applied sciences

MDPI

Article

A 3D Vision-Based Weld Seam Extraction Method for Arbitrarily Placed Spatial Planar Weldments

Bo Yang [1,2], Zhengtuo Wang [1,2], Yuetong Xu [1,2], Songyu Hu [1,2] and Jianzhong Fu [1,2,*]

[1] State Key Laboratory of Fluid Power and Mechatronic Systems, College of Mechanical Engineering, Zhejiang University, Hangzhou 310027, China; 11725028@zju.edu.cn (B.Y.); xyt@zju.edu.cn (Y.X.)

[2] Key Laboratory of 3D Printing Process and Equipment of Zhejiang Province, College of Mechanical Engineering, Zhejiang University, Hangzhou 310027, China

* Correspondence: fjz@zju.edu.cn

Abstract: Vision-based weld seam extraction poses a significant challenge for weldments with complex spatial structures in automated welding. Existing research primarily focuses on identifying weld seams from weldments with given positions and postures, while practical weld path planning requires multiple weld seams identified within arbitrarily placed weldments. This paper proposes a methodology that identifies weld seams from arbitrarily placed spatial planar weldments in a single run. First, by introducing a turntable calibrated with respect to a 3D camera, we perform 3D reconstruction on an arbitrarily placed spatial planar weldment. Second, an improved RANSAC algorithm based on Euclidean clustering is proposed to carry out plane segmentation, focusing on segmentation accuracy. Finally, we present a novel weld seam extraction algorithm leveraging the half-edge data structure to efficiently compute weld seams from the segmented planes. The experiments conducted in this study demonstrate that the average segmentation errors (as an indirect indicator of weld seam extraction error) are reduced by 90.3% to 99.8% over conventional segmentation methods, and the standard deviations are reduced by 64.8% to 97.0%.

Keywords: weld seam extraction; point cloud registration; plane segmentation; Euclidean clustering

check for updates

Citation: Yang, B.; Wang, Z.; Xu, Y.; Hu, S.; Fu, J. A 3D Vision-Based Weld Seam Extraction Method for Arbitrarily Placed Spatial Planar Weldments. *Appl. Sci.* **2024**, *14*, 8493. https://doi.org/10.3390/app14188493

Academic Editor: César Vasques

Received: 23 August 2024
Revised: 8 September 2024
Accepted: 19 September 2024
Published: 20 September 2024

1. Introduction

The introduction of visual sensors enables the industrial robot as an actuator to perceive the positional information of the surrounding environment. Therefore, vision-based automatic weld seam extraction (WSE) and weld seam tracking have become an indispensable part of robotic welding, which requires the weldment's posture and position to be known to the robot.

According to our research on actual production, many welding applications must handle randomly placed weldments with uncertain position information. For these applications, the additional human labor and time cost brought by fixing the weldments with satisfactory accuracy for WSE are unacceptable in terms of efficiency and economy, especially in welding applications with multi-variety and small batches, as exemplified by the samples in Section 4.2. There are two major steps to achieve this goal. The first step is model reconstruction, from which a 2D or 3D model representing the geometric information of the weldment is obtained; the second step is to perform WSE on the model.

Model reconstruction can be straightforward for simple weldments. A common approach is capturing a 2D image using a laser profile sensor and extracting the weld seams through pixel analysis [1–5]. For applications with a more complex weld seam or multiple weld seams to be recognized, a monocular or binocular structured light 3D camera that outputs point clouds is more commonly used [6–9]. The problem is that when the weldment is too large for the camera's FOV or when there are occlusions between multiple welds, a single frame of point cloud cannot fully express all the necessary geometric features. In this regard, the studies mentioned above focus on the WSE of a single weld seam or several weld seams in a single frame of a point cloud, referenced as local weld seams in this paper. There are also scholars who work on solving this practical problem and obtaining global weld seams. To perform 3D reconstruction on a large-size tube sheet, Wang et al. [10] collect 2D images and laser rangefinder data simultaneously through the sensors installed on a robot. They perform SIFT feature matching on the 2D images to obtain the positional relationship between different frames of images, which, combined with the depth data, leads to a complete point cloud. Yan et al. [11] first convert the 3D point cloud into a 2D grayscale image and use an edge detection algorithm to detect the pixel set corresponding to the weld seam from the 2D image. The weld seams are then fitted with spline curves. The position errors are within ±0.3 mm, and the standard deviations are less than 0.4 mm. Considering the need to extract geometric features from weldments of complex spatial structures, in this paper, we arbitrarily mount the assembled weldments (One of the factors influencing the quality of welded joints is the appropriate sequence and order of assembly [12]) on a one-axis turntable. By rotating a weldment along with the turntable and capturing multiple frames of point clouds of the weldment using a binocular structured light 3D camera, a point cloud that represents the complete geometric features of the weldment is created through point cloud registration.

WSE can be divided into two types according to the vision sensor used. For research with 2D vision employed, weld seams are mostly obtained by running pixel analysis on 2D images scanned with a laser profile sensor. This approach is particularly suitable for weldments assembled by two workpieces since their geometric features can be expressed in 2D images with decent accuracy. As an example, Guo, Zheng, Lian and Yao [3] perform WSE on V-shaped butt joints by extracting pixels that represent the center line of a weld seam, with an average positional error of 0.147 mm. Chen, Ma, He, Sun and Ren [4] use a neural network to perform semantic segmentation on the 2D image of a narrow butt joint and extract the weld seam with an average position error of 0.207 mm. As for 3D-vision-based applications, most researchers choose to identify geometric elements from the point clouds. Some accomplish it through edge extraction. Zhang, Geng, Tian, Zheng, Jiang and Lai [9] use K-Means clustering to separate boundary points identified by a new feature descriptor. This method's advantage is its strong adaptability (it can be used for free surfaces). However, its 78.57% corner recognition rate for planar weldments is still insufficient for practical use. More studies choose to segment planes from the point cloud first. Yuankai, Yong, Xincheng, Xiaolong, Yusen and Min [6] study three cases of weldments composed of two parallel planes and use vanilla RANSAC to detect and remove the edge points of the two planes, and then use least squares fitting to calculate the weld seams as the center lines of the two plane point clouds. Yang, Li, Long, Fan and Liang [7] use the least squares method to fit the common plane of the two butted workpieces. The weld seams are calculated by polynomial fitting on the points farthest from the plane. The RMSE error of the fitted weld seams is less than 0.7 mm. Yang et al. [13] apply vanilla RANSAC on the point clouds to fit TEE joints and lap joints in simple weldments. Geng et al. [14] undertake similar work with more complex planar weldments, optimizing RANSAC to enhance the computational efficiency and fitting accuracy compared to vanilla RANSAC. The successful implementation of these methods all has a common prerequisite, that is, either the structure of the weldment is simple enough for the weld seams to be extracted through a single frame of image or the weldments must be fixed with a certain accuracy in advance and the camera must be located (at least roughly) in a position where the weld features can be captured. In addition, since the research objects are all single weld seams or multiple weld seams in a frame of a point cloud, only local weld seams can be obtained in the end rather than all global weld seams. Further research is warranted to address the global welding path planning challenge for spatial planar weldments. Hence, this paper is dedicated to proposing a WSE method for arbitrarily placed spatial planar weldments, i.e., weldments with spatial structures comprised of plate-like materials commonly observed in construction machinery, steel structures, carriage plates, and container plate manufacturing applications [14].

In this paper, we propose a new approach that uses an improved RANSAC algorithm based on Euclidean clustering to carry out plane segmentation, and a novel weld seam extraction algorithm leveraging the half-edge data structure is developed to compute weld seams from the segmented planes efficiently. Due to the difference in weldment sizes and types, equipment types and accuracy levels, and error evaluation methods in the existing literature, we cannot create a unified benchmark for various methods for comparison. However, the process in this paper successfully solves the problem of WSE in a scenario that is more complex than similar articles and greatly improves the accuracy compared to traditional feature extraction methods. It provides an effective reference for applying robot automatic welding of complex planar welded parts without precise clamping.

The structure of this paper is as follows: Section 1 introduces the research background and briefly reviews the literature. Section 2 presents the 3D reconstruction method for point cloud registration and proposes an improved RANSAC algorithm tailored for consecutive plane segmentation. Section 3 introduces a WSE method based on the results of plane segmentation. Section 4 describes several experiments conducted to practice and validate the proposed methods. Finally, Section 5 offers concluding remarks on the study presented in this paper.

2. Improved RANSAC for Plane Segmentation

2.1. Three-Dimensional Reconstruction

To obtain the 3D model that represents the global geometric features of a weldment, we grab multiple point clouds of the weldment, which is randomly fixed on a turntable and rotates with it. Three-dimensional reconstruction is carried out by point cloud registration on all the point clouds. The setup is depicted in Figure 1.

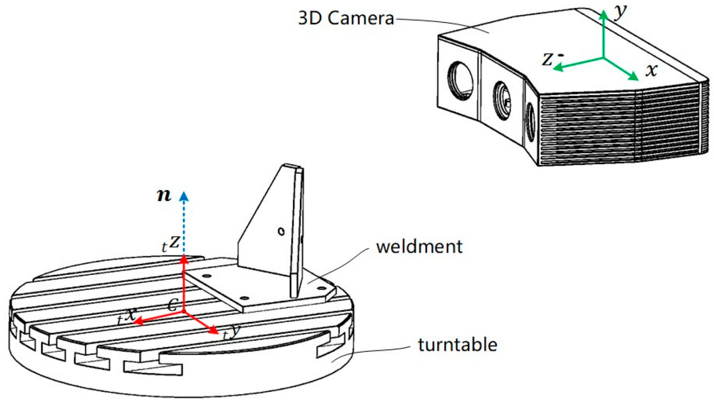

Figure 1. Capturing point clouds of the weldment on a turntable.

For any point fixed on the turntable, the coordinates with respect to the turntable system and the world base system (also the camera system) are ${}_t x$ and x, which satisfy

$$x = \boldsymbol{P}(\boldsymbol{n})\,{}_t x + c, \tag{1}$$

where $\boldsymbol{n}$ is the coordinate of the rotation axis, $\boldsymbol{P}(\boldsymbol{n})$ is the angle transform matrix between the two bases and c is the coordinate of the turntable center point with respect to the world base.

After the turntable rotates about its z axis (with respect to the turntable system) with angle θ, the coordinate becomes

$$\begin{aligned} {}_t x_\theta &= \boldsymbol{R}(z,\theta)\,{}_t x \\ \Rightarrow x_\theta &= \boldsymbol{P}(\boldsymbol{n})\,{}_t x_\theta + c \\ &= \boldsymbol{P}(\boldsymbol{n})\boldsymbol{R}(z,\theta)\,{}_t x + c \\ &= \boldsymbol{P}(\boldsymbol{n})\boldsymbol{R}(z,\theta)\boldsymbol{P}^{-1}(\boldsymbol{n})(x-c) + c \\ &= R(\boldsymbol{n},\theta)x + (\boldsymbol{I} - \boldsymbol{R}(\boldsymbol{n},\theta))c, \end{aligned} \tag{2}$$

where

$$\boldsymbol{R}(z,\theta) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{3}$$

and

$$\boldsymbol{R}(n,\theta) = \boldsymbol{P}(\boldsymbol{n})\boldsymbol{R}(z,\theta)\boldsymbol{P}^{-1}(\boldsymbol{n}) \tag{4}$$

are the rotation matrices with respect to rotation axis z and $\boldsymbol{n}$, respectively.

So, the registered point cloud of multiple frames grabbed on the turntable can be expressed as

$$C_\theta = \bigcup_j \bigcup_k \left[\boldsymbol{R}(\boldsymbol{n},\theta-\theta_j)x_k + \left(\boldsymbol{I} - \boldsymbol{R}(\boldsymbol{n},\theta-\theta_j)\right)c\right], \tag{5}$$

where θ_j is the angular position of the weldment in frame j and θ is that of the registered point cloud; x_k is the position of the kth point in the point cloud.

2.2. Improved RANSAC

Before extracting weld seams from the registered point cloud, we employ RANSAC for single-plane segmentation. As a widely utilized method in geometric feature extraction and shape detection, RANSAC has proved its effectiveness in multiple fields such as SLAM [15], Geographic detection in nature [16] or urban areas [17,18], and so on. As outlined in the algorithm proposed in [19], vanilla RANSAC iteratively fits a plane from a point cloud over k iterations. Each iteration entails four sequential steps (referred to as steps (1) to (4) in Figure 2) to obtain a candidate plane, maximizing the count of inliers—points with distances to the candidate plane smaller than a preset threshold.

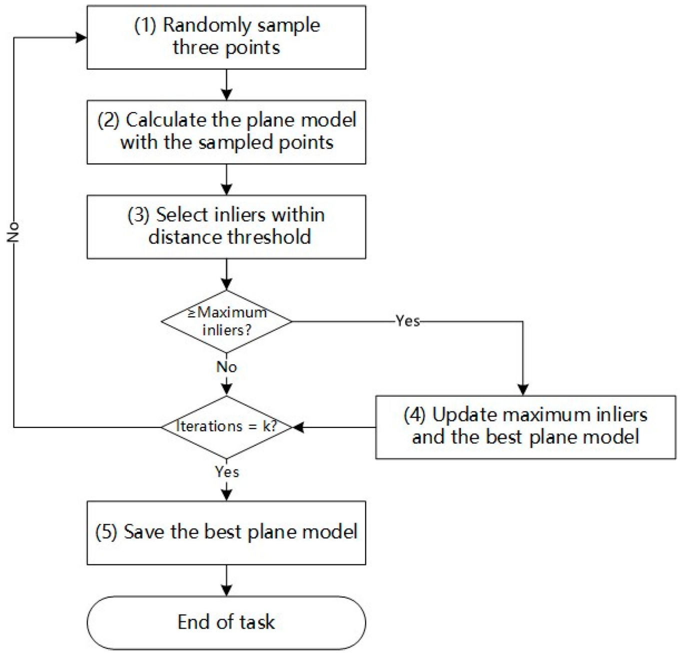
(1) Randomly sample three points
(2) Calculate the plane model with the sampled points
(3) Select inliers within distance threshold
≥Maximum inliers?
Yes
No
(4) Update maximum inliers and the best plane model
Iterations = k?
No
Yes
(5) Save the best plane model
End of task

Figure 2. Flowchart of vanilla RANSAC for single plane segmentation.

The iteration number k is defined as

$$k = \frac{\log(1-p)}{\log(1-\omega^3)}, \tag{6}$$

where ω is the inlier ratio, i.e., is the number of inliers divided by the number of points, and the constant variable p is normally set as 0.99 [20].

Despite its effectiveness in various applications, vanilla RANSAC encounters challenges when applied to plane segmentation tasks in spatial planar weldments. To address these issues, we propose several rules tailored to the different steps of the vanilla RANSAC, and the refined algorithm is simply called "improved RANSAC" in this paper.

2.2.1. Optimization for Point Sampling

The initial challenge is "invalid sampling," as Figure 3a illustrates. This scenario depicts a situation where, instead of extracting the intended candidate plane, S_1, an invalid plane, S_2, is obtained as it outperforms S_1 by including more inliers.

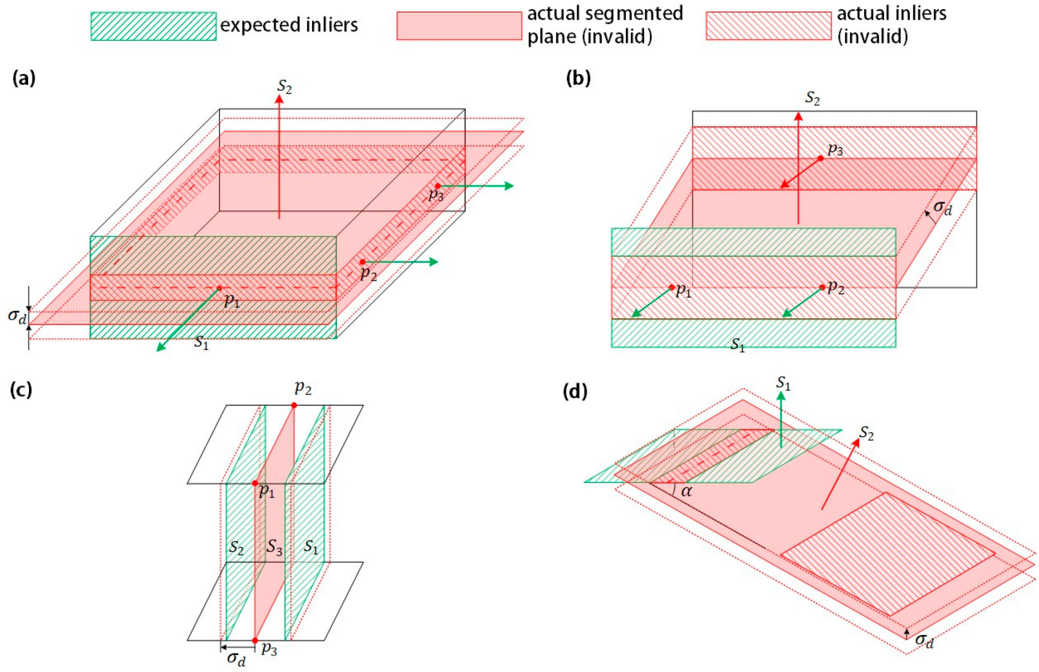

Figure 3. Situations of invalid segmentation: (**a**) invalid sampling; (**b**) invalid candidate plane; (**c**) closely spaced planes; (**d**) erroneously included inliers.

Rule 1. Sample validation.

In the sampling step (step 1) of the algorithm in Figure 2, any pair of the three sampled points must have similar normal vectors, which can be estimated with covariance matrices [21]. That is to say, for any three sampled points p_1, p_2, and p_3, the sample is valid only when

$$\alpha_{ij} = \arccos\left(\frac{|\boldsymbol{n}_i \cdot \boldsymbol{n}_j|}{\|\boldsymbol{n}_i\|\|\boldsymbol{n}_j\|}\right) < \sigma_a, \quad i, j \in \{1, 2, 3\}, i \neq j, \tag{7}$$

where $\sigma_a > 0$ is the preset angle threshold and $\boldsymbol{n}_i$ is the corresponding surface normal vector at p_i in Figure 3a.

By incorporating Rule 1, cases illustrated in Figure 3a are effectively bypassed. This further prevents the generation of unqualified candidate planes, reducing the effort of candidate plane validation.

2.2.2. Optimization for Candidate Plane Validation

The second challenge, termed "invalid candidate plane", arises after the plane model calculation step (step 2) of the algorithm depicted in Figure 2. Occasionally, invalid candidate planes may emerge despite the sampled points passing Rule 1, as illustrated in Figure 3b. In this scenario, while the sampled points possess similar normal vectors, the candidate plane's normal vector exhibits a significant disparity.

Rule 2. Candidate plane validation.

For any candidate plane calculated using the sampled points, its normal vector must be close to those of the sampled points. That is to say, candidate plane S is valid only when

$$\arccos\left(\frac{|\boldsymbol{n}_i \cdot \boldsymbol{n}_S|}{\|\boldsymbol{n}_i\|\|\boldsymbol{n}_S\|}\right) < \sigma_a, \quad i \in \{1, 2, 3\}, \tag{8}$$

where $\boldsymbol{n}_S$ is the normal vector of S.

2.2.3. Optimization for Inlier Validation and Update Criterion

In the irregular scenario depicted in Figure 3d, a correctly segmented plane erroneously includes inliers belonging to other planes, thereby causing interference with the remaining points to be segmented. This situation underscores the potential for a candidate plane to occupy points intended for other planes.

Rule 3. Inlier validation.

Besides the distance threshold, the points selected as inliers must have similar normal vectors as the segmented plane. Under this rule, a point p_i is a valid inlier of plane S only if the following conditions are satisfied:

$$\begin{cases} \text{distance}(p_i, S) < \sigma_d, \\ \arccos\left(\frac{|\boldsymbol{n}_i \cdot \boldsymbol{n}_S|}{\|\boldsymbol{n}_i\|\|\boldsymbol{n}_S\|}\right) < \sigma_a. \end{cases} \quad i \in \{1, 2, \ldots, l\}, \tag{9}$$

where σ_d is the distance threshold and

$$\text{distance}(p_i, S) = \frac{|Ax_i + By_i + Cz_i + D|}{\sqrt{A^2 + B^2 + C^2}}, \tag{10}$$

for point $p_i = (x_i, y_i, z_i)$ and plane S representing $Ax + By + Cz + D = 0$.

Another irregular situation, exemplified in Figure 3c, arises when the distance between two parallel planes is less than the distance threshold σ_d. In this case, the central plane, marked as S_3 in Figure 3, is prioritized over S_1 and S_2 to be chosen as the candidate plane, as per the iteration update in Figure 2, which selects a new candidate plane if it has more inliers than the historical best one. While a straightforward solution would be to use a smaller σ_d, altering such a significant variable in RANSAC could introduce unexpected variability in the output. Hence, there is a need to revise the update criterion accordingly.

To begin with, we introduce variable $\boldsymbol{e}$ as the local segmentation error of a candidate plane:

$$e = \sum_{i=1}^{l}\left(\gamma \text{distance}(p_i, S) + (1-\gamma)\frac{|\boldsymbol{n}_i \cdot \boldsymbol{n}_S|}{\|\boldsymbol{n}_i\|\|\boldsymbol{n}_S\|}\right), \tag{11}$$

where l is the number of inliers of the current candidate plane S, $\boldsymbol{n}_i$ is the normal vector at p_i and $\gamma \in [0, 1]$ is the weight to control the significance of distance errors and angle errors.

Rule 4. Update criterion.

For a new candidate plane S_j in the jth iteration, it outperforms the historical best iteration k only when

$$\rho_j = \frac{l_j}{e_j} > \frac{l_k}{e_k}. \tag{12}$$

The introduction of the new objective value ρ provides a refined measure of the priority of a candidate plane, considering not only its number of inliers but also the distance and angle errors. This modification substantially mitigates situations akin to Figure 3c, where parallel planes may erroneously interfere with each other.

2.2.4. Optimization for Final Output

Rule 3 and Rule 4 generally perform well for normal situations; however, there are instances where they fall short, particularly when the angle α in Figure 3d is smaller than the σ_a defined in Rule 3. As illustrated by S_1 and S_2 in Figure 4, the current RANSAC algorithm would only segment one plane (e.g., plane S' in Figure 4b) since the inliers of both S_1 and S_2 fall within the distance and angle thresholds. This outcome inevitably introduces additional segmentation errors in the resulting fitted planes and weld seams.

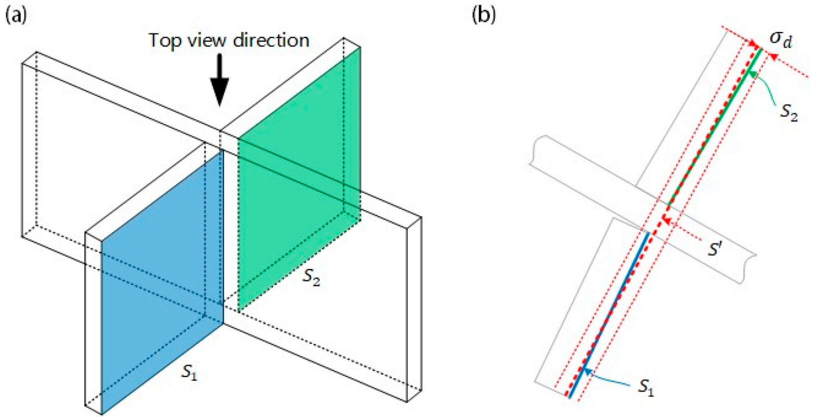

Figure 4. Segmentation error brought by similar planes: (**a**) 3D view; (**b**) top view. In the top view, S_1 and S_2, represented with blue and green solid lines respectively, are the planes to be segmented. In contrast, the mistakenly segmented plane S' is represented with a dotted red line. σ_d is the distance threshold for inlier selection in RANSAC.

The points in S_1 and S_2 can be easily clustered into two groups using Euclidean clustering: Point sets Φ_i and Φ_j are two clusters, if

$$\min\left(\mathrm{distance}\left(p_i, p_j\right)\right) \geq \sigma_d \tag{13}$$

holds for any i and j, where σ_d is the distance threshold [22]. This definition also indicates that for any point p_i in a cluster that has more than one element, there exists at least one point that distanced less than σ_d to p_i.

Rule 5. Final refinements.

After inlier validation, Euclidean clustering on the inliers is performed, resulting in m clusters of inliers $\left(p_1^1, \ldots, p_{l_1}^1\right), \left(p_1^2, \ldots, p_{l_2}^2\right), \left(p_1^m, \ldots, p_{l_m}^m\right)$. Choose only one cluster as the inliers for the final segmented plane according to the following steps.

(a) Least-square fitting is performed on each cluster of inliers, leading to a corresponding number of plane models. That is to say,

$$S_j = \mathrm{argmin} \sum_{i=1}^{l} \left\| \mathrm{distance}\left(p_i, S_j\right) \right\|^2, \tag{14}$$

where p_i annotates the ith inlier of plane S_j.

(b) Choose a plane as the final output that satisfies

$$S_j = \underset{j}{\mathrm{argmin}}\left(e_j\right), \tag{15}$$

where e_j is defined as e in (11). The inliers that do not belong in the selected cluster return to the point pool for sampling in the next iteration.

Figure 5 illustrates how Rule 5 affects the segmentation results of the case in Figure 4. Without the refinements in Rule 5, spatially isolated planes with a small angle are recognized as a single plane. Also, the segmented planes may look incomplete if there are points (that should be inliers of the segmented planes) distanced farther than σ_d to them. Both problems are eliminated with the introduction of Rule 5.

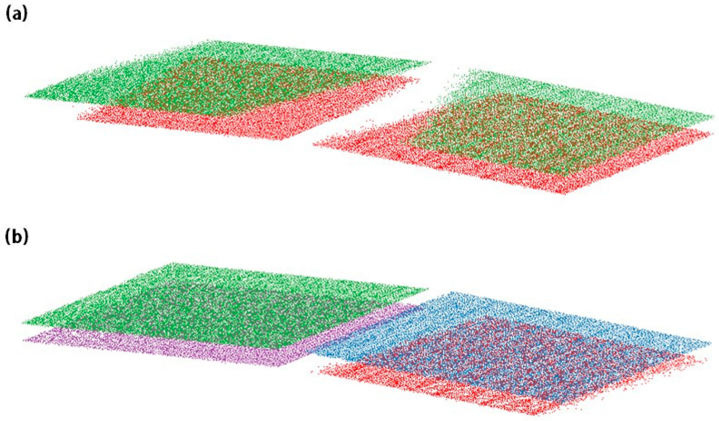

Figure 5. Segmentation before (**a**) and after (**b**) using Rule 5, with each segmented plane rendered in a different color.

2.3. Consecutive Plane Segmentation

With the proposed improved RANSAC algorithm for single-plane segmentation, consecutive segmentation becomes achievable. To consecutively segment multiple planes from a point cloud, the fundamental concept involves iteratively executing the algorithm on the point cloud with the inliers of the last segmented plane removed until all valid planes are segmented from the point cloud. Figure 6 illustrates this process. The output of the consecutive plane segmentation comprises a list of plane models segmented from the input point cloud, along with the corresponding inliers of each segmented plane.

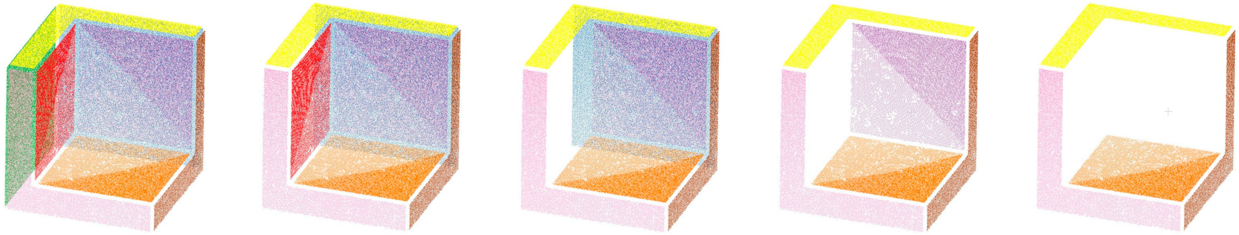

Figure 6. The process of consecutive plane segmentation: Each step of the segmentation fits a plane and removes its inliers. The inliers of each plane are rendered in a different color. The color contrast of the inliers belonging to the same plane is caused by the inconsistent density of points in the point cloud (caused by occlusion during point cloud capturing).

In consecutive plane segmentation, two termination conditions can be considered: the number of segmented planes and the number of remaining points. Since the former requires clear user expectations, making automation applications less flexible, the latter termination condition is adopted in this paper.

In the proposed improved RANSAC algorithm, comparisons of normal vectors occur frequently, leading to time-consuming duplication of efforts. Particularly in the sampling process, numerous invalid iterations are spent on sampling points with different normal vectors. This inefficiency could be mitigated by clustering points based on normal vectors. To explore this approach, the normal vectors of the point cloud are visualized in a unit ball (Figure 7a), and the result is rendered into a temperature map representing the probability density of the normal vectors (Figure 7b). It becomes apparent that the points can be clustered based on their normal vectors. This insight inspires us that most points in a point cloud can be divided into several clusters according to their normal vectors.

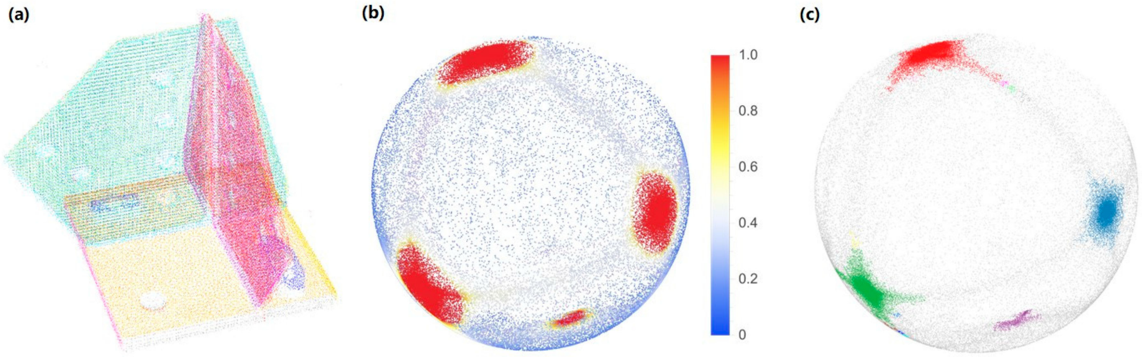

Figure 7. Normal vector clustering: (**a**) the original point cloud; (**b**) the temperature map representing the probability density of the normal vectors; (**c**) normal vector clustering using Euclidean clustering.

To precisely obtain cluster information, Euclidean clustering is applied to the unit ball in Figure 7b. The result (Figure 7c) resembles the probability density result in Figure 7b. The three major clusters in Figure 7c correspond to the three groups of orthogonal planes in Figure 7a.

Normal vector clustering offers two significant merits. Firstly, by clustering points, sampling is performed from a cluster rather than the entire point cloud in each iteration. This not only enhances sampling efficiency by reducing the sample space but also decreases the number of points to be considered in subsequent steps. Secondly, since points with different normal vectors are segregated into distinct clusters, they do not interfere with each other during the selection and counting of inliers, leading to improved segmentation results.

The detailed steps of consecutive plane segmentation based on normal vector clustering are elucidated in the pseudo-code provided in Section 2.4.2.

2.4. Algorithm Implementation

For more clarified elaboration, the algorithm's pseudo-code is separated into two parts: the first for the improved RANSAC Algorithm and the second for the consecutive plane segmentation.

2.4.1. Implementation for Improved RANSAC Algorithm

The pseudo-code for the improved RANSAC algorithm for a single segmentation is provided in Algorithm 1 in Python style.

Line 5 initiates the main loop (lines 5 to 24) that segments a single plane. Lines 8 to 14 implement Rule 1 (sample validation) in Section 2.2.1, where unqualified samples are bypassed before calculating the candidate plane. Line 15 calculates a candidate plane from the sampled points. Lines 16 to 17 implement Rule 2 (candidate plane validation) in Section 2.2.2. Lines 18 to 20 implement Rule 3 (inlier validation) in Section 2.2.3. Lines 21 and 22 update the current best candidate plane and its inliers by implementing Rule 4 (update criterion) in Section 2.2.3. Lines 23 and 24 check whether the termination condition in Formula (6) is satisfied. Lines 25 and 26 implement Rule 5 in Section 2.2.4, performing Euclidean clustering on the inliers and choosing the best cluster for final segmentation.

Algorithm 1 Improved RANSAC Algorithm

```
Input: the point cloud points and the corresponding normal vectors normals;
Output: segmented plane model plane and its inliers inliers
max_rho = 0
inlier_indices = []
for i in range(max_iteration):
    inlier_indices = []
    [a, b, c] = [−1, −1, −1]
    for j in range(max_sampling):
        [a1, b1, c1] = sample_3_random_points_indices(points)
        if valid_sample(normals[a], normals[b], normals[c] ):
            [a, b, c] = [a1, b1, c1]
            break
    if $a < 0$:
        continue
    plane = get_plane(points[a], points[b], points[c])
    if not valid_candidate_plane(plane):
        continue
    for j in range(len(points)):
        if valid_inlier(j):
            inlier_indices.append(j)
    if get_rho(inlier_indices) > max_rho:
        max_rho = get_rho(inlier_indices)
    if i > get_k(len(inlier_indices)):
        break
clusters = euclidean_cluster(points, inlier_indices)
plane, inliers = choose_from_clusters(clusters, points, normals)
return plane, inliers
```

2.4.2. Implementation for Consecutive Plane Segmentation

The pseudo-code of Consecutive Plane Segmentation is listed in Algorithm 2.

Line 5 performs Euclidean clustering on the normal vectors, which returns a list of clusters containing the point indices. Line 6 begins the main loop, where the biggest cluster is used as the input for single-plane segmentation in each iteration. The improved RANSAC method in line 10 refers to the code block in Algorithm 1.

Algorithm 2 Consecutive Plane Segmentation Algorithm

```
1  Input: the point cloud points and the corresponding normal vectors normals;
2  Output: segmented plane models planes and their corresponding inliers inliers_groups
3  planes = []
4  inliers_groups = []
5  clusters = euclidean_cluster(normals)
6  while True:
7      cluster_index = argmax([len(cluster) for cluster in clusters])
8      if len(clusters[cluster_index]) < least_points_for_segmentation:
9          break
10     plane, inliers = improved_RANSAC(
       points[clusters[cluster_index]], normals[clusters[cluster_index]])
11     if len(inliers) == 0:
12         break
13     clusters[cluster_index] = [index for index in clusters[cluster_index] if index not in inliers]
14     planes.append(plane)
15     inliers_groups.append(inliers)
16 return planes, inliers_groups
```

3. Weld Seam Extraction

3.1. Intersection Point Calculation

Consider all the possible topology combinations (Figure 8) of three planes.

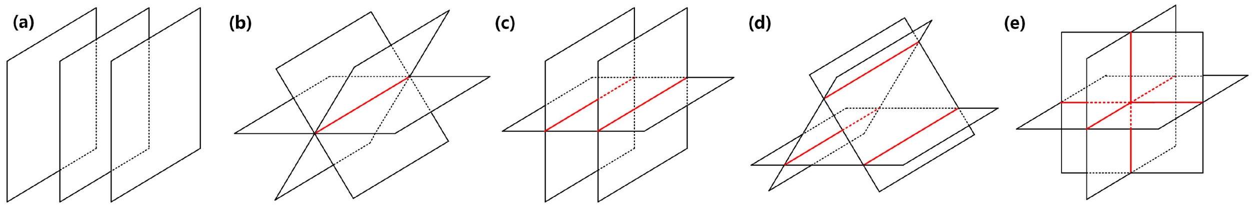

Figure 8. The possible topologies of three planes: (**a**) no intersection lines; (**b**) one intersection line; (**c**) two intersection lines; (**d**) three parallel intersection lines; (**e**) three intersection lines intersecting at one point. The intersection lines are represented with red lines.

Upon initial observation, situations in Figure 8b–e all appear to potentially produce weld seams. However, considering the physical thickness of any planar object, almost all situations can be expressed as Figure 8e. Consequently, a weld seam can be conceptualized as a set of lines connecting two intersection points, each formed by three planes. This problem can be addressed by solving a simple linear equation with three variables. Since Figure 8e is the only one to be considered, each combination of three planes can be validated using the following criterion:

$$\alpha_{ij} = \arccos\left(\frac{|\boldsymbol{n}_i \cdot \boldsymbol{n}_j|}{\|\boldsymbol{n}_i\|\|\boldsymbol{n}_j\|}\right) < \sigma_a,\ \ i,j \in \{1,2,3\}, i \neq j, \tag{16}$$

where $\boldsymbol{n}_i$ is the normal vector of plane i. Formula (16) looks identical to (7), but the latter is used to validate sampled points in the RANSAC algorithm.

As noted, "almost" all situations can be simplified to Figure 8e, with one exception: the base plane, where the weldment is mounted on the fixture, is typically obstructed from the camera's view. Consequently, no points in the point cloud serve as inliers for the base plane. This issue can be readily resolved by manually adding the base plane to the list of segmented planes.

3.2. Vertex Validation

Based on the topology combination analysis described above, every weld seam comprises a line set connecting a starting point and an ending point, both of which are intersected points (vertices) formed by three planes. By iterating through all $C(n,\ 3)$ combinations of the segmented planes, all possible starting and ending points can be calculated. However, not every vertex is a valid start point or end point of a weld seam. As an illustration, Figure 9 shows a point p_1 intersected by planes S_1, S_2 and S_3, which obviously cannot serve as a vertex of a weld seam.

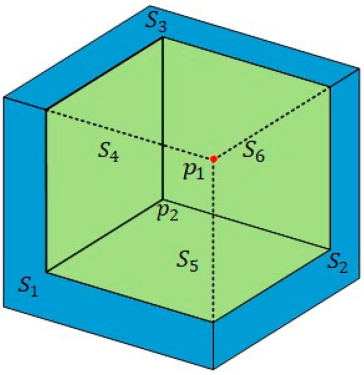

Figure 9. Example of an invalid intersection point (p_1).

By comparing p_1 with a valid vertex p_2 in Figure 9, a quick discovery is that invalid vertices are not located in the inliers of any of the planes intersecting. To rule out invalid intersections for weld seam extraction, each vertex is validated by the following criterion:

$$\min\left(\text{distance}\left(p, P_{S_i}\right)\right) < \sigma_d, \ \ i \in \{1,2,3\} \tag{17}$$

where P_{S_i} is the cluster of inliers of plane S_i. This criterion would likely fail without adopting Rule 5 proposed in Section 2.2.4 because isolated noisy points will be included in the inliers of a segmented plane, making the criterion satisfied even in case of an invalid intersection.

3.3. Edge Convexity Validation

With all the valid vertices obtained, a weld seam can be seen as an edge connecting two vertices that share two intersection planes. However, not all the edges are valid weld seams. A simple example is shown in Figure 10, and except p_1p_2 (in red color), all the other edges are invalid weld seams.

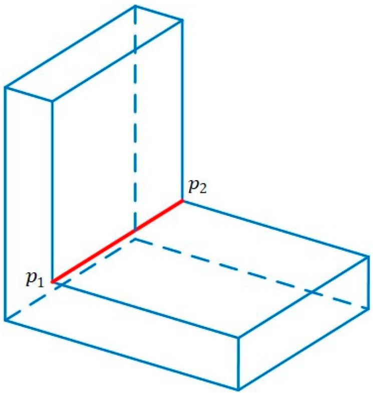

Figure 10. Edges connected by vertices.

By intuition, we recognize that weld seams must represent "concave edges," while the remaining edges are "convex edges." A concave edge is formed by two surfaces with an angle between 0° to 180°, and the angle has to be outside of the entity. Through the use of the half-edge data structure, also known as the doubly connected edge list [23], we observe the following pattern. As illustrated in Figure 11, for two adjacent planes, $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are the twin half-edges of their common edge p_1p_2. $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$ are the corresponding normal vectors. We introduce variable ς as an indicator of the convexity of an edge:

$$\varsigma = (\boldsymbol{n}_1 \times \boldsymbol{n}_2)\cdot\boldsymbol{e}_1. \tag{18}$$

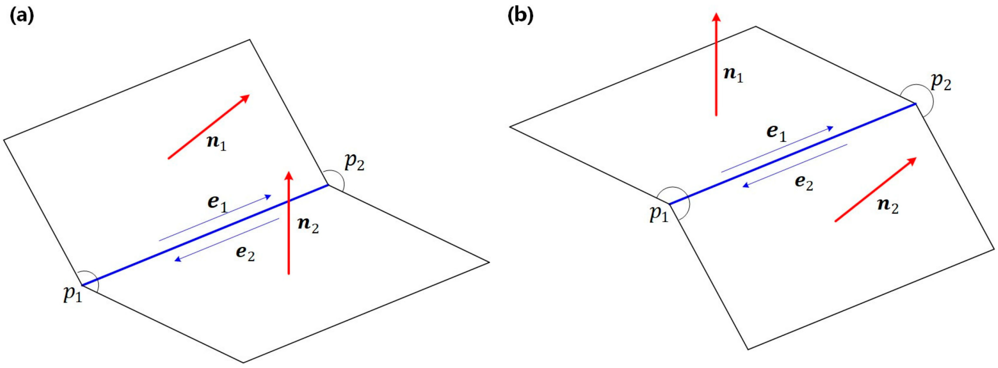

Figure 11. Concave edge (**a**) and convex edge (**b**).

The conclusion is as follows: when $\varsigma < 0$, edge p_1p_2 is concave; when $\varsigma > 0$, edge p_1p_2 is convex; when $\varsigma = 0$, the two planes forming edge p_1p_2 are coplanar. With this principle, all the convex edges can be ruled out.

By expressing the intersection points in each plane with the half-edge data structures, all the intersection points and edges constitute a half-edge mesh, where the edge convexity indicator is used to validate concave edges as valid weld seams.

3.4. Algorithm Implementation for WSE

The implementation of WSE is depicted in Algorithm 3. The loop from lines 7 to 18 iterates all the three-plane combinations and calculates the valid vertices, where line 11 implements the criterion (18). The edges are calculated from line 19 to line 24. Line 25 creates a half-edge mesh from the edges, from which lines 26 to 33 select the valid weld seams from the edges of the half-edge mesh using edge convexity validation in Section 3.3.

Algorithm 3 WSE Algorithm

```
1  Input: the point cloud points; the segmented planes planes and their inliers inliers_planes (each element in
   inliers_planes is a list of point indices representing the inliers of the corresponding plane).
2  Output: list of vertices vertices; list of binary tuples seams representing the indices of the points to be connected
   as weld seams.
3  vertices = []
4  edges = []
5  vertices_per_plane = [[] for _ in range(len(planes))]
6  planes_per_vertex = []
7  for comb_3_planes in combinations(range(len(planes)), 3):
8    if not valid_planes_for_intersection(comb_3_planes):
9      continue
10   vertex = get_intersect_point(planes[comb_3_planes[0]], planes[comb_3_planes[1]],
   planes[comb_3_planes[2]])
11   if not valid_intersection_point(vertex, points, inliers_planes[comb_3_planes[0]],
   inliers_planes[comb_3_planes[1]], inliers_planes[comb_3_planes[2]], sigma_d):
12     continue
13   vertex_idx = len(vertices)
14   vertices.append(vertex)
15   vertices_per_plane[comb_3_planes[0]].append(vertex_idx)
16   vertices_per_plane[comb_3_planes[1]].append(vertex_idx)
17   vertices_per_plane[comb_3_planes[2]].append(vertex_idx)
18   planes_per_vertex.append(comb_3_planes)
19 for comb_2_vertices in combinations(range(len(vertices)), 2):
```

4. Experimental Verification

4.1. Experiment Setup

To evaluate the proposed method with weldments, an experimental platform consisting of a turntable and a 3D structured light camera is built, depicted in Figure 12. The 3D camera (DM Series production of Hanchine) has a FOV of 110 × 70 mm (close end) to 300 × 210 mm (far end), with a resolution of 2.3 million and an average error of 0.15%. The turntable diameter is 250 mm, and the repeatability is ±0.05°.

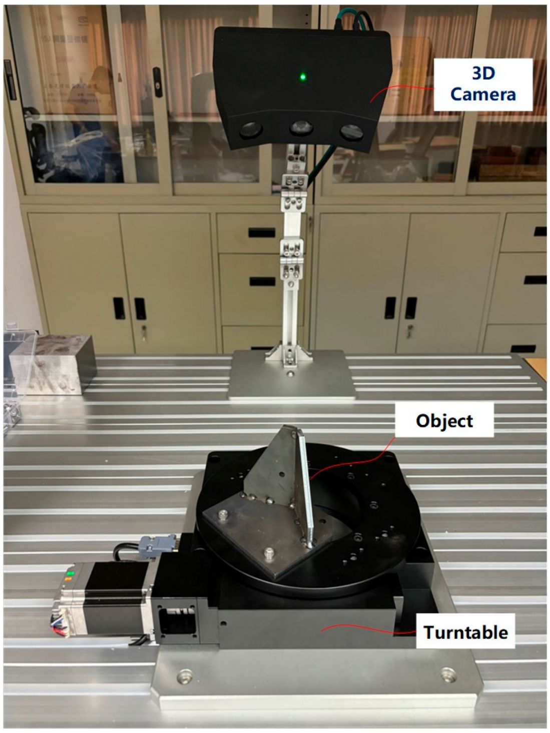

Figure 12. Experiment setup for 3D reconstruction.

According to Formula (5), the accurate turntable axis vector $\boldsymbol{n}$ and center point c with respect to the camera coordination are required for 3D reconstruction. We employ the method proposed by Yang et al. [24] to calibrate and verify the parameters in the following steps.

(1) A standard metal cube (of size 80 × 100 × 120 mm^3) is used as the reference object for calibration. The setup is displayed in Figure 13, where 36 frames of point clouds of the cube are captured with the turntable rotating 10° for each frame (Figure 14a).

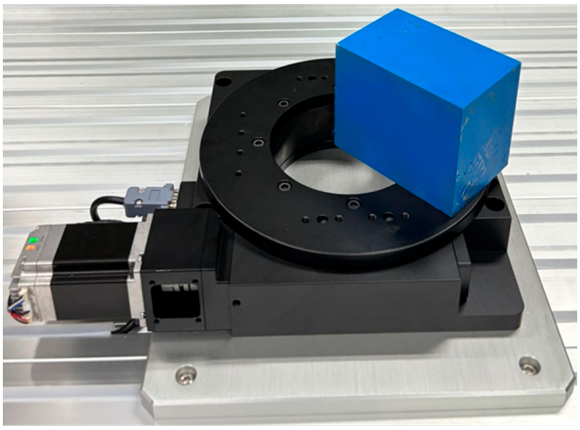

Figure 13. Experiment setup for turntable calibration.

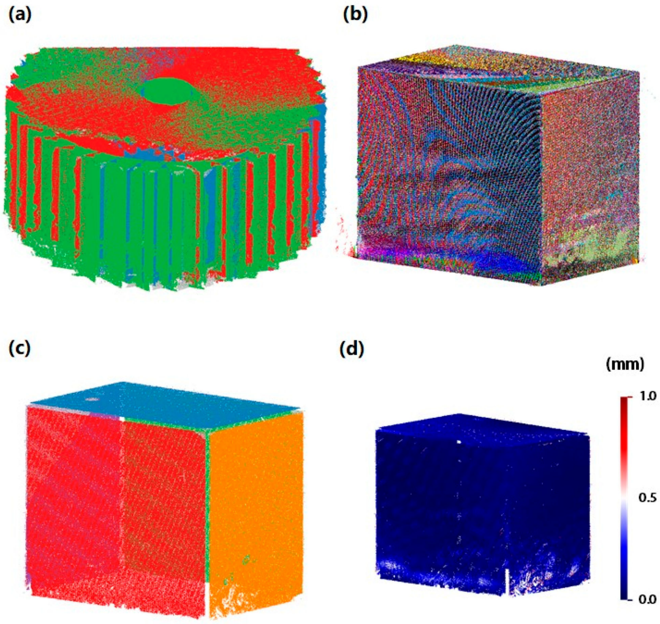

Figure 14. Verification experiments with the calibrated turntable: (**a**) the multiple point clouds grabbed from the standard cube; (**b**) the registered point cloud; (**c**) plane segmentation result of the registered point cloud; (**d**) temperature map representing segmentation error of the plane segmented planes.

(2) All frames of point clouds are merged into one with point cloud registration, where $\boldsymbol{n}$ and c are the main parameters. The calculation of $\boldsymbol{n}$ and c is turned into the optimization of the point cloud registration that leads to the minimum registration error, which is further decomposed into axis calibration and center point calibration. The result is recorded in Table 1.

Table 1. Calibration result of the turntable.

Variable	Result
$\boldsymbol{n}$	$[0.0202, -0.681, -0.732]$
c (mm)	$[73.72, -45.10, 466.60]$

(3) Point cloud registration is performed on all the point clouds obtained in step (1) with the parameters calculated in step (2), as depicted in Figure 14b.

(4) Perform plane segmentation on the registered point cloud in step (3) (Figure 14c). The registration error in step (3) is measured as the plane segmentation error, as visualized in Figure 14d. Since the segmentation error is small, we reasonably infer that the calibration is accurate enough.

4.2. Experiment for 3D Reconstruction

To test the proposed 3D reconstruction and WSE algorithms, we choose some typical spatial planar workpieces from a signal tower base manufacturer as weldments. As displayed in Figure 15, the samples meet the characteristics of "multi-variety and small batches" described in Section 1.

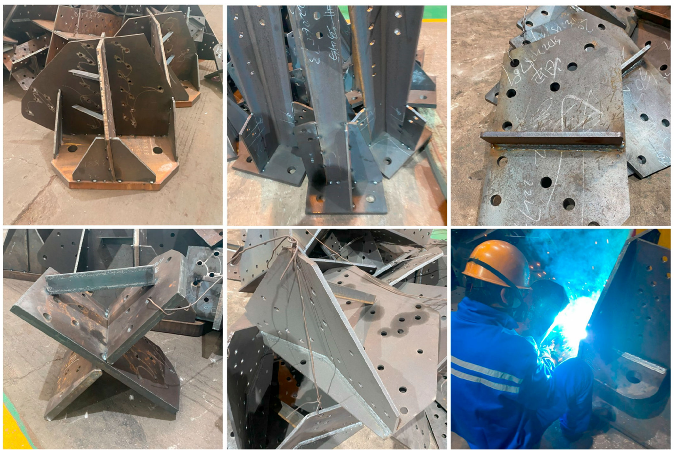

Figure 15. Spatial planar weldments from a signal tower base manufacture.

With the turntable calibrated, point cloud registration with multiple frames of point clouds is possible. To reconstruct the geometric characteristics of the weldment, point cloud registration is applied with Formula (5). Figure 16 displays the point clouds before and after registration.

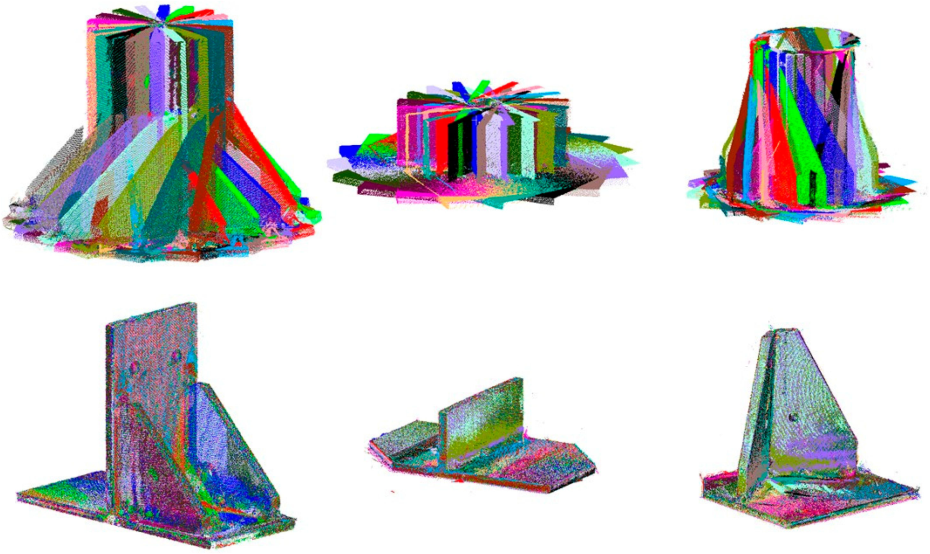

Figure 16. Point clouds before (**first row**) and after (**second row**) registration, with each point cloud frame rendered in a different color.

4.3. Experiment for Plane Segmentation

Several experiments are conducted to evaluate the performance of the plane segmentation algorithm proposed in Section 3.

The standard cube serves as the experimental object for the initial comparison. Table 2 presents a detailed comparison of segmentation results between vanilla RANSAC, RANSAC without normal vector clustering (non-clustering RANSAC hereafter), and the improved RANSAC. To ensure a fair comparison, all implementations utilize identical configurations for variables such as downsampling, distance threshold, max iteration, etc. No parallel computation, such as AVX, is employed (the same goes for the following comparisons). The reported results are averaged over 20 runs. For vanilla RANSAC, the PCL implementation [25] is utilized.

Table 2. Segmentation result comparison with a standard cube.

Variable	Vanilla RANSAC	Non-Clustering RANSAC		Improved RANSAC (Proposed)	
	Value	Value	Comparison *	Value	Comparison *
average error (mm)	0.28	0.40	+42.9%	0.072	−74.3%
standard deviation	0.0023	0.0030	+30.4%	0.00081	−64.8%
execution time (s)	4.85	30.46	+528%	16.79	+246%
correct planes segmented	5/5	5/5	0%	5/5	0%
incorrect planes segmented	0	0	0%	0	0%
total points (downsampled)	140, 022	140, 022	0%	140, 022	0%
inliers	139, 003	134, 471	−3.2%	134, 591	−3.17%

* The comparisons are made against Vanilla RANSAC.

Analysis of Table 2 reveals that the improved RANSAC significantly outperforms vanilla RANSAC regarding segmentation accuracy, although this difference may not be readily apparent in Figure 17. Vanilla RANSAC exhibits faster execution compared to the proposed algorithm, as the surface normal estimation and Euclidean clustering utilized in the proposed algorithm are relatively time-consuming. However, compared to non-clustering RANSAC, the improved RANSAC demonstrates superior efficiency.

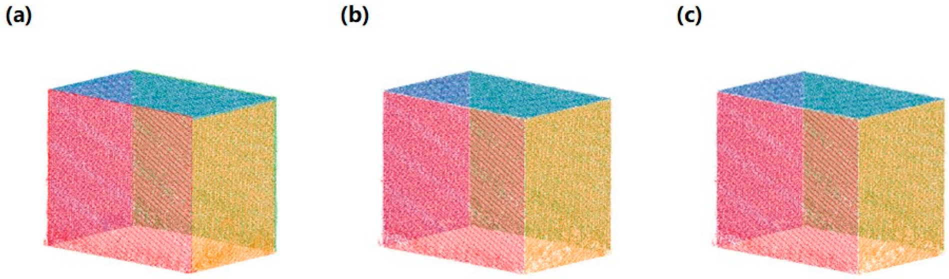

Figure 17. Segmentation comparison over a standard cube. (**a**) Vanilla RANSAC; (**b**) non-clustering RANSAC; (**c**) improved RANSAC.

In the subsequent experiment, an icosahedral column, serving as a relatively complex standard workpiece, is employed as the experimental object. The comparison results are depicted in Figure 18 and summarized in Table 3. In this set of experiments, both vanilla RANSAC and non-clustering RANSAC exhibit poor performance in accurately segmented planes, with numerous incorrect planes generated. With further tuning in the configuration, the latter may perform better in this experiment. However, we retain this result as, with the same configuration, the improved RANSAC efficiently identifies all the planes.

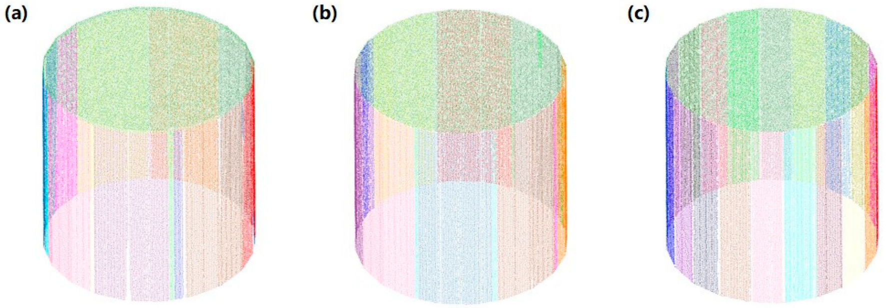

Figure 18. Segmentation comparison over an icosahedral column. (**a**) Vanilla RANSAC; (**b**) non-clustering RANSAC; (**c**) improved RANSAC.

Table 3. Segmentation result comparison with an icosahedral column.

Variable	Vanilla RANSAC	Non-Clustering RANSAC		Improved RANSAC (Proposed)	
	Value	Value	Comparison	Value	Comparison
average error (mm)	1.02	1.26	+23.5%	0.0030	−99.7%
standard deviation	0.022	0.026	+18.2%	0.00065	−97.0%
execution time (s)	13.42	104.93	+682%	20.80	+55.0%
correct planes segmented	15/21	13/21	−13.3%	21/21	+40%
incorrect planes segmented	10	9	−10%	0	−100%
total points (downsampled)	85,501	85,501	0%	85,501	0%
inliers	84,677	84,395	−0.33%	81,002	−4.3%

It is to be noted that in Tables 1–3, the total planes to be segmented (displayed as denominators in a row "correct planes segmented") is not equal to "correct planes segmented" plus "incorrect planes segmented". As an example, for a given point cloud with three planes to be segmented, if the segmentation mistakenly has two of the planes recognized as one plane and the left one plane recognized as three planes, the number of incorrect planes reaches four, which is bigger than the total planes to be segmented (three). Unreasonable segmentation may output more planes than needed, as actually happens to Vanilla RANSAC and non-clustering RANSAC in Tables 3 and 4. This phenomenon can be more intuitively observed in Figure 18a,b.

Table 4. Segmentation result comparison with a spatial planar weldment.

Variable	Vanilla RANSAC	Non-Clustering RANSAC		Improved RANSAC (Proposed)	
	Value	Value	Comparison	Value	Comparison
average error (mm)	1.35	0.31	−77.0%	0.0030	−99.8%
standard deviation	0.013	0.0023	−82.3%	0.00065	−95%
execution time (s)	74.77	597.63	+699%	578.42	+674%
correct planes segmented	15/21	16/21	+6.67%	21/21	+40%
incorrect planes segmented	7	0	−100%	0	−100%
total points (downsampled)	394,437	394,437	0%	394,437	0%
inliers	384,093	345,932	−9.94%	354,087	−7.81%

In the final comparison group, one of the spatial planar weldments is utilized for weld seam extraction as the object. Table 4 presents the data comparison of the results. The improved RANSAC successfully captures all the planes required. Conversely, vanilla RANSAC fails to segment nearly one-third of the necessary planes and produces several incorrect planes, as demonstrated in Figure 19.

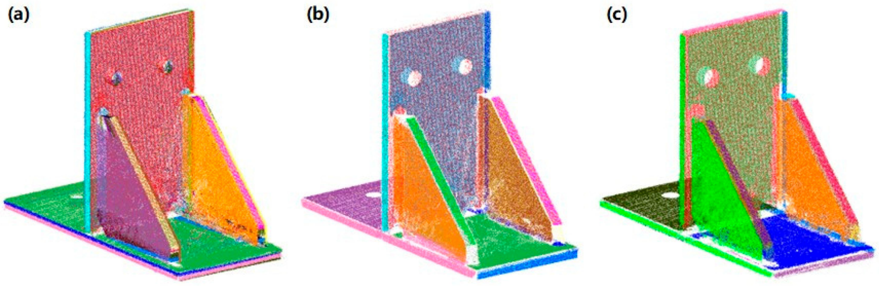

Figure 19. Segmentation comparison over a spatial planar weldment. (**a**) Vanilla RANSAC; (**b**) non-clustering RANSAC; (**c**) improved RANSAC.

By applying the improved RANSAC algorithm, consecutive plane segmentation is performed on the weldments, Figure 20 illustrates the process.

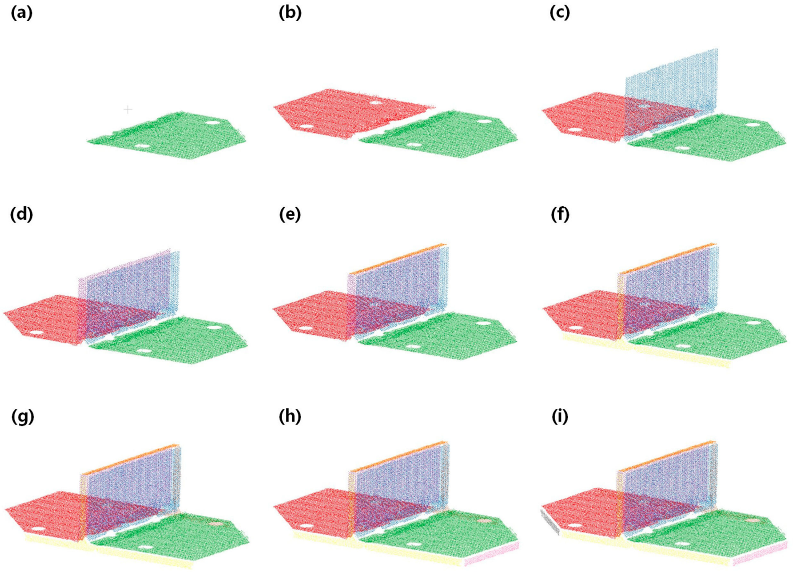

Figure 20. Part of the process of consecutive plane segmentation. One plane is segmented in each stage of (**a**–**i**).

The segmentation results of the weldments are displayed in Figure 21, where photographs of the weldments, segmentation results, and the corresponding normal vector clustering results are arranged for comparison.

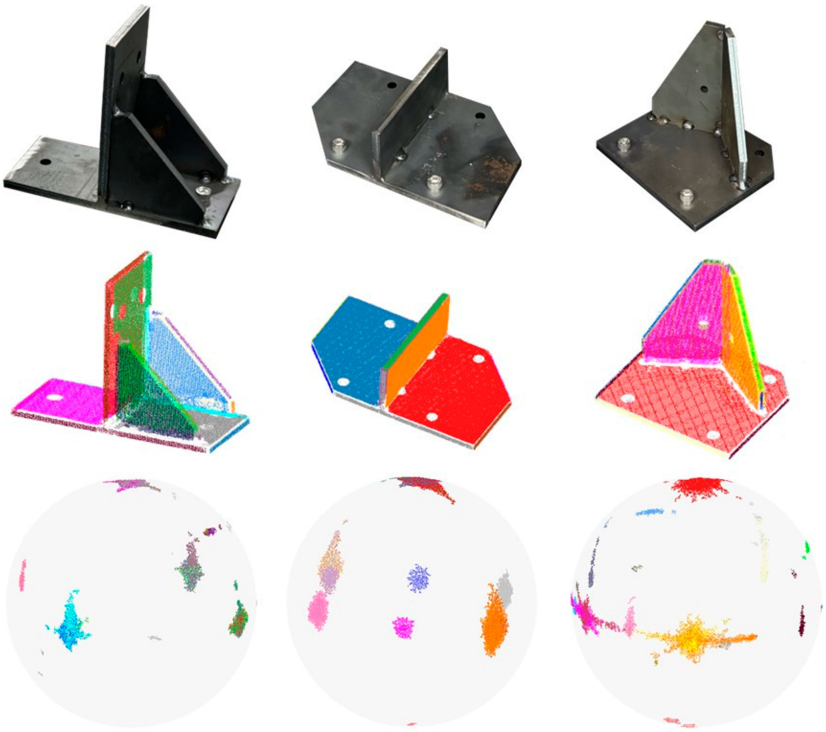

Figure 21. Plane segmentation for several weldments. (**Row one**): photos of the physical weldments. (**Row two**): segmentation results rendering each plane with a different color. (**Row three**): normal vector clustering results of the corresponding segmentations.

4.4. Experiment for Weld Seam Extraction

After obtaining the segmented planes, the calculation of weld seams can be executed using the WSE algorithm proposed in Section 3. The outcomes, comprising valid weld seams and invalid edges excluded based on the edge convexity criterion, are visualized in Figure 22.

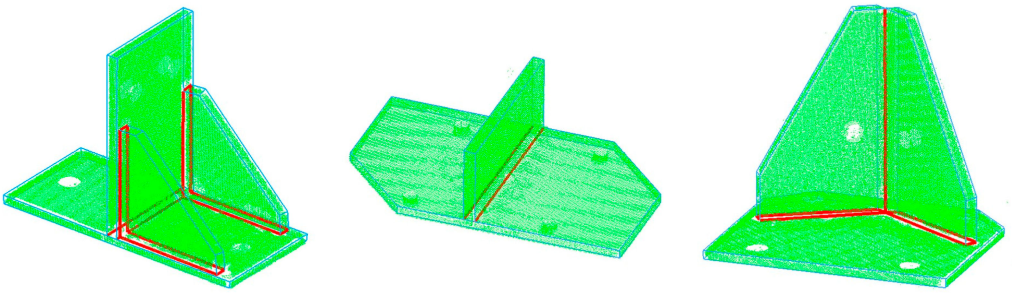

Figure 22. The WSE results: valid weld seams are marked with thick red lines and invalid weld seams are marked with thin blue lines.

4.5. Discussion

The 3D reconstruction for the weldments demonstrates that a point cloud that represents the complete geometric features of a complex spatial planar weldment can be generated by point cloud registration, with the incorporation of a well-calibrated turntable. Notably, this methodology remains effective regardless of the initial position and orientation of the weldments, thus addressing the issue outlined in the introductory section.

Although welding is a relatively low-precision processing process, extracting weld seams from point clouds is still challenging. The reason not only lies in the difficulty of ensuring numerical accuracy when extracting geometric features but also in the fact that conventional feature extraction algorithms such as vanilla RANSAC tend to fail for complex workpieces. Experiments on plane segmentation reveal that conventional RANSAC algorithms exhibit commendable performance and acceptable accuracy solely for objects with rudimentary shapes. When confronted with intricate spatial structures, these algorithms either fail to detect certain planes or return inaccurate results. In contrast, the proposed improved RANSAC algorithm showcases the ability to accurately recognize planes from spatial workpieces, even those characterized by multiple angles and complex intersecting planes.

The improvement of the proposed method over the conventional method varies from case to case. In the experiments conducted in this study, the average errors are reduced by 90.3% to 99.8%, and the standard deviations are reduced by 64.8% to 97.0%. Moreover, thanks to the optimizations introduced in Section 2.2, the improved RANSAC greatly eliminates the problems encountered by conventional RANSAC when dealing with complex workpieces. As shown in the experimental results, the proposed segmentation method ensures the extraction of necessary planes while avoiding the generation of erroneous planes. Since the conventional methods basically fail to generate the weld seams properly, we are unable to provide comparisons on WSE over the proposed method and conventional ones.

5. Conclusions

This paper introduces a global weld seam extraction method tailored for arbitrarily placed spatial planar weldments. The key contributions and outcomes of this research endeavor are summarized as follows:

(1) By introducing point cloud registration with multiple frames of point clouds captured from a weldment, a three-dimensional reconstruction of the weldment is carried out without knowing the position and posture of the weldment. Due to the universality of this method, it may be applied to most scenarios that require a complete 3D digital twin model.

(2) We propose an improved RANSAC algorithm designed to execute consecutive plane segmentation with remarkable accuracy. This algorithm is adept at handling the plane segmentation of complex spatial planar weldments, a task challenging for conventional similar algorithms. As demonstrated by the experiments, the average errors are reduced by 90.3% to 99.8% over the traditional RANSAC algorithm, and the standard deviations are reduced by 64.8% to 97.0%. This method can also be used in other processing techniques that require obtaining planar features through point clouds.
(3) Additionally, we present a weld seam extraction (WSE) algorithm for multiple weld seam extraction, which calculates all the edges from the segmented planes and selects valid weld seams with vertex validation and edge convexity validation utilizing the half-edge data structure.

The proposed methodologies offer an elegant solution for automatically and accurately extracting weld seams from complex weldments without precise fixing. Nevertheless, our approach still has limitations and areas that deserve extension.

(1) This research uses a single-axis turntable to expand the 3D camera's FOV. When the workpiece shape is particularly complex, there could still be geometric features that are difficult to capture. To further achieve the 3D reconstruction capability, we consider introducing multi-axis turntables as a replacement. The calibration of multi-axis turntables will become a research focus.
(2) The WSE method proposed in this paper approximates the weld seams as the intersection lines of adjacent planes in the workpiece. For the case where welding grooves are to be concerned, further microscopic analysis and modeling of the grooves is required before it is used for robot welding path planning. Also, due to the poor performance of RANSAC (even the improved version proposed in the paper) in identifying narrow and long planes, this method only applies to plate structures with a thickness of more than 2 mm.
(3) To improve the method's scope of application, our future investigations will center on weldments featuring free surfaces.

Author Contributions: Conceptualization, B.Y. and J.F.; Data curation, B.Y. and Z.W.; Formal analysis, B.Y.; Funding acquisition, J.F.; Investigation, B.Y. and Z.W.; Methodology, B.Y.; Project administration, J.F.; Resources, J.F.; Software, B.Y.; Supervision, J.F.; Validation, Y.X. and S.H.; Visualization, B.Y.; Writing—original draft, B.Y.; Writing—review and editing, Y.X. and S.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by the Key Research and Development Plan of Zhejiang Province (grant number 2022C01015).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

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

References

1. Johan, N.F.; Mohd Shah, H.N.; Sulaiman, M.; Naji, O.A.A.M.; Arshad, M.A. Weld seam feature point extraction using laser and vision sensor. *Int. J. Adv. Manuf. Technol.* **2023**, *127*, 5155–5170. [CrossRef]
2. Muhammad, J.; Altun, H.; Abo-Serie, E. A robust butt welding seam finding technique for intelligent robotic welding system using active laser vision. *Int. J. Adv. Manuf. Technol.* **2018**, *94*, 13–29. [CrossRef]
3. Guo, F.; Zheng, W.; Lian, G.; Yao, M. A V-shaped weld seam measuring system for large workpieces based on image recognition. *Int. J. Adv. Manuf. Technol.* **2023**, *124*, 229–243. [CrossRef]
4. Chen, X.; Ma, Q.; He, Z.; Sun, X.; Ren, Y. Real-time detection and localization method for weld seam of narrow butt joint based on semantic segmentation. *Meas. Sci. Technol.* **2023**, *35*, 035205. [CrossRef]
5. Wang, H.; Chen, C.; Liu, Y.; Ren, B.Q.; Zhang, Y.W.; Zhao, X.H.; Chi, Y.P. A novel approach for robotic welding trajectory recognition based on pseudo-binocular stereo vision. *Opt. Laser Technol.* **2024**, *174*, 18. [CrossRef]

6. Yuankai, Z.; Yong, J.; Xincheng, T.; Xiaolong, X.; Yusen, G.; Min, L. A point cloud-based welding trajectory planning method for plane welds. *Int. J. Adv. Manuf. Technol.* **2023**, *125*, 1645–1659. [CrossRef]
7. Yang, L.; Li, E.; Long, T.; Fan, J.; Liang, Z. A novel 3-D path extraction method for arc welding robot based on stereo structured light sensor. *IEEE Sens. J.* **2018**, *19*, 763–773. [CrossRef]
8. Lu, Z.; Fan, J.; Hou, Z.; Deng, S.; Zhou, C.; Jing, F. Automatic 3D seam extraction method for welding robot based on monocular structured light. *IEEE Sens. J.* **2021**, *21*, 16359–16370. [CrossRef]
9. Zhang, Y.K.; Geng, Y.S.; Tian, X.C.; Zheng, F.Q.; Jiang, Y.; Lai, M. A Feature Extraction Approach Over Workpiece Point Clouds for Robotic Welding. *IEEE Trans. Autom. Sci. Eng.* **2024**, 1–10. [CrossRef]
10. Wang, H.; Huang, Y.; Zhang, G.; Rong, Y. A novel method for dense point cloud reconstruction and weld seam detection for tubesheet welding robot. *Opt. Laser Technol.* **2023**, *163*, 109346. [CrossRef]
11. Yan, M.; Zhang, K.; Liu, D.; Yang, H.; Li, Z. Autonomous programming and adaptive filling of lap joint based on three-dimensional welding-seam model by laser scanning. *J. Manuf. Process.* **2020**, *53*, 396–405. [CrossRef]
12. Suszyński, M.; Peta, K.; Černohlávek, V.; Svoboda, M. Mechanical Assembly Sequence Determination Using Artificial Neural Networks Based on Selected DFA Rating Factors. *Symmetry* **2022**, *14*, 1013. [CrossRef]
13. Yang, L.; Liu, Y.; Peng, J.; Liang, Z. A novel system for off-line 3D seam extraction and path planning based on point cloud segmentation for arc welding robot. *Robot. Comput.-Integr. Manuf.* **2020**, *64*, 101929. [CrossRef]
14. Geng, Y.; Lai, M.; Tian, X.; Xu, X.; Jiang, Y.; Zhang, Y. A novel seam extraction and path planning method for robotic welding of medium-thickness plate structural parts based on 3D vision. *Robot. Comput.-Integr. Manuf.* **2023**, *79*, 102433. [CrossRef]
15. Bahraini, M.S.; Bozorg, M.; Rad, A.B. SLAM in dynamic environments via ML-RANSAC. *Mechatronics* **2018**, *49*, 105–118. [CrossRef]
16. Giulietti, N.; Allevi, G.; Castellini, P.; Garinei, A.; Martarelli, M. Rivers' Water Level Assessment Using UAV Photogrammetry and RANSAC Method and the Analysis of Sensitivity to Uncertainty Sources. *Sensors* **2022**, *22*, 5319. [CrossRef]
17. El-Sayed, E.; Abdel-Kader, R.F.; Nashaat, H.; Marei, M. Plane detection in 3D point cloud using octree-balanced density down-sampling and iterative adaptive plane extraction. *IET Image Process.* **2018**, *12*, 1595–1605. [CrossRef]
18. Xing, Y.; Lv, C.; Chen, L.; Wang, H.; Wang, H.; Cao, D.; Velenis, E.; Wang, F.Y. Advances in Vision-Based Lane Detection: Algorithms, Integration, Assessment, and Perspectives on ACP-Based Parallel Vision. *IEEE/CAA J. Autom. Sin.* **2018**, *5*, 645–661. [CrossRef]
19. Hast, A.; Nysjö, J.; Marchetti, A. Optimal RANSAC-towards a repeatable algorithm for finding the optimal set. *J. WSCG* **2013**, *21*, 21–30.
20. Derpanis, K.G. Overview of the RANSAC Algorithm. *Image Rochester NY* **2010**, *4*, 2–3.
21. Berkmann, J.; Caelli, T. Computation of surface geometry and segmentation using covariance techniques. *IEEE Trans. Pattern Anal. Mach. Intell.* **1994**, *16*, 1114–1116. [CrossRef]
22. Rusu, R.B. Semantic 3D object maps for everyday manipulation in human living environments. *KI-Künstliche Intell.* **2010**, *24*, 345–348. [CrossRef]
23. Muller, D.E.; Preparata, F.P. Finding the intersection of two convex polyhedra. *Theor. Comput. Sci.* **1978**, *7*, 217–236. [CrossRef]
24. Yang, B.; Wang, Z.; Xu, Y.; Hu, S.; Fu, J. An SVD-based turntable calibration method with multiple reference planes segmented from point clouds. *Meas. Sci. Technol.* **2023**, *35*, 015002. [CrossRef]
25. Rusu, R.B.; Cousins, S. 3D is here: Point cloud library (PCL). In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1–4.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.