Voting and Attention-Based Pose Relation Learning for Object Pose Estimation From 3D Point Clouds

Dinh-Cuong Hoang, *Member, IEEE*, Johannes A. Stork, and Todor Stoyanov, *Member, IEEE*

***Abstract*—Estimating the 6DOF pose of objects is an important function in many applications, such as robot manipulation or augmented reality. However, accurate and fast pose estimation from 3D point clouds is challenging, because of the complexity of object shapes, measurement noise, and presence of occlusions. We address this challenging task using an end-to-end learning approach for object pose estimation given a raw point cloud input. Our architecture pools geometric features together using a self-attention mechanism and adopts a deep Hough voting scheme for pose proposal generation. To build robustness to occlusion, the proposed network generates candidates by casting votes and accumulating evidence for object locations. Specifically, our model learns higher-level features by leveraging the dependency of object parts and object instances, thereby boosting the performance of object pose estimation. Our experiments show that our method outperforms state-of-the-art approaches in public benchmarks including the Siléane dataset [35 and the Fraunhofer IPA dataset [36]. We also deploy our proposed method to a real robot pick-and-place based on the estimated pose.**

***Index Terms*—6D object pose estimation, 3D point cloud, robot manipulation.**

I. Introduction

DEEP Neural Network (DNN) based methods taking RGB or RGB-D images as input have achieved state-of-the-art performance on the object pose estimation task [1]–[3]. However, in a number of cases, color information may not be available — for example, when the input is point cloud data from laser range finders or industrial high-resolution 3D sensors. In addition, deep learning-based methods require large amounts of labeled training data. Collecting images of objects from the real world under various conditions and annotating the images with 6DOF object poses is time-consuming and requires a significant human effort.

Manuscript received 24 February 2022; accepted 21 June 2022. Date of publication 7 July 2022; date of current version 12 July 2022. This letter was recommended for publication by Associate Editor M. Li and Editor M. Vincze upon evaluation of the reviewers' comments. This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement 101017274 (DARKO). The work of Johannes A. Stork and Todor Stoyanov was supported by the Knut and Alice Wallenberg Foundation through Wallenberg AI, Autonomous Systems and Software Program (WASP). *(Corresponding author: Dinh-Cuong Hoang.)*

Dinh-Cuong Hoang is with the ICT Department, FPT University, Hanoi, Vietnam (e-mail: cuonghd7@fe.edu.vn).

Johannes A. Stork is with the Centre for Applied Autonomous Sensor Systems, Orebro University, 70 281 Örebro, Sweden (e-mail: johannesandreas.stork@oru.se).

Todor Stoyanov is with the Centre for Applied Autonomous Sensor Systems, Orebro University, 70 281 Örebro, Sweden, and also with the Department of Computing and Software, McMaster University, Hamilton ON L8S 4L8, Canada (e-mail: todor.stoyanov@oru.se).

Digital Object Identifier 10.1109/LRA.2022.3189158

A promising alternative is the use of synthetic data for training such deep neural networks. Nonetheless, it is still difficult for methods requiring appearance information where the domain gap between synthetic training data and real test images is severe. Compared to color images, the domain gap between the synthetic and real data is considerably smaller for 3D point clouds. In addition, synthesizing data with only geometric information is less expensive in terms of time and hardware storage, as there is no texture or illumination present in the data. This allows us to scale data generation up to a large number of objects, which is often desirable in practical applications. Therefore, it is feasible to develop DNN models for object pose estimation from point clouds that can be trained purely in simulation and be deployed seamlessly in the real world.

In this letter, we propose a novel DNN architecture to address the problem of estimating the 6DOF pose of multiple rigid objects in a cluttered scene, using only a 3D point cloud of the scene as an input. We build our network on top of a deep Hough voting architecture for 3D object detection [4], VoteNet. The voting mechanism allows our model to perform reliable detection under clutter and occlusion. However, extending VoteNet for an object pose estimation task is not straightforward. Although VoteNet achieves state of the art performance on 3D object detection tasks, it fails to explore the correlation between objects (instances) and local parts of objects (parts), which are crucial for the 6D pose estimation task. Intuitively, when predicting the accurate pose of an object in a cluttered scene, the geometric relation between features on and around the object carry valuable information. Therefore, we exploit this contextual information by encoding the dependency of object parts and object instances into features. To this end, we introduce two high-level feature learning modules ($\mathcal{M}_\mathcal{O}$ and $\mathcal{M}_\mathcal{P}$) into the VoteNet architecture to model the pose relation between both object instances and object parts in the scene. The module $\mathcal{M}_\mathcal{O}$ first generates object-centric proposals using a per-point voting scheme from VoteNet. We then feed proposal features into a self-attention module in order to enable higher-order interactions between neighboring proposal features and learn instance-to-instance correlations. Similarly, the module $\mathcal{M}_\mathcal{P}$ votes for object part centers and then learns part-to-part correlations with self-attention.

The main contributions of this work are: 1) An end-to-end network for 6D object pose estimation from 3D point clouds: robust to noise and occlusion, and able to deal with multiple objects in cluttered scenes; 2) Two high-level feature learning modules for modeling part-to-part and instance-to-instance correlations with self-attention to improve the performance of object pose estimation; 3) Extensive experiments show that the developed framework is effective and efficient for deployment.

II. Related Work

A. Object Pose Estimation in 3D Point Clouds

Conventional methods are mainly based on the matching of local or global features extracted from a measured point cloud to features in a model of the object. Global feature-based approaches utilize the whole geometric appearance of the object surface to define a single feature vector that effectively and concisely describes the entire 3D object [5]–[7]. On the contrary, local feature-based methods exploit the geometric properties around specific keypoints [8]. While global methods are able to handle objects with self-similar surface parts, such as planar patches, spheres and cylinders, local approaches are more suitable for detecting and estimating the pose of complex objects in cluttered scenes. As a compromise solution, Drost *et al.* [9] build a global model description using point pair features and match that model locally using a fast voting scheme. The proposed point pair feature (PPF) describes the relative position and orientation of two oriented points. The algorithm has been successfully deployed in a number of industrial applications and has been thouroughly improved and extended [10]–[15].

Despite a wide range of applications, PPF-based approaches share some common problems: 1) relying on searching a large set of paired feature correspondences severely limits their speed; 2) they are sensitive to measurement noise, heavy occlusion and background clutter. Recent work has attempted to leverage the power of deep learning to achieve fast and robust 6D pose estimation [3], [16]–[19]. However, these approaches either still require color information in the process or rely on a prior instance segmentation step. As the quality of the segmentation mask greatly influences the performance of pose estimation, these approaches are often limited by poor performance of segmentation in the presence of clutter and occlusion. Unlike existing methods, our model consumes unordered 3D point cloud data without color information and does not require segmentation masks. Our method relies on a voting mechanism to perform reliable detection and learns the dependency of object parts and object instances to boost the performance of pose estimation.

B. Self-Attention

Attention mechanisms force deep learning models to focus on important regions within a context and have been widely applied in a variety of tasks [20]–[22]. Self-attention learns the correlation between inputs by allowing them to interact with each other. The ability of self-attention to directly model long-distance interactions and to reduce sequential computation have revolutionized machine translation and natural language processing [23], [24]. This has inspired applications of self-attention to computer vision tasks such as image recognition [25], semantic segmentation [26], and image captioning [27]. As discussed in [28], self-attention can be viewed as a form of the non-local mean [29]. The non-local module [28] is designed to capture long-range spatio-temporal dependencies in images and videos. It can be integrated into many computer vision architectures. The authors demonstrated that it can greatly improve the performances of existing deep neural networks on many benchmarks for video classification and static image recognition tasks. Motivated by the self-attention network [28], the Compact Generalized Non-local Network (CGNL) [30] is proposed to generalize the non-local module by learning explicit correlations among all of the elements across channels. We find that CGNL is well suited to explicitly model rich correlations between object parts and object instances. Therefore, we opt for CGNL as our self-attention module.

III. Learning to Aggregate Context for Pose Estimation

Given a 3D point cloud of a cluttered scene, we are interested in detecting objects and estimating their orientations and translations in 3D space. The problem is akin to 3D object detection, where the goal is to estimate oriented 3D bounding boxes of physical objects from sensor data. The box bounds the complete object and is parameterized by its size (height, width, length), center, and orientation. However, the information from bounding boxes is often not applicable to robot manipulation. Rather than detect objects with bounding boxes, aligning a full 3D model of the target object to its incomplete measured data would be more useful for manipulation tasks. To this end, we assume that an accurate 3D model of the object is available and the object coordinate system $\mathcal{O}$ is defined in the 3D space of the model. Object pose is represented by a rigid transformation from the object coordinate system to a reference coordinate system $\mathcal{G}$ (often camera coordinate system). The rigid transformation consists of a rotation $R \in SO(3)$ and a translation $t \in \mathbb{R}^3$, $\xi = [R|t]$.

A. Overview

In this work, we introduce a deep network for 6D object pose estimation, which is illustrated in Fig. 1 and Fig. 2. The model comprises four components. The first one takes as input a 3D point cloud and extracts seeds as high-dimensional features. The second component $\mathcal{M}_{\mathcal{P}}$ operates on the high-dimensional features to learn part proposals and use a self-attention module to enable higher-level interactions between part proposals. Similarly, the third one $\mathcal{M}_{\mathcal{O}}$ takes seeds as input to learn object proposals and then encode the instance-to-instance correlation information through the self-attention module. The voting part in both $\mathcal{M}_{\mathcal{P}}$ and $\mathcal{M}_{\mathcal{O}}$ is based on VoteNet [4] for 3D object detection. Last, a pose estimation module combines feature maps from the self-attention modules and generates the pose of objects.

B. Feature Extraction

In order to extract geometric features, we utilize the PointNet++ architecture with multi-scale grouping as our backbone network. As the core of our base network, its set abstraction level is composed of three layers: a sampling layer, a grouping

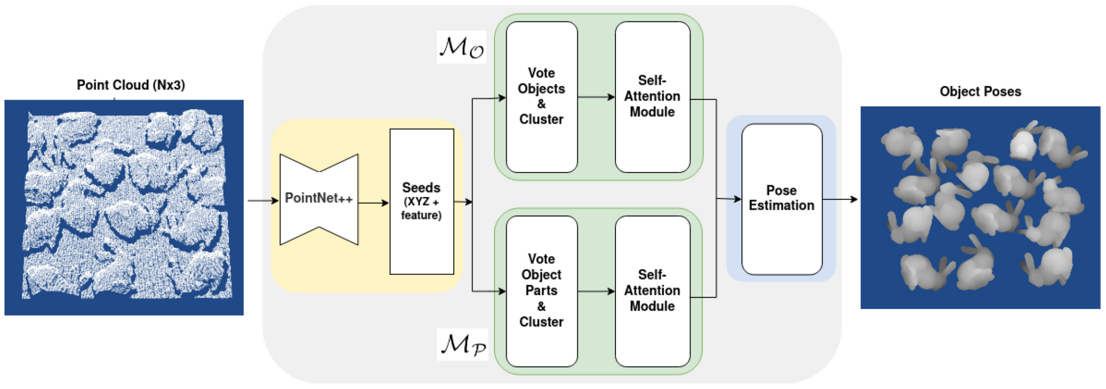


Fig. 1. The architecture of the proposed network for 6-DOF object pose estimation in point cloud data. Our model builds on a deep Hough voting neural network [4] to vote for object and part centers. Especially, we introduce self-attention modules for encoding the dependency of object parts and object instances into features to boost the performance of object pose estimation. The output is a rotation R and a translation t for each object (object pose $\xi = [R|t]$) which can be used to align object models to the scene (right).

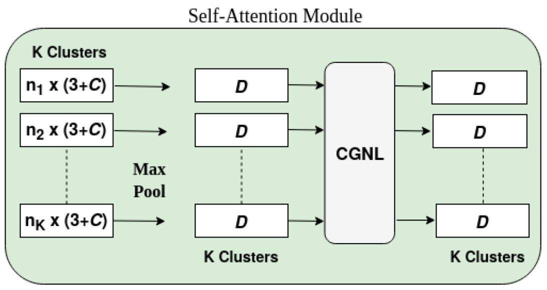


Fig. 2. Self-attention module with generaized non-local network (CGNL) [30].

layer, and the PointNet [31] based learning layer. Based on PointNet++, we are able to capture fine geometric structures from the neighborhood of each point. The base network selects M interest points (called seed points) and enriches them with high-dimensional features $\{s_i\}_{i=1}^M$ where $s_i = [x_i; f_i]$ with $x_i \in \mathbb{R}^3$ being the seed location in 3D space and $f_i \in \mathbb{R}^C$ being a feature vector.

C. Learning Part-to-Part Correlation

Given the high-dimensional features $\{s_i\}_{i=1}^M$, a part detection module $\mathcal{M}_\mathcal{P}$ is used to generates a fixed number of J proposals. A proposal is a tuple (z_i, h_i, s_i) consisting of a position $z_i \in \mathbb{R}^3$, a proposal features vector $h_i \in \mathbb{R}^D$ and a set of points s_i associated with the proposal. To be specific, each seed point s_i is fed into a shared Multi-Layer Perceptron (MLP) [32], [33] to predict a feature offset $\Delta f_i \in \mathbb{R}^C$ and a relative 3D offset $\Delta x_i \in \mathbb{R}^3$ between the seed point position $x_i \in \mathbb{R}^3$ and its corresponding ground truth part center $c_i \in \mathbb{R}^3$. Then the vote can be denoted as $v_i = [y_i; g_i] \in \mathbb{R}^{3+C}$ with $y_i = x_i + \Delta x_i$ and $g_i = f_i + \Delta f_i$. To supervise the learning of the 3D offset Δx_i, we apply an regression loss:

$$L_{part-vote} = \frac{1}{M_{pos}} \sum_i \|x_i + \Delta x_i - c_i^p\|_H \cdot \mathbb{1}(x_i) \tag{1}$$

where M_{pos} is the count of the total number of seeds on the object surface, $\|\cdot\|_H$ is the Huber norm and $\mathbb{1}(\cdot)$ is a binary function indicating whether a seed point s_i belongs to an object part. After each seed point has voted for a part center, we obtain a distribution over object part centers. The next step is to form J vote clusters by uniform sampling and finding neighboring votes within a certain Euclidean distance. Then votes from J clusters are aggregated to generate J feature proposals $\{h_i \in \mathbb{R}^D\}_{i=1}^J$ using a PointNet-like module as described in [4]. At this stage, we have J proposals composed of 3D positions $z_i = x_i + \Delta x_i$ located near part centers, proposal features $h_i \in \mathbb{R}^D$ describing the local geometry, and a set of seed points s_i associated with each proposal.

So far, the proposal feature maps $H = \{h_i\}_{i=1}^J$ encode only local information of the point cloud. In order to enable features to become aware of their global neighborhood, we explicitly model higher-order interactions between proposal features, and it can be formulated as the non-local operation:

$$H_{part-part} = f(\theta(H)\phi(H))g(H) \tag{2}$$

where $\theta(\cdot)$, $\phi(\cdot)$, $g(\cdot)$ are learnable transformations on the input feature map H, and $f(\cdot)$ encodes the relation between any two parts. To this end, we opt for the compact generalized non-local network (CGNL) [30] as our self-attention module to explicitly model rich correlations between parts and to provide higher-level feature learning in addition to the lower-level point features. Fig. 3 illustrates object part center votes and relation between parts.

D. Learning Instance-to-Instance Correlation

Similarly, an object detection module $\mathcal{M}_\mathcal{O}$ is used to generate K clusters from the high-dimensional features $\{s_i\}_{i=1}^M$ and a set of object centers. The loss is defined as:

$$L_{object-vote} = \frac{1}{M_{pos}} \sum_i \|x_i + \Delta x_i - c_i^o\|_H \cdot \mathbb{1}(x_i) \tag{3}$$

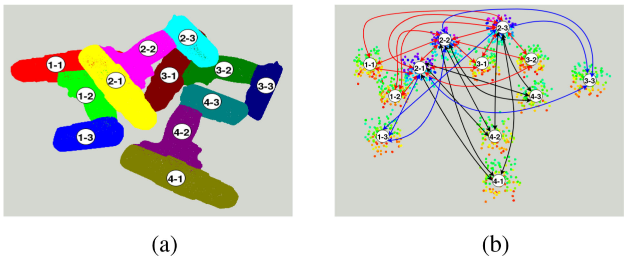


Fig. 3. An example to illustrate learning of object part center votes and seft-attention module. The red, blue and black arrows indicate high, fair and low correlations respectively.

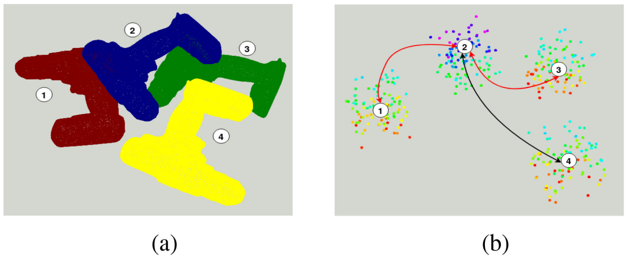


Fig. 4. An example to illustrate learning of object center votes and seft-attention module. The red, blue and black arrows indicate high, fair and low correlations respectively.

where Δx_i is a Euclidean space offset between the point position x_i and its corresponding ground-truth object center c_i^o. To encode the relationship of objects in the scene into features and exploit information outside of local regions, instead of processing each cluster independently, our network computes a new feature map from all clusters to learn higher-level features that consider the relationships between all object instances. Fig. 4 illustrates object center votes and relation between object instances. By leveraging the self-attention mechanism, we can combine features from clusters to enable higher-order interaction between proposals. We thus make use of the compact generalized non-local network (CGNL) [30]. CGNL allows for explicit modeling of rich interdependencies between clusters in feature space in a fast and low-complexity computation flow. The CGNL based self-attention module takes K clusters $\mathbf{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$ as inputs. Then votes from each cluster are processed by an MLP before being max-pooled to a single feature vector and passed to CGNL. The self-attention mechanism allows the features from different clusters to interact with each other ("self") and find out who they should pay more attention to ("attention"). The output is a new feature map $H_{obj-obj} = \{H^k_{obj-obj}\}$ with $k = 1, \ldots, K$.

$$H_{obj-obj} = CGNL(\max_{i=1,\ldots n}(MLP(v_i))) \tag{4}$$

E. Pose Estimation With Multi-Task Loss

Given new feature maps $H_{part-part}$ and $H_{obj-obj}$ generated by the respective self-attention modules, max-pooling is first applied to get two vectors including the part feature vector and the object feature vector, combining information from all the part and object candidates. These two vectors are then concatenated to form a single feature vector. An MLP layer is applied to further aggregate global information. While we can learn the translation part of the pose in the Euclidean space, representing the rotation part is more complicated. The common options of using Euler angles-based and quaternion-based representations are discontinuous and difficult for neural networks to learn as explained by [34]. In [34], the authors show that the 3D rotations have continuous representations in 5D and 6D, which are more suitable for learning. Therefore, we make use of 6D continuous representation for 3D rotations. Following [34], we map the 6D vectors in representation space produced by the network into the original rotation space and minimize the L2 loss between the output and the ground-truth rotation matrices.

We supervise the learning of modules jointly with a multi-tasks loss:

$$L = \lambda_1 L_{part-vote} + \lambda_2 L_{object-vote} + \lambda_3 L_{pose} \tag{5}$$

where λ_1, λ_2 and λ_3 are the weights for each task. The loss includes a voting part loss $L_{part-vote}$ (1), a object vote loss $L_{object-vote}$ (3), and a pose loss L_{pose}. We define the pose loss function as follows:

$$L_{pose} = L_t + \alpha L_{rot} + \beta L_{obj} + \gamma L_{sem} \tag{6}$$

where α, β, and γ are weights that scale the losses to similar scales. The pose loss is composed of a translation loss L_t (regression), an L2 loss between the output and the ground-truth rotation matrices, an objectness loss L_{obj} and a semantic classification loss L_{sem}. The objectness loss is a cross-entropy loss for two classes (an object or not). The semantic classification loss is also a cross-entropy loss of semantic classes. However, the above loss L_{rot} for rotation is only appropriate to asymmetric objects. For symmetric objects having multiple correct 3D rotations, given the ground truth rotation $\bar{R}$ and translation $\bar{T}$ and the estimated rotation $\hat{R}$ and translation $\hat{T}$ we compute L_{rot} as:

$$L_{rot} = \frac{1}{m} \sum_{x_1 \in M} \| \min_{x_2 \in M} (\bar{R}x + \bar{T} - \hat{R}x + \hat{T}) \| \tag{7}$$

where M denotes the set of 3D model points and m is the number of points. The loss is calculated as the average distance from vertices of the object model in the ground-truth pose to closest vertices of the model in the estimated pose. That way the loss is minimized when the two 3D models are aligned with each other.

IV. Evaluation

We evaluated our proposed approach on data from the Siléane dataset [35] and the Fraunhofer IPA dataset [36]. These datasets consist of multiple rigid texture-less objects of the same type under cluttered scenes with multiple and heavy occlusions. A comparison against the most closely related works is also performed here.

A. Datasets

The Siléane dataset [35] consists in total of 2,601 independent scenes, fully annotated. Depending on the object, the dataset comprises 46 to 325 scenes and is too small to train our model. Therefore, we used this dataset only for testing. For model training, we relied on the Fraunhofer IPA Bin-Picking dataset [36]. It includes 520 fully annotated point clouds and corresponding depth images of real-world scenes (for testing) and about 206,000 synthetic scenes (for training). The dataset comprises eight objects from the Siléane dataset and two newly introduced ones (gear shaft and ring screw). Training data for the coffee cup (C.cup) was not generated in the Fraunhofer IPA dataset. Hence, we precisely rebuild the setup from the dataset in simulation and produced 10,000 scene point clouds depicting various numbers of C.cup instances in bulk.

B. Implementation Details

Network Architecture: In our implementation, we randomly choose $N = 50$ k points from each raw point cloud and set λ_1=0.5, λ_2=1.0, λ_3=0.1 in (5), α=β=γ=1.0 in (6). We then apply the PointNet++ [33] based feature learning network, which has 4 set abstraction layers (SA) and 2 feature propagation layers (FP). The FP2 outputs seeds that will be transformed to votes. The voting module generates $J = 2$ votes per seed with an MLP layer spec: [256, 256, 259 $\times$ 2], 1 vote for the object center and 1 vote for the part center. In the learning part-to-part and instance-to-instance correlation modules, we form 1024 clusters by finding neighboring votes and finally output a new feature map for each cluster. In the last step, 256 proposals are generated from 256 vote clusters sampled from the 1024 part-centric vote clusters and 1024 instance-centric vote clusters in the previous step.

Training the network: Our proposed model is trained from scratch in an end-to-end manner using an Adam optimizer. We train the entire network with the batch size 48 and learning rate 0.001 for 200 epochs. It takes around 10 hours for training the Siléane and Fraunhofer IPA dataset [35], [36] using six Nvidia Tesla V100 32 GB GPUs. In regard to inference time, it takes around 150 ms to process an input containing 50 k points on a consumer GPU.

C. Evaluation Metric

The error of an estimated pose $\hat{P}$ with respect to the ground-truth pose $\bar{P}$ of an object model M is measured by the most widely used pose-error function Average Distance of Model Points (ADD) [37]. The error is calculated as the average distance from vertices of the object model in the ground-truth pose to vertices of the model in the estimated pose. Given the ground truth rotation $\bar{R}$ and translation $\bar{t}$ and the estimated rotation $\hat{R}$ and translation $\hat{t}$, the average distance is defined as:

$$ADD = \frac{1}{m} \sum_{x \in M} \parallel (\bar{R}x + \bar{t} - \hat{R}x + \hat{t}) \parallel \tag{8}$$

where m is the number of model points. For objects with symmetric views, we adapt the metric by computing the average distance using the closest point distance following prior works [35]. We report the accuracy of predictions in average precision (AP) following [35], given the goal of retrieval of instances less than 30% occluded. A 6D pose estimate is considered to be true positive if the average distance is smaller than 0.1 times the diameter of the smallest bounding sphere.

TABLE I
AP Values of Different Algorithms on Objects From the Siléane dataset [35] and the Fraunhofer IPA dataset [36]

	[37]	[9]	[15]	[17]	[18]	[19]	Ours
Brick	0.10	0.19	0.37	0.42	0.36	0.38	**0.48**
Bunny	0.38	0.27	0.44	0.54	0.46	0.52	**0.61**
C.stick	0.37	0.15	0.50	0.54	0.50	0.52	**0.60**
C.cup	0.08	0.27	0.40	0.45	0.41	0.44	**0.52**
Gear	0.21	0.28	0.40	0.51	0.42	0.50	**0.64**
Pepper	0.04	0.06	0.28	0.30	0.24	0.25	**0.39**
Tless 20	0.11	0.21	0.31	0.38	0.35	0.30	**0.44**
Tless 22	0.09	0.13	0.22	0.29	0.20	0.23	**0.37**
Tless 29	0.19	0.28	0.39	0.35	0.40	0.37	**0.46**
Gear shaft	0.18	0.32	0.38	0.48	0.41	0.51	**0.65**
Ring screw	0.27	0.40	0.54	0.56	0.45	0.54	**0.67**
MEAN	0.18	0.23	0.38	0.44	0.38	0.41	**0.53**

D. Results

Fig. 5 demonstrates qualitative results. Table I summarizes the comparison results between our method and current point cloud-based approaches on 9 objects in the Siléane dataset [35] and 2 objects (gear shaft and ring screw) in the Fraunhofer IPA dataset [36]. The experiments are based on publicly available implementations by the authors [17], [19], [37], [38], and on our own implementation of the rest. Our proposed approach achieves state-of-the-art results and outperforms others on all objects, obtaining an average precision of 53%. As can be seen from the obtained results, the proposed method results in an improvement of +15% to +35% on AP compared with the conventional approaches [9], [15], [37], [38]. The AP score of our method also surpasses the existing deep learning-based methods [17]–[19] with a significant margin. Moreover, in order to evaluate the robustness of algorithms towards occlusion and noise, we experiment with estimating the pose of objects under an increasing amount of hidden surface and added Gaussian noise. Fig. 6 and Fig. 7 illustrate how methods are influenced by different levels of occlusion and noise. As shown, our proposed method performs well even when objects are heavily occluded or input data are noisy, while the results of the previous approaches indicate high sensitivity to occlusion and noise.

E. Ablation Study

Furthermore, we ran a number of ablations to analyze our network without learning part-to-part correlation (Ours (- $\mathcal{M}_{\mathcal{P}}$)) and without learning object-to-object correlation (Ours (- $\mathcal{M}_{\mathcal{O}}$)). We modified VoteNet architecture to directly estimate the 6D pose of objects from the vote aggregated features and used it as the baseline method. We also verify the success of the multi-task loss. As seen in Table II, our proposed model achieves superior performance compared to the modified VoteNet. We observe that in all cases learning part-to-part and object-to-object correlations improved the accuracy of the pose estimation over the baseline method. We see an improvement of +17% over the baseline method with our proposed model, from 36% to 53%. This suggests that our framework makes efficient use of the object-to-object correlation information to improve pose estimation.

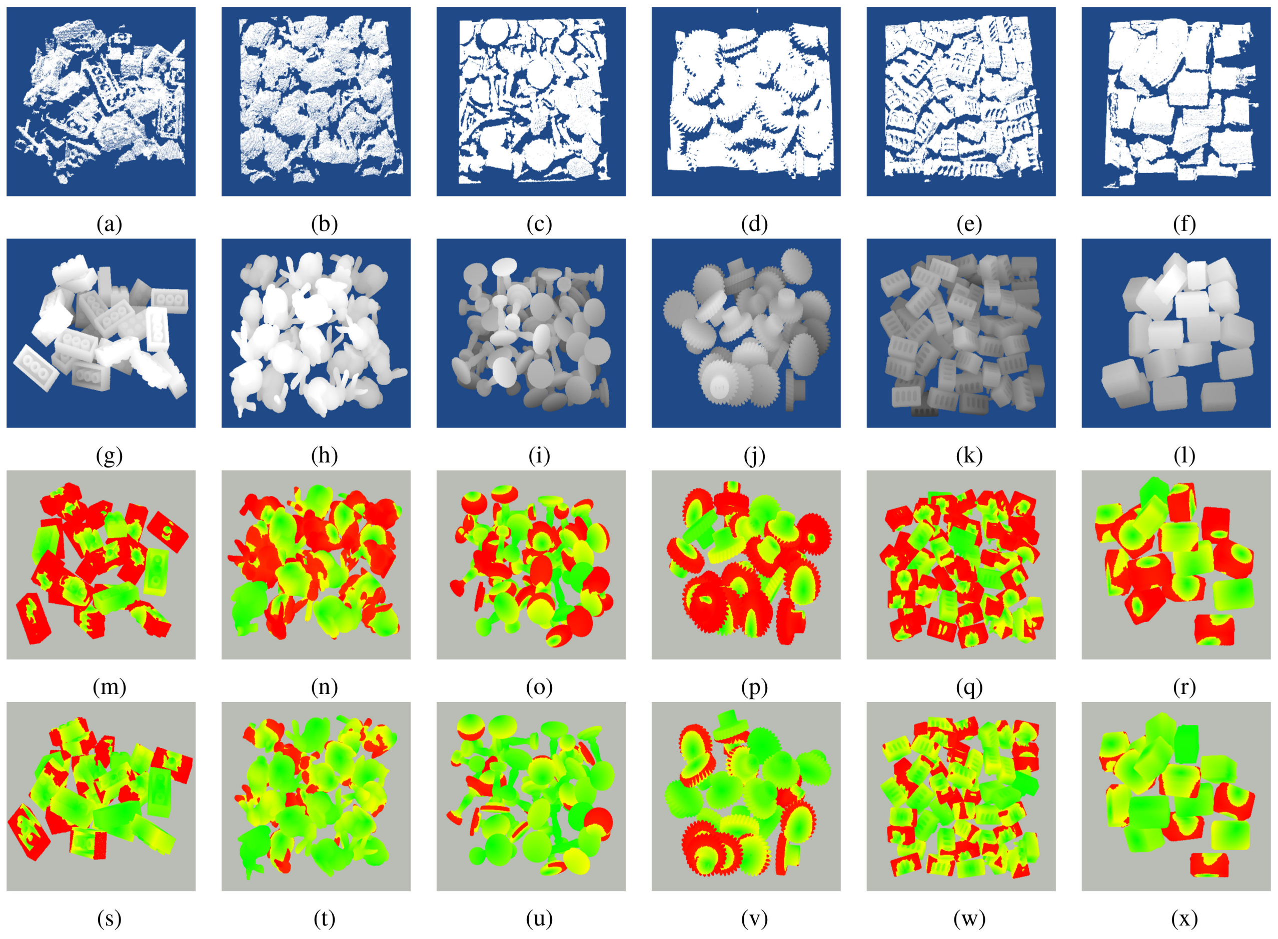


Fig. 5. Visualization for pose estimation results: (a)–(f) 3D point cloud inputs; (g)–(l) ground truth poses; (m)–(r) retrieved pose hypotheses by our baseline method (modified VoteNet [4]); (s)–(x) retrieved pose hypotheses by the proposed method. Color-coded visualization of point-wise distance error ranging from 0 (green) to greater than 0.2 times the diameter of the object (red). Predicted poses with ADD more than 0.2 times the diameter of the object have been removed for visualization purposes.

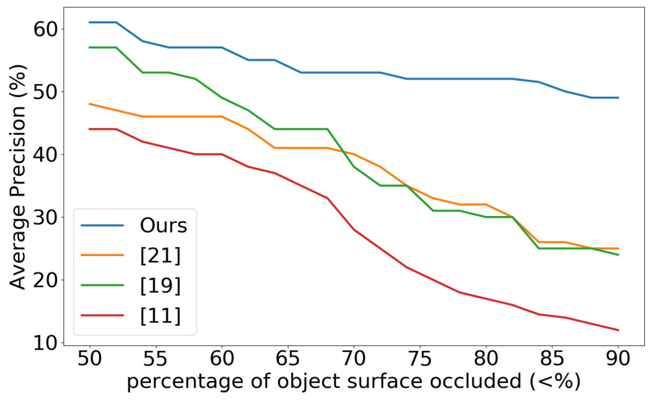


Fig. 6. Performance of different approaches under increasing levels of occlusion on the test set of Siléane dataset [35] and Fraunhofer IPA dataset [36].

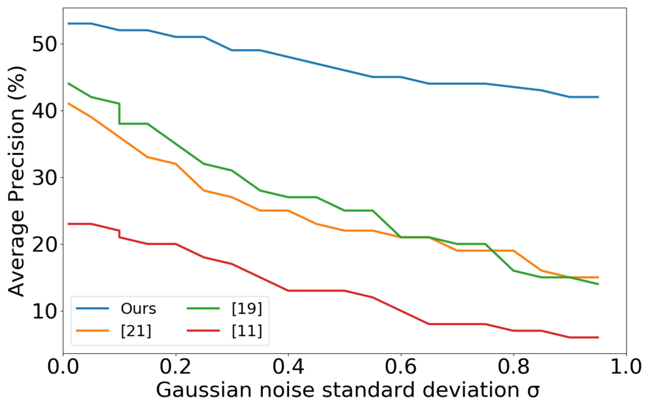


Fig. 7. Performance of different approaches under increasing Gaussian noise levels.

F. Real Robot Experiments

In our robot experiments, we evaluate the performance of the proposed approach and related works on a pick-and-place task. We collect 150 instances of 3 objects as shown in Fig. 8. Given 3D models of the objects, we develop an automated pipeline for synthetic point cloud generation and object pose annotation which runs on the platforms provided by Blender [39]. The synthetic point cloud generation method samples training examples using two distributions. The first distribution is a state distribution that randomizes over objects instances, object

TABLE II
ABLATION STUDY. WE REPORT AP VALUES OF MODIFIED VOTENET [4] FOR 6DOF POSE ESTIMATION (VOTENET), OUR PROPOSED NETWORK ALGORITHM WITHOUT LEARNING OBJECT-OBJECT CORRELATIONS (OURS (-$\mathcal{M}_{\mathcal{O}}$)), WITHOUT LEARNING PART-PART CORRELATIONS (OURS (-$\mathcal{M}_{\mathcal{P}}$)), WITH FULL OPTIONS (OURS). WE TRAINED MODELS USING EITHER ONLY LOSS IN (6) OR BOTH (6) AND (7)

Loss	VoteNet	Ours (- $\mathcal{M}_{\mathcal{O}}$)	Ours (- $\mathcal{M}_{\mathcal{P}}$)	Ours
Eq.6	0.31	0.40	0.41	**0.48**
Eq.6 + Eq.7	0.36	0.44	0.45	**0.53**

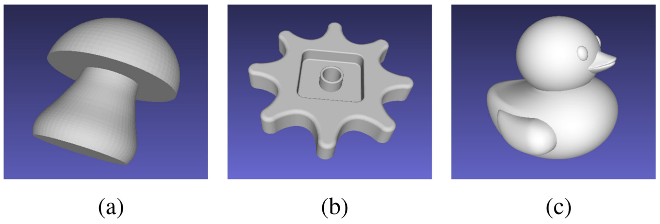
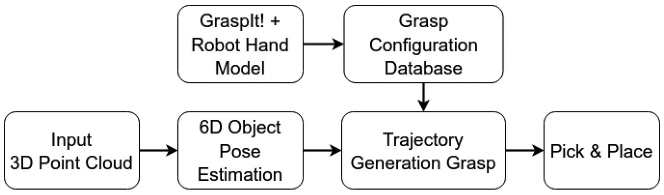


Fig. 8. 3D models of objects in real robot experiments. (a) Wood sponge. (b) Plastic gear. (c) Rubber duck.


GraspIt! + Robot Hand Model
Grasp Configuration Database
Input 3D Point Cloud
6D Object Pose Estimation
Trajectory Generation Grasp
Pick & Place
Fig. 9. A block diagram of the proposed pipeline for the robotic pick and place system.

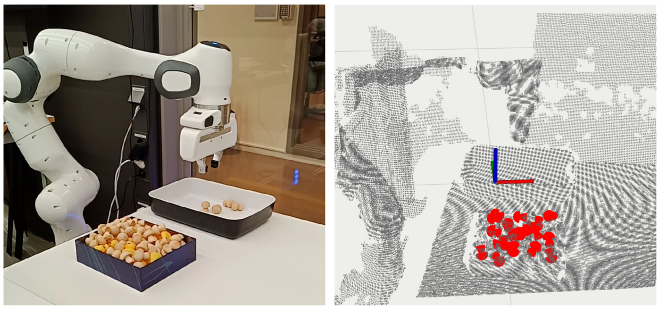


Fig. 10. Real robot experiments on pick-and-place task.

poses, and camera parameters. The second distribution is an observation distribution that models sensor operation and noise. We sample synthetic depth images using rendering and then compute point clouds from these images. We train our network model and baseline models on the synthetic data with hyperparameters as described in section IV-B and in the corresponding original papers. The trained models are then integrated into the pick-and-place system (Fig. 9) for estimating the pose of objects.

The experiments are carried on a Panda robot arm with 7-DOF, equipped with a parallel-jaw gripper as shown in Fig. 10. We use ASUS Xtion PRO LIVE sensor to capture input point clouds. The whole system is implemented using the ROS and MoveIt! frameworks. The 150 object instances are put together randomly.

TABLE III
RESULTS OF REAL ROBOT EXPERIMENTS WITH 150 OBJECTS FOR EACH METHOD. THE COMPUTATION TIME (TIME) IS ONLY FOR GRASP GENERATION PER POINT CLOUD

Method	Attempts	Success Rate	Time (s).
Linemod [37]	342	43.9%	1.60
Linemod+PP [38]	295	50.8%	3.21
PPF [9]	287	52.3%	1.52
PPF+PP [38]	270	55.6%	3.00
[15]	256	58.6%	0.30
G2L-net [17]	238	63.0%	0.21
[18]	252	59.6%	0.18
[19]	248	60.5%	0.20
Ours	**207**	**72.5%**	**0.15**

Given predicted object poses, a set of grasp configurations is then selected from a database of pre-computed grasps. Robot picks objects on top first and repeats the process until all the objects are taken away. In the following we report the success rate of the integrated system, defined as the number of objects picked divided by the total number of picking attempts made.

The most difficult object to grasp is the wood sponge. One possible reason is that there are no directly opposing faces on the object. This causes problems for two-finger grippers and leads to a failed grasp even with an accurate estimated pose. We report the success rate in Table III. Our system shows 72.5% success rate, and outperforms baseline methods by a large margin. It proves that our pose estimation model can successfully transfer to real robot pick-and-place task. Lastly, as evidenced by Table III our method also compares favorably to the SOTA in terms of computation time for generating candidate grasp configurations.

V. CONCLUSION

In this work, we introduced a new method for 6D object pose estimation from 3D point clouds. Our core idea is to employ correlation between poses of object instances and object parts to improve the performance of object pose estimation. We make use of self-attention mechanism and feature fusion to model the part-to-part and object-to-object relationships. We first produce a number of proposals using part-centric and object-centric voting-based VoteNet. We then allow higher-order feature interactions between proposals via the non-local network CGNL. Ablation studies demonstrate the effectiveness of the proposed modules to improve the accuracy of pose estimation. Experiments further show that our architecture outperforms previous approaches in benchmarks and real robot experiments.

REFERENCES

[1] K. Park, T. Patten, and M. Vincze, "Pix2pose: Pixel-wise coordinate regression of objects for 6D pose estimation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 7677–7686.
[2] Z. Li, G. Wang, and X. Ji, "CDPN: Coordinates-based disentangled pose network for real-time RGB-based 6-dof object pose estimation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 7677–7686.
[3] C. Wang *et al.*, "Densefusion: 6D object pose estimation by iterative dense fusion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3343–3352.
[4] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3D object detection in point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9276–9285.

[5] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D recognition and pose using the viewpoint feature histogram," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 2155–2162.
[6] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3D object classification," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2011, pp. 2987–2992.
[7] A. Aldoma *et al.*, "Cad-model recognition and 6dof pose estimation using 3D cues," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2011, pp. 585–592.
[8] 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.
[9] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3D object recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 998–1005.
[10] C. Choi, Y. Taguchi, O. Tuzel, M.-Y. Liu, and S. Ramalingam, "Voting-based pose estimation for robotic assembly using a 3D sensor," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 1724–1731.
[11] T. Birdal and S. Ilic, "Point pair features based object detection and pose estimation revisited," in *Proc. Int. Conf. 3D Vis.*2015, pp. 527–535.
[12] S. Hinterstoisser, V. Lepetit, N. Rajkumar, and K. Konolige, "Going further with point pair features," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 834–848.
[13] H. Wang, H. Wang, and C. Zhuang, "6D pose estimation from point cloud using an improved point pair features method," in *Proc. 7th Int. Conf. Control Automat. Robot.*, 2021, pp. 280–284.
[14] J. Guo *et al.*, "Efficient center voting for object detection and 6D pose estimation in 3D point cloud," *IEEE Trans. Image Process.*, vol. 30, pp. 5072–5084, 2021.
[15] D. Li, H. Wang, N. Liu, X. Wang, and J. Xu, "3D object recognition and pose estimation from point cloud using stably observed point pair feature," *IEEE Access*, vol. 8, pp. 44335–44345, 2020.
[16] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun, "PVN3D: A deep point-wise 3D keypoints voting network for 6dof pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11629–11638.
[17] W. Chen, X. Jia, H. J. Chang, J. Duan, and A. Leonardis, "G2l-Net: Global to local network for real-time 6D pose estimation with embedding vector features," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4232–4241.
[18] F. Hagelskjær and A. G. Buch, "PointvoteNet: Accurate object detection and 6 DOF pose estimation in point clouds," in *Proc. IEEE Int. Conf. Image Process.*, 2020, pp. 2641–2645.
[19] G. Gao, M. Lauri, Y. Wang, X. Hu, J. Zhang, and S. Frintrop, "6D object pose regression via supervised learning on point clouds," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 3643–3649.
[20] V. Mnih *et al.*, "Recurrent models of visual attention," *Adv. Neural Inf. Process. Syst.*, vol. 27, pp. 2204–2212, 2014.
[21] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *Int. Conf. Learn. Representations*, pp. 577–585, 2015.
[22] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," *Adv. Neural Inf. Process. Syst.*, vol. 28, pp. 577–585, 2015.
[23] A. Vaswani *et al.*, "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 30, pp. 6000–6010, 2017.
[24] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguist.*, 2019, pp. 2978–2988.
[25] H. Hu, Z. Zhang, Z. Xie, and S. Lin, "Local relation networks for image recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3463–3472.
[26] J. Fu *et al.*, "Dual attention network for scene segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3141–3149.
[27] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola, "Stacked attention networks for image question answering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 21–29.
[28] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7794–7803.
[29] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2005, vol. 2, pp. 60–65.
[30] K. Yue, M. Sun, Y. Yuan, F. Zhou, E. Ding, and F. Xu, "Compact generalized non-local network," *Adv. Neural Inf. Process. Syst.*, 2018, pp. 6510–6519.
[31] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.
[32] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep. ICS Report 8506, 1985.
[33] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet: Deep hierarchical feature learning on point sets in a metric space," *Adv. Neural Inf. Process. Syst.*, vol. 30, pp. 5105–5114, 2017.
[34] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5738–5746.
[35] R. Brégier, F. Devernay, L. Leyrit, and J. L. Crowley, "Symmetry aware evaluation of 3D object detection and pose estimation in scenes of many parts in bulk," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2017, pp. 2209–2218.
[36] K. Kleeberger, C. Landgraf, and M. F. Huber, "Large-scale 6D object pose estimation dataset for industrial bin-picking," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 2573–2578.
[37] S. Hinterstoisser *et al.*, "Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes," in *Proc. Asian Conf. Comput. Vis.*, 2012, pp. 548–562.
[38] A. Aldoma, F. Tombari, L. Di Stefano, and M. Vincze, "A global hypotheses verification method for 3D object recognition," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2012, pp. 511–524.
[39] Blender Documentation Team, 2019. *Blender 2.81 Reference Manual*. [Online]. Available: https://docs.blender.org/manual/en/2.81/