PLOS ONE

RESEARCH ARTICLE

Automatic point cloud registration algorithm based on the feature histogram of local surface

Jun Lu[1,2]*, **Zhuo Wang**[1,2], **Bowen Hua**[1,2], **Kun Chen**[1,2]

1 College of Automation, Harbin Engineering University, Harbin, China, **2** Key laboratory of Intelligent Technology and Application of Marine Equipment, Harbin Engineering University, Ministry of Education, Harbin, China

* lujun0260@sina.com

Abstract

In this paper, we present an efficient algorithm for point cloud registration in presence of low overlap rate and high noise. The proposed registration method mainly includes four parts: the loop voxel filtering, the curvature-based key point selection, the robust geometric descriptor, and the determining and optimization of correspondences based on key point spatial relationship. The loop voxel filtering filters point clouds to a specified resolution. We propose a key point selection algorithm which has a better anti-noise and fast ability. The feature descriptor of key points is highly exclusive which is based on the geometric relationship between the neighborhood points and the center of gravity of the neighborhood. The correspondences in the pair of two point clouds are determined according to the combined features of key points. Finally, the singular value decomposition and ICP algorithm are applied to align two point clouds. The proposed registration method can accurately and quickly register point clouds of different resolutions in noisy situations. We validate our proposal by presenting a quantitative experimental comparison with state-of-the-art methods. Experimental results show that the proposed point cloud registration algorithm has faster calculation speed, higher registration accuracy, and better anti-noise performance.

Citation: Lu J, Wang Z, Hua B, Chen K (2020) Automatic point cloud registration algorithm based on the feature histogram of local surface. PLoS ONE 15(9): e0238802. https://doi.org/10.1371/journal.pone.0238802

Editor: Qichun Zhang, University of Bradford, UNITED KINGDOM

Received: December 26, 2019

Accepted: August 23, 2020

Published: September 11, 2020

Data Availability Statement: All relevant data are within the manuscript and its Supporting Information files.

Funding: This research was supported, in part, by the Natural Science Foundation of Heilongjiang Province of China in the form of a grant awarded to JL (No.F201123). No additional external funding was received for this study. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

1. Introduction

With the development of novel sensing technologies, such as Kinect, 3D LiDAR [1, 2] and terrestrial laser scanners (TLS), 3D point cloud becomes more convenient to acquire. And those technologies have been used widespreadly in the fields of 3D reconstruction, archaeology, medical image analysis etc. Point cloud processing has become a research hotspot. To reconstruct a complete 3D model, it is necessary to obtain the point cloud from different viewpoints. But each point cloud is in different coordinate systems. Therefore, point clouds of multi-view in different coordinate systems should be transformed to one coordinate system. This process is called point cloud registration. Point cloud registration is a key step in point cloud processing and has the profound value in computer vision, computer graphics, robotics and so on.

Competing interests: The authors have declared that no competing interests exist.

According to the initial conditions and accuracy, point cloud registration can be divided into coarse registration and fine registration. The coarse registration can quickly estimate a rough transformation matrix without strict requirements of initial spatial positions of point clouds. The fine registration can obtain a good result of registration. There are numerous algorithms for point cloud registration proposed by scholars. Of these algorithms, the Iterative Closest Point (ICP) algorithm is an important registration method for fine registration [3]. The ICP algorithm proposed by Besl et al. can obtain the best transformation matrix according to correspondences iteratively. However, the ICP algorithm also has some shortcomings, such as high requirements for initial positions of point clouds. Chen et al. presented a new approach which works on range data directly and aligns successive scans with enough overlapping area to get an accurate transformation between scans [4].

Ji et al. proposed a hybrid algorithm which integrated the GA algorithm and the ICP algorithm [5]. In the literature [6], Zhu et al. deployed an improved Iterative Closest Point (ICP) algorithm in which an adaptive threshold was introduced to accelerate iterative convergence. Meng combined kd-tree and extrapolation to improve the speed and accuracy of the ICP algorithm [7]. In order to improve the accuracy of point cloud registration and the convergence speed of registration, Liu et al. took point pairs with smaller Euclidean distances as the points to be registered, and designed the depth measurement error model and weight function [8]. Agamennoni et al. presented a point cloud registration method based on probability distribution which is another type of fine registration [9].

In general, common coarse registration methods are based on local geometric features description, which includes the extraction of geometric features and the determination of correspondences. Many approaches of extracting the feature point have been widely reported. Li proposed an improved Harris algorithm by combining the discrete curvature and the normal vector to extract feature [10]. The SIFT operator can reduce the influence of scale change on key point search, but its computation is complex [11, 12]. In the paper [13], a registration method combining with color moment information improves the registration accuracy. In the literature [14], the future points are obtained via 3D Difference of Gaussians over geometric scalar values of the points which ensures obtaining salient features. Prakhya S M calculated the HoNo (Histogram of Normal Orientations) at every point and detected the key point by evaluating the properties of both the HoNo and the neighborhood covariance matrix [15]. The point feature histogram (PFH) algorithm and the fast point feature histogram (FPFH) algorithm are popular algorithms of feature description [16–18], which generate a feature histogram for each point based on feature information. Prakhya S M et al. applied a binary quantization method on a state-of-the-art 3D feature descriptor [19], SHOT [20], and created a new binary 3D feature descriptor, B-SHOT. Kleppe A L introduced a descriptor of key point using conformal geometric algebra [21]. Instead of feature descriptor's calculating and feature matching, the 4-Points Congruent Sets (4PCS) and semantic-key point based 4PCS (SK-4PCS) determine the corresponding four-point base sets by exploiting the rule of intersection ratios [22, 23]. Mellado et al. improved 4PCS and proposed SUPER 4PCS and speedups the registration process [24]. Another idea of coarse registration is Sample Consensus algorithm. For example, Ye et al. used Random Sample Consensus (RANSAC) algorithm to eliminate the wrong matches [25]. In the literature [26], during coarse registration stage, Random Sample Consensus (RANSAC) algorithm is used to obtain the transformation between two 3D point clouds. The Normal distributions transform (NDT) algorithm is used to solve 2-D registration problem in the paper [27]. And Magnusson applied it in a 3-D space [28]. The NDT algorithm uses statistical probability method to determine the corresponding point pairs according to the normal distribution. Hong et al. proposed a probabilistic normal distributions transform (PNDT) representation which improves the accuracy of point cloud registration by using the probabilities of point samples [29]. Huan Lei et al. present a robust global approach for point cloud registration from uniformly sampled points, based on eigenvalues and normals computed from multiple scales [30].

Different from the above methods, this paper presents a key point selection algorithm which has a better anti-noise and fast ability. The feature descriptor of key points is highly exclusive which is based on the geometric relationship between the neighborhood points and the center of gravity of the neighborhood. We validate our proposal by presenting a quantitative experimental comparison with state-of-the-art methods. Experimental results show that the proposed point cloud registration algorithm has faster calculation speed, higher registration accuracy, and better anti-noise performance.

The rest of the paper is structured as follows. In Section 2, We introduce the principle of the algorithm in detail. In Section 3, the effectiveness of the algorithm is shown by experiment. Section 4 concludes this paper.

2. Point cloud registration based on the feature histogram of local surface

The registration process in our method mainly includes Loop voxel filtering, Finding key points, The Feature Descriptor, Point cloud registration and other parts. The flow chart for the registration process is shown in Fig 1.

2.1. Loop voxel filtering

The resolutions between different point clouds by using different acquisition equipment for different objects have a large difference, which leads that the multiple parameters should be set manually during the registration process. If the point clouds have too many points, the registration time will greatly increase. The point cloud filtering can deal with above problems. Compared to the filtered point cloud, original dense point cloud uses more points to describe the object surface. As shown in Fig 2, in order to describe the same surface, it requires 17 points in the original dense point cloud, but in filtered point only 7 points. Over-filtered point cloud cannot describe the surface correctly as shown in Fig 2(C).

The resolution of a point cloud is the average of the distances between each point and its nearest neighborhood point in the point cloud. The resolution describes the sparsity of point clouds. The greater the resolution, the sparser the point cloud. In order to achieve fast automatic registration, the point cloud resolution should to be calculated first:

$$s = \frac{\sum_{i=1}^{n} \|\boldsymbol{p}_i - \boldsymbol{p}_i'\|_2}{n} \tag{1}$$

where $\boldsymbol{p}_i$ is the i-th point in the point cloud, $\boldsymbol{p}_i'$ is its nearest neighbor point and n is the number of points in the point cloud.

In order to reduce point cloud size, voxel filtering will be used. The three-dimensional voxel grid is created in which each point is represented by the center of gravity of the grid. In this paper, an automatic voxel filtering on the point cloud is designed.

To improve registration efficiency, point clouds are filtered with uniform resolution of 1.0 mm. Automatic loop voxel filtering is adopted which calculates the maximum and minimum values of the x, y and z axis of the input point cloud, and establishes a three-dimensional bounding box according to these values and divides the bounding box into small cubes with the assigned voxel size, and represents all points in the small cube with the center of gravity of

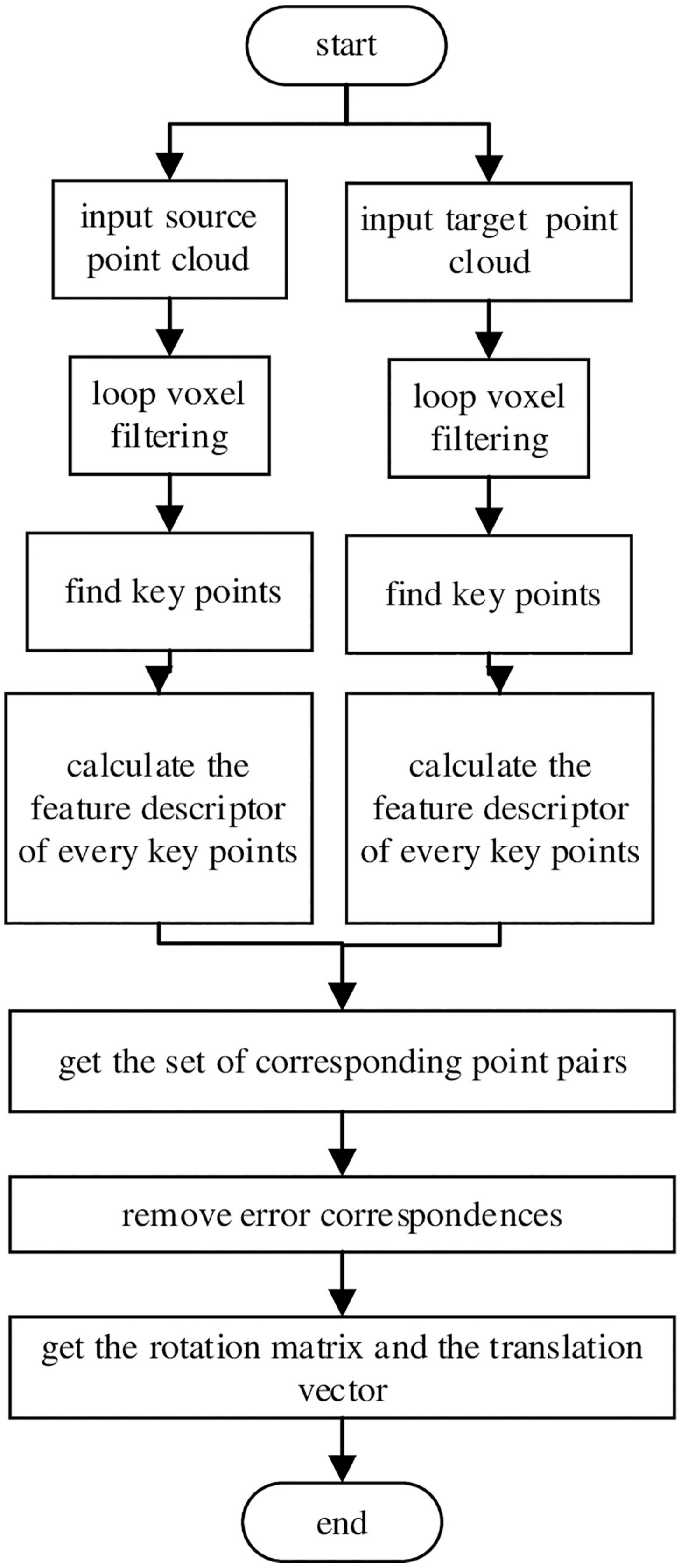

start
input source point cloud
input target point cloud
loop voxel filtering
loop voxel filtering
find key points
find key points
calculate the feature descriptor of every key points
calculate the feature descriptor of every key points
get the set of corresponding point pairs
remove error correspondences
get the rotation matrix and the translation vector
end

Fig 1. Flow chart of registration process.

https://doi.org/10.1371/journal.pone.0238802.g001

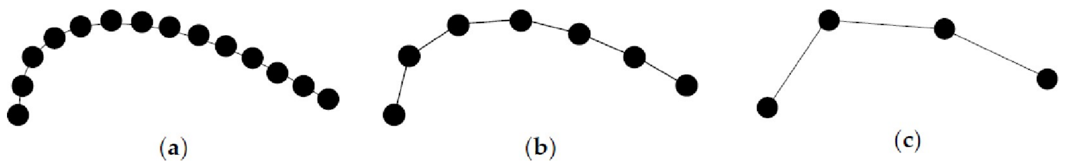


Fig 2. Point clouds of different densities. (a) The original dense point cloud; (b) the filtered point cloud; (c) the over-filtered point cloud.

https://doi.org/10.1371/journal.pone.0238802.g002

the small cube. In this way, multiple points inside the voxel are represented by one point, and the point cloud is reduced.

The filtering algorithm in this paper is implemented as follows:

1. Filter the original point cloud by using voxel filtering, and set the final voxel size, s_{target}, to 1.0 mm.
2. Calculate the resolution of filtered point cloud, s_{now}.
3. If 1.02* s_{now} is greater than s_{target}, end the filtering process, else take voxel filtering on the filtered point cloud again, and the voxel size s_{loop} is determined by:

$$s_{loop} = s_{target} + 0.2 \times (s_{target} - s_{now}) \tag{2}$$

4. Go to step 2.

The filtered point cloud after above steps will be used as the initial point cloud for registration in the following sections, and its resolution is represented by s_n which will be used in the following sections also.

2.2. Finding key points

After the point cloud filtering, the points are still redundant for the registration of point clouds. Most of the points locate at locations where local features are not apparent such as flat region. To improve the speed of registration, the key points are found for the registration. In order to find key points, classic algorithms based on a single point feature is sensitive to noise. In order to strengthen the resistance to noise, a finding algorithm of key point based on the biggest mean curvature of the pre-keypoint in its neighborhood is proposed. The algorithm proposed in this paper has better performance than the algorithm which only relies on single point's curvature value. The key points finding algorithm is shown in Fig 3.

The key points are obtained based on point neighborhood. The neighborhood of a point p_i is defined as the set which includes all points within the sphere with center p_i and the radius r, where $r = 5^* s_n$. s_n is the current point cloud resolution. The covariance matrix E with dimension 3*3 and the eigenvalues λ_1, λ_2, λ_3 based on the neighborhood of p_i are calculated:

$$\boldsymbol{E} = \frac{1}{m}\sum_{a=1}^{m}(\boldsymbol{p}_i^a - \bar{\boldsymbol{p}}_i)\cdot(\boldsymbol{p}_i^a - \bar{\boldsymbol{p}}_i)^T \tag{3}$$

$$\boldsymbol{E}\cdot\boldsymbol{v}_j = \lambda\cdot\boldsymbol{v}_j, j\in\{1,2,3\} \tag{4}$$

where $\boldsymbol{p}_i^a$, a $\in(1, m)$ is the a-th point in the Neighborhood of p_i and m is the number of points in the neighborhood of the point $\boldsymbol{p}_i$. $\bar{\boldsymbol{p}}_i$ is the centroid of the neighborhood of $\boldsymbol{p}_i$. λ_j and $\boldsymbol{v}_j$ are the eigenvalue and eigenvector of the covariance matrix $\boldsymbol{E}$, correspondingly. λ_1 is the smallest

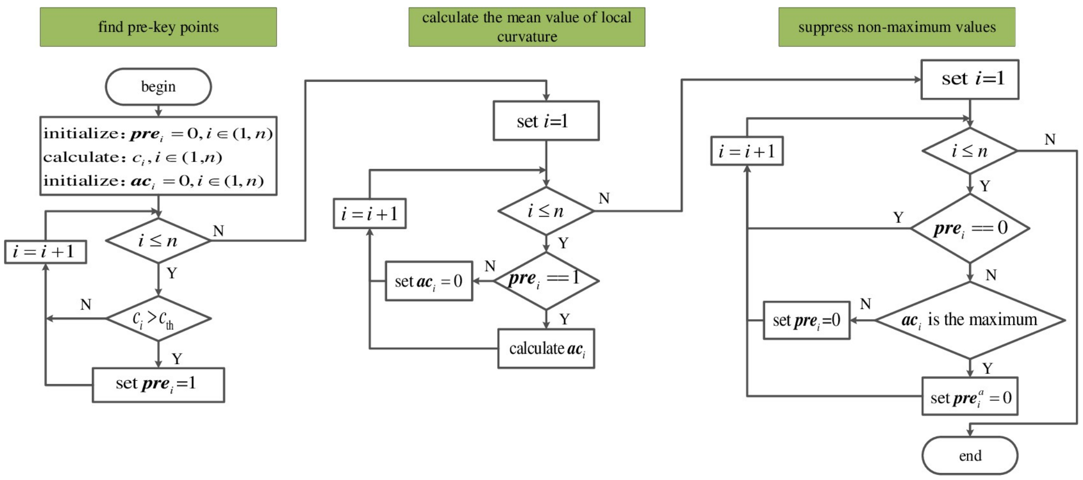


Fig 3. Key points finding algorithm.

https://doi.org/10.1371/journal.pone.0238802.g003

eigenvalue. The curvature can be estimated from the above eigenvalues. The curvature c_i of the point $\boldsymbol{p}_i$ is obtained by the following formula:

$$c_i = \frac{\lambda_1}{\lambda_1 + \lambda_2 + \lambda_3} \tag{5}$$

To speed up searching key points, the points whose curvatures are greater than the threshold c_{th} are chosen as candidate key points. The threshold of curvatures is $c_{th} = c_{max}$—(c_{max}-c_{min})/3, where c_{max} and c_{min} are the maximum and minimal curvatures in the whole cloud points, respectively. An n-dimensional column vector ***pre*** is established to store the flags which indicate whether each point in the point cloud is a candidate key point. Where n is the number of points of the point cloud. The initial value of ***pre*** is an all-zero vector, i.e. $pre_i = 0$, $i \in (1, n)$. It means that the i-th point is not a pre-key point. If the curvature of i-th point is greater than c_{th}, pre_i is set to 1 and make it as a pre-key point.

We use the symbol ac_i to represent the mean value of curvature of all points in the neighborhood of the i^{th} point. The neighborhood radius is r. If the point $\boldsymbol{p}_i$ is not a pre-key point, its curvature mean value is set to 0, $ac_i = 0$. If the point $\boldsymbol{p}_i$ is a pre-key point, the mean curvature of its neighborhood ac_i would be calculated:

$$ac_i = \frac{\sum_{a=1}^{m} c_i^a}{m} \tag{6}$$

where c_i^a is the curvature of the point $\boldsymbol{p}_i^a$ and m is the number of neighborhood points. The neighborhood point is denoted by $\boldsymbol{p}_i^a$.

In the process of determining whether the pre-key point has the largest mean value of the curvature, the point's pre-key point flag pre_i is set to 0 when its curvature mean ac_i is less than its surrounding points' value. It can reduce the number of pre-key points and accelerate the calculation of curvature mean. The mean value of the curvature of the pre-key point would be compared with those of its neighbor points. If the mean value of the curvature of the pre-key point is larger than all its neighbor points, the pre-key point flag would be set to 1 and the pre-key point flag of its neighbor points would be set to 0. After above procedure, the points whose pre-key point flag are still 1 are taken as the final key points p_k.

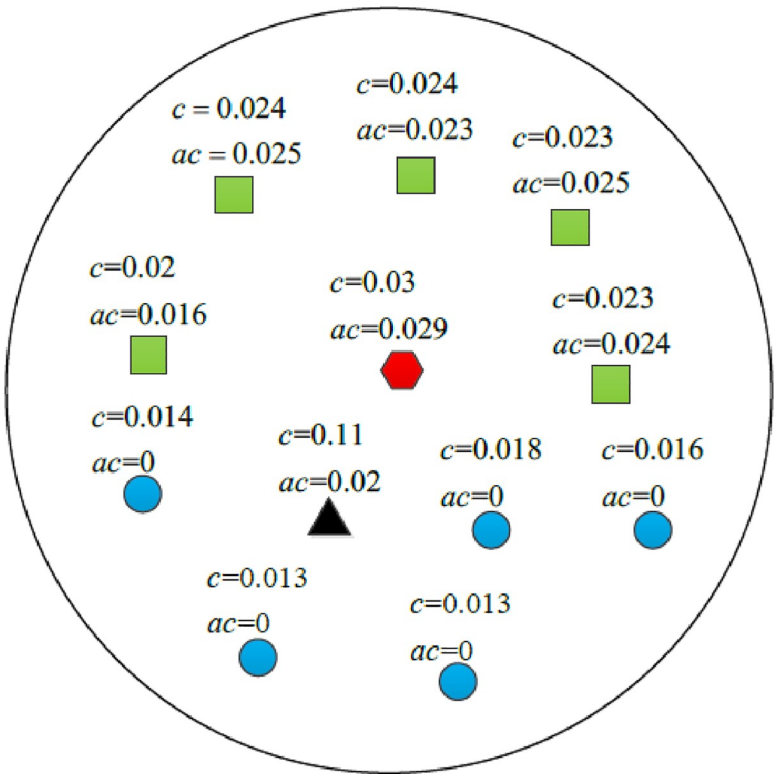


Fig 4. Anti-noise principle of finding algorithm of key points.

https://doi.org/10.1371/journal.pone.0238802.g004

The anti-noise principle of the finding algorithm of key point is shown in Fig 4, where $c_{th} = 0.02$. The curvature of the circular points are less than 0.02, which indicates these points are normal. Since curvature of the square points are greater than 0.02, they are pre-key points. The curvature of the triangle point is abnormal, it is a noise point. The mean curvature of the neighborhood of the hexagonal point is the maximum in its neighborhood, which is the key point. Due to the abnormal curvature of the noise point, the noise point would be mistaken as a key point according to its curvature only. Selecting the key point according to the mean of the neighborhood curvature can improve the anti-noise ability of the key point search algorithm. Although the curvature of the noise point is large, the mean of the neighborhood curvature of the neighborhood points are increased, the red hexagonal point can still be correctly selected as the key point.

2.3. The feature descriptor

The classical feature descriptor depends on the relationship between the key point and its neighborhood points. Due to abnormal information such as the normal of the noise point, when noise is mistaken as a key point, feature descriptors cannot correctly describe the geometric features of key point based on its neighbor information. For this reason, we propose a feature descriptor in which the local surface histogram is calculated according to the distance between the neighbor points and the gravity center of neighborhood of the key point, as well as normal of points in the neighborhood.

The radius of the neighborhood of the key point $\boldsymbol{p}_k$ is denoted by r. The center of gravity $\bar{\boldsymbol{p}}_k$ of the neighborhood is shown in Fig 5. The d_a is the distance from the neighborhood point $\boldsymbol{p}_k^a$

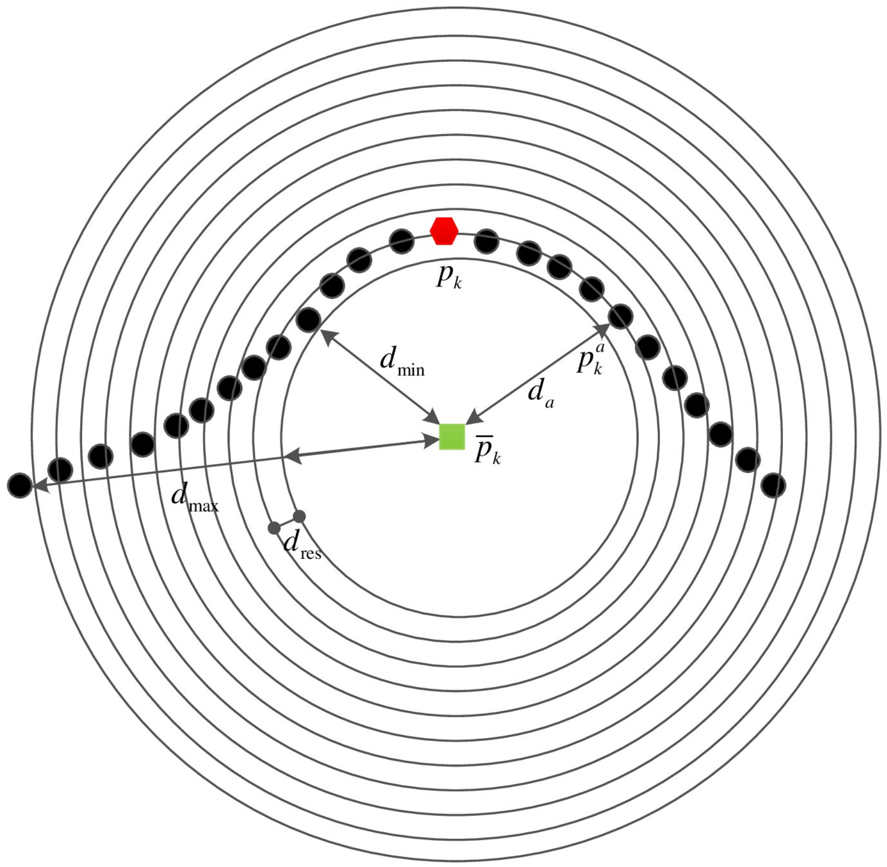


Fig 5. Grouping schematic of distances.

https://doi.org/10.1371/journal.pone.0238802.g005

to the center of gravity $\bar{\boldsymbol{p}}_k$. The nearest distance is $d_{\min}$ and farthest distance is $d_{\max}$. The $d_{\max}$-$d_{\min}$ is divided into 10 parts.

The length of each part d_{res} is:

$$d_{res} = \frac{d_{\max} - d_{\min}}{10} \tag{7}$$

For the neighborhood point $\boldsymbol{p}_k^a$, according to the distance d_a, $bin_d^a \in (1, 10)$ is computed as:

$$bin_d^a = \left\lceil \frac{d_a - d_{\min}}{d_{res}} \right\rceil \tag{8}$$

where $\lceil\rceil$ means to round up to an integer.

The $c_a \in$(-1,1) is cosine of the angle between the normal of the neighborhood point $\boldsymbol{p}_k^a$ and the line from $\boldsymbol{p}_k^a$ to the center of gravity $\bar{\boldsymbol{p}}_k$, as shown in Fig 6. Where the hexagonal point $\boldsymbol{p}_k$ is key point. The cosine value c_a is averagely divided into 12 parts.

Each part c_{res} is calculated as:

$$c_{res} = \frac{(+1) - (-1)}{12} \tag{9}$$

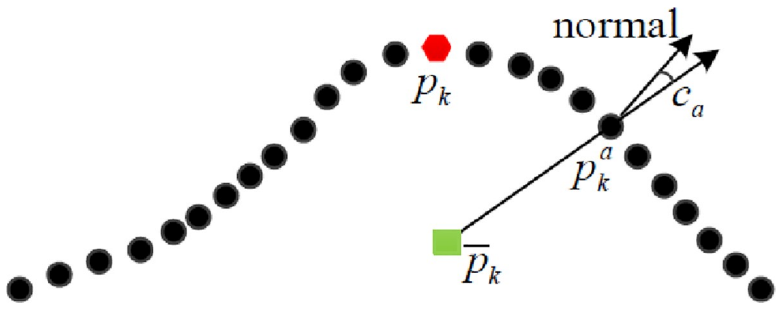


Fig 6. Schematic diagram of grouping cosine value.

https://doi.org/10.1371/journal.pone.0238802.g006

For the neighbor point $\boldsymbol{p}_k^a$, its cosine value group number, $bin_c^a \in (1,12)$ is calculated as

$$bin_c^a = \left\lceil \frac{c_a + 1}{c_{res}} \right\rceil \tag{10}$$

The feature descriptor is calculated according to the geometric relationship between the center of gravity and neighbor points in the neighborhood of the key point. So the effect, when a noise is mistakenly chosen as a key point, can be reduced.

The anti-noise principle of the feature descriptors is shown in Figs 7 and 8, respectively. Where the hexagonal point $\boldsymbol{p}_k$ is the true key point and the square point $\bar{\boldsymbol{p}}_k$ is the gravity center of its neighborhood. The triangle point $\boldsymbol{p}_k'$ is a noise point and the diamond point $\bar{\boldsymbol{p}}_k'$ is the center of gravity of the neighborhood when the noise point $\boldsymbol{p}_k'$ is mistakenly chosen as a key point. As it can be seen from the figures, when the noise point is mistaken as a key point, the distinction between the two centers of gravity is small. At the same time, for the neighbor point $\boldsymbol{p}_k^a$, the values (bin_c^a, bin_d^a) calculated by the wrong center of gravity also has small difference. The feature descriptor based on the noise point $\boldsymbol{p}_k'$ can still correctly describes the neighborhood.

A two-dimensional array $f_{12\times10}$ is used to store neighborhood information of the key point with 12×10 zeroes as initial values.

According to the values of bin_c and bin_d of neighbor point $\boldsymbol{p}_k^a$, the value in the corresponding position of the 2D array $f_{12\times10}$ are added by one. As shown in Fig 9, the $[bin_c, bin_d]$ of the neighbor point $\boldsymbol{p}_k^a$ is [2, 3]. So the value in the position [2, 3] of the feature descriptor $f_{12\times10}$ of the key point $\boldsymbol{p}_k$ are added by one.

To normalize the value in each position of the two-dimensional array $f_{12\times10}$, it is divide by the number of neighborhood points. After all points in the neighborhood of the key point are traversed, the two-dimensional array $f_{12\times10}$ is obtained and it is flattened to a column vector $\boldsymbol{f}$ of 120 rows. The column vector $\boldsymbol{f}$ is used as the feature descriptor for the key point p_k.

2.4. Point cloud registration

The correspondences are determined based on the Euclidean distances of the descriptors of key points. The feature vector of key point p_k^s of the source point cloud is represented by symbol $\boldsymbol{f}_k^s$ and the feature vector of key point p_k^t of the target point cloud is $\boldsymbol{f}_k^t$:

$$\boldsymbol{f}_k^s = [f_{k1}^s \quad f_{k2}^s \quad \cdots \quad f_{k120}^s]^T \tag{11}$$

$$\boldsymbol{f}_k^t = [f_{k1}^t \quad f_{k2}^t \quad \cdots \quad f_{k120}^t]^T \tag{12}$$

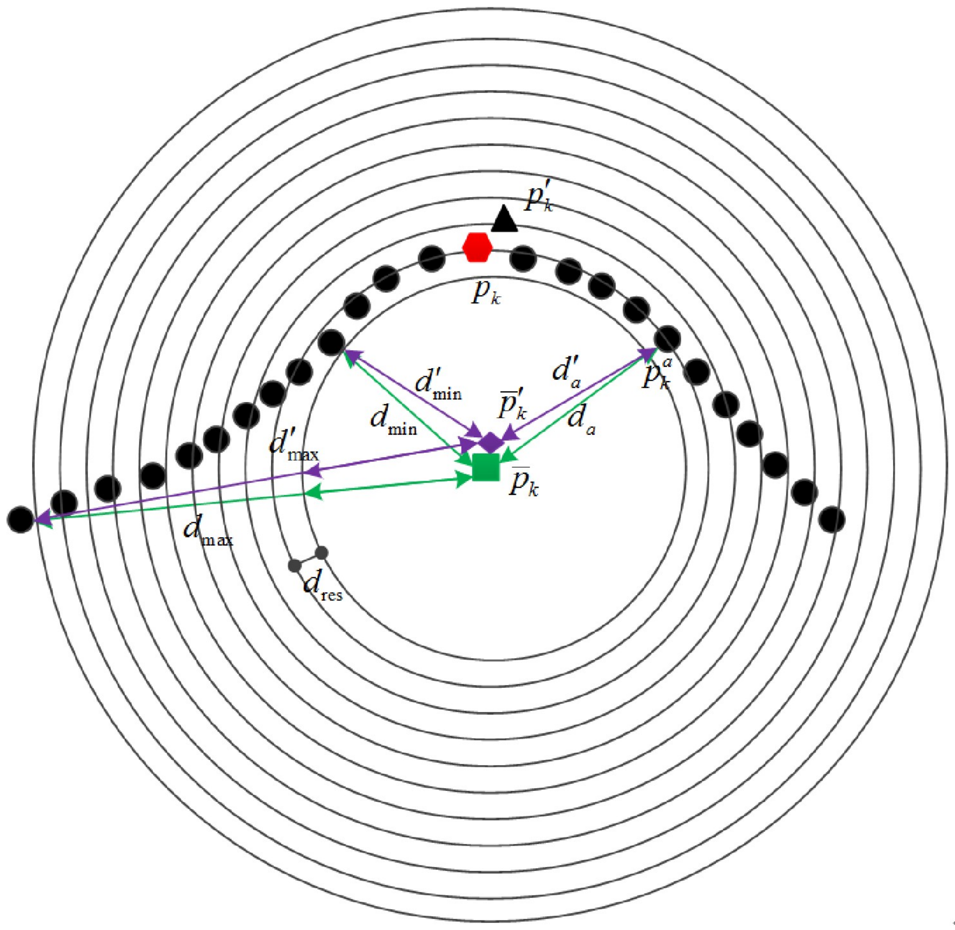


Fig 7. Anti-noise principle of distance grouping.

https://doi.org/10.1371/journal.pone.0238802.g007

The Euclidean distance between the feature vectors $\boldsymbol{f}_k^s$ and $\boldsymbol{f}_k^t$ is calculated by:

$$d(\boldsymbol{f}_k^s, \boldsymbol{f}_k^t) = \sqrt{\sum_{j=1}^{120} (f_{kj}^s - f_{kj}^t)^2} \tag{13}$$

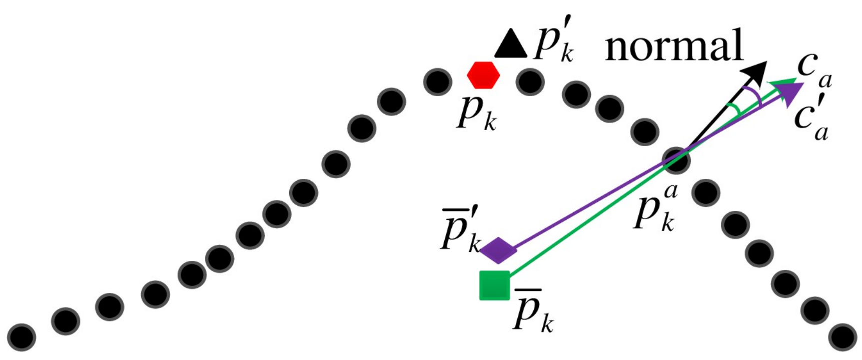
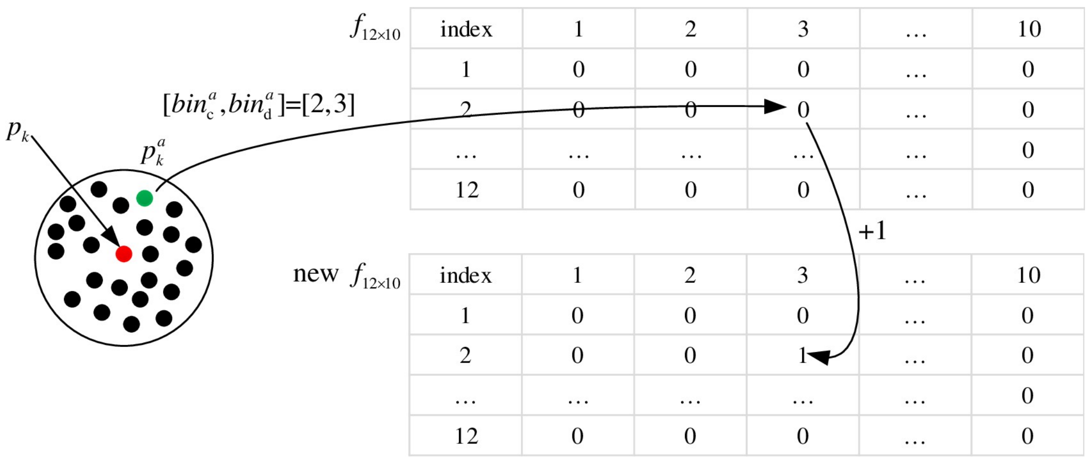


Fig 8. Anti-noise principle of cosine value grouping.

https://doi.org/10.1371/journal.pone.0238802.g008



index	1	2	3	…	10
1	0	0	0	…	0
2	0	0	0	…	0
…	…	…	…	…	0
12	0	0	0	…	0

index	1	2	3	…	10
1	0	0	0	…	0
2	0	0	1	…	0
…	…	…	…	…	0
12	0	0	0	…	0

Fig 9. Calculation principle of the feature descriptor.

https://doi.org/10.1371/journal.pone.0238802.g009

Since the feature descriptor has been normalized already, the average value of each dimension of the 120-dimensional feature descriptor, $\boldsymbol{f}_{\text{avg}}$, is:

$$f_{avg} = \frac{1}{120} \tag{14}$$

When the difference between the key point feature descriptors of source point cloud and target point cloud is less than 0.5* $\boldsymbol{f}_{\text{avg}}$, the mean square error *mse* satisfies:

$$mse < 120 \times (0.5 \times f_{avg})^2 \approx 0.002 \tag{15}$$

The kd-tree based on the descriptors of key points in source point cloud is generated. The closest key point in the target point cloud is searched in the generated kd-tree. If the mean square error of key point in the target point cloud is less than 0.002, the corresponding point pair will be added to the initial correspondence set *O*.

Because the feature descriptor describes the neighborhood information of the key point, if the neighborhoods of different key points are similar, some incorrect initial correspondences would be generated. In order to remove the incorrect correspondences, the neighborhood composite feature of the initial matching point pair is proposed in this paper.

Using the information of the Euclidean distance and feature descriptor of the nearest key point as a combined feature, the incorrect correspondence relationship is discarded according to the combined features. The nearest neighbor point $\boldsymbol{p}_k^{son}$ for the key point $\boldsymbol{p}_k^{so}$ in source point cloud is found, as shown in Fig 10. The d_k^{son} is the distance between the point $\boldsymbol{p}_k^{son}$ and $\boldsymbol{p}_k^{so}$. The mean value of the descriptors of the point $\boldsymbol{p}_k^{so}$ and the point $\boldsymbol{p}_k^{son}$ is taken as the neighborhood composite feature $\boldsymbol{f}_k^{so}$ of the point $\boldsymbol{p}_k^{so}$. The nearest neighbor $\boldsymbol{p}_k^{ton}$ for the key points $\boldsymbol{p}_k^{to}$ in the target point cloud is found. The d_k^{ton} is the distance between the points $\boldsymbol{p}_k^{ton}$ and $\boldsymbol{p}_k^{to}$. The mean value of the descriptors of the point $\boldsymbol{p}_k^{to}$ and the point $\boldsymbol{p}_k^{ton}$ is taken as the neighborhood composite feature $\boldsymbol{f}_k^{to}$ of the point $\boldsymbol{p}_k^{to}$. If the absolute value of the difference between d_k^{son} and d_k^{ton} is greater than 10 times resolution of source point cloud, the correspondence will be discarded. Otherwise, if the Euclidean distance between the vectors $\boldsymbol{f}_k^{so}$ and $\boldsymbol{f}_k^{to}$ is greater than 0.002, the correspondence is discarded. After above procedures, the final correspondence set is obtained. As shown in Fig 10, solid lines represent mismatches and dashed lines represent correct matches. Although the mean square error of feature descriptors of points $\boldsymbol{p}_k^{so}$ and $\boldsymbol{p}_k^{to}$ is small, the closest points distance d_k^{son} and d_k^{ton} are quite different. And the neighborhood composite

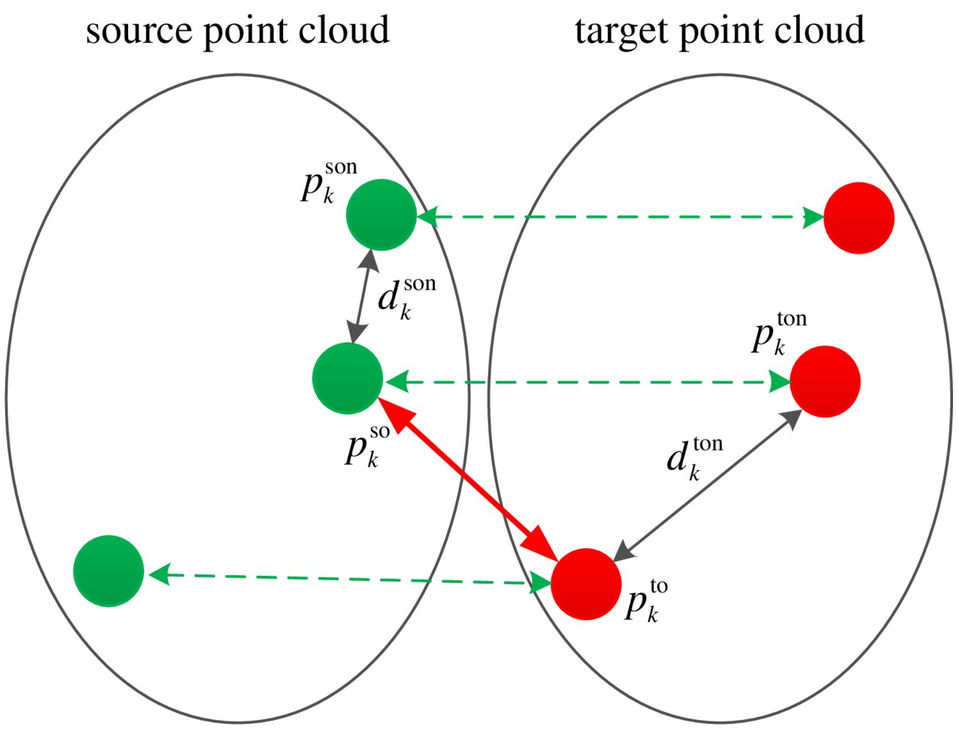


Fig 10. The removal principle of error correspondences.

https://doi.org/10.1371/journal.pone.0238802.g010

features $\boldsymbol{f}_k^{so}$ and $\boldsymbol{f}_k^{to}$ are quite different. The incorrect correspondence can be effectively removed by comparing the closest point distance of the initial corresponding point pair with the neighborhood composite feature.

According to the final correspondence set, the rotation matrix and the translation vector between source point cloud and target point cloud are calculated by using the SVD algorithm and coarse registration is completed. Then the fine registration is finished by using ICP algorithm.

3. Experiment

The initial positions of the point clouds are shown in Fig 11. The cheff_source, dragon_source, armadillo_source, happy_source and boy_source are source point clouds represented by green color. The cheff_target, dragon_target, armadillo_target, happy_target and boy_target are target point clouds represented by blue color.

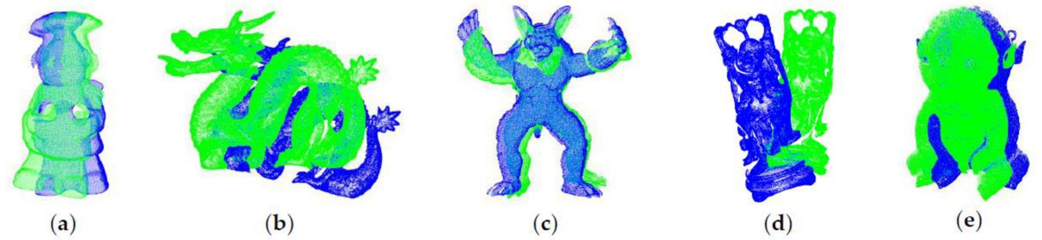


Fig 11. Initial positions of point clouds. (a) Cheff; (b) Dragon; (c) Armadill; (d) Happy; (e) Boy.

https://doi.org/10.1371/journal.pone.0238802.g011

Table 1. Number of points in the process of loop voxel filtering.

Counts of iteration	dragon_source		dragon_target		armadillo_ source		armadillo_ target	
	point number	resolution (mm)	point number	resolution (mm)	point number	resolution (mm)	point number	resolution (mm)
0	100250	0.221882	43572	0.218354	172974	0.457438	83636	0.453643
1	59515	0.745332	26330	0.745497	54787	0.717815	27926	0.708482
2	45820	0.864262	20347	0.861433	39987	0.859273	20350	0.847701
3	40086	0.935212	17873	0.933344	33827	0.934583	17362	0.926101
4	36500	0.989653	16370	0.984749	30572	0.985521	15567	0.982849
Consuming time (ms)	218		94		203		109	

https://doi.org/10.1371/journal.pone.0238802.t001

The dense point clouds can be simplified to a specified resolution by using proposed algorithm in this paper. Table 1 shows the resolutions of point cloud during loop filtering where the specified resolution is 1mm. After several loops of voxel filtering, the point cloud can be automatically adjusted to the specified resolution. So the registration can deal with point clouds obtained by different scanners from different distance automatically, eliminating manual turning registration parameters which are based on point clouds with different sizes. After filtering, the resolution of source point cloud is almost the same as the target cloud with the error about 1.2%.

The key point distributions of point cloud dragon_source under Gaussian noise with variance $\sigma^* s_n$ are shown in Fig 12. It can be seen that the key points obtained by the proposed algorithm based on the mean curvature of points in its neighborhood are more evenly distributed. The key points are located in the surface where the curvature changes greatly. After adding Gaussian noise, the key points are found in same place almost for different σ values that means the obtained key points are robust to noise.

The registration error is defined as the average distance between corresponding point pairs in the source point cloud and the target point cloud after registration. The smaller the registration error, the better the registration result. Table 2 shows that the registration algorithm proposed in this paper is fast and has high registration accuracy. When the noise is small, the registration accuracy of these registration algorithms is similar. With the increase of noise, the registration algorithm proposed in this paper is more accurate than PFH, FPFH and SHOT.

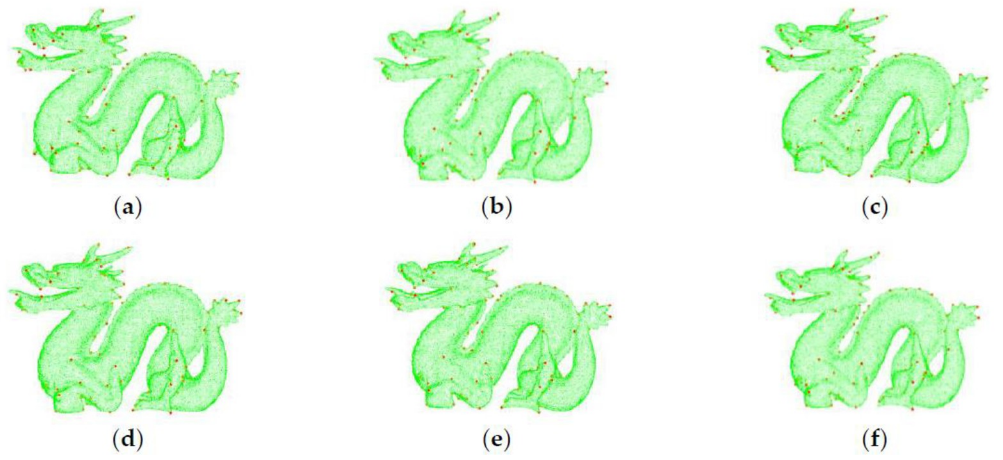


Fig 12. Key point distribution under noises. (a) $\sigma = 0$; (b) $\sigma = 0.1$; (c) $\sigma = 0.2$; (d) $\sigma = 0.3$; (e) $\sigma = 0.4$; (d) $\sigma = 0.5$.

https://doi.org/10.1371/journal.pone.0238802.g012

Table 3 shows the calculation time of feature descriptors for different radiuses. As the radius of the neighborhood increases, the calculation time of PFH and FPFH grows faster, while the calculation time of SHOT and the algorithm proposed in this paper grows slower.

When the same accuracy is achieved, the correspondence optimizing algorithm proposed in this paper has a shorter time than RANSAC, as shown in Table 4.

The descriptors of two correct corresponding points obtained by using the PFH, FPFH, SHOT and our method are roughly similar. Figs 13–17 shows the wrong corresponding points and their feature descriptors obtained by these methods. The left side is the source point cloud, the right side is the target point cloud, and the big point is the corresponding point. Because the local features are similar, PFH, FPFH, and SHOT cannot distinguish the wrong corresponding points, but the feature descriptor proposed in this paper can still distinguish subtle differences. The feature descriptor proposed in this paper has better distinct ability with fewer dimensions.

As shown in Fig 18, there are many error correspondences through first matching. By using our method to remove the error correspondences, the final correspondences are basically correct.

The Table 5 is parameters and results of registration used by the algorithm proposed in this paper. The registration process is implemented automatically without human intervention.

Table 2. Registration error under noises.

Noises	Registration error of cheff (mm)				Registration error of dragon (mm)			
	PFH	FPFH	SHOT	our method	PFH	FPFH	SHOT	our method
0	0.224574	0.273543	0.213564	0.192364	0.163234	0.152315	0.125342	0.134274
0.1	0.374634	0.425436	0.323743	0.204072	0.214734	0.225462	0.152362	0.147342
0.2	0.574965	0.567244	0.434583	0.321691	0.324374	0.337345	0.274245	0.252324
0.3	0.824546	0.765364	0.534296	0.384844	0.454536	0.473745	0.394554	0.355267
0.4	1.011268	0.950696	0.845454	0.515503	0.573454	0.601235	0.523512	0.512315
0.5	1.157674	1.157645	1.012543	0.634767	0.734572	0.953742	0.585596	0.561351
Time (ms)	6231	3257	2526	1423	4231	3021	2834	1637

https://doi.org/10.1371/journal.pone.0238802.t002

Table 3. The comparison of time of calculating feature descriptors.

Neighborhood radius (mm)	Time spent on calculating 300 feature descriptors (ms)			
	PFH	FPFH	SHOT	Our method
10	1346	970	186	124
20	14500	6406	250	164
30	76504	14312	410	228
40	240594	24906	532	376
50	556628	37532	906	504

https://doi.org/10.1371/journal.pone.0238802.t003

Table 4. Results of RANSAC and proposed algorithm for removing the error correspondence.

Algorithm	Consuming time (ms)		Registration error (mm)	
	cheff	dragon	cheff	dragon
Our method	332	316	0.141291	0.134632
RANSAC	1109	938	0.149769	0.132017

https://doi.org/10.1371/journal.pone.0238802.t004

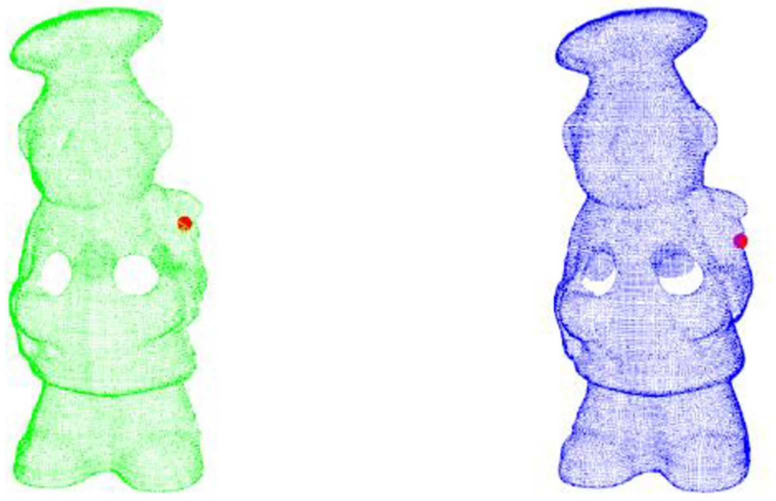


Fig 13. Wrong corresponding points in cheff.

https://doi.org/10.1371/journal.pone.0238802.g013

Table 6 shows the results of registration by tuning algorithm's parameters manually. The filtering algorithm is voxel filtering. The voxel size is determined by multiple trials. The finding key point algorithm is based on uniform sampling. The sampling interval is 20 times the voxel size. The feature descriptor SHOT is adopted. The fault correspondences are removed by the RANSAC algorithm. Finally, ICP fine registration is performed.

Compared with the manually turning parameter algorithm, the filtering algorithm of this paper can automatically adjust the parameters according to the point cloud resolution. Despite the merely increased time, it is suitable for registration without manual intervention. Fig 19 shows the registration results of our algorithm with high registration accuracy.

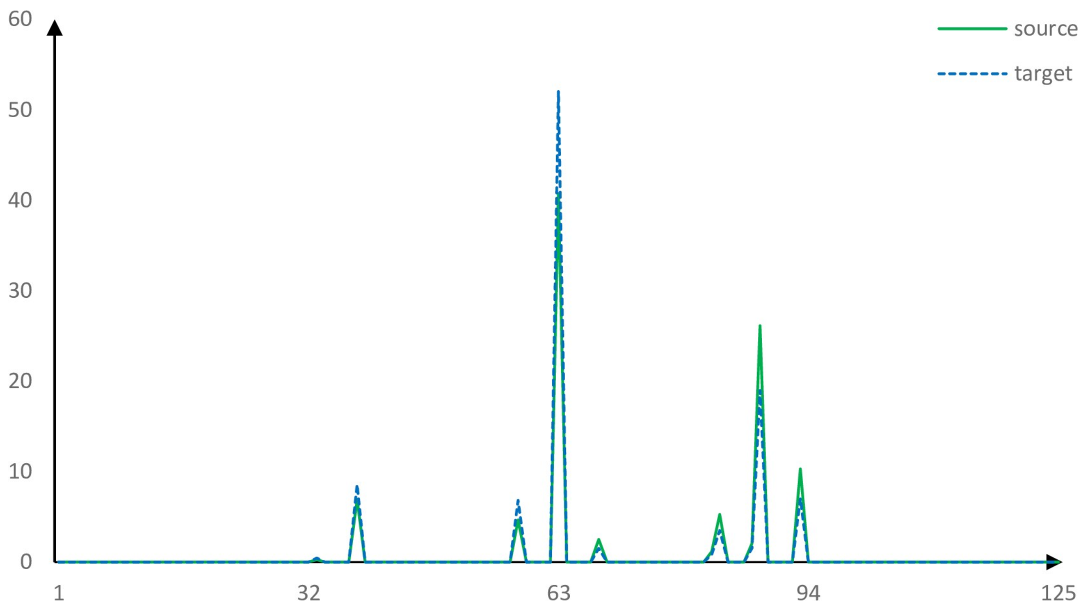


Fig 14. Corresponding points feature descriptors using PFH.

https://doi.org/10.1371/journal.pone.0238802.g014

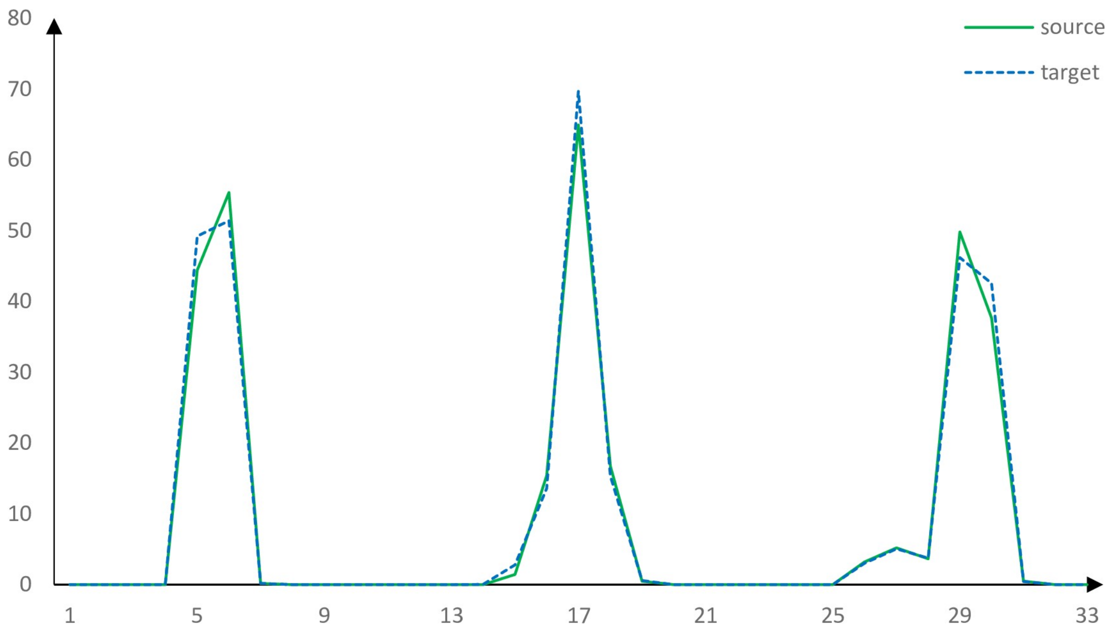


Fig 15. Corresponding points feature descriptors using FPFH.

https://doi.org/10.1371/journal.pone.0238802.g015

Fig 20 shows the six different scan directions to generate six point clouds of cheff. The six point clouds have different overlap rates. The overlap rate of two aligned point clouds is calculated as following. First search the closest point pairs in two aligned point clouds. When the distance between two closest points is less than five times the point cloud resolution, the points are viewed as overlapping point. The number of overlapping points is divided by the number of points in the point cloud as the overlap rate.

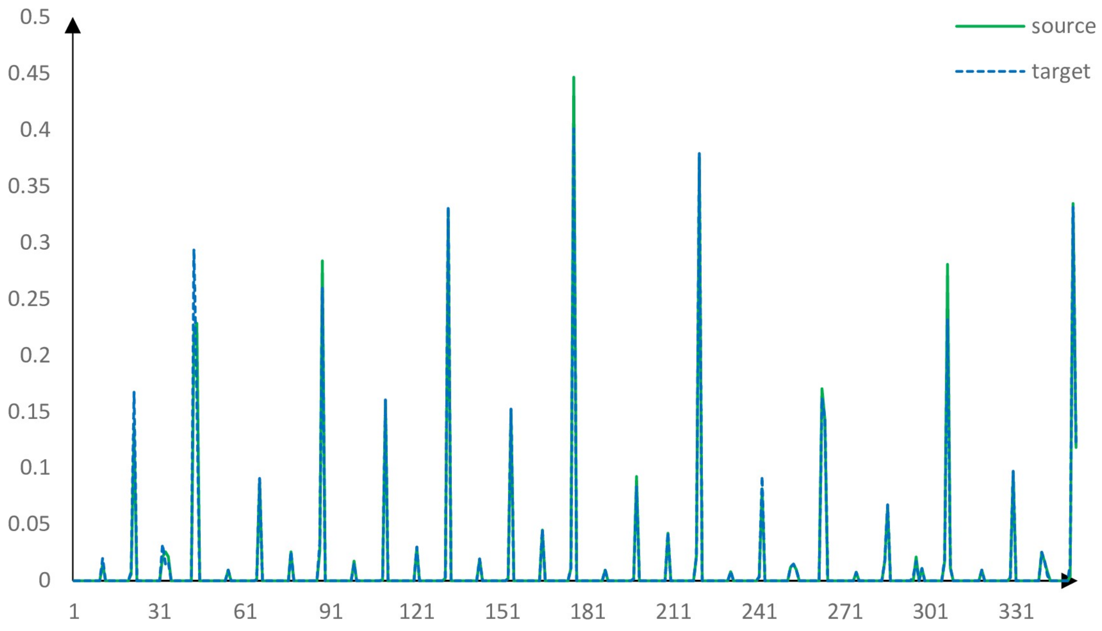


Fig 16. Corresponding points feature descriptors using SHOT.

https://doi.org/10.1371/journal.pone.0238802.g016

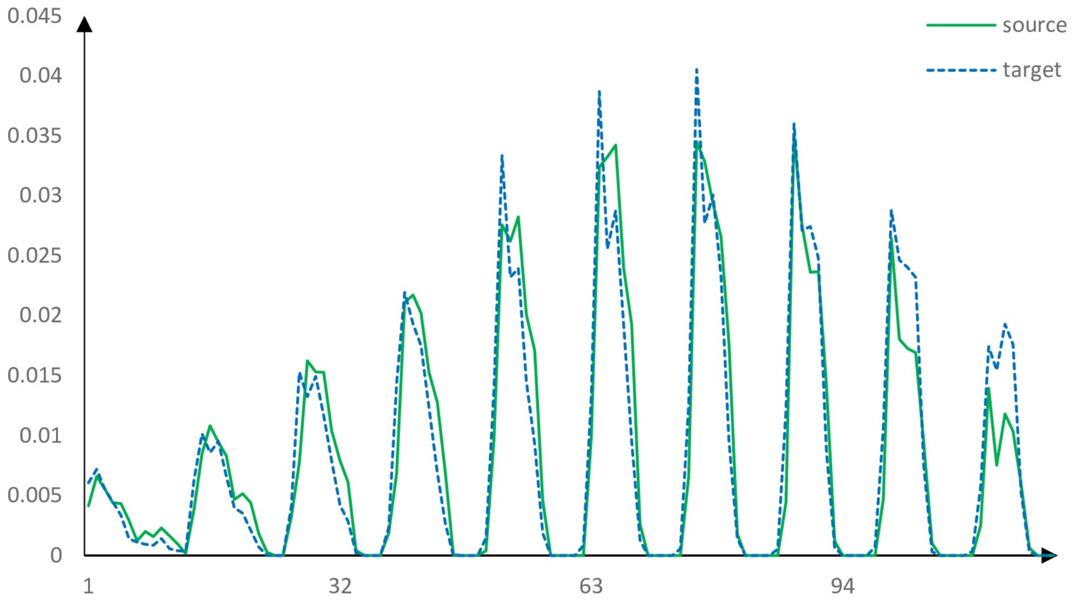


Fig 17. Corresponding points feature descriptors using our method.

https://doi.org/10.1371/journal.pone.0238802.g017

As shown in Table 7, Figs 21 and 22, when the overlap rate is greater than or equal to 43.72%, our register algorithm has better accuracy. When the overlap rate is less than or equal to 37.52%, our algorithm fails to register. When the overlap rate is greater than or equal to 57.58%, the PFH algorithm can accurately register. When the overlap rate is less than or equal to 43.72%, the PFH algorithm fails to register. Due to the small number of key points in our registration algorithm, the difficulty of key point matching can be reduced and the registration effect can be maintained for the situation with a low overlap rate.

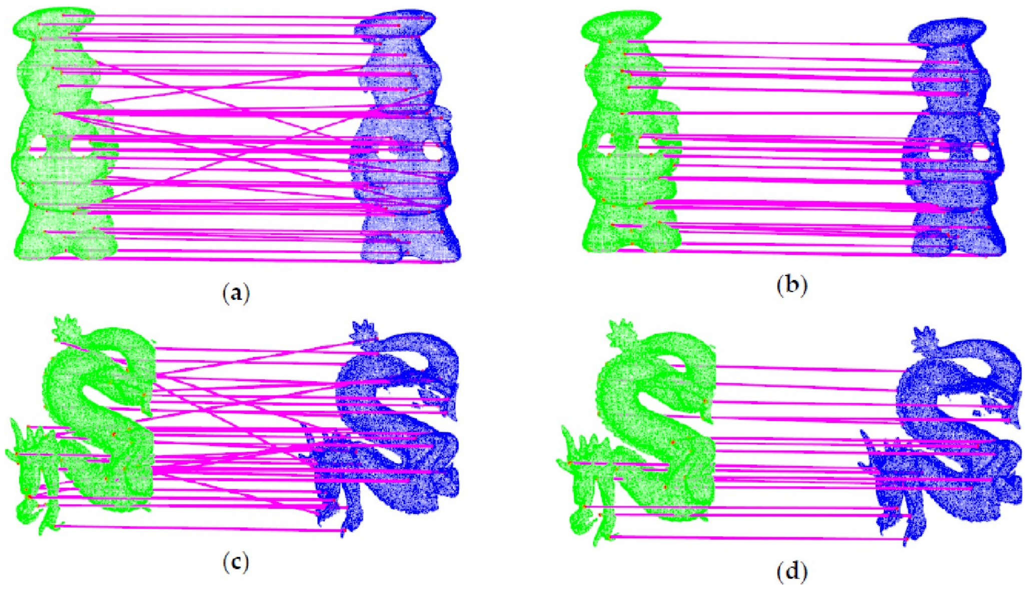


Fig 18. Comparison of the initial correspondences and correspondences after removing the errors. (a) Initial Correspondences Between a_s and a_t; (b) Final Correspondences Between a_s and a_t; (c) Initial Correspondences Between d_s and d_t; (d) Final Correspondences Between d_s and d_t.

https://doi.org/10.1371/journal.pone.0238802.g018

Table 5. Parameters and results of registration.

Point cloud	cheff_s	cheff_t	dragon_s	dragon_t
Point number of clouds	89024	85794	187121	176930
Initial resolutions (mm)	0.59141	0.59243	0.22188	0.23245
Point number after filtering	35123	32634	31674	29457
Filtering time (ms)	145	132	215	228
Resolutions after filtering (mm)	0.983531	0.982354	0.987564	0.990124
Number of key points	226	215	263	274
The time of computing feature (ms)	156	163	175	179
Initial correspondences (pair)	142		157	
The time of determining initial correspondences (ms)	32		36	
Optimized correspondences (pair)	84		79	
The time of optimizing correspondences (ms)	146		147	
The time of SVD (ms)	344		323	
The time of ICP (ms)	425		585	
Registration error (mm)	0.190132		0.136213	

https://doi.org/10.1371/journal.pone.0238802.t005

Table 6. Parameters and results of registration.

Point cloud	cheff_s	cheff_t	dragon_s	dragon_t
Point numbers of clouds	89024	85794	187121	176930
Initial resolutions (mm)	0.59141	0.59243	0.22188	0.23245
Voxel size	1.45	1.45	1.35	1.35
Point numbers after filtering	24019	23508	22412	21536
Filtering time (ms)	47	35	74	71
Resolutions after filtering (mm)	1.04143	1.04626	0.951742	0.922374
Number of key points	160	155	184	186
The time of computing feature (ms)	231	215	205	195
Initial correspondences (pair)	196		161	
The time of determining initial correspondences (ms)	42		63	
Correspondences after removal process (pair)	74		84	
The time of removal process (ms)	1134		1237	
The time of ICP (ms)	863		910	
Registration error (mm)	0.210131		0.124432	

https://doi.org/10.1371/journal.pone.0238802.t006

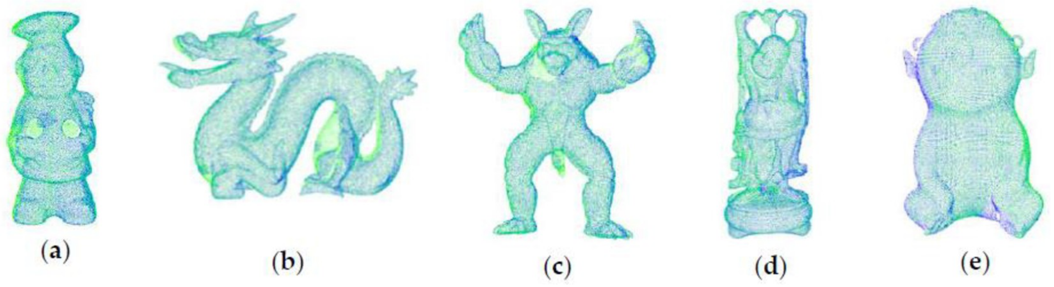


Fig 19. The results of point cloud registration. (a) Cheff; (b) Dragon; (c) Armadill; (d) Happy; (e) Boy.

https://doi.org/10.1371/journal.pone.0238802.g019

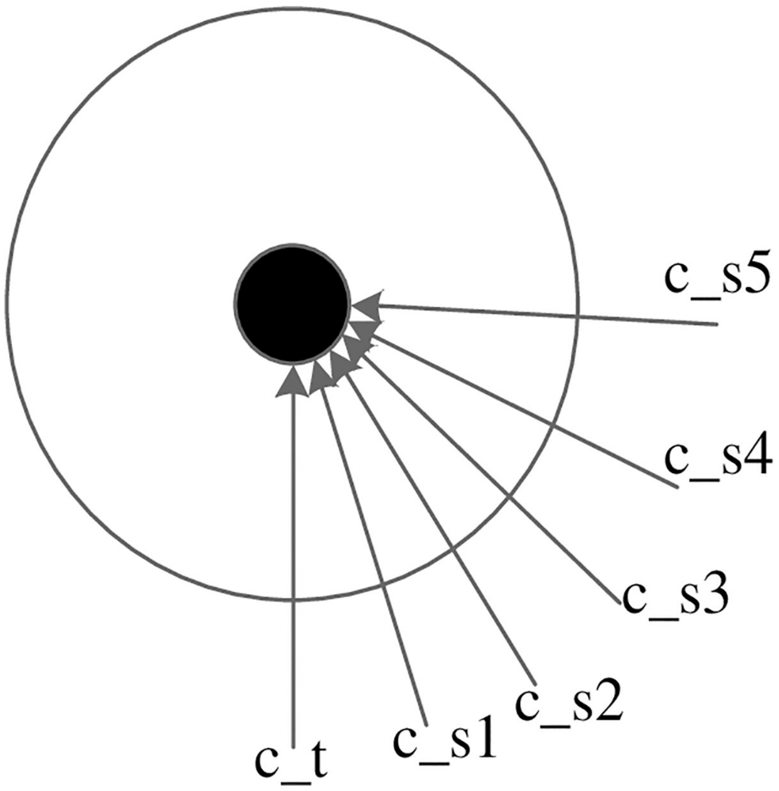


Fig 20. Six scan directions.

https://doi.org/10.1371/journal.pone.0238802.g020

4. Conclusion

In this paper, the filter parameters are adaptively turned according to the resolution of the point cloud. A key point finding algorithm based on the mean value of the curvature of the neighborhood of the pre-keypoint is proposed. It did not adopt common key point finding algorithms which rely on single point curvature values and enhances robustness to noise, reduces the repetitiveness of key points at the same local region. In this paper, we proposed a computation method of feature descriptor based on distances and normal relationship between the center of gravity and each points in its neighborhood. Robustness to noise and uniqueness of the descriptor are improved. The wrong correspondences are removed effectively based on neighborhood combined feature of original matching point pair. It ensures accuracy of registration and reduces time of ICP. The proposed registration algorithm has good accuracy, computing efficiency and robustness to noise. It is suitable for automatic registration of point clouds with low overlapping rate and big noise.

Table 7. Registration data of different overlap rate.

	c_t	c_s1	c_s2	c_s3	c_s4	c_s5
Point numbers of clouds	86884	90019	82949	76890	77467	81933
Rate of overlap (%)		89.74	71.55	57.58	43.72	37.52
Registration error of our paper		0.120578	0.143751	0.218378	0.242352	0.967535
Registration error of SHOT		0.132231	0.263515	0.326346	0.936345	0.977345

https://doi.org/10.1371/journal.pone.0238802.t007

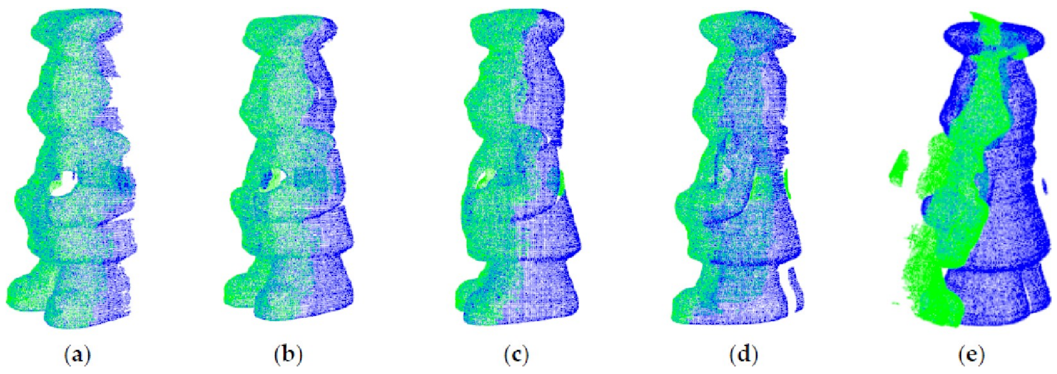


Fig 21. Registration results by using proposed algorithm. (a) c_s1 Registration Result; (b) c_s2 Registration Result; (c) c_s3 Registration Result; (d) c_s4 Registration Result; (e) c_s5 Registration Result.

https://doi.org/10.1371/journal.pone.0238802.g021

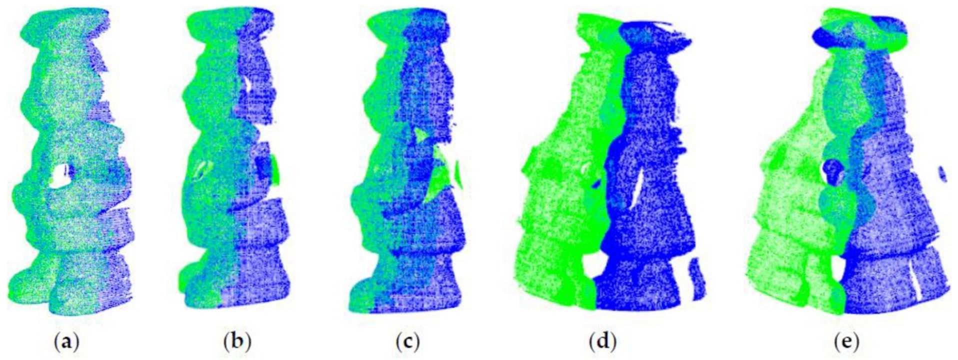


Fig 22. Registration results by using PFH algorithm. (a) c_s1 Registration Result; (b) c_s2 Registration Result; (c) c_s3 Registration Result; (d) c_s4 Registration Result; (e) c_s5 Registration Result.

https://doi.org/10.1371/journal.pone.0238802.g022

Supporting information

S1 File. PCDATA. Point cloud data used in the paper.
(RAR)

Author Contributions

Conceptualization: Jun Lu.

Data curation: Zhuo Wang, Bowen Hua, Kun Chen.

Methodology: Jun Lu.

Project administration: Jun Lu.

Resources: Jun Lu, Zhuo Wang, Kun Chen.

Software: Bowen Hua.

Supervision: Jun Lu.

Validation: Bowen Hua, Kun Chen.

Visualization: Bowen Hua.

Writing – original draft: Zhuo Wang.

Writing – review & editing: Zhuo Wang.

References

1. Han M, Zhou B, Qian K. 3Dlocalization and mapping of outdoor mobile robots using a LIDAR. Journal of Huazhong University of ence & Technology. 2015.
2. Gézero L, Antunes C. An efficient method to create digital terrain models from point clouds collected bymobile lidar systems. ISPRS Int. 2017.
3. Besl P.J., Mckay H.D. A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Mach. Intell, 1992.
4. Chen Y, Gérard Medioni. Object modeling by registration of multiple range images. Image and Vision Computing. 1992; 10(3):145–155.
5. Ji S, Ren Y, Ji Z. An improved method for registration of point cloud. Optik—International Journal for Light and Electron Optics. 2017; S0030402617300517.
6. Zhu QY, Wu JJ. LIDAR point cloud registration for sensing and reconstruction of unstructured terrain. APPLIED SCIENCES-BASEL. 2018; https://doi.org/10.3390/app8112318
7. Meng JG, Li JL. An accelerated ICP registration algorithm for 3D point cloud data. 9th International Symposium on Advanced Optical Manufacturing and Testing Technologies (AOMATT)—Optical Test, Measurement Technology, and Equipment. Chengdu, PEOPLES R CHINA, JUN 26–29, 2018, 10.1117/12.2504772.
8. Liu S, Gao D, Wang P, Guo X, Xu J, Liu D-X. A depth-based weighted point cloud registration for indoor scene. Sensors. 2018; 18, 3608.
9. Agamennoni G, Fontana S, Siegwart, R.Y., Sorrenti, D.G. Point clouds registration with probabilistic data association. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2016; 4092–4098.
10. Li Q, Huang X. Feature extraction from point clouds for rigid aircraft part inspection using an improved Harris algorithm. Measurement Science and Technology. 2018; https://doi.org/10.1088/1361-6501/aadff6
11. Lowe D.G. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision. 2004; 60(2):91–110.
12. Xian Y, Xiao J, Wang Y. A fast registration algorithm of rock point cloud based on spherical projection and feature extraction. Frontiers of Computer Science. 2019; 13(1).
13. Chen WL, Yang Y. Registration of color point cloud by combining with color moments information. IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE Syst Man & Cybernet Soc, Miyazaki, JAPAN, OCT 07–10, 2018; 2102–2108.
14. Bueno M, Gonzalez-Jorge H, Martinez-Sanchez J. Automatic point cloud coarse registration using geometric keypoint descriptors for indoor scenes. Automation in Construction. 2017; 81(sep.):134–148.
15. Prakhya S M, Liu B, Lin W. Detecting keypoint sets on 3D point clouds via Histogram of Normal Orientations. Pattern Recognition Letters. 2016; 83:42–48.
16. Amberg B, Romdhani S, Vetter T. Optimal step nonrigid ICP algorithms for surface registration. Computer Vision & Pattern Recognition IEEE Computer Society Conference on. 2007;1–8.
17. R.B. Rusu, N. Blodow, M. Beetz. Fast point feature histograms (FPFH) for 3D registration. International Conference on Robotics and Automation (ICRA). 2009; 12–17.
18. Wang X, Zhang X. Rigid 3D point cloud registration based on point feature histograms. 2nd International Conference on Machinery, Electronics and Control Simulation (MECS). Taiyuan, PEOPLES R CHINA, JUN 24–25. 2017; 138: 543–550.
19. Prakhya S M, Liu B, Lin W. B-SHOT: a binary 3D feature descriptor for fast Keypoint matching on 3D point clouds. Autonomous Robots. 2016.
20. Salti S., Tombari F., Di Stefano L. SHOT: unique signatures of histograms for surface and texture description. Computer Vision and Image Understanding. 2014; 125, pp. 251–264.
21. Kleppe A L, Egeland O. A curvature-based descriptor for point cloud alignment using conformal geometric algebra. Advances in Applied Clifford Algebras. 2018; 28(2):50.
22. Aiger D, Mitra N J, Cohen-Or D. 4-points congruent sets for robust pairwise surface registration. ACM Press ACM SIGGRAPH 2008 papers—Los Angeles. 2008.
23. Ge X. Automatic markerless registration of point clouds with semantic-keypoint-based 4-points congruent sets ISPRS. Photogram. 2017; 130, pp. 344–357.

24. Mellado N, Aiger D, Mitra N J. Super 4PCS fast global point cloud registration via smart indexing. Computer Graphics Forum. 2014; 33(5):205–215.

25. Qin Ye, Hang Liu, Yuhang Lin. Study of RGB-D point cloud registration method guided by color information. Tenth International Conference on Information Optics and Photonics. 2018.

26. Chang WC, Andini DP, Pham VT. An implementation of reinforcement learning in assembly path planning based on 3D point clouds. International Automatic Control Conference (CACS). 2018; 04–07.

27. P. Biber, W. Strasser. The normal distributions transform: a new approach to laser scan matching. Intelligent Robots and Systems, IEEE, pp. 2003; 2743–2748.

28. M. Magnusson. The three-dimensional normal-distributions transform—an efficient representation for registration, surface analysis, and loop detection. Ph.D. dissertation, Örebro University. 2009.

29. Hyunki Hong, B.H. Lee. Probabilistic normal distributions transform representation for accurate 3D point cloud registration. International Conference on Intelligent Robots and Systems (IROS) Sept. 2017; 24–28.

30. Lei H, Jiang G, Quan L. Fast descriptors and correspondence propagation for robust global point cloud registration. IEEE transactions on image processing: a publication of the IEEE Signal Processing Society. 2017; 26(8):3614.