.; Zhang, G.; Bao, H. Robust monocular SLAM in dynamic environments. In Proceedings of the 2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Adelaide, Australia, 1–4 October 2013; pp. 209–218.
12. Engel, J.; Koltun, V.; Cremers, D. Direct Sparse Odometry. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 611–625.
13. Snavely, N.; Seitz, S.M.; Szeliski, R. Modeling the World from Internet Photo Collections. *Int. J. Comput. Vis.* **2008**, *80*, 189–210.
14. Zhang, H.; Wei, Q.; Jiang, Z. 3D Reconstruction of Space Objects from Multi-Views by a Visible Sensor. *Sensors* **2017**, *17*, 1689.
15. Vosselman, G.; Gorte, B.G.; Sithole, G.; Rabbani, T. Recognising structure in laser scanner point clouds. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2004**, *46*, 33–38.
16. Li, Y.; Wu, X.; Chrysathou, Y.; Sharf, A.; Cohen-Or, D.; Mitra, N.J. GlobFit: Consistently fitting primitives by discovering global relations. In *ACM SIGGRAPH 2011 Papers, Proceedings of the SIGGRAPH '11, Vancouver, BC, Canada, 7–11 August 2011*; ACM: New York, NY, USA, 2011; pp. 1–12.
17. Limberger, F.A.; Oliveira, M.M. Real-time detection of planar regions in unorganized point clouds. *Pattern Recognit.* **2015**, *48*, 2043–2053.

18. Chen, D.; Zhang, L.; Mathiopoulos, P.T.; Huang, X. A methodology for automated segmentation and reconstruction of urban 3-D buildings from ALS point clouds. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 4199–4217.
19. Cao, R.; Zhang, Y.; Liu, X.; Zhao, Z. Roof plane extraction from airborne lidar point clouds. *Int. J. Remote Sens.* **2017**, *38*, 3684–3703.
20. Rabbani, T.; Van Den Heuvel, F. Efficient hough transform for automatic detection of cylinders in point clouds. *ISPRS WG III/3, III/4* **2005**, *3*, 60–65.
21. Qiu, R.; Zhou, Q.Y.; Neumann, U. Pipe-Run Extraction and Reconstruction from Point Clouds. In *Computer Vision—ECCV 2014, Proceedings of the 13th European Conference, Zurich, Switzerland, 6–12 September 2014*; Springer International Publishing: Cham, Switzerland, 2014; pp. 17–30.
22. Pang, G.; Qiu, R.; Huang, J.; You, S.; Neumann, U. Automatic 3D industrial point cloud modeling and recognition. In Proceedings of the 2015 14th IAPR International Conference on Machine Vision Applications (MVA), Tokyo, Japan, 18–22 May 2015; pp. 22–25.
23. Borrmann, D.; Elseberg, J.; Lingemann, K.; Nüchter, A. The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design. *3D Res.* **2011**, *2*, 3.
24. Schnabel, R.; Wahl, R.; Klein, R. Efficient RANSAC for Point-Cloud Shape Detection. *Comput. Graph. Forum* **2007**, *26*, 214–226.
25. Yang, M.Y.; Förstner, W. Plane detection in point cloud data. In Proceedings of the 2nd International Conference on Machine Control Guidance, Bonn, Germany, 9–11 March 2010; Volume 1, pp. 95–104.
26. Isack, H.; Boykov, Y. Energy-Based Geometric Multi-model Fitting. *Int. J. Comput. Vis.* **2012**, *97*, 123–147.
27. Wang, L.; Shen, C.; Duan, F.; Guo, P. Energy-Based Multi-plane Detection from 3D Point Clouds. In *Neural Information Processing. ICONIP 2016. Lecture Notes in Computer Science*; Hirose, A., Ozawa, S., Doya, K., Ikeda, K., Lee, M., Liu, D., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 715–722.
28. Delong, A.; Osokin, A.; Isack, H.N.; Boykov, Y. Fast Approximate Energy Minimization with Label Costs. *Int. J. Comput. Vis.* **2012**, *96*, 1–27.
29. Boykov, Y.; Veksler, O.; Zabih, R. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.* **2001**, *23*, 1222–1239.
30. Wei, Q.; Jiang, Z.; Zhang, H.; Nie, S. Spacecraft Component Detection in Point Clouds. In Proceedings of the 12th Chinese Conference on Advances in Image and Graphics Technologies, IGTA 2017, Beijing, China, 30 June–1 July 2017; pp. 210–218.
31. Eberly, D.H. Geometric Tools. Available online: https://www.geometrictools.com (accessed on 18 November 2017).
32. Knapitsch, A.; Park, J.; Zhou, Q.Y.; Koltun, V. Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction. *ACM Trans. Graph.* **2017**, *36*, doi:10.1145/3072959.3073599.
33. Schöps, T.; Schönberger, J.L.; Galliani, S.; Sattler, T.; Schindler, K.; Pollefeys, M.; Geiger, A. A Multi-View Stereo Benchmark with High-Resolution Images and Multi-Camera Videos. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.