Speedup 3-D Texture-Less Object Recognition Against Self-Occlusion for Intelligent Manufacturing

Yang Cong, Senior Member, IEEE, Dongying Tian, Yun Feng, Baojie Fan, and Haibin Yu

Abstract—Realtime 3-D object detection and 6-DOF pose estimation in clutter background is crucial for intelligent manufacturing, for example, robot feeding and assembly, where robustness and efficiency are the two most desirable goals. Especially for various metal parts with a textless surface, it is hard for most state of the arts to extract robust feature from the clutter background with various occlusions. To overcome this, in this paper, we propose an online 3-D object detection and pose estimation method to overcome self-occlusion for textureless objects. For feature representation, we only adopt the raw 3-D point clouds with normal cues to define our local reference frame and we automatically learn the compact 3-D feature from the simple local normal statistics via autoencoder. For a similarity search, a new basis buffer k-d tree method is designed without suffering branch divergence; therefore, ours can maximize the GPU parallel processing capabilities especially in practice. We then generate the hypothesis candidates via the hough voting, filter the false hypotheses, and refine the pose estimation via the iterative closest point strategy. For the experiments, we build a new 3-D dataset including industrial objects with heavy self-occlusions and conduct various comparisons with the state of the arts to justify the effectiveness and efficiency of our method.

Index Terms—Hough voting, hypothesis generation, k-d tree, local reference frame (LRF), object recognition, pose estimation.

I. INTRODUCTION

Detection and pose estimation of everyday objects arising in many practical applications is still a challenging problem [1]–[3], due to noise, low resolution, occlusion, clutter, or other interference factors. In this paper, we focus on 3-D industrial object recognition and 6-DOF pose estimation [4], [5] for online intelligence manufacturing, for example, industrial robot manipulation as shown in Fig. 1. Therefore, both the accuracy and efficiency are our pursuit goals in practice. Generally, our problem has two unique features.

1) Self-Occlusion: In comparison with the existing methods [6]–[8] recognizing on “full” 3-D models, we focus on “online partial” 3-D recognition from a single camera view, where the self-occlusion issue is inevitable, especially for industrial objects with many holes inside.

2) Textureless and Colorless: In contrast to RGB-D-based object recognition methods [9], [10], most industrial objects are metallic with simple shapes, so we only adopt the 3-D point clouds without color information.

Generally, there are three key stages involved.

1) 3-D Object Representation: While most existing methods [6]–[8] extract various features from 3-D point clouds directly, we adopt an efficient virtual camera projection to overcome self-occlusion, which can be more realistic to use the 3-D model to simulate the real scene. In order to maintain efficiency, any complicated computations are abandoned, for example, moment calculation [6], and only the simple point clouds and normal information are adopted to define our local reference frame (LRF) and learn the discriminative 3-D feature automatically.

2) Similarity Search: So far, the only tactical for large scale problems is still based on the nearest neighbor (NN) search; in contrast to supervised learning (e.g., deep
learning), it offers the possibility to trivially add new
objects or remove old ones arbitrarily. Compared with
the CPU-based similarity search, we intend to resort the
graphics processing units (GPUs) for parallelly accelerat-
ing the search, and design a new basis buffer k-d tree
method without traversing the whole tree as [11].

3) Hypothesis Generation and Verification: We first ge-
generate the hypothesis candidates via the hough voting,
filter the false hypotheses, and refine the pose estima-
tion via the iterative closest point (ICP) strategy. A new
3-D industrial objects dataset is also built to justify the
performance of our proposed method.

Generally, the main contributions of this paper are as
follows.

1) Depending on the virtual camera projection, a simple
but robust 3-D feature is proposed without any compli-
cated computation, in which we define a new LRF and
automatically learn a compact low-dimensional feature
via the stacked autoencoder (AE).

2) For similarity search, a new basis buffer k-d tree based
on GPU is designed without suffering branch divergence
as most existing methods, which can achieve an NN
search efficiently.

3) A new 3-D industrial objects dataset is built with heavy
self-occlusion; moreover, by adopting two additional
public 3-D datasets, we justify both the effectiveness
and efficiency of our method accordingly.

The rest of this paper is organized as follows. In Section II,
we review the related works. In Section III, we introduce
our framework, including 3-D feature representation, similarity
search, and hypotheses generation and refinement, respec-
tively. In Section IV, we compare our model with the other
3-D object recognition methods and conclude this paper in
Section V.

II. RELATED WORKS

3-D object detection and pose estimation is a hot topic,
where [12] and [13] survey the 3-D object recognition with
local surface features and content-based 3-D shape retrieval
methods, respectively. Generally, the framework can be sum-
morized into three steps.

A. 3-D Object Representation

The state of the arts can be roughly subdivided into three
categories.

1) Feature-Based Methods: Earlier methods extract 2-D
features [14]–[17] from the RGB image and then project
them to 3-D space [18]–[20]. Recently, most of feature-
based methods depend on the local 3-D surface [8],
such as spin image [21], SHOT [22], normal his-
togram [23], fast geometric point [24], and rotational
projection statistics (RoPS) [6], adjustable sensitivity
feature [25], BOLD [26].

2) Template-Based Methods: These methods usually
combine the RGB and depth information together,
for example, RGB-D kinect camera. For example,
LineMOD [9], [10] performs robust 3-D object detection
based on matching templates generated from rendered
views of the proposed 3-D models and extracting both
image contours and normal orientations. However, the
model complexity increases linearly with the number of
models. To improve the efficiency, [27] achieves ten
times speedup by optimizing the matching via a cas-
daced classification scheme. Kehl et al. [28] proposed
a hashing matching approach based on LineMOD tem-
plates. Another recent approach [29] searches 3-D chairs
depending on part-based 2-D-3-D alignment of the CAD
models.

3) Learning-Based Methods: These methods try to learn
more representative features via various machine-
learning technologies, especially deep learning,
to improve the matching accuracy in practice.

B. Similarity Search

This intends to find the most similar training features related
to queries, which can be considered a K-NN search problem.
However, an absolute K-NN search is too time consuming
and cannot be used for real applications. In order to improve the
efficiency, many works focus on the approximate NN search,
such as local sensitive hashing, latent-class hough forests [4],
various kinds of k-d tree methods, random forest. Recently,
several GPU-based k-d tree methods are proposed [11] to
speed up the search parallelly, for example, the buffer k-d
tree [11]. However, the bottleneck is to suffer the branch
divergence and irregular tree traversal. Therefore, we intend
to avoid the frequent branch split and design a basis buffer
k-d tree method to overcome such an issue.

C. Hypothesis Generation and Verification

Given the pairwise feature matching set, the hypothesis
candidates can be generated by hough voting techniques or
RANSAC. However, false hypothesis candidates are unavoidable
due to various disturbance issues, such as clutter back-
ground, occlusions, or scalable multiobjects. Most existing
methods focus on how to filter false 3-D correspondences and
refine the pose estimation accordingly, for example, the ICP-
based methods [37], [38] achieve this with multimodel cue
integration, [39] proposing a local and global scale outlier
detection method, [40] filtering the 2-D-to-3-D correspondence
efficiently, and [41] designing a global hypotheses verification
method.
Fig. 2. Demonstration of our 3-D object recognition method against self-occlusion. (a) For model training, the 3-D model is projected via virtual camera technology (512 virtual scenes) to overcome self-occlusion. (b) For each testing scene, the LRFs are first detected (red arrows) from local 3-D surface (green rectangle). We then calculate the 147th local normal statistic from $7 \times 7$ local patches, and learn the compact $75d$ feature via AE accordingly. (c) Similarity search is sped up by our basis buffer k-d tree based on GPUs. (d) Effective pairwise correspondences are invited for hough voting, generate the hypotheses, and refine the results accordingly.

Fig. 3. Demonstration of self-occlusion, where some 3-D point clouds of (a) are occluded and are invisible from the camera center view in (b).

III. OUR METHOD

In this section, we describe our 3-D object recognition method. Assume the testing scene is $S$ and the 3-D model is $M$, our problem can be summarized as follows.

1) whether the scene $S$ has the 3-D model $M$ or not
2) if so, how many $M$ in $S$
3) where are they, that is, finding the corresponding 6-DOF pose.

The characteristic of our problem lies in that we focus on the identification of textureless industrial components with self-occlusion, and wish to use it in real applications with a clutter background. Therefore, the efficiency, accuracy, and robustness to self-occlusion are our most desirable goals.

Fig. 2 demonstrates the framework of our method. Generally, there are three key components included: 1) feature representation; 2) similarity search; and 3) hypothesis generation and refinement. In comparison with most state of the arts (e.g., RoPS and Spin image) always collecting feature information from 3-D points directly, the virtual camera projection helps to improve both the accuracy and efficiency of feature matching and boost the performance accordingly. In comparison with the traditional LRF [6], we propose a new simple LRF that is designed without suffering branch divergence as traditional methods, which demonstrates the advantages of the GPU parallel processing capabilities for the similarity search.

3) For hypothesis generation and refinement, the validate pairwise correspondences are invited to generate the hypothesis candidates via hough voting; the false hypothetical candidates are filtered depending on the self-occlusion assumption; then the pose is refined via the ICP strategy.

A. 3-D Feature Representation

1) Local Reference Frame: The LRF [42] intends to define the local three axes of the reference point by adopting its surrounding 3-D neighbor points. An efficiency, repeatable, and robust LRF is crucial for 3-D object recognition, which helps to improve both the accuracy and efficiency of feature matching and boost the performance accordingly. In comparison with the traditional LRF [6], we propose a new simple LRF by adopting the 3-D points of the local surface.

Given the reference point as $p \in \mathbb{R}^3 = [x, y, z]^T$ and a support radius $r$, we first find its $R$-NN from the whole 3-D scene using the sphere of radius $r$ centered at $p$ as $Q = \{q_1, \ldots, q_R\}$ ($R$ is the number of local points). The LRF related to $p$ is denoted as $L = \{\vec{v}_{xp}, \vec{v}_{yp}, \vec{v}_{zp}\}$ ($\vec{v} \in \mathbb{R}^3$).

1) We first decompose $Q$ via the principal component analysis, where the eigenvalues are denoted as $\{\lambda_1, \lambda_2, \lambda_3\}$. The first eigenvector with the smallest eigenvalue is defined as the $z$-axis $\vec{v}_{zp}$ of our LRF.

2) We calculate the average normal of all $Q$ as $\vec{n}_Q$, and define the $y$-axis $\vec{v}_{yp}$ of our LRF as the $\vec{v}_{yp} = \vec{v}_{zp} \times \vec{n}_Q$.

3) We then define $\vec{v}_{xp}$ as $\vec{v}_{xp} = \vec{v}_{yp} \times \vec{v}_{zp}$ in order to achieve an orthogonal coordinate system.

with normal information are used to define our LRF, and then a more condensed 3-D feature is learned via stacked AE with the dimension as $75d$.

2) For similarity search, a new basis buffer k-d tree method is designed without suffering branch divergence as traditional methods, which demonstrates the advantages of the GPU parallel processing capabilities for the similarity search.

3) For hypothesis generation and refinement, the validate pairwise correspondences are invited to generate the hypothesis candidates via hough voting; the false hypothetical candidates are filtered depending on the self-occlusion assumption; then the pose is refined via the ICP strategy.
Moreover, the less informative points, for example, related points on symmetrical local surfaces, are filtered to improve the efficiency depending on the criterion as

\[
\lambda_1 \lambda_2 \lambda_3 \sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2} > \tau_{lf}\ (\lambda_0 + \lambda_2 + \lambda_3)^2
\]

where \(\tau_{lf} = 0.01\) in this paper. It is worth noting that our LRF is very efficient without complicated computation, and we will verify its effectiveness and robustness compared with several existing LRFs [42] in Section IV-D.

2) 3-D Feature Learning: In this section, we propose how to extract the rough 147d feature from normal statistic, and learn a final compact 75d feature via stacked AE for 3-D object presentation.

Given the reference point \(p\) and the related LRF \(L\), we first eliminate the rotation invariance by transforming the surrounding points \(Q\) with respect to \(L\) as \(Q' = \{q'_1, \ldots, q'_k\}\), and the normal of each \(q'_i\) as \(\pi_{q'_i}\). All of the transformed local point clouds \(Q'\) and related normal \(\pi_{q'_i}\) are projected on the XOY plane with respect to the LRF \(L\). A 2-D bounding rectangle is obtained and divided into \(E \times E\) bins (\(E = 7\) in this paper). We represent each bin by the normal statistic calculated by the average normal of local 3-D points falling into each bin, and then concatenate all of the \(E \times E\) bins together to generate a rough feature with the dimension as \(E \times E \times 3 = 147d\) in our case.

In order to represent the 3-D feature well, we adopt the stacked AE [43] to learn a compact feature space in an unsupervised way. Specifically, the AE is a multilayer neural network mainly including an encoder part and decoder part, and encoder projects input sample \(d^l \in \mathbb{R}^{d^l}\) to the hidden layer with a nonlinear function

\[
h^{l+1} = s(W^l d^l + \tau^l)
\]

where \(W^l \in \mathbb{R}^{d^l \times d^{l+1}}, \tau^l \in \mathbb{R}^{d^l}\), and \(h \in \mathbb{R}^{d^{l+1}}\) are the encoder weights, the corresponding bias, and the hidden layer, respectively, and \(s(\cdot)\) is the sigmoid function. The model parameters are optimized by the belief propagation by considering both low reconstruction cost and sparsity regularization. In our case, \(l \in \{1, \ldots, 7\}\) and \(d^7 = 147\) expected for the middle layer \(d^4 = 75\), which is considered as the learned 3-D compact feature.

B. Similarity Search via the Basis Buffer k-d Tree

In this section, we intend to find the pairwise correspondences for 3-D object recognition. Since our training feature size \(N\) is huge to handle self-occlusion via the virtual camera projection, we intend to resort the GPU to parallelly accelerate the search instead of a CPU-based similarity search. In comparison with the parallelized brute-force k-NN, the GPU-based k-d tree is more efficient. For example, the buffer k-d tree [11] delays the querying process by propagating the query samples into the buffers and performing several iterations concurrently to achieve parallel speedup. However, [11] is unsuitable to handle a high-dimensional feature search well, due to the GPU architecture being low efficient for too frequency branch divergence.

Motivated by [11], we propose a new basis buffer k-d tree to maximize the GPU parallel-processing capabilities. Our basic idea is to select the most discriminative bases to represent each leaf, where the tree search is achieved by directly calculating the similarity between the query sample and the bases of each leaf instead of traversing the whole tree as traditional methods. As shown in Fig. 2(c), four parts consist of our basis buffer k-d tree: 1) the top k-d tree, offline trained by CPU-based method; 2) the leaf structure, only the bottom leaves are used in our case; 3) the basis selection model, to select the most representative bases from the corresponding leaf; and 4) a set of buffers, one buffer per leaf, which is used to store query samples propagated in the corresponding leaf. The specific details are as follows.

1) Model Training: The k-d tree is first initialized offline based on the CPU version k-d tree. Let us define the bottom leaves of the k-d tree as \(l_1, l_2, \ldots, l_{k_f}\) (\(L\) is the total leaves number), the number of feature contained in each leaf \(l_i\) as \(N_{li}\), the training size \(N = \sum_{i=1}^{L} N_{li}\). We intend to select the most representative bases from each leaf as \(b_j \in \mathbb{R}^d, j = 1, \ldots, K_b\) with the corresponding radius as \(r_{bh} \in \mathbb{R}\), which can be considered as a dictionary selection problem [44].

2) Online Search: In contrast to [11] combining CPU+GPU for the search, the online search stage of ours is completely based on GPU. The first step is to determine which leaf the query sample \(f\) belongs to, where we only calculate the similarity between query \(f\) and the bases of each leaf instead of traversing the whole tree as traditional methods

\[
\text{flag} = \begin{cases} 
1 & \text{if } \|f - b_j\| \leq r_{bh} \\
0 & \text{otherwise.}
\end{cases}
\]

Then, if flag = 1, we consider it similar to the leaf and propagate this query into a corresponding buffer. To speed up, we split the testing samples into several min-batches, and repeat this process iteratively to fill the corresponding buffer with the buffer size as \(B\), which is similar to an R-NN search. As soon as the buffers are full, the second step is to calculate the nearest similarity between all collected query samples and the training features stored in the corresponding leaf, which is a brute-force K-NN manner. In our case, we set the number of the most representative bases from each leaf as \(K_b = 1\) for efficiency and the buffer size as \(B = 1024\).

Generally for online search, we can see that our basis buffer k-d tree does not encounter frequency branch divergence by calculating the R-NN search, and can maximize GPU parallel processing capabilities, which will be much more efficient especially for high-dimensional data.

C. Hypotheses Generation and Refinement

Suppose there are \(M\) testing features \(f_1\) extracted in the scene, and we search the their pairwise K-NN from \(N\) model features via Section III-B (\(K = 2\) in this paper). We first filter the pairwise correspondences by

\[
s_f = 1 - \frac{d(f, f'_1)}{d(f, f'_2)}
\]

where \(f'_1\) and \(f'_2\) denote the closest and second-closest neighbors, and \(s_f \in [0, 1]\). If the score \(s_f\) is greater than a threshold
The valid pairwise correspondences are invited to vote the model by the general hough voting procedure in the 6-DOF voting space \([x, y, z, \text{yaw}, \text{pitch}, \text{roll}]\), which is grouped into several clusters accordingly. The nonmaximum suppress is adopted to find the local maximums depending on the voting score, where each denotes a hypothesis candidate \((R_c, T_c)\). Due both the true and false model candidates being involved inevitably, we have to exclude the false hypothesis candidates and refine the pose estimation depending on the following criteria:

1) The voting score should be greater than a threshold.
2) Occlusion Assumption: If the hypothesis \((R_c, T_c)\) is valid, the transformed model cannot occlude the background from the camera center view.
3) ICP Refinement: Each hypothesis candidate is refined by the ICP method to calculate the residual score \(\epsilon\). The \(\epsilon\) of the valid hypothesis should be greater than a threshold \(\tau_{icp} = 300\).

The hypothesis candidates only satisfying all of the aforementioned three conditions could be considered as a true detection; and vice versa.

IV. Experiments

In this section, we carry out various comparisons with several representative 3-D object recognition methods, such as RoPS [6], tensor [7], spin image [7], keypoint [8], VD-LSD [45], and the EM-based method [46]. The experiments are first conducted on two public 3-D object recognition datasets (the UWA dataset [8] and the Queen’s dataset [47]) and then on our new 3-D industrial object datasets, which contain multiple objects in each testing scene with clutter background and heavy self-occlusions.

A. UWA Dataset and Results

This dataset [8] includes five 3-D models and 50 real scenes, where each scene is scanned by a Minolta VIVID 910 laser scanner by placing 4–5 different models in the scene.

The statistic recognition results are shown in Table I, where the overall average recognition rate of our method is 98.8%, in comparison with RoPS (98.8%), spin image (87.8%), tensor (96.6%), and EM-based algorithm (97.5%), respectively. Ours also generate a good result for the T-Rex model with heavy occlusion, contrast, most existing methods (e.g., spin image) fail and do not provide the corresponding results. Generally, there are only 2 out of the total 188 objects in the 50 scenes mistakenly recognized with no false positive occurring for other objects in the experiments. These results verify that our method could recognize 3-D objects in complex scenarios with heavy clutter, occlusion, and mesh resolution variation. Moreover, most existing methods, for example, RoPS [6], exclude the Rhino model since it contains large holes and cannot be recognized well; however, ours still generates the recognition rate as 86.2% since ours can handle self-occlusion well.

We then define the occlusion rate as [7]

\[
\text{occlusion} = 1 - \frac{\#\text{model surface patch area in scene}}{\#\text{total model surface area}}.
\]

Fig. 4 demonstrates the recognition rate by varying the occlusion rate from 60% to 90%. Ours outperforms all state of the arts, and achieves a recognition rate as 100% with up to 90% occlusion (only two false recognitions occurred for about a 95% occlusion rate), where RoPS merely gets the recognition rate as 93.1% for 80% occlusion. Fig. 5 demonstrates our recognition results on two sample scenes on the UWAs dataset, where our proposed method can accurately recognize all objects with heavy amounts of occlusion and clutter. The support radius is set as 20 mm for feature extraction. We extract an average of 4000 and 140 000 feature points in the testing scene and the model by virtual camera projection, respectively.

B. Queen’s Dataset and Results

This dataset [47] contains 5 models and 80 real scenes, where each model is generated by registering several range images and each scene is scanned by an LIDAR sensor from a viewpoint by randomly putting 1–5 models in the scene.

The statistic recognition results are proposed in Table II, where the average recognition rate of our proposed method is 97.6, outperforming all the state of the arts, for example, the second best one RoPS (95.4), the third best one VD-LSD (SQ) (83.8), and so on. Especially, ours gets the best result for 4 out of all 5 models, that is, Angle, Gnome, Kid, and Zoe; and
achieves about 100 accuracy for 2 out of 5 models, that is, Angle and Kid. Moreover, for the model of Kid and Zoe, ours generates a great advantage over the second best one, RoPS. Fig. 6 shows our recognition results on two sample scenes on the Queens dataset. We can see that our proposed method can accurately recognize all objects. All of these results justify the robustness and effectiveness of our method especially with heavy amounts of occlusion and clutter. The support radius is 20 mm for feature extraction. We extract an average of 3500 and 100,000 3-D feature points in the testing scene and the proposed 3-D model by virtual camera projection, respectively.

C. Our Industrial 3-D Dataset and Results

We build a new industrial 3-D dataset composed of three industrial models with heavy self-occlusions, for example, several holes inside. These three models are all made by metal materials with textureless and colorless surface. We then scan these three models to generate the 3-D point cloud model. We capture about 24 different scenes by both kinect and industrial-structured light, where each scene contains 2–6 randomly placed objects from multiple models. The 6-DOF of each object in different scenes is labeled groundtruth, so we can compare it for fair evaluation.

Two criterion are adopted for evaluation.

1) The single object recognition rate: if any object is accurately recognized in the scene, it is considered a true positive, and the ratio is defined as

$$\text{Single-acc} = \frac{\# \text{accurately recognized scenes}}{\# \text{total scenes}}. \quad (6)$$

2) The multiple objects recognition rate

$$\text{Multi-acc} = \frac{\# \text{accurately recognized objects}}{\# \text{total objects}}. \quad (7)$$

We compare ours with the RoPS feature [6] using PCL library; for a fair comparison, the procedure of both the similarity search and hypothesis generation are the same as ours. The statistical results are shown in Table III, and we can see that ours is more accurate than RoPS with respect
TABLE III

RECOGNITION RATE ON OUR 3-D INDUSTRIAL DATASET IN COMPARISON WITH THE BASELINE METHOD, RoPS, WHERE BOTH THE SINGLE OBJECT RECOGNITION RATE AND MULTIPLE OBJECTS RECOGNITION RATE ARE EVALUATED. THE BEST RESULTS ARE MARKED BY RED COLOR

| Method | Axle | Gearbox1 | Gearbox2 | Single-obj Avg | Axle | Gearbox1 | Gearbox2 | Multi-object Avg |
|--------|------|----------|----------|----------------|------|----------|----------|------------------|
| Ours   | 53.9 | 71.4     | 90.0     | 74.5           | 59.6 | 77.1     | 85.9     | 75.9            |
| RoPS   | 53.9 | 71.4     | 85.0     | 72.3           | 57.4 | 70.8     | 77.4     | 70.0            |

We demonstrate the recognition results on six different scenes in Fig. 7, where the RoPS fails more times than ours to detect the objects. Since the models in our industrial 3-D dataset contain many holes inside and deduce self-occlusion...
Fig. 8. Compare the robustness of our LRF with the traditional LRFs, where the horizontal axis is the radius varying from 0.1 to 0.4, and the vertical axis denotes the matching score, that is, the greater the value, the better the corresponding method performs. (a) Captured by kinect sensor and (b) captured by structured light.

Fig. 9. Compare the time consumption of ours with the existing buffer k-d tree [11] by (a) tuning the feature dimension \( d \) and (b) increasing the training data size.

Fig. 10. Percentage of the total time occupied by each stage of our method, including 3-D feature extraction, our GPU-based similarity search, hypothesis generation, and hypothesis refinement.

Accordingly, all of these results above can justify the ability of ours to overcome self-occlusion and clutter background for 3-D object recognition.

D. Compare the Robustness of Our LRF

In this section, we intend to justify the robustness of our LRF in comparison with two existing LRFs (SHOT [22] and PS [42]) via our industrial 3-D dataset. For the same scenario, we capture it from two different positions, and use the ICP method to estimate their corresponding transformation parameters as \([R, T]\). We then extract LRFs from the first scene as \( L_{1,i} \) \((i \in \{1, \ldots, M\})\) with the center point as \( p_{1,i} \), project them to the second scene by \([R, T]\), and get the projected LRFs as \( L'_{1,i} \). We also extract LRFs from each corresponding center points \( p_{2,i} \) of the second scene as \( L_{2,i} \). Whether the
Fig. 12. Compare the 6-DOF \((x, y, z, \alpha, \beta, \gamma)\) estimation error of our method using the Queen’s dataset (top row) and UWA dataset (bottom row), where we compare the error of the center position \((x, y, z)\) and each attitude \((\alpha, \beta, \gamma)\), respectively. (a) and (e) Probability of the angle error. (b) and (f) Probability of the position error with the horizontal-axis as the ratio (%) of the center error/minimal external radius. (c) and (g) Accumulate probability of the angle error. (d) and (h) Accumulate probability of the position error.

corresponding LRF is repeatedly detected, it is defined as

\[
score = \sum_{i=1}^{M} sgn_{i}, \quad sgn_{i} = \begin{cases} 
1 & \text{if } \| L'_{1,i} - L_{2,i} \|_F < \tau_c \\
0 & \text{otherwise}
\end{cases}
\]

where \(\tau_c = 0.1\) in our case. Obviously, the higher the value of the matching score, the more robust the corresponding LRF will be. Two cases are shown in Fig. 8: our LRF is compared with the LRF defined in SHOT [22] and PS [42], where ours is more robust than the existing LRFs in both cases by varying the radius from 0.1 to 0.4. Moreover, due to the pointed clouds captured by structured light being more accurate and dense than kinect, the matching scores for all methods in Fig. 8(b) are greater than Fig. 8(a).

E. Compare the Efficiency of the Similarity Search

In order to justify the efficiency of the similarity search, we compare our basis buffer k-d tree with the GPU-based buffer k-d tree [11]. By increasing the feature dimension \(d\) from 10 to 128 and fixing the training and query data sizes as 100k and 10k, the time consumed by different similarity search methods is plotted in Fig. 9(a), and we can see that ours is more efficient than the baseline method, GPU-based buffer k-d tree [11]. We also fix the dimension as 32, the queries data as 10,000, and increase training data size from 100k to 700k with the step as 100k, as shown in Fig. 9(b), and we can observe that by increasing the training data size, our basis buffer k-d tree is more efficient than the traditional methods [11] as well. Due to the average training data size for 3-D object recognition being about 100k similar to our simulated experiments (140k for UWA dataset in Section IV-A and 10k for Queen’s dataset in Section IV-B), we can conclude that our basis buffer k-d tree is more suitable for 3-D object detection in our application. This is because we only search the bases of each leaf instead of suffering the branch divergence as [11], which can be considered as an \(R\)-NN procedure and is more suitable for GPU architecture. Therefore, ours has more advantages for real practical applications. All of the times are recorded using the public SIFT128 dataset.

F. Compare the Time Consumption

We present the time consumption with a mixed CPU+GPU implementation of our method, where the GPU is only adopted for similarity search. As shown in Fig. 10, we record the individual time consumption of various stages, including feature representation, similarity search, and hypothesis generation and refinement. Especially, as shown in Fig. 11, the total consuming time is 0.47 s for our method with about 3200 testing features and 140,000 model features. In comparison, the RoPS occupies about 7.2 s in [6] using MATLAB, and PS [42] and SHOT [22] consume about 791 and 1233 ms, respectively. Moreover, we record 1000 times of feature extraction without LRF extraction, where ours (3.3 ms) achieves approximately 13 times faster than RoPS (41.9 ms), and about three times faster than both PS (8.3 ms) and SHOT (9.9 ms), respectively (RoPS is also coded by C++ using PCL library here). Therefore, ours is more efficient and more suitable in practice.

All experiments are executed on a standard PC with a Genuine Intel CPU at 3.40 GHz (four cores, eight hardware threads), 8 GB RAM, and a GeForce GTX TITAN Black/PCle/SSE2 GPU with 2880 shader units (6 GB RAM). The operating system is Windows 10 (64 Bit). All components of our proposed 3-D object recognition method are implemented by C++.
G. Compare the 6-DOF Estimation Error

One of the major tasks for 3-D object recognition in intelligent manufacturing scenarios is to estimate the 6-DOF \((x, y, z, \alpha, \beta, \gamma)\) of the object from the cluttered background for industrial robot operation, for example, grasping the objects like human ourselves. Therefore, in this section, we evaluate 6-DOF estimation error by comparing the position error and attitude error separately. The results are shown in Fig. 12 by adopting the Queen’s dataset (top row) and UWA dataset (bottom row).

For the position error, we define the position error as the ratio of the center position error with the minimal external radius \(R_{\text{Ext}}\) of the corresponding model

\[
\text{pre}_{\text{pos}} = \frac{\| p_m - p_t \|_F}{R_{\text{Ext}}} \times 100\%
\]

where \(p_m = [x_m, y_m, z_m]^T\) and \(p_t = [x_t, y_t, z_t]^T\) are the center position of the groundtruth and the result estimated by our method, respectively. Specifically, we evaluate the probability of the ratio of the center position error in Fig. 12(b) and (f) and for the corresponding accumulate probability in Fig. 12(d) and (h), we can conclude that more than 95% results of the center position error are less than 2% (Queen’s dataset) and 0.5% (UWA dataset) of the minimal external radius of the related model.

For the attitude error, we denote the attitude error of \(\alpha, \beta, \gamma\) independently by comparing the attitude of the groundtruth with our method

\[
\text{pre}_{\text{ang}} = \frac{\| \text{Ang}_m - \text{Ang}_t \|_F}{F}
\]

where \(\text{Ang}_m\) and \(\text{Ang}_t\) stand for the attitude (i.e., \(\alpha, \beta, \gamma\)) of the groundtruth and the result estimated by our method, respectively. Specifically, we evaluate the probability of the attitude \((\alpha, \beta, \gamma)\) error in Fig. 12(a) and (e) and for the corresponding accumulate probability in Fig. 12(c) and (g), we can see that over 95% results of the attitude error are less than 2° for both the Queen’s dataset and UWA dataset.

Therefore, we can conclude the 6-DOF estimation error of our method is acceptable for robot operation in intelligent manufacturing scenarios.

V. CONCLUSION

In this paper, we intend to design a new 3-D object recognition and 6-DOF pose estimation method especially for textureless and colorless industrial objects. For feature representation, only raw 3-D point clouds with normal information are adopted to define our LRF and we then learn a compact 75d feature via the stacked AE. For a similarity search, a new basis buffer k-d tree based on GPU is designed without traversing the whole tree as traditional methods. We then vote the hypothesis using the validate pairwise correspondences, generate the hypothetical candidates and refine the pose estimation via the ICP strategy. For experiments, a new 3-D dataset is built including several industrial objects with heavy self-occlusions, which will be released soon. To evaluate the accuracy, our method achieves the performance of UWA (98.8%), Queen (97.6%), and our industrial 3-D dataset (75.9%), which all outperform the state of the arts, such as RoPS, Spin image. By concerning the efficiency, ours occupies 0.47 s per frame in contrast to RoPS 7.2 s [6] per frame, which is partially caused by our simple low-dimensional 75d feature and GPU-based basis buffer k-d tree. Moreover, ours is more robust to self-occlusion as well, for example, 3-D objects with many holes inside. All of these justify both the effectiveness and efficiency of our 3-D object recognition method.

REFERENCES

[1] O. Lopes, M. Reyes, S. Escalera, and J. González, “Spherical blurred shape model for 3-D object and pose recognition: Quantitative analysis and HCI applications in smart environments,” IEEE Trans. Cybern., vol. 44, no. 12, pp. 2379–2390, Dec. 2014.

[2] O. S. Gedik and A. A. Alatan, “3-D rigid body tracking using vision and depth sensors,” IEEE Trans. Cybern., vol. 43, no. 5, pp. 1395–1405, Oct. 2013.

[3] M. Liang, H. Min, R. Luo, and J. Zhu, “Simultaneous recognition and modeling for learning 3-D object models from everyday scenes,” IEEE Trans. Cybern., vol. 45, no. 10, pp. 2237–2248, Oct. 2015.

[4] R. Kouskouridas, A. Teljani, A. Doumanoglou, D. Tang, and T.-K. Kim, “Latent-class hough forests for 6 DoF object pose estimation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 1, pp. 119–132, Jan. 2018.

[5] Y. Xiao, H. Lu, and C. Sun, “Pose estimation based on pose cluster and candidates recombination,” IEEE Trans. Circuits Syst. Video Technol., vol. 25, no. 6, pp. 935–943, Jun. 2015.

[6] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, “Rotational projection statistics for 3D local surface description and object recognition,” Int. J. Comput. Vis., vol. 105, no. 1, pp. 63–86, 2013.

[7] A. S. Mian, M. Bennamoun, and R. Owens, “Three-dimensional model-based object recognition and segmentation in cluttered scenes,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 10, pp. 1584–1601, Oct. 2006.

[8] A. Mian, M. Bennamoun, and R. Owens, “On the repeatability and quality of keypoints for local feature-based 3D object retrieval from cluttered scenes,” Int. J. Comput. Vis., vol. 89, nos. 2–3, pp. 348–361, 2010.

[9] S. Hinterstoisser et al., “Model-based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes,” in Proc. ACCV, Daejeon, South Korea, 2012, pp. 548–562.

[10] S. Hinterstoisser et al., “Gradient response maps for real-time detection of textureless objects,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 5, pp. 876–888, May 2012.

[11] F. Gieseke, J. Heinermann, C. E. Oancea, and C. Igel, “Buffer k-d trees: Processing massive nearest neighbor queries on GPUs,” in Proc. ICML, 2014, pp. 172–180.

[12] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, and J. Wan, “3D object recognition in cluttered scenes with local surface features: A survey,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, no. 11, pp. 2270–2287, Nov. 2014.

[13] J. W. Tangelder and R. C. Veldkamp, “A survey of content based 3D shape retrieval methods,” in Proc. IEEE Shape Model. Appl., Genova, Italy, 2004, pp. 145–156.

[14] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 31, no. 4, pp. 615–630, Oct. 2005.

[15] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” Int. J. Comput. Vis., vol. 60, no. 2, pp. 91–110, 2004.

[16] Q. Wang, J. Gao, and Y. Yuan, “Embedding structured contour and projection statistics for 3D local surface description and object recognition,” Pattern Recognit., vol. 40, no. 1, pp. 119–132, Jan. 2018.

[17] D. G. Lowe, “Local feature view clustering for 3D object recognition,” in Proc. CVPR, vol. 1, 2001, pp. 682–688.

[18] K. Pauwels, L. Rubio, J. Diaz, and E. Ros, “Real-time model-based rigid object pose estimation and tracking combining dense and sparse visual cues,” in Proc. CVPR, Portland, OR, USA, 2013, pp. 2347–2354.

[19] M. Ulrich, C. Wiedemann, and C. Steger, “Combining scale-space and similarity-based aspect graphs for fast 3D object recognition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 10, pp. 1902–1914, Oct. 2012.
[21] A. E. Johnson and M. Hebert, “Using spin images for efficient object recognition in cluttered 3D scenes,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 21, no. 5, pp. 433–449, May 1999.

[22] F. Tombari, S. Salti, and L. Di Stefano, “Unique signatures of histograms for local surface description,” in Proc. ECCV, 2010, pp. 356–369.

[23] G. Hetzel, B. Leibe, P. Levi, and B. Schiele, “3D object recognition from range images using local feature histograms,” in Proc. CVPR, vol. 2, 2001, pp. 356–369.

[24] R. B. Rusu, A. Holzbach, N. Blidow, and M. Beetz, “Fast geometric point labeling using conditional random fields,” in Proc. IROS, St. Louis, MO, USA, 2009, pp. 7–12.

[25] F. R. Al-Osaimi, “A novel multi-purpose matching representation of local 3D surfaces: A rotationally invariant, efficient, and highly discriminative approach with an adjustable sensitivity,” IEEE Trans. Image Process., vol. 25, no. 2, pp. 658–672, Feb. 2016.

[26] F. Tombari, A. Franchi, and L. Di Stefano, “BOLD features to detect texture-less objects,” in Proc. ICCV, Sydney, NSW, Australia, 2013, pp. 1265–1272.

[27] R. Rins-Cabrer and T. Tuytelaars, “Discriminatively trained templates for 3D object detection: A real time scalable approach,” in Proc. ICCV, Sydney, NSW, Australia, 2013, pp. 2048–2055.

[28] W. Kehl, F. Tombari, N. Navab, S. Ilic, and V. Lepetit, “Hashmod: A hashing method for scalable 3D object detection,” in Proc. BMVC, 2015, pp. 1–12.

[29] W. Kehl, S. Maturana, A. A. Efros, B. C. Russell, and J. Sivic, “Seeing 3D chairs: Exemplar-based 2D–3D alignment using a large dataset of CAD models,” in Proc. CVPR, 2014, pp. 3762–3769.

[30] E. Brachmann et al., “Learning 6D object pose estimation using 3D object coordinates,” in Proc. ECCV, Zürich, Switzerland, 2014, pp. 536–551.

[31] A. Tejani, D. Tang, R. Koussouridas, and T.-K. Kim, “Latent-class hough forests for 3D object detection and pose estimation,” in Proc. ECCV, Zürich, Switzerland, 2014, pp. 462–477.

[32] P. Wohlhart and V. Lepetit, “Learning descriptors for object recognition and 3D pose estimation,” in Proc. CVPR, 2015, pp. 3109–3118.

[33] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, “Learning rich features from RGB-D images for object detection and segmentation,” in Proc. ECCV, 2014, pp. 345–360.

[34] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik, “Aligning 3D models to RGB-D images of cluttered scenes,” in Proc. CVPR, Boston, MA, USA, 2015, pp. 4731–4740.

[35] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab, “Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation,” in Proc. ECCV, Amsterdam, The Netherlands, 2016, pp. 205–220.

[36] A. Doumanoglou, R. Koussouridas, S. Malassiotis, and T.-K. Kim, “Recovering 6D object pose and predicting next-best-view in the crowd,” in Proc. CVPR, Las Vegas, NV, USA, 2016, pp. 3583–3592.

[37] P. J. Besl and N. D. McKay, “A method for registration of 3-D shapes,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 14, no. 2, pp. 239–256, Feb. 1992.

[38] A. Aldoma et al., “Multimodal cue integration through hypotheses verification for RGB-D object recognition and 6DOF pose estimation,” in Proc. ICRA, Karlsruhe, Germany, 2013, pp. 2104–2111.

[39] A. G. Buch, Y. Yang, N. Krüger, and H. G. Petersen, “In search of inliers: 3D correspondence by local and global voting,” in Proc. IEEE CVPR, Columbus, OH, USA, 2014, pp. 2075–2082.

[40] Q. Hao et al., “Efficient 2D-to-3D correspondence filtering for scalable 3D object recognition,” in Proc. CVPR, Portland, OR, USA, 2013, pp. 899–906.

[41] A. Aldoma, F. Tombari, L. Di Stefano, and M. Vincze, “A global hypotheses verification method for 3D object recognition,” in Proc. ECCV, Florence, Italy, 2012, pp. 511–524.

[42] A. Petrelli and L. Di Stefano, “On the repeatability of the local reference frame for partial shape matching,” in Proc. IEEE ICCV, Barcelona, Spain, 2011, pp. 2244–2251.

[43] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in Proc. ACM TISM, Helsinki, Finland, 2008, pp. 906–1103.

[44] Y. Cong, J. Yuan, and J. Luo, “Towards scalable summarization of consumer videos via sparse dictionary selection,” IEEE Trans. Multimedia, vol. 14, no. 1, pp. 66–75, Feb. 2012.

[45] B. Taati and M. Greenspan, “Local shape descriptor selection for object recognition in range data,” Comput. Vis. Image Understand., vol. 115, no. 5, pp. 681–694, 2011.

[46] P. Borti, J. Konrad, G. Schwartz, and K. Nishino, “3D geometric scale variability in range images: Features and descriptors,” Int. J. Comput. Vis., vol. 99, no. 2, pp. 232–255, 2012.

[47] B. Taati, M. Bondy, P. Jasiodzcki, and M. Greenspan, “Variable dimensional local shape descriptors for object recognition in range data,” in Proc. ICCV, 2007, pp. 1–8.