photonics

MDPI

Article

Fast Point Cloud Registration Algorithm Based on 3DNPFH Descriptor

Bo You [1,2,*], Hongyu Chen [2], Jiayu Li [1,2], Changfeng Li [3] and Hui Chen [3]

[1] Heilongjiang Provincial Key Laboratory of Complex Intelligent System and Integration, Harbin University of Science and Technology, Harbin 150080, China; lijiayu@hrbust.edu.cn
[2] School of Automation, Harbin University of Science and Technology, Harbin 150080, China; 1920510072@stu.hrbust.edu.cn
[3] Changzhou Mingseal Robot Technology Co., Ltd., Changzhou 213164, China; lichangfeng@mingseal.com (C.L.); chenhui@mingseal.com (H.C.)
* Correspondence: youbo@hrbust.edu.cn

Abstract: Although researchers have investigated a variety of approaches to the development of three-dimensional (3D) point cloud matching algorithms, the results have been limited by low accuracy and slow speed when registering large numbers of point cloud data. To address this problem, a new fast point cloud registration algorithm based on a 3D neighborhood point feature histogram (3DNPFH) descriptor is proposed for fast point cloud registration. With a 3DNPFH, the 3D key-point locations are first transformed into a new 3D coordinate system, and the key points generated from similar 3D surfaces are then close to each other in the newly generated space. Subsequently, a neighborhood point feature histogram (NPFH) was designed to encode neighborhood information by combining the normal vectors, curvature, and distance features of a point cloud, thus forming a 3DNPFH (3D + NPFH). The descriptor searches radially for 3D key point locations in the new 3D coordinate system, reducing the search coordinate system for the corresponding point pairs. The "NPFH" descriptor is then coarsely aligned using the random sample consensus (RANSAC) algorithm. Experiment results show that the algorithm is fast and maintains high alignment accuracy on several popular benchmark datasets, as well as our own data.

Keywords: local surface descriptors; descriptor matching; point cloud registration; iterative closest point

Citation: You, B.; Chen, H.; Li, J.; Li, C.; Chen, H. Fast Point Cloud Registration Algorithm Based on 3DNPFH Descriptor. *Photonics* **2022**, *9*, 414. https://doi.org/10.3390/photonics9060414

Received: 15 May 2022
Accepted: 13 June 2022
Published: 15 June 2022

1. Introduction

There has been a recent and rapid development of three-dimensional (3D) laser scanning technology for use in reverse engineering, digital cities, deformation monitoring, and other applications [1]. Owing to limitations in the field of view of 3D laser scanning equipment, and the complex geometry of the scanned objects themselves, the point cloud data collected from each viewpoint only partially cover the geometric information of the scanned object surfaces. To obtain complete information regarding such surfaces, it is necessary to register the point clouds from individual scans into the same reference coordinate system.

To date, the classic iterative nearest point (ICP) [2] remains the most widely-used point cloud matching algorithm. Many researchers have made improvements to the ICP algorithm [3–6], which relies on a good initial alignment position. Without this, it quickly falls into the local optimum and cannot achieve a good registration effect. Therefore, as common practice, the initial alignment method is used to obtain a good initial alignment position and then the ICP algorithm is applied to achieve an accurate alignment. Due to the irregular nature of 3D point clouds, designing a local surface descriptor with high overall performance is a challenge. Three-dimensional point cloud descriptors fall into two main categories: handcrafted and deep learning based methods. The incredible power of deep learning techniques has resulted in a breakthrough in the obtainment of 3D point cloud

descriptors. As an example, Deng et al. [7] proposed the Point Pair Feature NETwork, which uses a new N-tuple loss and an architecture that naturally injects global information into the local descriptors, achieving good results in terms of accuracy, speed, and other factors. In addition, Ao et al. [8] proposed Spinnet, which first introduces a spatial point transformer to map the input local surface into a well-designed cylindrical space and then applies a powerful point-based and 3D cylindrical convolutional neural layer feature extractor to derive compact and representative descriptors for matching. Spinnet has an excellent ability to generalize well under unseen scenarios. Finally, Fabio et al. [9] normalized their estimated local reference system by extracting point cloud patches and encoding rotationally invariant compact descriptors using a robust deep neural network based on PointNet. Although deep learning has made significant progress in the area of point cloud alignment, it inherently requires a large amount of training data, and a separate training process is needed to learn a separate feature extraction network, which is time-consuming and results in significant hardware requirements.

For handcrafted methods, such descriptors are divided into two categories depending on whether a local reference frame (LRF) is used. Although many efficient methods have been proposed for descriptors without an LRF, point pair feature (PPF) based descriptors are the most classical approaches for a 3D surface description. Johnson and Heber [10] proposed using spin image (SI) features, in which the normal vectors of the key points are first applied as reference axes and the local neighborhood points are then projected onto a 2D surface both horizontally and vertically. Although an SI is a frequently cited descriptor, it has limited descriptive power and is sensitive to changes in the data resolution. In addition, a point feature histogram (PFH) [11] has high discriminatory power but is extremely time-consuming. To address this problem, Rusu et al. [12] constructed a fast point feature histogram (FPFH) using a simplified point feature histogram (SPFH), which is characterized as fast and discriminative. Albarelli et al. [13] defined a descriptor for low-dimensional surface hashing and applied it to the surface matching problem in a game-theoretic framework, where the surface hash features are mainly computed based on multiscale statistical histograms of the local properties, including normal vector pinch angles, integral volumes, and their combination. Moreover, Zhao et al. [14] proposed a novel normal surface reorientation drawing upon a Poisson-disc sampling strategy to address the problem of data redundancy during data preprocessing. Subsequently, a new technique is used to divide the local point pairs for each key point into eight regions, where each local point pair distribution region is applied to construct the corresponding sub-features. Finally, an extracted histogram of the point pair features is generated by concatenating all the subfeatures into a single vector.

For LRF-based features, Stein and Medioni [15] designed the local characteristics of point clouds as a consequence of the offset angle, torsion angle, and curvature relationships between key points and those within the geodetic neighborhood. Frome et al. [16] also suggested using a 3D shape context (3DSC) by extending the 2D shape context (SC) [17] into the 3D domain. The proposed method first divides the local spherical domain into multiple subspaces and then computes the feature descriptors by calculating the percentage of points in each subspace. Zaharescu et al. [18] compute gradient vectors for each neighborhood point and project them onto three orthogonal planes of the LRF. Each plane is divided into four quadrants, and each quadrant corresponds to an 8-dimensional feature. In addition, Prakhya et al. [19] form a histogram by accumulating the points in each interval divided by a "3D" descriptor. As a concrete process, the local surface of the key points is aligned to the defined LRF by employing a "3D" descriptor, and the range between the minimum and maximum x-coordinate values of the points on the surface is then divided into D intervals along the x-axis. The same process is repeated along the y- and z-axes, and a 3D histogram of the point distribution (3DHoPD) is generated by concatenating these histograms. Although this method is fast, it is poorly descriptive. Guo et al. [20] first rotate the local surface in the LRF, project the surface after each rotation to calculate the projected point density statistics, and concatenate these statistics to obtain rotational projection

statistics (RoPs). Although the use of RoPS has achieved the highest matching performance for several datasets, its shortcomings include a poor descriptiveness and time-consuming computations for data with uneven point distributions. Similar to the rotational projection mechanism of RoPS, Guo et al. [21] use the three orthogonal axes of the LRF to compute three spin map features and obtain triple spin images (TriSI) that are more resistant to occlusions than RoPS but are still more time-consuming.

In this paper, a point cloud alignment algorithm based on the 3D point neighborhood feature histogram (3DNPFH) descriptor is proposed to address the current problems inherent to an alignment algorithm. After uniform sampling using the voxel grid method, the algorithm transfers the key points to the new 3D coordinate system for the point cloud. The point pairs to be matched are obtained using the principle in which the key points are generated by similar numbers of 3D surfaces that are close to each other. We construct the neighborhood point feature histogram (NPFH) descriptor by calculating the normal vector, curvature, and distance features within the neighborhood of the key points, and finally apply the RANSAC algorithm to achieve a coarse alignment and obtain a good initial alignment position.

2. Proposed Methods

In general, in a 3D surface description, a unique high-dimensional feature vector is used to describe a local 3D surface at a 3D key point. In this study, a 3D neighborhood point feature histogram (3DNPFH) is proposed to represent a local 3D surface. Generating the 3DNPFH descriptor involves two main steps: encoding the 3D key point positions to form a pre-3D descriptor and obtain a list to be matched, and finding an exact match by computing the NPFH descriptor. Let us consider an input source point cloud P_{source} upon which I key points, K_i, where $i = \{1, 2, \ldots, I\}$, are extracted using a uniform sampling. We then create a 3DNPFH descriptor from its radius neighborhood surface, $Surface_{ik}$, including K points, where $k = \{1, 2, \ldots, K\}$. The neighborhood surface from which the descriptor is constructed is determined based on its support radius r. A key point K_i has $Surface_{ik}$, where i represents the index of the key points, and k represents the index of the points within the surface of the neighborhood of the key points.

2.1. Encoding 3D Key Point Locations

The uniformly-sampled key points are transformed into a new 3D coordinate system, and their 3D coordinates are recorded in the first three dimensions of the 3DNPFH descriptor. The extraction of feature descriptors around the key points and a comparison to find their correspondences are conducted to locate exact 3D key point correspondences generated from a similar 3D surface neighborhood. Therefore, the key points are transformed into a new 3D coordinate system in which the key points of similar surfaces are close to one another. We first calculate the centroid coordinates $mean_{pt}$ in the neighborhood coordinate system $Surface_{ir}$ of surface radius r of key point K_i, and then subtract the centroid coordinates $mean_{pt}$ from all point coordinates in $Surface_{ir}$, effectively generating a new surface $Surface_{ir-mean_{pt}}$, and letting the key points K_i subtract the centroid coordinate $mean_{pt}$ to obtain $K_{i-mean_{pt}}$. Finally, we have a new surface, $Surface_{ir-mean_{pt}}$, used to calculate the local reference coordinate system {a}. The three axes of {a} form the rotation matrix $[RF]_{3\times3}$, and the key point $K_{i-mean_{pt}}$ is then transferred to a new 3D coordinate system, as shown in (1).

$$[K_{iRF}]_{3\times1} = [RF]_{3\times3}\left[K_{i-mean_{pt}}\right]_{3\times1} \quad (1)$$

where its new coordinates, K_{iRF}, constitute the first three dimensions of the descriptors proposed herein.

Local reference coordinate system: This algorithm establishes a local reference frame, such as SHOT [22], and the corrected covariance matrix of the point cloud is then calculated using the 3D surface $Surface_{ir-mean_{pt}}$, as shown in (2). Within the radius neighborhood of the considered key points, weighting is conducted according to the distance $K_{i-mean_{pt}}$ of

the key points from the centroid. For better readability, we use q_r to represent the point within the surface $Surface_{ir-mean_{pt}}$ around the key points, and q to represent the distance between the key points and the centroid $K_{i-mean_{pt}}$.

$$COV = \frac{1}{\sum\limits_{r:d_r \le r} (r - d_m)} \sum_{i:d_m \le r} (r - d_r)(q_r - q)(q_r - q)^T \tag{2}$$

Here, $d_r = \|q_r - q\|_2$. To create a unique local reference coordinate system, it is necessary to disambiguate the direction of the normal. We use the orientation of the local x- and z-axes toward the principal directions of the vectors they represent, and finally obtain the local y-axis through the cross product of z and x, where y = z × x. By solving the covariance matrix COV, we can obtain the eigenvalues and the eigenvectors corresponding to the eigenvalues, which can be sorted in descending order to obtain $\mu_1 > \mu_2 > \mu_3$. The corresponding eigenvectors x, y, and z represent the three coordinate axes, and the rotation matrix is constructed as follows:

$$[RF]_{3\times3} = [x^T y^T z^T]^T \tag{3}$$

However, because there may be various anomalies, we cannot state that the point closest to the key point in the new 3D coordinate system is the correct match; for example, noisy point cloud data may be acquired by the sensor or similar local reference coordinate systems. However, the correct match lies within the neighborhood of the key points, significantly reducing the search coordinate system for correctly matched points.

Figure 1 shows a schematic of the 3DNPFH descriptor. The key points of the source point cloud are shown in blue, and the key points of the target point cloud are shown in red. The blue and red spheres represent the LRF and the support size of the constructed descriptor, respectively. Then, for each model key point, the list of key points in the scene (shown as pink spheres in the new 3D coordinate system) is retrieved through a radial search of the threshold radius. From this retrieved list of possible key point matches of a scene, the one closest to the NPFH descriptor is considered an exact match.

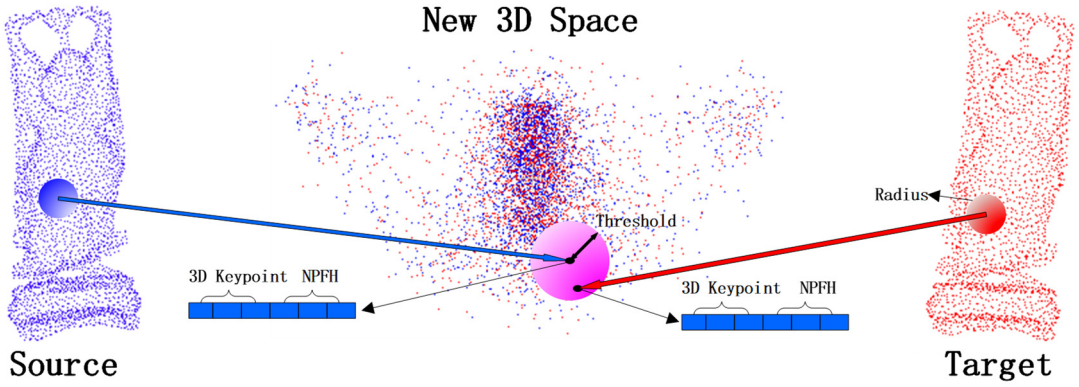


Figure 1. Schematic of 3DNPFH descriptor.

2.2. Calculation of Neighborhood Point Feature Histogram (NPFH)

Geometric features such as the curvature, surface normals, and distances reflect the most basic geometry of a point cloud, which is the key to expressing the local features of the cloud. In this section, based on the radius neighborhood of the key point, the neighborhood curvature sum, normal vector angle sum, and distance sum corresponding to the key point are calculated. A 3D vector consists of the sum of the curvature of the neighborhood, the sum of the angle between the normal vectors, and the sum of the distance. This 3D vector is an NPFH descriptor.

2.2.1. Curvature

The normal and curvature information of a point cloud surface are important geometric features for 3D object recognition. The curvature is invariant to rotation, translation, and scaling and is therefore used as a feature element. The curvature value reflects the degree of concavity of the point-cloud surface. The sharp features of the point cloud have a more significant curvature. By contrast, the non-featured parts of the point cloud exhibit a relatively slight curvature.

In this study, the method described in [23] is used to estimate the norm and curvature of the data points by analyzing the covariance of k neighborhood points within radius r. We analyze the covariance of a given point of the neighborhood point covariance for point cloud datasets and solve the covariance matrix. The direction of the eigenvector corresponding to the smallest eigenvalue is defined as the normal value of the point. The curvature of the point is then estimated based on the surface variation of the point in the local area. The curvature Cur_i is calculated as follows:

First, the following is calculated:

$$C_i \cdot v_j = \lambda_j \cdot v_j \tag{4}$$

This is transformed into solving the eigenvalues and eigenvectors of matrix C_i, where matrix C_i is a covariance matrix of k points, and (v_1, v_2, v_3) are the eigenvectors of the matrix. Each data point K_i in the point cloud corresponds to matrix C_i as follows:

$$C_{\mathrm{i}} = \begin{bmatrix} K_{i1} - \overline{K_i} \\ K_{ik} - \overline{K_i} \end{bmatrix}^T \begin{bmatrix} K_{i1} - \overline{K_i} \\ K_{ik} - \overline{K_i} \end{bmatrix} \tag{5}$$

where C_i is a positive semi-definite third-order symmetric matrix, $\overline{K_i}$ is the centroid of the point within the neighborhood of the key points K_i, and K_{ik} is the coordinate of the point in the radius neighborhood of K_i. The three eigenvalues λ_1, λ_2, and λ_3 of matrix C_i and the corresponding unit eigenvectors e_1, e_2, and e_3 are then calculated. Assuming that $\lambda_1 \leq \lambda_2 \leq \lambda_3$, λ_1 describes the change in the surface in the normal direction and represents the distribution of data points on the tangent plane, the curvature can then be expressed as

$$Cur_i = \frac{\lambda_1}{\lambda_1 + \lambda_2 + \lambda_3} \tag{6}$$

2.2.2. Deviation Angle between Normals

Previous studies have demonstrated that a representation based on the deviation angle between two normals has a high discriminative power [9]. The normal directions of K_i and K_{ik} are n_{K_i} and $n_{K_{ik}}$, respectively. The cosine of the normal angle between K_i and K_{ik} can be expressed as

$$cos\,\theta_{K_{ik}} = \frac{n_{K_i} \cdot n_{K_{ik}}}{\left|n_{K_i}\right|\left|n_{K_{ik}}\right|} \tag{7}$$

The sum of the angles normal to the key points and all points in the radius neighborhood is calculated as follows:

$$\omega_a(K_i) = \sum_{K_{ir} \in Surface_{ir}} \theta_{K_{ik}} \tag{8}$$

Figure 2 shows a schematic of the angle between the normal vector of the feature and non-feature regions. In Figure 2a, the normal angle in the neighborhood of the key point K_i is larger, which generally forms the feature region of the point cloud. In Figure 2b, the normal angle within the neighborhood of key point K_i is smaller, which generally forms the non-featured region of the point cloud. Therefore, the normal vector angle is also used as a parameter to calculate the feature description of the point cloud.

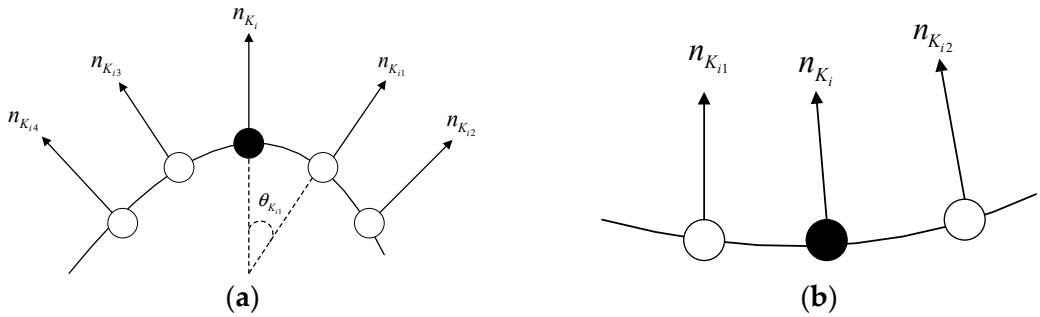


Figure 2. Angle between the normal vector of the feature and non-feature areas. (**a**) the normal vector of the feature area; (**b**) the normal vector of the non-feature area.

2.2.3. Sum of Distances between Neighborhood Points

For the radius neighborhood of a key point, the number of points within the neighborhood generally differs; therefore, the local 3D geometric features can be described by the distance sum of the neighborhood points.

The sum of the distances between the key points and the neighboring points reflects the characteristics of the point cloud. The point cloud feature is distinguished using the sum of the distances between the key points and neighboring points as the distance parameter.

The distance from a point in the neighborhood to a point in the radius neighborhood can be expressed using the following formula:

$$\omega_d(K_i) = \sum_{K_{ir} \in Surface_{ir}} |K_{ik} - K_i| \tag{9}$$

Figure 3 shows a diagram of the neighborhood point distances, where K_i is the key point, and K_{ik} is the point within the radius neighborhood of K_i.

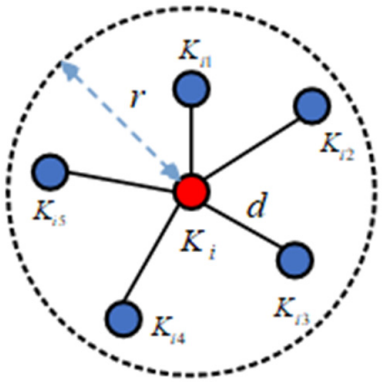


Figure 3. Diagram of neighborhood point distances.

After proposing three local geometric features, three sub-histograms are obtained: the sum of the distances between neighborhood points, the sum of the curvatures, and the sum of angles between the normal direction of the key point and the normal direction of the neighborhood points. Figure 4 shows a schematic of the NPFH feature-description mechanism. A 3DNPFH descriptor is created by combining the three sub-histograms above into a single histogram. The following are the three primary characteristics of the NPFH descriptor: The NPFH descriptor is computationally efficient, the three local geometric characteristics of the NPFH descriptor are low-dimensional and computationally efficient, and the computational cost of the NPFH descriptor is O(k), where k is the number of points within the radius neighborhood of key point K_i.

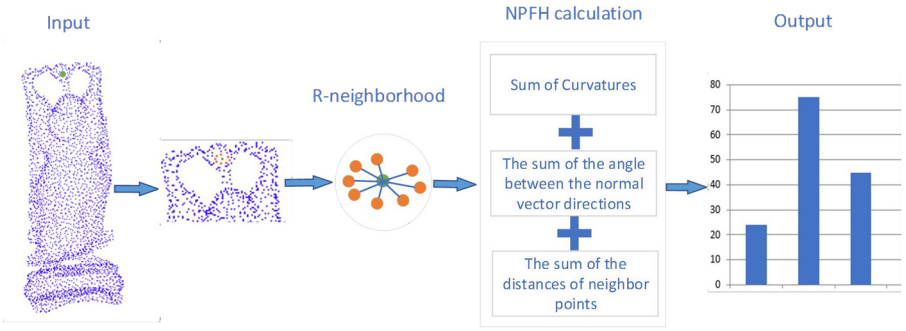


Figure 4. Schematic representation of NPFH characterization mechanism.

2.3. Matching

Taking advantage of the fact that key points generated by similar 3D surfaces are close to each other in the new 3D coordinate system, we apply the following matching of the 3DNPFH descriptors. First, the extracted key points are transformed into a new 3D coordinate system by constructing local reference coordinate systems for the key point neighborhood surfaces in the source and target point clouds. Then, for each source key point $K_{i-source}$ in the new 3D coordinate system, we find a list of the closest target key points $K_{i-target}$ in the new system that lie within the radius threshold T_d. Finally, we find the most accurate nearest neighbors in the 3DNPFH descriptor using a Euclidean distance search to form the corresponding point pairs in the to-be-matched list.

2.4. Mismatch Rejection of RANSAC

The corresponding point pairs are identified using the 3DNPFH feature-based descriptors. This study uses the RANSAC algorithm to eliminate mismatched pairs and improve the alignment accuracy of two-point clouds. Assuming that n matching pairs are obtained after the above feature matching, the RANSAC algorithm is used to reject false matches and obtain the transformation relationships between the point clouds as follows:

Step 1: Randomly select three non-collinear points from the source point cloud P, denoted as $\{p_1, p_2, p_3\}$, and search for their corresponding points from the target point cloud Q, which are denoted as $\{q_1, q_2, q_3\}$, $\{p_1, p_2, p_3\}$, and $\{q_1, q_2, q_3\}$, as samples.

Step 2: Use the samples to estimate the rigid body transformation matrix H.

Step 3: Using the model estimation matrix, transform the remaining points in the source point cloud P and calculate the distance error between all points in the transformed source point cloud P and the target point cloud Q if the distance error of one point pair is below the set threshold. If there is an error, the point is added to the interior point set U; otherwise, it is an outlier.

Step 4: Repeat the above process until the number of points in the interior point set reaches the set threshold or the iteration number becomes greater than the maximum iteration number, and then stop the iterative calculations.

Step 5: Select the model parameter H with the most significant number of interior points in the interior point set as the optimal model parameter and use the optimal model parameter H to achieve a rough registration of the point cloud.

Because of the increased proportion of correct points in the set of correspondence points optimized using the RANSAC algorithm, the resulting rigid body transformation matrix is more accurate and effectively reduces the error in the point cloud alignment. However, RANSAC requires multiple iterations in the algorithm, and thus a lengthy time is required to obtain optimal results when the number of correspondence point pairs is large. Therefore, when using the RANSAC algorithm, care should be taken to set the threshold error and the number of iterations, as reasonable settings can help improve the rejection of erroneous point pairs.

3. Results

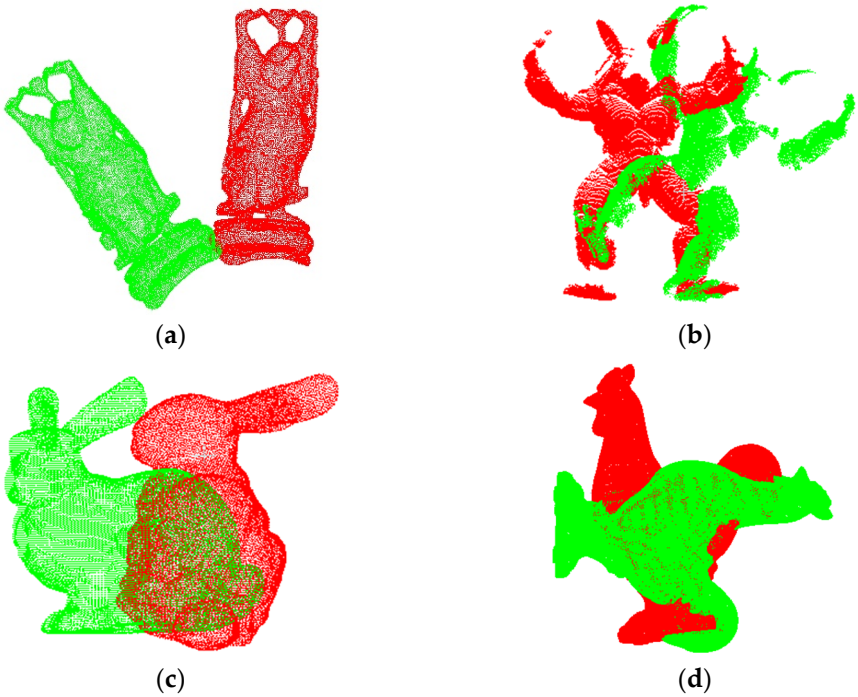
This section describes a performance test of our algorithm on four public datasets and on our own data, and compares the results with those of existing state-of-the-art descriptors. All experiments were conducted on a laptop with an Intel 2.40-GHz i5-10200H processor and 16 GB of memory, and the algorithms were implemented on Microsoft Visual Studio 2019 and the Point Cloud Library (PCL). To verify the effectiveness of the algorithm, in this study, the FPFH and 3DHoPD algorithms are compared under similar environments.

3.1. Experimental Data

This experiment first uses the open-source "Happy Buddha," "Bunny," "Armadillo" [24] and "Chicken" point cloud data [25], taken from the "Bologna" object recognition dataset, the "UWA" dataset, and the "random view" dataset. The numbers of points in the point cloud data in the experiment were as follows: Happy Buddha (32,316, 31,424), Bunny (34,834, 34,618), Armadillo (28,885, 32,385), and Chicken (135,142, 135,142). We then used a locally scanned mobile phone camera module point cloud for testing to further verify the effectiveness of the algorithm. The number of points in the point cloud of the "Camera Module" is (171,996, 171,979). The relevant information regarding the data used in this study is listed in Table 1, and the initial pose of the point cloud data is shown in Figure 5.

Table 1. Data used in the experiment.

No.	Model	Acquisition	Quality	Dimension	Data Set
1	Happy Buddha	Synthesis	High	3D	Bologna
2	Armadillo	Minolta vivid	Medium	2.5D	Random View
3	Bunny	Synthesis	High	3D	Bologna UWA
4	Chicken	Synthesis	High	3D	Bologna
5	Camera Module	SR7080	High	3D	Ours



(**a**) (**b**) (**c**) (**d**)

Figure 5. *Cont.*

(**e**)

Figure 5. Initial pose graph of experimental data (the source point cloud is shown in green, and the target point cloud is shown in red). (**a**) Happy Buddha data; (**b**) Armadillo data; (**c**) Bunny data; (**d**) Chicken data; (**e**) Camera Module data.

3.2. Selection of Main Parameters

The choice of relevant parameters directly affects the registration results. To measure the registration effect under different parameters, we use the most commonly applied root-mean-square error (RMSE) to measure the accuracy of the point cloud registration. This is defined as

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \|P_i - Q_i\|^2} \tag{10}$$

where P_i and Q_i are two corresponding points, and N is the number of matching point pairs. The smaller the RMSE is, the better the alignment of the two-point clouds.

In this study, uniform sampling is used to extract the key points. Table 2 lists the number of sampling points for the "Happy Buddha" dataset for the different grid sizes. To reduce the running time of the algorithm, there are generally several hundred key points. Figure 6 shows the registration results for the different voxels. As shown in the figure, with an increase in the grid size, the registration time and number of feature points gradually decrease, and the RMSE is the lowest when the voxel size is 0.015 m. Therefore, we set the voxel size to 0.015 m.

Table 2. Number of key points under different voxel sizes.

Grid Size/m	0.01	0.013	0.015	0.017	0.02
Source points (P)	640	375	279	279	146
Target points (Q)	671	377	388	288	141

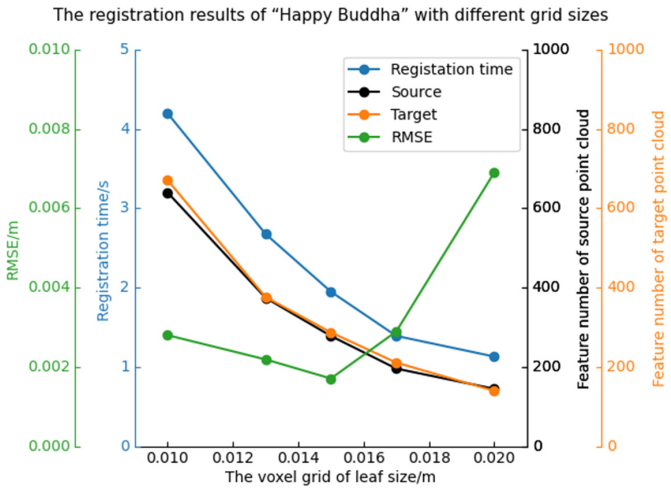


Figure 6. Registration results under different voxel sizes.

Table 3 shows the registration results of the Happy Buddha image dataset under different values of ε_{ransac}. When the value of ε_{ransac} is too large, the distance threshold of the corresponding point pair increases, leading to an increase in the calculation error. When the value of ε_{ransac} is too small, the distance threshold of the corresponding point pair becomes smaller, leading to a decrease in the number of corresponding points; thus, the model parameters cannot be accurately estimated, and the registration fails. When ε_{ransac} is 0.01, the registration time of the algorithm is the shortest, and the root-mean-square error is the smallest.

Table 3. Registration errors under different values of ε_{ransac}.

ε_{ransac}/m	0.002	0.003	0.004	0.005	0.01
RMSE/m	—	0.00632	0.00172	0.00172	0.00172
Time/s	3.081	3.002	2.902	2.632	2.552

3.3. Evaluation

To better test the effectiveness of the proposed descriptors, this section compares the algorithm with the well-performing FPFH and 3DHoPD algorithms. The registration results are first evaluated through a visualization and then by applying the well-known RMSE. The RMSE is defined through (5). Figure 7 shows the registration effect of the FPFH, 3DHoPD, and our proposed algorithms. The target point cloud is shown in red, and the registered point cloud is shown in blue. Table 4 lists the coarse registration RMSE values for each algorithm for the five models. The best results are highlighted in bold. From the experimental results, it can be seen that for all test point cloud data, our proposed algorithm achieves the smallest registration error for the five models, which further demonstrates the effectiveness of the proposed 3DNPFH descriptor. If a precise ICP registration is required in the future, our algorithm will provide a smaller pose registration, which requires fewer iterations to achieve better results.

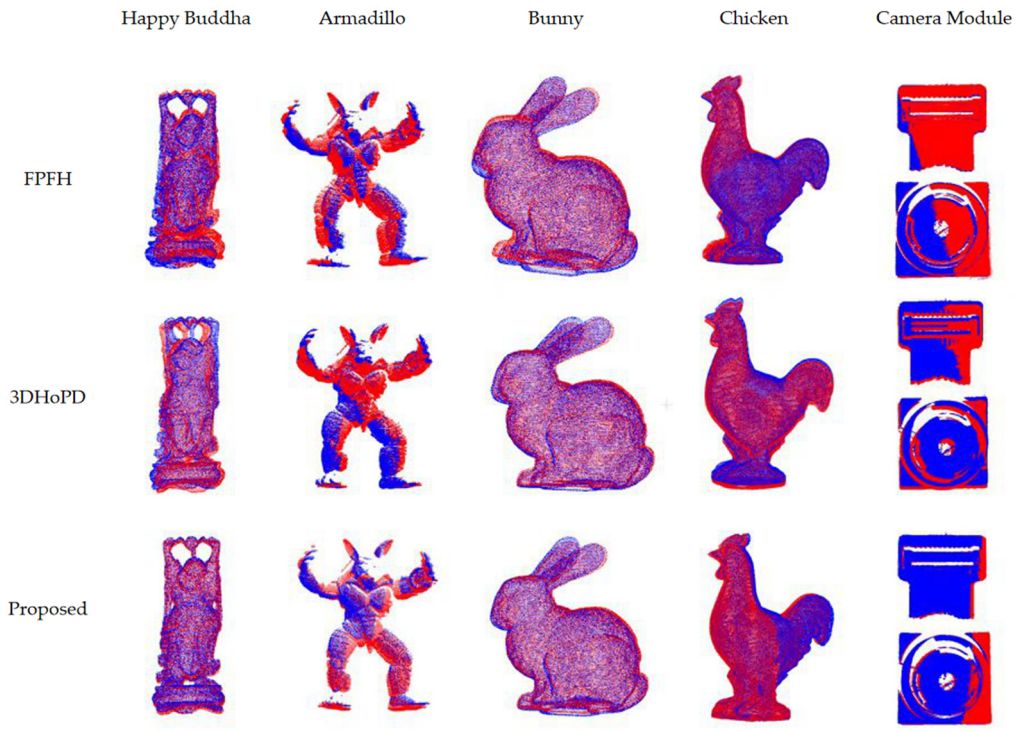


Figure 7. Registration results of the different algorithms.

Table 4. Registration errors of five models.

Model	FPFH	3DHoPD	Proposed
Happy Buddha	0.00544	0.00364	**0.00172**
Armadillo	0.00436	0.00410	**0.00314**
Bunny	0.00317	0.00331	**0.00148**
Chicken	0.00259	0.00302	**0.00248**
Camera Module	0.05895	0.05992	**0.05739**

Table 5 lists the registration times of the five models under the three different algorithms. The proposed method is the most effective. This is mainly due to the low number of dimensions of our proposed descriptor, i.e., only six. In addition, we significantly reduced the search space of the corresponding point pairs through "3D" descriptors, avoiding a complex corresponding point pair search. The total registration time of our method for the five models was only 1/6 that of the FPFH method and 7/10 that of the 3DHoPD approach. For the FPFH algorithm, which has 33 dimensions, its extraction time is significantly higher than that of the other descriptors because the supported radius size for the descriptor extraction doubles when calculating the SPFH. This was verified on the "Chicken" and "Camera Module" point clouds. Because of the higher density in these clouds, the number of points in the 3D sphere increases exponentially with the increase in the support radius size. Therefore, the computational cost of the FPFH is high in high-density point clouds or with larger support radius sizes [26]. For the 3DHoPD algorithm, which has 18 dimensions, the calculation of the histogram of point distributions (HoPD) descriptor considers the changes in the surface along the x-, y-, and z-directions, and then calculates the number of points by normalizing the histogram. However, this cannot explain the small surface changes (using a normal is helpful in this case).

Table 5. Comparison of the registration times of all algorithms for the five models.

Model	FPFH	3DHoPD	Proposed
Happy Buddha	3.693	2.620	**1.828**
Armadillo	3.286	2.507	**1.514**
Bunny	3.076	2.415	**1.217**
Chicken	63.781	7.611	**6.262**
Camera Module	14.432	5.875	**4.232**
Total	88.268	21.028	**15.053**

The above experiments revealed that the FPFH algorithm, 3DHoPD algorithm, and our proposed method successfully registered five of the models, and the registration error of the proposed algorithm was the smallest. Additionally, our approach requires the shortest registration time and exhibits a higher efficiency. Therefore, the proposed algorithm achieves a significant improvement over other previous algorithms.

4. Conclusions

We propose a point cloud registration algorithm based on the 3DNPFH descriptor. The algorithm first uniformly samples the point cloud to extract the key points, transfers the key points to a new 3D coordinate system by constructing a local reference coordinate system of the point cloud, and significantly reduces the coordinate search system during feature matching based on key points of similar surfaces that are close to each other. Then, by combining the density, curvature, and normal vector information of the point cloud, a neighborhood point feature histogram is constructed to find exact matches. Comparing popular point cloud registration algorithms on different datasets, we deduce that this algorithm is faster and maintains a similar level of registration accuracy, making it more suitable for point cloud registration systems than existing algorithms.

Author Contributions: Conceptualization, B.Y. and H.C. (Hongyu Chen); methodology, H.C. (Hongyu Chen); software, H.C. (Hongyu Chen); validation, B.Y., H.C. (Hongyu Chen) and J.L.; formal analysis, B.Y.; investigation, H.C. (Hongyu Chen); resources, C.L.; data curation, J.L.; writing—original draft preparation, H.C. (Hongyu Chen); writing—review and editing, B.Y.; visualization, H.C. (Hongyu Chen); and supervision, H.C. (Hui Chen). All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Chen, B.; Gao, H.; Li, H.; Ma, H.; Gao, P.; Chu, P.; Shi, P. Indoor and Outdoor Surface Measurement of 3D Objects under Different Background Illuminations and Wind Conditions Using Laser-Beam-Based Sinusoidal Fringe Projections. *Photonics* **2021**, *8*, 178. [CrossRef]
2. Besl, P.J.; McKay, N.D. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* **1992**, *14*, 239–256. [CrossRef]
3. He, Y.; Liang, B.; Yang, J.; Li, S.; He, J. An Iterative Closest Points Algorithm for Registration of 3D Laser Scanner Point Clouds with Geometric Features. *Sensors* **2017**, *17*, 1862. [CrossRef] [PubMed]
4. Koide, K.; Yokozuka, M.; Oishi, S.; Banno, A. Voxelized Gicp for Fast and Accurate 3d Point Cloud Registration. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 11054–11059.
5. He, Y.; Lee, C.-H. An Improved ICP Registration Algorithm by Combining PointNet++ and ICP Algorithm. In Proceedings of the 2020 6th International Conference on Control, Automation and Robotics (ICCAR), Singapore, 20–23 April 2020; pp. 741–745.
6. Wu, Z.; Chen, H.; Du, S.; Fu, M.; Zhou, N.; Zheng, N. Correntropy Based Scale ICP Algorithm for Robust Point Set Registration. *Pattern Recognit.* **2019**, *93*, 14–24. [CrossRef]
7. Deng, H.; Birdal, T.; Ilic, S. PPFNet: Global Context Aware Local Features for Robust 3D Point Matching. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 195–205.
8. Ao, S.; Hu, Q.; Yang, B.; Markham, A.; Guo, Y. SpinNet: Learning a General Surface Descriptor for 3D Point Cloud Registration. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 19–25 June 2021; pp. 11748–11757.
9. Poiesi, F.; Boscaini, D. Distinctive 3D Local Deep Descriptors. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 5720–5727.
10. Johnson, A.E.; Hebert, M. Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes. *IEEE Trans. Pattern Anal. Mach. Intell.* **1999**, *21*, 433–449. [CrossRef]
11. Rusu, R.B.; Blodow, N.; Marton, Z.C.; Beetz, M. Aligning Point Cloud Views Using Persistent Feature Histograms. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008; pp. 3384–3391.
12. Rusu, R.B.; Blodow, N.; Beetz, M. Fast Point Feature Histograms (FPFH) for 3D Registration. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 3212–3217.
13. Albarelli, A.; Rodolà, E.; Torsello, A. Fast and Accurate Surface Alignment through an Isometry-Enforcing Game. *Pattern Recognit.* **2015**, *48*, 2209–2226. [CrossRef]
14. Zhao, H.; Tang, M.; Ding, H. HoPPF: A Novel Local Surface Descriptor for 3D Object Recognition. *Pattern Recognit.* **2020**, *103*, 107272. [CrossRef]
15. Stein, F.; Medioni, G. Structural Indexing: Efficient 3-D Object Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **1992**, *14*, 125–145. [CrossRef]
16. Frome, A.; Huber, D.; Kolluri, R.; Bülow, T.; Malik, J. Recognizing Objects in Range Data Using Regional Point Descriptors. In *Proceedings of the European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 224–237.
17. Belongie, S.; Malik, J.; Puzicha, J. Shape Matching and Object Recognition Using Shape Contexts. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 509–522. [CrossRef]
18. Zaharescu, A.; Boyer, E.; Varanasi, K.; Horaud, R. Surface Feature Detection and Description with Applications to Mesh Matching. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 22–24 June 2009; pp. 373–380.
19. Prakhya, S.M.; Lin, J.; Chandrasekhar, V.; Lin, W.; Liu, B. 3DHoPD: A Fast Low-Dimensional 3-D Descriptor. *IEEE Robot. Autom. Lett.* **2017**, *2*, 1472–1479. [CrossRef]

20. Guo, Y.; Sohel, F.; Bennamoun, M.; Wan, J.; Lu, M. An Accurate and Robust Range Image Registration Algorithm for 3D Object Modeling. *IEEE Trans. Multimed.* **2014**, *16*, 1377–1390. [CrossRef]
21. Guo, Y.; Sohel, F.A.; Bennamoun, M.; Lu, M.; Wan, J. Trisi: A Distinctive Local Surface Descriptor for 3d Modeling and Object Recognition. In Proceedings of the International Conference on Computer Graphics Theory and Applications, Barcelona, Spain, 21–24 February 2013; pp. 86–93.
22. Tombari, F.; Salti, S.; Stefano, L.D. Unique Signatures of Histograms for Local Surface Description. In *Proceedings of the European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 356–369.
23. He, Y.; Yang, J.; Hou, X.; Pang, S.; Chen, J. ICP Registration with DCA Descriptor for 3D Point Clouds. *Opt. Express* **2021**, *29*, 20423–20439. [CrossRef] [PubMed]
24. The Stanford 3D Scanning Repository. Available online: http://graphics.stanford.edu/data/3Dscanrep/ (accessed on 30 May 2022).
25. On the Repeatability and Quality of Keypoints for Local Feature-Based 3D Object Retrieval from Cluttered Scenes. Available online: https://link.springer.com/article/10.1007/s11263-009-0296-z (accessed on 5 June 2022).
26. Guo, Y.; Bennamoun, M.; Sohel, F.; Lu, M.; Wan, J.; Kwok, N.M. A Comprehensive Performance Evaluation of 3D Local Feature Descriptors. *Int. J. Comput. Vis.* **2016**, *116*, 66–89. [CrossRef]