Article

Efficient 3D Object Recognition from Cluttered Point Cloud

Wei Li ¹, Hongtai Cheng ^{2,*} and Xiaohua Zhang ¹

¹ Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, China; davidlee83@mail.dlut.edu.cn (W.L.); xh_zhang@dlut.edu.cn (X.Z.)

² School of Mechanical of Engineering and Automation, Northeastern University, Shenyang 110167, China

* Correspondence: chenght@me.neu.edu.cn

Abstract: Recognizing 3D objects and estimating their postures in a complex scene is a challenging task. Sample Consensus Initial Alignment (SAC-IA) is a commonly used point cloud-based method to achieve such a goal. However, its efficiency is low, and it cannot be applied in real-time applications. This paper analyzes the most time-consuming part of the SAC-IA algorithm: sample generation and evaluation. We propose two improvements to increase efficiency. In the initial aligning stage, instead of sampling the key points, the correspondence pairs between model and scene key points are generated in advance and chosen in each iteration, which reduces the redundant correspondence search operations; a geometric filter is proposed to prevent the invalid samples to the evaluation process, which is the most time-consuming operation because it requires transforming and calculating the distance between two point clouds. The introduction of the geometric filter can significantly increase the sample quality and reduce the required sample numbers. Experiments are performed on our own datasets captured by Kinect v2 Camera and on Bologna 1 dataset. The results show that the proposed method can significantly increase (10–30×) the efficiency of the original SAC-IA method without sacrificing accuracy.

Citation: Li, W.; Cheng, H.; Zhang, X. Efficient 3D Object Recognition from Cluttered Point Cloud. *Sensors* **2021**, *21*, 5850. <https://doi.org/10.3390/s21175850>

Academic Editor: Marco Leo

Received: 14 July 2021

Accepted: 26 August 2021

Published: 30 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: object recognition; point cloud; SAC-IA; RANSAC

1. Introduction

Object recognition and localization are essential functionalities for autonomous robot working in unstructured applications [1]; generally, fast and accurate 3D posture information retrieval is important to tasks, such as material handling, manipulation, assembly, welding, etc [2].

Although 2D image [3]/3D point cloud can all be used to achieve such goal, image-based methods are sensitive to ambient lighting conditions, object texture, and have to estimate 3D information indirectly; hence, point cloud-based methods are becoming promising especially with the emerging of many affordable depth sensors, such as Microsoft Kinect and Intel Realsense cameras.

According to the used features, there are two kinds of pipelines for recognizing 3D objects from point clouds: the global pipeline and the local pipeline. Global feature descriptors, such as ensemble of shape functions (ESF) [4], global fast point feature histogram (GFPFH) [5], Viewport Feature Histogram (VFH) [6], and cluster view feature histogram (CVFH) [7], can be used to evaluate an object observed from different view angles. By generating candidate model clouds all around the object/CAD and extracting their features, the segmented scene cloud can be evaluated and recognized, given some thresholds. The coarse postures can further be refined using Iterative Closest Point (ICP) method [8]. Global feature-based algorithms are efficient in computation time and memory consumption. However, these methods require segmenting the object point cloud from the scene and are difficult to be applied in a cluttered environment with occlusion and complex foreground and background [9].

On the contrary, the local feature descriptors only model the surface characteristics around a single point. Because the points are generally located all around the object

and cannot be blocked entirely, hence, they are more robust in occlusion and cluttered environment. The local pipeline mainly consists of keypoint detection, feature matching, hypothesis generation, and evaluation.

Johnson et al. [10] proposed a pipeline using “Spin image (SI)” to identify the objects. SI is a data level shape descriptor which transforms the point cloud into a stack of 2D spin images, and then uses them to represent the 3D model. By matching the model/scene spin images, and filtering and grouping these correspondences, the transformation can be established.

Guo et al. [11] introduced a local descriptor Rotational Projection Statistics (RoPS) to recognize object in the scene, by constructing local reference frames and projecting neighbor points on the reference plane and using statistical values to describe the local feature. It is robust to noise and varying mesh resolution. By selecting some feature points and calculating their feature descriptors, the candidate models are evaluated one-by-one to recognize possible objects.

Rusu proposes Point Feature Histograms (PFH) [12] and Fast Point Feature Histograms (FPFH) [13] descriptors and Sample Consensus Initial Alignment (SAC-IA) initial align method for point cloud registration and object recognition. The use of histograms makes the descriptor invariant view port differences and is robust to noises, hence its widely being used in a lot of registration and matching tasks (usages).

In order to improve the performance, Buch proposes to use both 2D image data and 3D contextual shape data to increase the quality of the correspondences [14]. Results show that the efficiency is significantly improved.

It is noticed that, although local featured-based algorithms have advantage in dealing with occlusion, they require additional time to establish correspondence between model and scene. This process is typically done through Random Sample Consensus (RANSAC). Commonly used initial registration methods for the point cloud mainly include SAC-IA and Super4PCS. Both of these algorithms are implemented based on RANSAC [15]. The RANSAC algorithm mainly uses the idea of multiple sampling iterations to obtain the optimal solution. The algorithm can provide reliable parameter estimates when there are abnormal points in the sample data. The algorithm is highly robust, so it is widely used in machine vision. The field of image registration and object recognition. However, if the number of abnormal points exceeds half, the number of iterations will increase exponentially. These shortcomings will make the algorithm difficult to be applied in actual engineering.

To overcome above shortcomings, some researchers have improved the classic RANSAC algorithm in terms of model solving, sample pre-test, sample selection, and post-stage optimization. The improved method for model solving is mainly for the improvement of the loss function. It includes the following: M-Estimation [16], LMedS [17] MLESAC [18,19], Mapsac [20], etc. M-Estimation is a method that adds weight to data and converts the model solving problem into a least-squares problem with weight added. LMedS uses the method of optimizing which requires the square of the data points residual and minimizing the median value to obtain the optimal solution of the model. However, the above two methods are useful only when the proportion of external points are less than half. Torr et al. used the distribution of inner point/outer point to evaluate the assumption and proposed a method based on maximum likelihood estimation to express the loss function MLESAC to find the optimal solution of the model. After that, the MLESAC method was improved by Torr et al., who proposed Mapsac method. In the improved method, the loss function has been described by Bayesian and maximum posterior probability. The new RANSAC method improves the robustness of the classic algorithm.

For the sample pre-test, the improvement is mainly checking the sample and eliminating the bad sample. The computing time can be decreased by this improvement. RANSAC-Tdd [21] adds the pre-test step and tests the accuracy of the model parameters that are built from the pre-sample. Mates uses conditional probability to describe the quality of each matching point, as well as uses the threshold to select the model, and

it proposes the RANSAC-SPRT algorithm [22,23]. The ARRSAC algorithm uses interior points to generate additional hypothesis evaluations after model parameter evaluation. This method uses a competitive evaluation framework to reduce the running time of local optimization, and it improves the efficiency of the algorithm [24].

For the sample selection, it can be divided into sorting method (PROSAC [25]) and positional relationship method (NAPSAC [26] and GROUPSAC [27]), etc., according to the selection strategy.

For the post-stage optimization method, the current optimal solution is used as a starting point to further optimize the model to reduce the number of iterations and improve the efficiency and stability of the algorithm. Some famous methods include LO-RANSAC [28] and OPTIMAL-RANSAC [29].

With the development of machine learning, some researchers combine the reinforcement learning and deep learning with RANSAC. DSAC [30] AND NG-SAC [31] have been proposed to improve the efficiency and accuracy.

In summary, to improve the efficiency and accuracy of point cloud coarse registration, and further improve the three-dimensional object recognition algorithm based on local features, it is necessary to start with the improved RANSAC algorithm.

SAC-IA algorithm is a commonly used architecture for local feature-based methods. However, its efficiency and accuracy is limited in complex cluttered environment. Therefore, the goal of this paper is to improve its performance. The paper analyzes the most time-consuming part of the SAC-IA algorithm: sample generation and evaluation, and proposes two improvements to increase efficiency. The contributions are summarized as follows.

1. In the initial aligning stage, instead of sampling the key points, the correspondence pairs between model and scene key points are generated in advance and chosen in each iteration, which reduces the redundant correspondence search operations.
2. A geometric filter is proposed to prevent the invalid samples to the evaluation process, which is the most time-consuming operation because it requires transforming and calculating the distance between two point clouds. The introduction of the geometric filter can significantly increase the sample quality and reduce the required sample numbers.

2. The Original SAC-IA Approach

A diagram of the SAC-IA is shown in Figure 1. The modifications lie at the sampling and fine refinement process. Instead of sampling the keypoints, the correspondence is sampled. In addition, a geometric constraint is proposed to filter the “impossible” samples in advance to increase the efficiency. In order to overcome the problem brought by a small object in a large scene and avoid to stuck in local minimum, an adaptive ROI extraction operation is proposed to balance restrict the ICP in a local area of the scene point cloud.

2.1. Keypoints Detection

Typically, there are thousands of points in a point cloud. Recognizing an object with all of them is complex and not necessary. A commonly used approach is to focus on the more distinctive points instead of ordinary ones. The simplest keypoints detection method is uniform subsampling. However, this operation does not consider the distinctive information among the points; hence, the obtained keypoints are not informative and lack in repeatability.

Figure 1. Diagram of the original SAC-IA algorithm and the proposed efficient SAC-IA algorithm.

Several more sophisticated keypoints detection methods are available. Narf 3D keypoint detector [32] is a method of extracting keypoints in the range image. It can only be used in range image captured from a single perspective and cannot be used in point cloud generated from multiple view angles. Harris 3D keypoint detector and SIFT Keypoints [33] follow similar ideas as their 2D versions. Harris 3D uses surface normals to replace the image gradient, while SIFT 3D still has the scaling and rotation invariant features. ISS3D keypoints [34] is an effective keypoints detection method, which mainly uses eigenvalues to select key points. A local coordinate frame is defined for each point using its neighbors within a radius r_{frame} . A 3×3 scatter matrix is generated, and the eigenvalue $\lambda_1, \lambda_2, \lambda_3 (\lambda_1 > \lambda_2 > \lambda_3)$ and vectors are obtained. Then, keypoints can be picked out by evaluating those eigenvalues with some thresholds

$$\begin{aligned} \lambda_2^2 / \lambda_1^2 &< \gamma_{21} \\ \lambda_3^2 / \lambda_2^2 &< \gamma_{32} \end{aligned} \quad (1)$$

In this paper, ISS3D keypoints is chosen because of its efficiency and robustness (details in comparison can be found in Reference [35]).

2.2. Local Shape Descriptors

In order to identify and estimate the pose of an object, it is necessary to find correspondence between model point cloud and scene point cloud. Different from global feature descriptors, local shape descriptors encode features of a point instead of the whole point clouds. Most of the descriptors use the histogram to achieve viewport invariance. They accumulate geometric or topological measurements into histograms according to a specific domain (e.g., point coordinates, geometric attributes) to describe the local surface.

The “spin image (SI)” [11], introduced by Johnson and Hebert, is a typical spatial distribution histogram-based descriptor. The “spin image” descriptor uses the 2D data to represent the 3D feature, which counts the 3D points in a surrounding supporting volume, and then produces a 2D histogram. Therefore, this method will lose valuable 3D shape information.

Signature of Histogram of Orientations (SHOT) [36] encodes histograms of the surface normals in different spatial locations. Therefore, it has the advantage of rotational invariance and robustness to noise. In addition, Prakhya made improvements by converting the real value vector to the binary value vector based on SHOT descriptor and named Binary Signatures of Histogram of Orientations (B-SHOT) [37].

Rusu et al. [13] proposed Point Feature Histogram (PFH) descriptor. This descriptor calculates the normal vector angle and distance between any two points in the search radius of the keypoints to generate the histogram. This descriptor has a computational complexity of $O(nk^2)$, hence its not being suitable for real-time application. Fast Point Feature Histogram (FPFH) is proposed by Rusu [14] to resolve this problem; FPFH only encodes the relation between the querying point and its k neighborhood points. Therefore, its computational complexity is $O(nk)$. FPFH descriptor is a robust and efficient feature to characterize the local geometry around a point. In this paper, it is used to measure the similarity between different keypoints.

2.3. Initial Alignment

For a solid object, at least three corresponding points are required to estimate the transformation matrix between the model and scene point cloud. It is not easy to find the correct match due to similarities in different key points and uncertainties in the point clouds. Although the number of keypoints is fewer than the original point cloud, the search space is still large. Assume there are M model key points and N scene keypoints, and, for each model keypoint, K candidates are considered; the total search space is $M(M-1)(M-2)K^3 \approx M^3K^3$, which is 125,000, even with 10 keypoints and 5 candidates.

Compared to brute grid search, the random search is more efficient. One such coarse registration method for 3D point clouds is Sample Consensus Initial Alignment (SAC-IA) [14]. It consists of the following four steps:

1. Keypoints Sampling. Select three sample points from the model point cloud, and ensure that their distances are greater than the user-defined minimum distance d_{min} .
2. Correspondence Searching. For each sample point, find a list of points (K candidates) in the scene point cloud with a similar local descriptor. This is usually done by searching KD-Tree. Finally, randomly choose one point from the candidates as the correspondence pair.
3. Transformation Matrix Estimation. With three correspondence pairs, it is able to estimate the transformation matrix T_j , where j is the sampling iteration index.
4. Performance Evaluating. Transfer the model point cloud with T_j and compare it against the scene point cloud. Compute an error metric based on those two point clouds using a Huber penalty measure

$$E_j = \sum_{i=1}^n H(l_i)1, \quad (2)$$

where E_j is the overall measure,

$$H(l_i) = \begin{cases} \frac{1}{2}l_i^2 & \|l_i\| < m_i \\ \frac{1}{2}m_i(2\|l_i\| - m_i) & \|l_i\| > m_i \end{cases}, \quad (3)$$

and m_i is a predefined threshold used to exclude the outliers, while l_i is the minimum distance between a model point and scene point cloud.

Finally, by repeating the above four processes, the true correspondence is assumed to definitely occur. The robustness is proportional to the trial number. However, too many trials will decrease the efficiency; hence, there is a trade-off between stability and efficiency. In spite of that, SAC-IA can still get a good result after many iterations and be widely used in the initial alignment process.

2.4. Fine Alignment

SAC-IA randomly chooses three corresponding points in a model and scene point cloud and obtains the best guess for the object location. However, due to the uncertainty in keypoint extraction and sensors, the obtained transformation matrix may deviate from the real one. Refinement has to be performed to the aligned model and scene with all points involved.

Iterative Closest Point (ICP) [9] is the commonly used registration method. It is based on point-to-point registration and uses LM to calculate the transformation matrix. ICP has good precision when aligning two similar point clouds with a small initial posture error. If the initial posture differs too much, it may stuck in local minimum. Therefore, typically, ICP follows a coarse alignment method, such as SAC-IA.

2.5. The Overall Workflow of Standard SAC-IA Algorithm

The above process can be summarized in the following Algorithm 1. By providing model point cloud and scene point cloud, the object pose is returned in the form of a transformation matrix.

Algorithm 1 Standard SAC-IA Pose Estimation Algorithm.

Input: The model point cloud: PC_m

The scene point cloud: PC_s

Output: The transformation matrix of the best match: T^*

- 1: Downsampling $PC_m \rightarrow \hat{PC}_m$ and $PC_s \rightarrow \hat{PC}_s$
 - 2: Detecting keypoints KP_m, KP_s
 - 3: Calculating FPFH descriptors PD_m, PD_s
 - 4: **for** $j = 1$ to n_{max} **do**
 - 5: Randomly choose three points $P1_m^j, P2_m^j, P3_m^j$
 - 6: Filter samples by distance threshold d_{min}
 - 7: For Pi_m^j , find n_K similar points Pi_s^j from KP_s
 - 8: Randomly choose one Pi_s^j for Pi_m^j
 - 9: Estimate the transformation T^j with $Pi_m^j \rightarrow Pi_s^j$
 - 10: Transform $\hat{PC}_m$, count inliers with threshold d_{max}
 - 11: **end for**
 - 12: Find the transformation matrix $\bar{T}^*$ with most inliers
 - 13: Using ICP to find the optimal transformation T^*
 - 14: **return** T^*
-

It is noted that the model processing, keypoint detection, and FPFH calculation can be done offline. Therefore, the scene point processing and the loops 3–8 take most of the computation time.

2.6. The Problems

The standard SAC-IA algorithm depends on RANSAC to estimate the model. Some limitations of conventional RANSAC will affect the efficiency.

1. Random sampling:
The sampling is performed on the keypoint level. For a typical configuration, a model set includes 50,000 points. To find their correspondence points, 5 candidates are explored. This means KD-tree search operation is performed totally, which will result in 750,000 times.
2. Transform matrix estimation and evaluation:
In standard SAC-IA, the models are represented as a transformation matrix. One has to optimize the matrix parameters and then transform all point clouds and calculate their distances to the scene point cloud. Therefore, this estimation and evaluation process is time-consuming. Moreover, this operation is repeated for n_{max} times for robustness consideration, which greatly restricts its efficiency

3. The Efficient SAC-IA Approach

3.1. The Workflow of Improved SAC-IA Algorithm

The efficiency is limited by the sampling process. In order to overcome this difficulties, two main aspects are improved. (1) Instead of picking samples from the key points, the correspondence pairs are randomly selected. (2) A geometric constraint is proposed to filter the “impossible” samples in advance, to increase the efficiency. The Efficient approach is improved on the basis of Algorithm 1, see below Algorithm 2.

Algorithm 2 Efficient SAC-IA Pose Estimation Algorithm.

Input: The model point cloud: PC_m

The scene point cloud: PC_s

Output: The transformation matrix of the best match: T^*

```

1: Downsampling  $PC_m \rightarrow \hat{PC}_m$  and  $PC_s \rightarrow \hat{PC}_s$ 
2: Detecting keypoints  $KP_m, KP_s$ 
3: Calculating FPFH descriptors  $PD_m, PD_s$ 
4: For each keypoint in  $KP_m$ , Build correspondence map  $C_{ij} = \langle P_{m_i}, P_{s_{ij}} \rangle, i = 1, \dots, n_m, j = 1, \dots, n_K$ 
5: for  $k = 1$  to  $n_{max}$  do
6:   repeat
7:     Choose  $C_k = [C_{i_1j_1}, C_{i_2j_2}, C_{i_3j_3}]$ ;
8:     Filter  $C_k$  by threshold and triangle similarity
9:   until  $C_k$  is valid
10:  Estimate the transformation  $T^k$  with  $C_k$ 
11: end for
12: Find the transformation matrix  $\hat{T}^*$  with most inliers
13: Using ICP to find the optimal transformation  $T^*$ 
14: return  $T^*$ 

```

3.2. The Sampling Process

In standard SAC-IA process, the sampling is performed on the keypoint level. Keypoints are randomly chosen firstly and then try to find their correspondences based on FPFH similarity. Usually, three points are used to estimate the transformation matrix. In order to increase the robustness, usually, n_k candidates are explored. This means the KD-tree search operation is performed for a total of $n_{max} \times n_k \times 3$ times.

It is noticed that much of these queries are redundant. As shown in Figure 2, since there are totally n_m keypoints in the model point cloud, the possible correspondence pairs are $n_m \times n_K$. Usually, n_m is significantly smaller than n_{max} ; hence, the above cost can be eliminated by changing the sampling target and sequences. If there are 100 keypoints in the model point cloud and $n_k = 5$, the search queries only 500 times.

Figure 2. Finding the correspondence pairs in scene point cloud.

Based on this observation, the following sampling method is proposed.

1. **One Shot Correspondence Map Building.** Before the random sampling iteration loop, for each keypoint P_{m_i} in KP_m , search their n_K most similar keypoints $P_{s_{ij}}$ in KP_s based on FPFH descriptor. This is done by performing nearest neighbor search in the KD-tree. Then, a correspondence map $C_{ij} = \langle P_{m_i}, P_{s_{ij}} \rangle, i = 1, \dots, n_m, j = 1, \dots, n_K$ can be derived.
2. **Correspondence Level Sampling.** Instead of sampling keypoints, the correspondence pairs are sampled directly. Since there is no need to evaluate their similarities, the sampling efficiency can be accelerated. In each iteration, three pairs $C_k = [C_{i_1j_1}, C_{i_2j_2}, C_{i_3j_3}]$ are randomly picked out. These samples need to be filtered before evaluation. First of all, their indices should not be equal to each other; second, their distances should be bigger than a threshold d_{min} . Third, those three points should not be in a line. In addition to the above constraints, this paper proposes a new filtering criteria to improve the correspondence quality: Triangle Similarity Constraint-based sample filter.

3.3. Triangle Similarity Constraint Based Sample Filter

In order to reduce the matching time, the samples have to be selected carefully before sending to estimation and evaluation. Standard SAC-IA only considers the distances between model keypoints by setting a minimum threshold. This can ensure that the sample points are far apart, which is beneficial for the matrix estimation process. However, this condition is too loose to filter out “bad” samples. These points can fulfill the filter conditions, but these points and their similarities in the scene cannot calculate the transformation matrix.

In this paper, another constraint is proposed based on triangle similarity. Because, in each iteration, three correspondences are selected which involve three model points and three scene points, they are expected to correspond to each other in pairs. As shown in Figure 3, this assumption leads to a fact that there are two triangles in the model and scene, and they should be similar/identical, i.e., their side lengths should be the same.

Figure 3. The assumption of geometric constraint, two similar triangles in a model and scene point cloud.

This geometric constraint is based on solid basis and can filter out many invalid samples. A direct benefit of filtering out many “bad” samples is avoiding many useless transformation matrix estimations and evaluations, increasing the quality of the samples and decreasing the iteration number n_{max} .

The Triangle Similarity-based sample generation and filtering process is as follows:

1. **Sampling:** Three pairs $C_k = [C_{i_1j_1}, C_{i_2j_2}, C_{i_3j_3}]$ are randomly picked out.
2. **Removing Same Points:** $i_1 \neq i_2, i_1 \neq i_3, i_2 \neq i_3, j_1 \neq j_2, j_1 \neq j_3, j_2 \neq j_3$.
3. **Filtering Close Samples:** $|P_m^{i_1}, P_m^{i_2}| > d_{min}, |P_m^{i_1}, P_m^{i_3}| > d_{min}, |P_m^{i_2}, P_m^{i_3}| > d_{min}$.

4. Filtering Dissimilar Samples: In order to increase the stability, the following measure is proposed to evaluate the similarity:

$$d_{ij} = \left\| \begin{bmatrix} P_m^{i1}, P_m^{i2} \\ P_m^{i1}, P_m^{i3} \\ P_m^{i2}, P_m^{i3} \end{bmatrix} - \begin{bmatrix} P_s^{j1}, P_s^{j2} \\ P_s^{j1}, P_s^{j3} \\ P_s^{j2}, P_s^{j3} \end{bmatrix} \right\|. \quad (4)$$

If $d_{ij} > d_{similar}$, this sample will be removed, and a resampling process is performed until a valid sample is acquired.

d_{min} and $d_{similar}$ are two parameters used in the candidate pair filtering process. They are tuned according to the sensor performance and object size. d_{min} should not smaller than the object radius and large enough to ensure the pairs can make robust estimation of the object posture. $d_{similar}$ should be small enough to filter out more “bad” samples. However, if it is too small, because of sensor noises, it may also reduce the possibility of finding potential objects. Therefore, in real application, these two parameters have to be tuned accordingly.

4. Experiments

4.1. Experimental Setup

A Kinect v2 Camera is used to capture the point clouds of object. The algorithm is implemented using C++ and Point Cloud Library (PCL) in Windows 10. The computer is equipped with an Intel Core i7-8650U CPU (2.11 GHz, 4 cores) and 16 GB memory.

A stackable storage bin is used as the test object, whose CAD is shown in Figure 4. By placing the box randomly on the table and grabbing it with different poses (with occlusion), point clouds are captured, as shown in Figure 4. Totally, there are 17 scene point clouds in the dataset.

4.2. Efficiency Verification Experiments

The proposed Efficient SAC-IA (E-SAC-IA) and original SAC-IA were both implemented for comparison. The used parameters are listed in Table 1.

The CAD file is converted into a point cloud by subdividing the triangular meshes; therefore, the point density is not even. To resolve this issue, a fine resolution is firstly used and then downsampled with a leaf size of 0.01 m, to match the scene point clouds captured with real sensor.

In the implementation of Efficient SAC-IA, instead of using specific keypoint detector, the generic subsampling operation is used. By setting the leaf size as 0.04 m, a sparse point cloud is generated from the original density one. The remaining points are treated as keypoints. For model point cloud, there are 160 key points, while, for the scene point clouds, typically, 370 keypoints are available.

Figure 4. The object CAD model and recorded point cloud datasets. (a) The CAD model. (b) Point cloud model of the object; different pose with occlusion is shown in (c,d).

Table 1. The used parameter values in the Algorithm.

Parameter	Value
Voxel Leaf Size (m)	0.01
Downsample Size (m)	0.04
Number of Points in Model (subsampling)	2101
Number of Points in Scene (subsampling)	≈6000
Number of Model Keypoints	160
Number of Scene Keypoints	≈370
Randomness (K)	5
Triangle Similarity Threshold $d_{similar}$ (m)	0.06
Minimum Sample Point Distance d_{min} (m)	0.03
Max Iteration Number (proposed)	50; 100; 200; 300; 400; 500; 600
Max Iteration Number (original)	50,000
Thread number for OpenMP	4

4.2.1. Performance against Max Iteration Number

Max iteration number is the key to control the robustness and efficiency of the RANSAC-based algorithms. For the proposed E-SAC-IA, the performance changes against max iteration number are evaluated. The 17 scene point clouds are fed into the algorithm with different max iteration numbers (50; 100; 200; 300; 400; 500; 600). The results are given in Figure 5.

Figure 5a shows the success rate and recognition time when changing the max valid sample number from 50 to 600. It is seen that, with the increase of max valid sample number, the success rate increases. When 500 more samples are used, the success rate is above 90% percent. In addition, the recognition time is proportional to the sample number. The average recognition time is around 200–300 ms.

Figure 5b shows the ratio between total samples and valid samples. The valid samples are samples that fulfill the minimum distance and triangular similarity constraints. It is found that the ratio is 100:1 and nearly keeps constant with the change of max valid sample number parameters.

Due to the randomness and variations between each scene point cloud, the actual recognition time may changes. Figure 5c shows that mean and standard division of the recognition time. The standard division is 30 ms.

In comparison, the original SAC-IA is also tested. The max sample number is chosen as 50,000, to guarantee the success rate. The success rate is 0.941, while the recognition time is shown in Figure 5d. It can be found that the recognition time is around 10 s, which is nearly 30 times that of the proposed E-SAC-IA (Algorithm 2).

Figure 5. The performance comparison between the traditional and improved SAC-IA algorithm.

4.2.2. Performance against Sampling Order

In spite of the iteration number in the RANSAC procedures, the sampling order and type also affect the efficiency. As previously mentioned, in conventional SAC-IA algorithm, the key points are randomly selected in each iteration, and then, by finding their nearest neighbors through Kd-Tree search, the correspondence is established online. This nearest neighbor search is queried for the same times as the iteration number, which is significantly larger than the keypoints number. Therefore, by moving the correspondence building process outside the loop and randomly selecting point pairs, this issue can be eliminated.

In Figure 5d, the recognition time of three configurations is given. The original SAC-IA with sample number is 50,000, and the E-SAC-IA with sample number is 500 (equivalent to about 50,000 total samples). The last one is the original SAC-IA with the above improved sample selection process, but without sample filters.

From the results, one can see that reorganizing the sampling order can also improve the overall efficiency. Compared to original SAC-IA, it saves about 2.5 s for 50,000 iterations, i.e., 50 μ s for each nearest neighbor search operation.

4.3. Test on Bologna 1 Dataset

In order to verify the performance on objects with different shapes, the algorithm is tested on Bologna 1 dataset [37]. These datasets, created from some of the models belonging to the Stanford 3D Scanning Repository, are composed of 6 models (armadillo, buddha, bunny, Chinese dragon, dragon, and statuette, as shown in Figure 6) and 45 scenes. Each scene contains a subset of the 6 models that are randomly rotated and translated. The 45 scenes belong to three types: 3 objects, 4 objects, and 5 objects. Each type has 15 scenes.

Figure 6. The object models in the Bologna 1 datasets.

In this experiment, ISS algorithm is used to detect the keypoints. The configuration is listed in Table 2.

Table 2. The used parameters in the Bologna experiments.

Parameter	Value
Voxel Leaf Size (m)	0.005
ISS Salient Radius (m)	0.018
ISS Non-Max Radius (m)	0.012
Number of Points in Model (subsampling)	2014; 2635; 3017; 3929; 1913; 2279
Number of Points in Scene (subsampling)	≈ 8000 ; $\approx 11,000$; $\approx 14,000$
Number of Model Keypoints	30; 46; 42; 44; 17; 48
Number of Scene Keypoints	≈ 110 ; ≈ 140 ; ≈ 180
Randomness (K)	5
Triangle Similarity Threshold $d_{similar}$ (m)	0.1
Minimum Sample Point Distance d_{min} (m)	0.03
Max Iteration Number (proposed)	1000
Max Iteration Number (original)	10,000
Thread number for OpenMP	4

For each scene, no matter whether it contains specific model, it is processes to detect all those models. The detection time is averaged and listed in Table 3. Because of the ISS keypoint detector, the keypoints are less than previous experiments. Therefore, the overall runtime is lower. With the increase of points in the scene, the detection time increases slightly. The recognition time is below 200 ms for the proposed E-SAC-IA method, while, for the original SAC-IA method, the recognition time is around 2000 ms.

Table 3. Recognition time of E-SAC-IA and Original SAC-IA Algorithm.

Model Name	E-SAC-IA Runtime (ms)			Original SAC-IA Runtime (ms)		
	Scene 3	Scene 4	Scene 5	Scene 3	Scene 4	Scene 5
armadillo	116.8	127.9	133.3	1905.6	1950.0	1979.0
buddha	170.0	180.0	183.4	2423.4	2471.0	2487.5
bunny	158.2	169.7	174.7	2290.6	2319.6	2359.9
Chinese Dragon	169.5	169.6	178.4	2333.3	2377.2	2428.6
Dragon	76.0	84.8	88.5	1500.0	1576.0	1587.3
statuette	183.0	193.0	197.4	2447.0	2508.1	2557.5

The acceleration and success ratio are listed in Table 4. One can find that it is within the range of 11–18 times.

Table 4. Efficiency and success ratio of the E-SAC-IA to original SAC-IA Algorithm.

Model Name	Scene 3	Scene 4	Scene 5	Success Rate of SAC-IA(%)	Success Rate of E-SAC-IA(%)
armadillo	15.31	14.24	13.84	100	100
buddha	13.25	12.72	12.56	96.42	100
bunny	13.47	12.66	12.50	100	100
Chinese Dragon	12.76	13.01	12.61	100	100
Dragon	18.73	17.58	16.93	82.14	82.14
statuette	12.37	11.99	11.95	91.30	91.30

One can find that the success rate is identical for original and improved SAC-IA algorithms. In most cases, the rate is above 90%. As shown in previous experiments, the success rate is related to several issues, such as the max iteration number, the quality of the keypoints, and fuzziness of the scene point cloud.

Therefore, it can be concluded that the proposed E-SAC-IA method indeed can increase the object recognition efficiency without scarifying the success rate. For typical

configuration, the recognition time is about 200–300 ms, i.e., 3–4 fps. Therefore, it can be applied in some real-time applications. Because the method still follows the local pipeline, the keypoint detection, feature calculation, and ICP restriction all cost some time (about 150 ms), and the space to decrease the recognition time is bounded if no GPU acceleration or multiple core paralleling techniques are used. The acceleration ratio is related to the keypoints number, randomness, and point numbers. According to the experimental results, the typical ratio is $10\times$ to $30\times$.

5. Conclusions and Future Works

This paper studies the problem of recognizing 3D objects from cluttered scenes with point cloud information. The bottleneck to the efficiency of the conventional SAC-IA method is identified: the sampling target and the sampling criteria. Accordingly, two improvements are made to reduce the sampling time. Firstly, the correspondences between model keypoints and scene keypoints are established beyond the sampling loop to reduce the feature matching and nearest neighbor search operation. Considering the magnitude difference between correspondence number and max sampling iteration number, this change can save amount of computing time; secondly, a triangle similarity-based geometric constraint is proposed to filter out “bad” samples to prevent them for further evaluation. Considering the computational complexity of sample evaluation operation, this filtering mechanism can significantly reduce the sampling time and increase the sampling efficiency.

Although the efficiency is in some degree improved, the robustness of the method is still restricted by the keypoint detector, feature descriptor and sample filter. In the future, RGB/texture information will be incorporated to increase the distinction between keypoints in the model and environment; The intensity/color/depth keypoints are fused and filtered to provide more robust keypoints for correspondence estimation.

Author Contributions: Methodology, W.L. and H.C.; Software, W.L., H.C.; Supervision, X.Z.; Visualization, X.Z.; Writing—original draft, W.L. All authors have read and agreed to the published version of the manuscript.

Funding: The research is jointly funded by Major Innovation Engineering Project of Shandong Province (2019JZZY010128) and Fundamental Research Funds for the Central Universities (N2104008 and N2003003).

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. Mikolajczyk, K.; Schmid, C. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *27*, 1615–1630. [[CrossRef](#)] [[PubMed](#)]
2. Li, D.; Liu, N.; Guo, Y.; Wang, X.; Xu, J. 3d object recognition and pose estimation for random bin-picking using partition viewpoint feature histograms. *Pattern Recognit. Lett.* **2019**, *128*, 148–154. [[CrossRef](#)]
3. Wang, F.; Liang, C.; Ru, C.; Cheng, H. An improved point cloud descriptor for vision based robotic grasping system. *Sensor* **2019**, *19*, 2225. [[CrossRef](#)] [[PubMed](#)]
4. Ulrich, M.; Wiedemann, C.; Steger, C. Cad-based recognition of 3d objects in monocular images. *ICRA* **2009**, *9*, 1191–1198.
5. Wohlkinger, W.; Vincze, M. Ensemble of shape functions for 3d object classification. In Proceedings of the 2011 IEEE International Conference on Robotics and Biomimetics, Karon Beach, Thailand, 7–11 December 2011; pp. 2987–2992.
6. Rusu, R.B.; Holzbach, A.; Beetz, M.; Bradski, G. Detecting and segmenting objects for mobile manipulation. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision Workshop, Kyoto, Japan, 27 September–4 October 2009; pp. 47–54.
7. Liang, X.; Cheng, H. RGB-D Camera based 3D Object Pose Estimation and Grasping. In Proceedings of the 2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems, Suzhou, China, 29 July–2 August 2019; pp. 1279–1284.

8. Aldoma, A.; Tombari, F.; Rusu, R.B.; Vincze, M. Our-cvfh-oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6DOF pose estimation. In *Joint DAGM and OAGM Symposium*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 113–122.
9. Besl, P.J.; McKay, N.D. Method for registration of 3-D shapes. In *Proceedings of the Robotics '91*, Boston, MA, USA, 14–15 November 1991.
10. Guo, Y.; Bennamoun, M.; Soheli, F.; Lu, M.; Wan, J. 3D object recognition in cluttered scenes with local surface features: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 2270–2287. [[CrossRef](#)] [[PubMed](#)]
11. 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](#)]
12. Guo, Y.; Bennamoun, M.; Soheli, F.; Lu, M.; Wan, J. Rotational projection statistics for 3D local surface description and object recognition. *Int. J. Comput. Vis.* **2013**, *105*, 63–86. [[CrossRef](#)]
13. 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.
14. Rusu, R.B.; Blodow, N.; Beetz, M. Fast point feature histograms (FPFH) for 3D registration. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Kobe, Japan, 12–17 May 2009; pp. 3212–3217.
15. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* **1981**, *24*, 381–395. [[CrossRef](#)]
16. Chen, J.H.; Chen, C.S.; Chen, Y.S. Fast algorithm for robust template matching with M-estimators. *IEEE Trans. Signal Process.* **2003**, *51*, 230–243. [[CrossRef](#)]
17. Meer, P.; Mintz, D.; Rosenfeld, A.; Dong, Y.K. Robust regression methods for computer vision: A review. *Int. J. Comput. Vis.* **1991**, *6*, 59–70. [[CrossRef](#)]
18. Torr, P.; Zisserman, A. MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. *Comput. Vis. Image Underst.* **2000**, *78*, 138–156. [[CrossRef](#)]
19. Xu, F.; Gao, D.; Wang, S.; Zhanwen, A. MLESAC Based Localization of Needle Insertion Using 2D Ultrasound Images. *J. Phys. Conf. Ser.* **2018**, *1004*, 012037. [[CrossRef](#)]
20. Torr, P.H.S. Bayesian Model Estimation and Selection for Epipolar Geometry and Generic Manifold Fitting. *Int. J. Comput. Vis.* **2002**, *50*, 35–61. [[CrossRef](#)]
21. Matas, J.; Chum, O. Randomized RANSAC with T(d,d) test. *Image Vis. Comput.* **2004**, *22*, 837–842. [[CrossRef](#)]
22. Chum, O.; Matas, J. Optimal Randomized RANSAC. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *30*, 1472–1482. [[CrossRef](#)] [[PubMed](#)]
23. Matas, J.; Chum, O. Randomized RANSAC with sequential probability ratio test. *IEEE Int. Conf. Comput. Vis.* **2005**, *2*, 1727–1732.
24. Raguram, R.; Chum, O.; Pollefeys, M.; Matas, J.; Frahm, J.M. USAC: A Universal Framework for Random Sample Consensus. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 2022–2038. [[CrossRef](#)] [[PubMed](#)]
25. Chum, O.; Matas, J. Matching with PROSAC—progressive sample consensus. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Diego, CA, USA, 20–25 June 2005.
26. Myatt, D.R.; Torr, P.; Nasuto, S.J.; Bishop, J.M.; Craddock, R. NAPSAC: High Noise, High Dimensional Model Parameterisation—It's in the Bag. Available online: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.371.8748&rep=rep1&type=pdf> (accessed on 27 August 2021).
27. Kai, N.; Jin, H.; Dellaert, F. GroupSAC: Efficient consensus in the presence of groupings. In *Proceedings of the 2009 IEEE 12th International Conference on Computer Vision*, Kyoto, Japan, 29 September–2 October 2009.
28. Chum, O.; Jiří, M.; Kittler, J. Locally Optimized RANSAC. In *Joint Pattern Recognition Symposium*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 236–243.
29. Hast, A.; Nysjö, J.; Marchetti, A. Optimal RANSAC—Towards a Repeatable Algorithm for Finding the Optimal Set. *J. WSCG* **2015**, *21*, 21–30.
30. Brachmann, E.; Krull, A.; Nowozin, S.; Shotton, J.; Michel, F.; Gumhold, S.; Rother, C. DSAC Differentiable RANSAC for Camera Localization. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 21–26 July 2017.
31. Brachmann, E.; Rother, C. Neural-Guided RANSAC: Learning Where to Sample Model Hypotheses. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea, 27 October–2 November 2019.
32. Steder, B.; Rusu, R.B.; Konolige, K.; Burgard, W. Point feature extraction on 3D range scans taking into account object boundaries. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, 9–13 May 2011; pp. 2601–2608.
33. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [[CrossRef](#)]
34. Zhong, Y. Intrinsic shape signatures: A shape descriptor for 3d object recognition. In *Proceedings of the 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, Kyoto, Japan, 27 September–4 October 2009; pp. 689–696.
35. Guo, Y.; Bennamoun, M.; Soheli, 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](#)]

-
36. Tombari, F.; Salti, S.; Di Stefano, L. Unique signatures of histograms for local surface description. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 356–369.
 37. Prakhya, S.M.; Liu, B.; Lin, W. B-SHOT: A binary feature descriptor for fast and efficient keypoint matching on 3D point clouds. In *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 28 September–2 October 2015; pp. 1929–1934.