Graspability-Aware Object Pose Estimation in Cluttered Scenes

Dinh-Cuong Hoang, Anh-Nhat Nguyen, Van-Duc Vu, Thu-Uyen Nguyen, Duy-Quang Vu, Phuc-Quan Ngo, Ngoc-Anh Hoang, Khanh-Toan Phan, Duc-Thanh Tran, Van-Thiep Nguyen, Quang-Tri Duong, Ngoc-Trung Ho, Cong-Trinh Tran, Van-Hiep Duong, and Anh-Truong Mai

***Abstract*—Object recognition and pose estimation are critical components in autonomous robot manipulation systems, playing a crucial role in enabling robots to interact effectively with the environment. During actual execution, the robot must recognize the object in the current scene, estimate its pose, and then select a feasible grasp pose from the pre-defined grasp configurations. While most existing methods primarily focus on pose estimation, they often neglect the graspability and reachability aspects. This oversight can lead to inefficiencies and failures during execution. In this study, we introduce an innovative graspability-aware object pose estimation framework. Our proposed approach not only estimates the poses of multiple objects in clustered scenes but also identifies graspable areas. This enables the system to concentrate its efforts on specific points or regions of an object that are suitable for grasping. It leverages both depth and color images to extract geometric and appearance features. To effectively combine these diverse features, we have developed an adaptive fusion module. In addition, the fused features are further enhanced through a graspability-aware feature enhancement module. The key innovation of our method lies in improving the discriminability and robustness of the features used for object pose estimation. We have achieved state-of-the-art results on public datasets when compared to several baseline methods. In real robot experiments conducted on a Franka Emika robot arm equipped with an Intel Realsense camera and a two-finger gripper, we consistently achieved high success rates, even in cluttered scenes.**

***Index Terms*—6D object pose estimation, grasp detection, robot manipulation.**

I. Introduction

FOR grasping objects in cluttered scenes, previous research can be mainly divided into two categories: model-free grasp pose detection [1], [2], [3], [4], [5] and model-based object pose estimation [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17]. The research in the first category mainly takes three-dimensional (3D) point clouds as inputs and directly outputs a set of grasp configurations without assuming a known 3D model of the object or pre-computed grasps. One of the key advantages of this approach is its ability to generalize to novel objects. Deep learning models can be trained on a diverse dataset of objects, allowing the system to learn general grasp strategies that can be applied to previously unseen objects [1], [2]. However, the potential drawback of relying solely on grasp detection is that in numerous real-world scenarios, robots don't merely need to grip objects; manipulation extends beyond mere grasping. Robots may have the requirement to initially secure an object and subsequently engage with it in a manner that necessitates a comprehensive understanding of the object's appearance and geometry [8], [9], [10]. Therefore, despite the limitation of model-based object pose estimation methods, which require known 3D object models and do not generalize well to novel objects, they play a pivotal role in real-world applications. For example, in applications where objects need to be precisely aligned or verified against Computer-Aided Design (CAD) models, model-based pose estimation is indispensable. Robots can align components with sub-millimeter accuracy, ensuring they fit together perfectly. This is crucial in industries where safety, performance, and reliability are paramount. In our work, we assume the availability of CAD models for objects and focus on the problem of recognizing and estimating the pose of objects in cluttered scenes. A distinctive aspect of our approach, differing from existing object pose estimation methods, is our exploration of grasp detection as an auxiliary task. This auxiliary task is designed to offer supplementary information, contributing to the enhancement of pose estimation performance.

In this study, we present a novel end-to-end deep learning approach for estimating the pose of multiple rigid objects in cluttered scenes for robot manipulation, using color and depth (RGBD) images as input. To effectively fuse RGB and depth information, we propose a fusion module, $\mathcal{M}_{fus}$, which learns object features from both color and depth images. Our method employs an adaptive fusion mechanism, allowing us to enhance or attenuate specific features based on their relevance to the primary task of recovering object poses. For efficient

Manuscript received 19 September 2023; accepted 29 January 2024. Date of publication 9 February 2024; date of current version 21 February 2024. This letter was recommended for publication by Associate Editor V. Krueger and Editor C. Cadena Lerma upon evaluation of the reviewers' comments. *(Corresponding author: Dinh-Cuong Hoang.)*

The authors are with IT Department, FPT University, Hanoi 10000, Vietnam (e-mail: cuonghd12@fe.edu.vn; nhatna3@fe.edu.vn; ducvvhe176438@fpt.edu.vn; uyennthe176614@fpt.edu.vn; quangvdhe163133@fpt.edu.vn; quannpgch211110@fpt.edu.vn; anhhnhe186401@fpt.edu.vn; toanpkhe17098-3@fpt.edu.vn; thanhtdhe176812@fpt.edu.vn; thiepnvhe173027@fpt.edu.vn; tridqgch210221@fpt.edu.vn; trunghnhe172033@fpt.edu.vn; trinhtche1609-16@fpt.edu.vn; hiepdvhe181185@fpt.edu.vn; truongmahe182474@fpt.edu.vn).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3364451, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3364451

grasping, we introduce a point-wise graspability segmentation network, $\mathcal{M}_{seg}$, enabling the system to focus on suitable grasping points, and reducing computational overhead. Extracted features from the segmentation network, capturing point characteristics and contextual information, feed into the graspability-aware feature enhancement module, $\mathcal{M}_{gfe}$, enhancing feature discriminability and robustness for improved pose estimation accuracy.

The main contributions of our work can be summarized as follows: 1) *Graspability-Aware Object Pose Estimation:* We introduce a framework that estimates object poses and identifies graspable regions in cluttered scenes, enhancing robotic interaction efficiency. 2) *Adaptive Fusion Module:* Our adaptive fusion module selectively combines appearance and geometry features from RGB and depth images, improving pose estimation accuracy. 3) *Graspability Segmentation Network:* We develop a network that identifies optimal grasp points on objects, reducing computational overhead and streamlining grasping. 4) *Feature Enhancement Module:* Our graspability-aware feature enhancement module improves feature quality, enhancing object pose estimation precision.

II. Related Work

A. Deep Learning-Based 6D Object Pose Estimation

The rapid advancement of deep learning techniques has spurred numerous efforts to address object pose estimation using convolutional neural networks (CNNs) applied to RGB images [6], [9], [13], [14]. However, the inherent loss of geometric information due to perspective projection imposes limitations on the performance of these approaches in demanding scenarios such as low-light conditions, scenes with low contrast, and objects lacking distinct textures. The emergence of affordable RGBD sensors has led to the availability of more RGBD datasets [1], [6]. This additional depth information enables the extension of 2D algorithms into the 3D realm, resulting in enhanced performance. One approach in existing research [6], [9] harnesses the advantages of both data sources through cascaded designs. These methods initially estimate an object's pose from RGB images and subsequently refine it using point clouds, employing techniques like the Iterative Closest Point (ICP) algorithm or multi-view hypothesis verification. However, these refinement steps are time-consuming and cannot be optimized in an end-to-end manner with the pose estimation from RGB images. In contrast, approaches such as those in [18], [19] utilize a point cloud network (PCN) and a CNN to extract dense features separately from cropped RGB images and point clouds. These extracted dense features are then combined for pose estimation. Notably, DenseFusion [8] introduced an improved fusion strategy, replacing simple concatenation with a dense fusion module, resulting in enhanced performance. However, DenseFusion operates solely on image patches cropped using the object mask's bounding box. Our approach aligns closely with RGBD-based methods, integrating geometric and appearance data in a heterogeneous architecture. Our adaptive fusion scheme outperforms state-of-the-art methods, and a graspability-aware feature enhancement module further improves pose estimation by enhancing feature quality.

B. Deep Learning-Based 6-DoF Grasp Detection

A 6-DoF grasp refers to a grasp where gripper poses are determined by both a 3D position and orientation, resulting in a total of 6 Degrees of Freedom (DoF). The exploration of deep learning approaches for 6-DoF grasp detection has seen substantial growth in recent years [1], [2], [4], [5]. These approaches can efficiently process the entire sample space using a single network to predict either a single or multiple grasps, including specific grasp properties such as parameters and quality, solely from color images or 3D point clouds. Employing an end-to-end design, these approaches utilize a single network to process the entire input and regress an output [2], [4], [5]. While extensive research has addressed 6D object pose estimation and 6-DoF grasp pose detection for robot grasping, few works have delved into combining these tasks through multi-task learning. Recognizing the high correlation between object pose estimation and grasp detection, we propose a multi-task deep network.

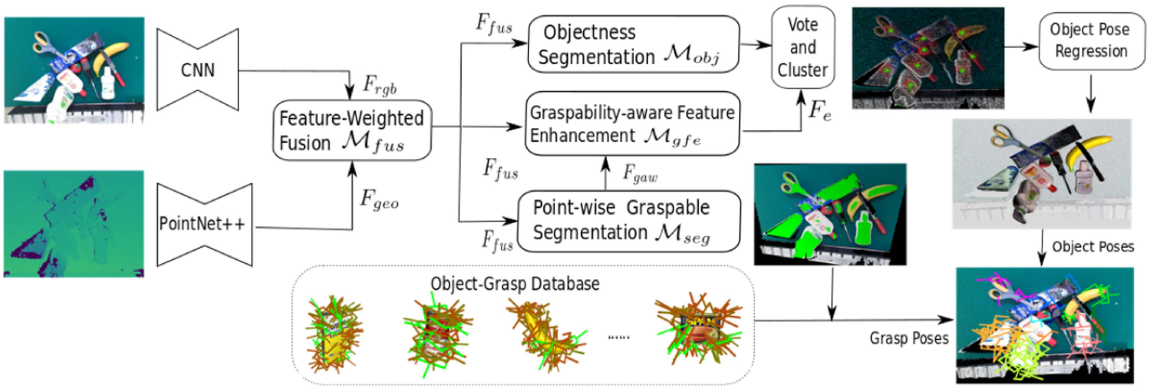
III. Methodology

An overview of the proposed method is presented in Fig. 1. Our approach consists of several key components, including feature-weighted fusion, point-wise graspable segmentation, object center voting, and multi-task loss for pose estimation. We provide a detailed explanation of each of these components and their role in our system's success.

A. Feature-Weighted Fusion

Given an input RGBD image I, we process color and depth data separately to extract unique features from each. We extract appearance features, denoted as F_{rgb}, and geometry features, denoted as F_{geo}, from a color image I_{rgb} with dimensions $h \times w \times 3$ and its corresponding depth map I_d with dimensions $h \times w \times 1$. The appearance features $F_{rgb} \in \mathbb{R}^{H \times W \times 256}$ are obtained through a two-stage process involving an encoder based on ResNet34 [20] and a decoder following the architecture of PSPNet [21]. On the other hand, geometry features, denoted as F_{geo}, are obtained from the depth map I_d, which is initially transformed into a 3D point cloud representation $P = \{p_1, \ldots, p_N\} \in \mathbb{R}^3$. This resulting point cloud is subsequently processed using PointNet++ [22] to extract geometry features, represented as $F_{geo} \in \mathbb{R}^{N \times 256}$.

PointNet++ extracts hierarchical and local features from 3D point clouds, capturing object shapes and spatial relationships. While RGB images offer rich color details, depth maps provide valuable geometry information, despite potential quality variations. Balancing the informativeness of appearance and geometry features across different scene positions poses a challenge. To address this, we propose an adaptive fusion module, incorporating weight matrices (M_{rgb} and M_{geo}) applied to appearance features (F_{rgb}) and geometry features (F_{geo}) before fusion. The



CNN
PointNet++
F_{rgb}
F_{geo}
Feature-Weighted Fusion $\mathcal{M}_{fus}$
F_{fus}
Objectness Segmentation $\mathcal{M}_{obj}$
Graspability-aware Feature Enhancement $\mathcal{M}_{gfe}$
Point-wise Graspable Segmentation $\mathcal{M}_{seg}$
F_{gaw}
Vote and Cluster
F_e
Object Pose Regression
Object Poses
Object-Grasp Database
Grasp Poses

Fig. 1. Overview of our network architecture. Features from RGB and depth data are integrated using $\mathcal{M}_{fus}$. Then F_{fus} is input into the Point-wise Graspable Segmentation module $\mathcal{M}_{seg}$ to identify grasp points. Graspability-aware features from $\mathcal{M}_{seg}$, along with F_{fus}, are processed by the Graspability-aware Feature Enhancement (GFE) module $\mathcal{M}_{gfe}$, producing enhanced features F_e. These enhanced features are then used for voting-based object pose estimation.

adaptive fusion process is defined by the equation:

$$F_{fus} = Concat(M_{rgb} \odot F_{rgb}, M_{geo} \odot F_{geo}) \tag{1}$$

Here, $M_{rgb} \in \mathbb{R}^{H\times W\times 1}$ and $M_{geo} \in \mathbb{R}^{N\times 1}$ are the weight matrices. These weights signify the importance of each location in contributing to the final prediction, and $\odot$ represents the Hadamard product. The concatenation operation $Concat()$ combines the outcomes of two tensor operations: $M_{rgb} \odot F_{rgb}$ and $M_{geo} \odot F_{geo}$, along the channel dimension, resulting in the fused feature tensor $F_{fus} \in \mathbb{R}^{N\times 512}$. Note that before concatenation we need to associate each point's geometric features with its corresponding image feature pixel by projecting them onto the image plane using known camera intrinsic parameters. Consequently, we obtain N corresponding RGB features ($\mathbb{R}^{N\times 256}$) for N point cloud features ($\mathbb{R}^{N\times 256}$), which are then concatenated.

To compute M_{rgb}, information from the feature map F_{rgb} is aggregated using average pooling along the channel axis, resulting in average-pooled features $F_{rgb}^{avg} \in \mathbb{R}^{H\times W\times 1}$. The matrix M_{rgb} is then computed as:

$$M_{rgb} = \sigma(f^{7\times 7}(AvgPool(F_{rgb}^{avg}))) \tag{2}$$

Here, σ denotes the sigmoid function, and $f^{7\times 7}$ represents a convolution operation with a filter size of 7×7. Choosing this relatively low number of parameters for the attention mechanism is a deliberate design decision to strike a balance between model complexity and effectiveness. It selectively focuses on certain spatial locations and assigns weights to different regions, highlighting the most relevant parts. The pooling and convolution operations can effectively combine global context awareness with local information. Despite having a low number of parameters, we found that this simple adaptive module was able to capture the necessary relationships between different parts of the input features.

Similarly, to calculate M_{geo}, information from the feature map F_{geo} is aggregated via average pooling along the channel axis, yielding average-pooled features $F_{geo}^{avg} \in \mathbb{R}^{N\times 1}$. The adaptive weight matrix M_{geo} is computed as follows:

$$M_{geo} = \sigma(MLP(AvgPool(F_{geo}^{avg}))) \tag{3}$$

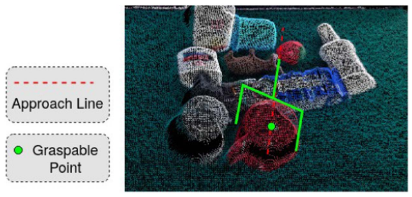


Fig. 2. Example of graspable point.

where σ represents the sigmoid function, while Multi-Layer Perceptron (MLP) is a single hidden layer network [23].

B. Point-Wise Graspable Segmentation

A graspable point is defined as the intersection point between the approach line and the object surface as shown in Fig. 2. The approach line is formed by connecting the grasp center with the origin of the gripper frame. Consequently, a graspable point is required to be located on the object's surface and visible within the sensor's field of view. The goal of point-wise graspable segmentation $\mathcal{M}_{seg}$ is to assign a label, either "graspable" or "not graspable," to each point in a 3D scene. This task is simplified compared to multi-class semantic segmentation, which typically involves labeling each point with one of several possible object categories. Following the fusion of features $\mathcal{M}_{fus}$, $\mathcal{M}_{seg}$ is approached using a MLP(512,128,2) network. "MLP" stands for multi-layer perceptron, and the numbers in brackets denote the sizes of layers [23]. Specifically, the fused features $F_{fus} \in \mathbb{R}^{N\times 512}$, serve as input to the MLP, which then produces graspability-aware features $F_{gaw} \in \mathbb{R}^{N\times 128}$ and classification scores of dimension $N \times 2$. We term these intermediate representations as graspability-aware features. To supervise the graspable segmentation task, the focal loss [24] is employed, defined as:

$$L_{graspable} = \frac{1}{N}\sum_{i=1}^{N}(-\alpha(1-q_i)^{\gamma}log(q_i)) \tag{4}$$

$$q_i = c_i \cdot l_i \tag{5}$$

The loss is computed over all N points. α is the α-balance parameter, γ is the focusing parameter, c_i represents the predicted confidence that point p_i belongs to each class, l_i is the one-hot representation of the ground truth class label, and q_i is calculated as the product of c_i and l_i. The network is explicitly supervised to predict point-wise graspability, necessitating the availability of ground-truth data during training.

To create training data, we first generate a set of grasp candidates $G_i = \{g_i^j \mid j = 1, \ldots, J\}$ from each point p_i by grid sampling along gripper depths and in-plane rotation angles. For each grasp g_i^j in G_i, a grasp quality score q_i is computed using a force analytic model [1]. Subsequently, a threshold q is manually set to filter out unsuccessful grasps based on the grasp quality score q_i. Once grasp configurations have been defined for individual object models, the graspability concept is extended to cluttered scenes. Cluttered scenes typically comprise multiple objects and irrelevant backgrounds. To calculate scene-level graspability, one straightforward approach involves directly projecting object-level graspability scores onto the scene using object 6D poses. However, this method has limitations. Firstly, a valid grasp of a single object might lead to collisions with the background or other objects when placed within a cluttered environment, rendering it ineffective. Secondly, depth cameras provide single-view partial point clouds, necessitating the association of the scene point cloud with the projected object points. To address these challenges, a method inspired by [1] is employed. This method involves reconstructing the scene using object 3D models and their corresponding 6D poses. Each grasp g_i^j is then subjected to a collision checking process, resulting in the assignment of a collision label c_i^j. The graspability scores are subsequently updated as follows:

$$s_i = \frac{1}{J}\sum_{j=1}^{J}(\mathbb{1}(q_i^j > q) \cdot \mathbb{1}(c_i^j)) \tag{6}$$

Here, $\mathbb{1}$ is an indicator function, ensuring that only successful grasps with quality scores exceeding the threshold and free of collisions contribute to the graspness score s_i. This comprehensive approach to creating training data for point-wise graspable segmentation accounts for the complexities of cluttered scenes.

C. Graspability-Aware Feature Enhancement

Given fused features $F_{fus} \in \mathbb{R}^{N\times 512}$ and graspability-aware features $F_{gaw} \in \mathbb{R}^{N\times 128}$, these features are initially concatenated along the channel axis to create a unified feature map $F \in \mathbb{R}^{N\times 640}$. Subsequently, a Graspability-aware Feature Enhancement (GFE) module is employed, which selectively integrates these two types of features using a Squeeze-and-Excitation (SE) block [25]. The SE block utilizes global average pooling to reduce each feature map to a single element, resulting in a 1D vector of length 640. This vector then undergoes processing through an MLP network with a hidden layer and sigmoid activation function [26]. The resulting output is element-wise multiplied with the original feature map F, thereby recalibrating the feature responses to emphasize important channels and suppress less relevant ones. The GFE module can be summarized as follows:

$$\mathcal{M}_{ca}(F) = \sigma(MLP(AvgPool(F)) \tag{7}$$

$$F_e = \mathcal{M}_{ca}(F) \otimes F \tag{8}$$

D. Object Center Voting and Clustering

Inspired by VoteNet [26], we employ the deep Hough voting scheme to generate pose proposals, aiming to enhance robustness to occlusion by accumulating evidence for object poses through voting. In the voting process, only object points are permitted to vote, necessitating a foreground-background segmentation (Objectness Segmentation $\mathcal{M}_{obj}$) before voting. This task, which categorizes points as either “object” or “not object,” is performed on the fused features F_{fuse} as input. Similar to point-wise graspable segmentation, this module utilizes an MLP and is trained with the focal loss [24] denoted as L_{seg}.

With the enhanced features $F_e = \{f_i\}$, a voting and grouping module $\mathcal{MP}$ as in [26] is employed to generate votes. Each vote represents both a 3D location that is supervised to be close to the object center and a feature vector learned for the final pose estimation task. Since only object points are allowed to vote, the output of the foreground/background segmentation $\mathcal{M}_{obj}$ is pointwise multiplied by the enhanced features $F_e = \{f_i\}$ to filter out features from background regions. Subsequently, each point p_i with feature f_i is processed through a shared MLP to predict a feature offset $\Delta f_i \in \mathbb{R}^{640}$ and a relative 3D offset $\Delta x_i \in \mathbb{R}^3$ between the point's position $x_i \in \mathbb{R}^3$ and its corresponding ground truth center $c_i \in \mathbb{R}^3$. Each vote is represented as $v_i = [y_i; h_i] \in \mathbb{R}^{3+640}$, where $y_i = x_i + \Delta x_i$ and $h_i = f_i + \Delta f_i$. To supervise the learning of the 3D offset Δx_i, a regression loss is applied:

$$L_{vote} = \frac{1}{M_{pos}}\sum_i \|x_i + \Delta x_i - c_i^p\|_H \cdot \mathbb{1}(x_i) \tag{9}$$

Here, M_{pos} represents the total number of points on the object surface, $\|\cdot\|_H$ is the Huber norm, and $\mathbb{1}(\cdot)$ is a binary function indicating whether a point belongs to an object. After voting, a distribution over object centers is obtained. Then we form K vote clusters $\mathbf{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$ by uniform sampling and finding neighboring votes.

E. Pose Estimation With Multi-Task Loss

We note that the input of Object Pose Regression module is not the 3D point cloud of individual objects, as in many other works. Instead, the input consists of vote clusters $\mathbf{C}$ from the voting and clustering module. A vote cluster $\mathcal{C}_k$ essentially represents a set of high-dimensional points, where each point v_i encapsulates both a 3D location y_i close to the object center and a learned feature vector h_i. We followed VoteNet [26] but regressing pose parameters instead of bounding boxes, which is inspired by our previous work [2]. Specifically, given a vote cluster $\mathcal{C}_k$, we perform sampling of 256 points $v_i = [y_i; h_i]$. Subsequently, these points undergo further processing via a multi-layer perceptron, which consists of three fully connected layers. All fully connected layers are sequentially followed by batch normalization and Rectified Linear Unit (ReLU) activation, except for the final prediction layer. We supervise the learning of the modules jointly with a multi-task loss:

$$L = \lambda_1 L_{pose} + \lambda_2 L_{vote} + \lambda_3 L_{seg} + \lambda_4 L_{graspable} \tag{10}$$

This loss function is composed of four individual task losses, each weighted by specific factors λ_1, λ_2, λ_3 and λ_4. We define the pose loss function as follows:

$$L_{pose} = \alpha_1 L_t + \alpha_2 L_{rot} + \alpha_3 L_{sem} \tag{11}$$

The pose loss L_{pose} consists of three components: translation loss L_t, rotation loss L_{rot}, and semantic classification loss L_{sem}, each scaled by α and β. While we can learn the translation part of the pose in Euclidean space, representing the rotation part is more complex. The rotation part is represented as a 6D continuous representation for 3D rotations [2]. The rotation loss is adapted for asymmetric objects, whereas symmetric objects employ a specialized rotation loss which computes the average distance between the vertices of the ground-truth and estimated object models [8]. Lastly, the semantic classification loss deals with assigning objects to semantic classes.

IV. Evaluation

In this section, we perform experiments to validate the effectiveness of the proposed approach on three widely used datasets including GraspNet-1Billion [1], YCB-Video [6] and Occluded-LINEMOD [27]. Furthermore, the trained models are deployed for conducting grasping experiments within real-world cluttered scenes, showcasing the practical applicability of our approach.

A. Datasets and Implementation Details

We train our network using the GraspNet-1Billion dataset [1], the only large-scale dataset publicly available with comprehensive annotations for object 6D poses and grasp poses. Featuring 88 objects with various shapes, this dataset includes 97,280 RGBD images across 190 cluttered scenes. Training and evaluating on a subset of 100 training scenes and 90 testing scenes, respectively, enables model assessment. For Occluded-LINEMOD [27] and YCB-Video [6], widely-used datasets for 6D object pose estimation, we lack grasp pose annotations. To address this, we applied the grasp pose annotation pipeline from [1]. The pipeline involves initial grasp point sampling for 21 objects in YCB-Video and 8 objects in Occluded-LINEMOD. Grasp view, in-plane rotation, and gripper depth are sampled and evaluated, and generated grasps are projected onto the scene using the 6D pose of each object. Collision detection is performed to avoid collisions between grasps and the background or other objects. We followed dataset splits for training and testing as per baselines [8], [13], [19].

Implementation Details: We standardized the preprocessing of images in all datasets, cropping and resizing them to dimensions of 256×256 pixels. Our implementation leverages a pre-trained ResNet34 model [20], which was originally trained on the ImageNet dataset [28], as the encoder for RGB images. Following the encoder, we employ a PSPNet decoder for feature processing [21]. For extracting features from point clouds, we perform a random sampling of 12,288 points from depth images and employ a PointNet++-based feature learning network [22]. Our implementations are realized using the PyTorch framework on a single Nvidia GeForce RTX 2080 Ti 11 GB GPU, running on a CUDA-enabled Linux operating system. We adopted a batch size of 8 and incorporated common data augmentation techniques. The initial learning rate was set to 0.001, and we trained the network for a total of 200 epochs. Learning rate decay occurred at epochs 80, 120, and 160, with corresponding decay rates of 0.1, 0.1, and 0.1, respectively. The training process, until convergence, takes approximately 15 hours for GraspNet-1Billion, 12 hours for YCB-Video, and 8 hours for Occluded-LINEMOD.

TABLE I
QUANTITATIVE RESUTLS ON GRASPNET-1BILLION [1], YCB-VIDEO [6] AND OCCLUDED-LINEMOD [27] DATASETS

	Object Pose (AP)			Grasp Pose (AP)		
Method	[27]	[6]	[1]	[27]	[6]	[1]
CloudPose [29]	52.1	53.4	44.6	26.6	27.5	23.3
PoseCNN [6]	55.6	58.2	55.0	30.8	31.2	29.0
PVNet [9]	70.1	71.9	61.3	37.0	37.5	33.6
SegDriven [30]	70.8	72.5	58.3	35.1	36.4	31.2
GDR-Net [13]	84.8	86.5	67.7	43.7	44.5	39.0
ZebraPose [14]	83.6	85.2	66.0	39.1	39.9	37.3
DenseFusion [8]	85.3	86.9	69.8	43.2	44.7	40.5
PVN3D [18]	85.3	88.5	71.3	44.0	45.6	41.2
FFB6D [19]	85.6	88.1	70.2	42.5	44.1	40.0
DFTr [31]	86.2	88.7	70.6	42.9	44.5	40.6
Ours($-\mathcal{M}_{fus}$)	84.1	86.8	64.8	71.6	72.4	68.7
Ours($-F_{gaw}$)	88.8	90.2	76.6	72.6	73.9	71.5
Ours($-\mathcal{M}_{gfe}$)	90.1	90.8	84.1	77.7	78.2	76.1
Ours	**92.3**	**92.2**	**88.7**	**84.5**	**85.7**	**82.3**

Ours($-M_{fus}$): our method without adaptive fusion module $-M_{fus}$. Ours($-f_{gaw}$): our method using the RGBD fused features F_{fus} without the integration of F_{gaw}. Ours($-M_{gfe}$): our method using the simple concatenated F_{fus} and graspability-aware feature F_{gaw} without M_{gfe}.

B. Evaluation on Datasets

We evaluate the performance of 6D object pose estimation using a commonly used metric ADD(-S) [6]. This metric quantifies the average distance between transformed model points, considering the estimated pose and the ground-truth pose as references. An estimated pose is considered accurate if the calculated distance is less than 10% of the model's diameter. For symmetric objects, we utilize the ADD-S variant [6], which calculates the mean distance based on the closest point distance. The evaluation results in Table I demonstrate the performance of our model when compared to state-of-the-art 6D object pose estimation methods. The experiments are based on publicly available implementations by the authors. The baselines were re-trained on the same datasets with the same settings. On GraspNet-1Billion, we achieve remarkable results, surpassing PVN3D [18] by 17.4% and DenseFusion [8] by 18.9%. Fig. 3 shows qualitative results. Additionally, the proposed approach attains AP scores of 92.3 and 92.2 on YCB-Video and Occluded-LINEMOD, respectively, outperforming all other methods.

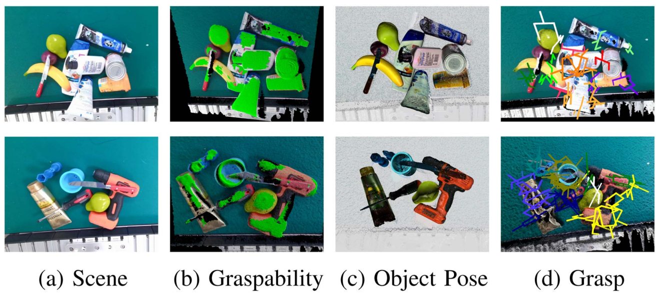


Fig. 3. Qualitative results on GraspNet-1Billion dataset [1]: (a) Input images; (b) Input 3D point cloud; (c) Point-wise graspable segmentation results; (d) Predicted poses; (e) Selected Grasp configurations.

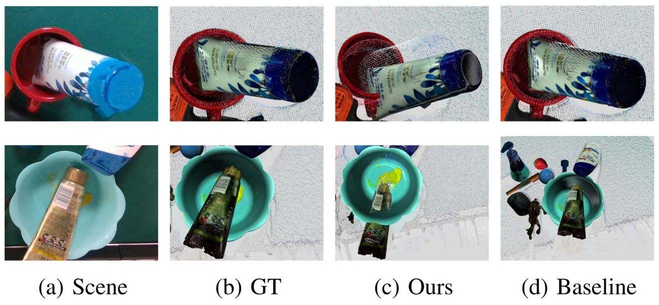


Fig. 4. Examples of our method's failures contrast with DenseFusion's capability to produce favorable outcomes. GT: Ground truth.

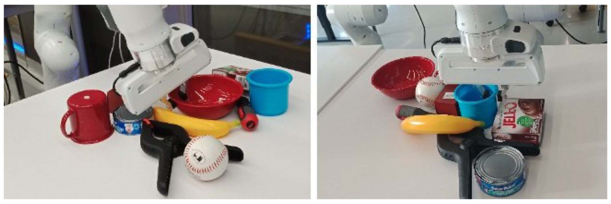


Fig. 5. Real robot experiments on cluttered scenes.

Effectiveness of the fusion module: In Table I, we observe that the fusion module $\mathcal{M}_{fus}$ plays a significant role in enhancing the performance of the proposed method. To evaluate its impact, the Ours($-\mathcal{M}_{fus}$) represents the results when $\mathcal{M}_{fus}$ is not utilized. Instead, the approach directly concatenates geometric and appearance features without the adaptive fusion module. The AP scores without $\mathcal{M}fus$ are 64.8, 86.8, and 84.1, whereas the full model with $\mathcal{M}_{fus}$ achieves substantially higher scores of 88.7, 92.2, and 92.3.

Effectiveness of F_{gaw} and graspability-aware feature enhancement module $\mathcal{M}_{gfe}$: To assess the impact of F_{gaw} and $\mathcal{M}_{gfe}$, we analyze results using only RGBD fused features F_{fus} (Ours($-F_{gaw}$)). On GraspNet-1Billion, AP performance increases from 76.6 without F_{gaw} to 88.7 with both F_{gaw} and $\mathcal{M}_{gfe}$, underscoring their critical role in achieving a 12.1% improvement. Additionally, we evaluate the effectiveness of simple concatenation of F_{fus} and F_{gaw} without $\mathcal{M}_{gfe}$ (Ours($-\mathcal{M}_{gfe}$)). Results showed a higher AP at 84.1 compared to Ours($-F_{gaw}$), emphasizing the positive impact of incorporating graspability-aware features. A direct comparison between Ours and Ours($-\mathcal{M}_{gfe}$) revealed a substantial increase in accuracy, emphasizing the crucial role of $\mathcal{M}_{gfe}$ in achieving superior 6D object pose estimation.

Evaluation on grasp pose generation: We follow prior work [1] to assess the performance of grasp pose generation using the $Precision@k$ metric. Pre-defined grasps on known object models are projected onto the scene using the predicted 6D pose of each object. We denote the average $Precision@k$ as AP_μ, where k ranges from 1 to 50 for a given friction coefficient μ. We report the average of AP_μ with $\mu = \{0.2, 0.4, 0.6, 0.8, 1.0\}$, referred to as AP. From the results presented in Table I, it is evident that the proposed method significantly benefits from the utilization of predicted graspable points in grasp pose generation.

Effect of the multi-task loss: We conducted a thorough analysis of the impact of hyperparameters on the performance of our method. Through experimentation, we determined that setting $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 0.3$ and $\alpha_1 = 1, \alpha_2 = 0.3, \alpha_3 = 0.3$ yielded the best results, as illustrated in Table I. Notably, any variation in these parameters resulted in a degradation of performance. For instance, changing λ_1 or λ_2 to 1 led to a notable decrease in the AP score for object pose, reducing it to 70.1% on average across the three datasets. Similarly, the grasp pose Average Precision (AP) dropped to 66.5%. This observation underscores the critical importance of maintaining a balance between the various losses for optimal performance.

Runtime: While achieving superior accuracy, our model maintains competitive time efficiency. With an inference time of 110 milliseconds, the proposed approach falls within a reasonable range compared to other methods. The experiments were conducted on an Intel Xeon E-2716 G CPU, 3.7 GHz, Nvidia GeForce RTX 2080 Ti GPU with 11 GB of memory.

Limitation: Despite the superior performance of our proposed method over baselines, it exhibits limitations in certain scenarios. Notably, as illustrated in Fig. 4, when the centers of objects are in close proximity, the votes for different objects may overlap, leading to inaccuracies in the predicted poses.

C. Experiments on Real Robot

In this experiment, we conducted an evaluation to assess the accuracy of the poses estimated by our approach for enabling robot grasping and manipulation. These experiments were carried out using a Franka Emika Panda robot arm equipped with a parallel-jaw gripper, as depicted in Fig. 5. To capture RGBD data, we utilized an Intel RealSense 435 camera. The MoveIt! is used for trajectory planning and control. We carefully selected objects that align with the gripper's shape and size capabilities. We then conducted experiments in cluttered scenes by randomly selecting and placing objects. Following the prediction of target object poses, we identified the grasp pose with the highest grasp score, subjected it to collision checking, and executed the grasp. Each of the evaluated methods underwent 100 grasp attempts. A grasp was considered a success if the object could be securely grasped and subsequently placed at a predefined drop point. The results are detailed in Table II. Our method achieved a success rate of 82%. This surpasses the success rates of the comparison

TABLE II
RESULTS OF REAL ROBOT EXPERIMENTS

Method	Attempt	Success	Rate	Collision Check
DenseFusion [8]	100	72	72%	137 (ms)
PVN3D [18]	100	73	73%	132 (ms)
FFB6D [19]	100	71	71%	135 (ms)
Ours($-\mathcal{M}_{fus}$)	100	76	76	55 (ms)
Ours($-F_{gaw}$)	100	78	78	52 (ms)
Ours($-\mathcal{M}_{gfe}$)	100	80	80	48 (ms)
Ours	100	**82**	**82**%	**45** (ms)

The networks were trained on the graspnet-1billion dataset. The table shows the number of attempts, successful attempts, grasp success rates, and collision checking runtime in milliseconds.

methods: DenseFusion achieved 72%, PVN3D achieved 73%, and FFB6D achieved 71%. Additionally, our model demonstrated remarkable efficiency, with a collision-checking runtime of just 45 milliseconds. In contrast, the other methods, including DenseFusion, PVN3D, and FFB6D, required significantly more time for collision checking, with respective runtimes of 137 ms, 132 ms, and 135 ms. These results affirm the advantages of integrating point-wise graspable segmentation into the pose estimation network. Leveraging detected graspable points, our method significantly reduces collision checking time, leading to more efficient and successful grasping operations.

V. Conclusion

In this paper, we presented a novel framework for graspability-aware object pose estimation in cluttered scenes, aiming to enhance the efficiency of autonomous robot manipulation. Our approach integrates depth and color information to estimate object poses and identify graspable regions. Key contributions include an adaptive fusion module for effective feature integration, a point-wise graspability segmentation network to streamline grasping computations, and a graspability-aware feature enhancement module for improved feature quality. Experimental results showcase the effectiveness of our method, achieving state-of-the-art performance on datasets and demonstrating high success rates in real robot experiments. Beyond enhancing grasp success rates, our approach exhibits efficient collision checking, making it a promising solution for real-world robotic tasks.

References

[1] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "GraspNet-1billion: A large-scale benchmark for general object grasping," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11444–11453.

[2] D.-C. Hoang, J. A. Stork, and T. Stoyanov, "Context-aware grasp generation in cluttered scenes," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 1492–1498.

[3] D.-C. Hoang et al., "Grasp configuration synthesis from 3D point clouds with attention mechanism," *J. Intell. Robot. Syst.*, vol. 109, no. 3, 2023, Art. no. 71.

[4] J. Cai, J. Cen, H. Wang, and M. Y. Wang, "Real-time collision-free grasp pose detection with geometry-aware refinement using high-resolution volume," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 1888–1895, Apr. 2022.

[5] R. Newbury et al., "Deep learning approaches to grasp synthesis: A review," *IEEE Trans. Robot.*, vol. 39, no. 5, pp. 3994–4015, Oct. 2023.

[6] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," 2017, *arXiv:1711.00199*.

[7] D.-C. Hoang, L.-C. Chen, and T.-H. Nguyen, "Sub-OBB based object recognition and localization algorithm using range images," *Meas. Sci. Technol.*, vol. 28, no. 2, 2016, Art. no. 025401.

[8] 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.

[9] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-wise voting network for 6DoF pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4561–4570.

[10] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6 D object pose and size estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2642–2651.

[11] M. Z. Irshad, T. Kollar, M. Laskey, K. Stone, and Z. Kira, "CenterSnap: Single-shot multi-object 3D shape reconstruction and categorical 6D pose and size estimation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 10632–10640.

[12] S. Agrawal, N. Chavan-Dafle, I. Kasahara, S. Engin, J. Huh, and V. Isler, "Real-time simultaneous multi-object 3D shape reconstruction, 6DoF pose estimation and dense grasp prediction," 2023, *arXiv:2305.09510*.

[13] G. Wang, F. Manhardt, F. Tombari, and X. Ji, "GDR-Net: Geometry-guided direct regression network for monocular 6D object pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 16611–16621.

[14] Y. Su et al., "Zebrapose: Coarse to fine surface encoding for 6DoF object pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6738–6748.

[15] D.-C. Hoang, J. A. Stork, and T. Stoyanov, "Voting and attention-based pose relation learning for object pose estimation from 3D point clouds," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 8980–8987, Oct. 2022.

[16] D.-C. Hoang, A. J. Lilienthal, and T. Stoyanov, "Object-RPE: Dense 3D reconstruction and pose estimation with convolutional neural networks," *Robot. Auton. Syst.*, vol. 133, 2020, Art. no. 103632.

[17] V.-D. Vu et al., "Occlusion-robust pallet pose estimation for warehouse automation," *IEEE Access*, vol. 12, pp. 1927–1942, 2024.

[18] 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. 11632–11641.

[19] Y. He, H. Huang, H. Fan, Q. Chen, and J. Sun, "FFB6D: A full flow bidirectional fusion network for 6D pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3003–3013.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[21] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2881–2890.

[22] C. R. Q. L. Y. Hao and S. L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," 2017, *arXiv:1706.02413*.

[23] 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/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.

[24] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.

[25] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

[26] 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. 9277–9286.

[27] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, "Learning 6D object pose estimation using 3D object coordinates," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 536–551.

[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[29] 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.

[30] Y. Hu, J. Hugonot, P. Fua, and M. Salzmann, "Segmentation-driven 6D object pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3385–3394.

[31] J. Zhou, K. Chen, L. Xu, Q. Dou, and J. Qin, "Deep fusion transformer network with weighted vector-wise keypoints voting for robust 6 d object pose estimation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 13967–13977.