On-Policy and Pixel-Level Grasping Across the Gap Between Simulation and Reality

Dixin Wang, Faliang Chang, Chunsheng Liu, Member, IEEE, Hengqiang Huan, Nanjun Li, and Rurui Yang

Abstract—Grasp detection in cluttered scenes is a very challenging task for robots. Generating synthetic grasping data is a popular way to train and test grasp methods, as is Dex-Net; yet, these methods sample training grasps on 3-D synthetic object models, but evaluate at images or point clouds with different sample distributions, which reduces performance due to covariate shift and sparse grasp labels. To solve existing problems, we propose a novel on-policy grasp detection method for parallel grippers, which can train and test on the approximate distribution with dense pixel-level grasp labels generated on RGB-D images. An Orthographic-Depth Grasp Generation (ODG-Generation) method is proposed to generate an orthographic depth image through a new imaging model of projecting points in orthographic; then this method generates multiple candidate grasps for each pixel and obtains robust positive grasps through flatness detection, force-closure metric and collision detection. Then, a comprehensive Pixel-Level Grasp Pose Dataset (PLGP-Dataset) is constructed, which is the first pixel-level grasp dataset, with the on-policy distribution. Lastly, we build a grasp detection network with a novel data augmentation process for imbalance training. Experiments show that our on-policy method can partially overcome the gap between simulation and reality, and achieves the best performance.

Index Terms—On-policy grasp, orthographic depth image, pixel-level grasp, grasp detection.

I. INTRODUCTION

OBJECT grasping is a core issue in the field of robots. Recently, many algorithms have claimed to be effective for handling stacked scenarios and grasping novel objects. There are currently the following two problems that are still unresolved, including on-policy and pixel-level grasping generation.

1) Some approaches train and test grasp policy on two different sample distributions [1], [2], [3], [4], [5], [6], and this process belongs to off-policy grasping strategy. Parts of these methods sample training grasps that are constrained to 3-D objects while the learned policy samples grasps from observations [1], [2], [3], [4]. The same grasp pose may have opposite labels (i.e., positive and negative) on two objects with different shapes but the same observation (Fig. 1). This means that grasp poses that are distinguishable in 3-D scenes may be indistinguishable in observations, making the distribution of training grasps different from testing grasps. Wang et al. [6] manually labeled grasps on the image, but humans cannot handle complex sample distributions. The difference in distribution would confuse a network trained on observations and reduce performance [7].

2) Previous methods may be able to obtain dense training grasps, but they struggle to obtain exhaustive grasps in the parameter space (i.e., each alternative grasping configuration is labeled as positive or negative), such as graspNet-1Billion [4] and Dex-Net [1]. GraspNet-1Billion samples the grasping configurations on the surface of object
model, and then maps all grasps into observation space (i.e., image and pointcloud). Quantitatively, GraspNet-1Billion contains a total of 1.1 billion positive and negative grasp labels, and the object masks in all images contain a total of 18.2 billion points, with one grasp labeled every 17 points on average. Our method traverses the points on the object mask to calculate the grasps, ensuring that each point is labeled with at least one grasp.

For the first problem, our method fills in invisible parts of the observation to form an orthographic depth image and computes dense 4-DOF grasp labels that are distinguishable in the observations (Fig. 1). Since the images based on the pinhole model may shrink or lose the space between objects, grasps that are actually positive may be labeled as negative due to collisions. We generate orthographic depth images by projecting points in the scene orthographic to the imaging plane, and take each pixel as the grasp center to calculate the grasp quality and generate training grasps. This method makes the training and testing grasps of the network contain approximate distributions of positive and negative samples. In RL, on-policy methods use the learned policy to collect data and avoid disparity in the distribution of target policy (i.e., training) and current policy (i.e., testing) [8]. Vishal et al. [7] introduced the on-policy concept to the domain of grasping detection to represent methods that train and test on samples from approximate distributions. However, they still used 3-D models to guide label collection, which is not a pure off-policy method.

For the second problem, we compute multiple robust grasp poses for each pixel in the image. Since computing the grasp centered on each voxel is extremely time consuming, previous methods use sampling instead of traversal. We traverse the pixels of the depth image to automatically generate dense pixel-level labels that covering the full grasp parameter space.

Based on the above solutions, we propose Orthographic-Depth Grasp Generation (ODG-Generation) method. The method first renders the orthographic depth image for the cluttered scene in the PyBullet simulation environment [9] based on the orthographic projection model. Then, candidate grasp poses are generated for each pixel. Finally, some checks based on orthographic depth image are used to distinguish between positive and negative grasps. Based on the automatic ODG-Generation method, we build the first Pixel-Level Grasp Pose Dataset (PLGP-Dataset) that can serve as a base for training and evaluating grasp detection algorithms. PLGP-Dataset contains 45 550 RGB-D images taken from the different viewpoints of cluttered scenes. Each image is densely annotated, bringing over 58 million grasp poses.

We further construct an end-to-end network for learning grasping. A novel data augmentation method is proposed to extend positive samples to neighboring pixels to reduce data imbalance in network training. Meanwhile, we contribute a simulation bin-picking benchmark that can provide synthetic RGB-D images and point clouds embedded with noise, and drive the simulated robotic arm to grasp object according to a given grasp pose. We design rich experiments to demonstrate that our on-policy method outperforms the off-policy methods, and the pixel-level grasp labels improve generalization. Besides, experiments demonstrate that our method can partially overcome the gap between simulation and reality.

The contributions of our study are summarized as follows.

1) We propose an on-policy grasp method that avoid covariate shift caused by training and testing the network on different distributions.
2) We propose an ODG-Generation method to collect RGB-D images and generate pixel-level grasp labels.
3) The first Pixel-Level Grasp Pose Dataset (PLGP-Dataset) is built which can serve as a base for training and evaluating grasp pose detection algorithms.
4) Based on the PLGP-Dataset, we train a grasp detection network which achieve the state-of-the-art performance.

II. RELATED WORK

The goal of grasp detection is to output a gripper pose using the visual information of cluttered scenarios, and the gripper can stably grasp the target when closing the jaws in this posture. In recent years, deep learning-based methods have attracted the attention of many researchers.

Dataset: In order to train the grasp detection network, many datasets are proposed. Cornell dataset [10] collects RGB-D images of scenes containing a single object, and manually annotates rectangle grasps. L. Pinto [11] and S. Levine [12] used real robots for random grasping, and record the successful grasps as labels. These methods are time consuming and insufficient to cover all the grasp parameter space. To avoid such problem, [13] used 3-D object models to build cluttered scenes in simulation environment, and used simulation robots to collect grasp labels. The simulation-based method improves efficiency, but the grasp labels are still sparse. Mahler et al. [17] first calculated 6-DOF grasps on the 3-D object models through force-closure metric, then obtained the 4-DOF grasp pose perpendicular to the table [2], [3]. To avoid the gap from simulation to reality, Fang et al. [4] constructed 3-D models of real objects to sample 6-DOF grasp poses, and recorded point clouds of real stacked scenes to train grasp detection networks. However, networks trained on these datasets need to fit functional relationships from images or point clouds to grasp poses generated based on 3-D models, and the difference in data distribution reduces the performance. On the contrary, we propose a label generation method that works directly on depth images and avoids differences in the data distribution of training network.

Method: Based on different grasp pose representations, grasp detection networks are mainly divided into three categories: 1) grasp quality classification networks; 2) 4-DOF networks based on RGB-D images; and 3) 6-DOF networks based on point clouds. Based on grasp quality classification networks, Dex-Net [1], [2], [3] first samples multiple candidate grasp poses on the depth image, and then uses the network to evaluate the quality of each grasp. This method is less efficient and the candidate grasps cannot cover the grasp parameter space. Based on 4-DOF networks, discrete or pixel-level 4-DOF grasp poses are predicted end-to-end [6], [18], [19], [20], [21]. Morrison et al. [22] proposed GGCNN, which is a generative grasping network and predicts the 4-DOF grasp pose at each pixel.
Our previous work AFFGA-Net [6] improved the accuracy of pixel-wise prediction using a hybrid feature fusion approach. Yu et al. [23] proposed SE-ResUNet, which introduced an attention mechanism to improve the utilization of important features. But fewer degrees of freedom limit the reach of the gripper. Based on 6-DOF networks, sparse 6-DOF grasp poses are predicted in a decoupled manner [4], [24], [25], [26], [27]. Ni et al. [28] used PointNet++ to generate and evaluate grasps. Zhao et al. [5] proposed an end-to-end network REGNet, which generated grasp poses from observed point cloud, and its performance was superior to several methods such as GPD [29], PointnetGPD [30], and S4G [27]. Fang et al. [4] proposed learning the approach direction and grasping parameters in a decoupled manner, which achieved good performance. However, the extreme imbalance of positive and negative samples makes the 6-DOF networks difficult to train. Note that our grasp pipeline is similar to most 4-DOF grasping methods, such as [31], the main differences of our method are the calculation method of grasp labels and network structure. 6-DOF grasping is more flexible. But in the bin-picking task we study, using 4-DOF grasp has the following advantages.

1) **Application Scenarios:** 4-DOF grasping is suitable for general grasp tasks, such as bin-picking [32] and stacking [33]. 6-DOF grasping is suitable for grasp tasks with downstream subtasks, such as handover [34] and reorientation [35].

2) **Cost:** 4-DOF grasping can be used for industrial robotic arms with fewer degrees of freedom, reducing operating costs, while 6-DOF grasping is only applicable to 6-DOF robot arms.

3) **Data Distribution:** Our 4-DOF ODG-Generation method can generate training grasps whose distribution approximates that of testing grasps, and the grasp labels cover the entire parameter space, i.e., all pixels. Most 6-DOF methods generate grasp labels for observation data based on 3-D models and sampling, resulting in nonexhaustive training grasps with a distribution different from that of testing grasps (Fig. 1).

4) **Computational Efficiency:** Most 4-DOF grasp detection networks directly predict the grasp poses end-to-end with high efficiency. Due to the difficulty of directly predicting high-dimensional parameters, most 6-DOF methods predict approach vectors and grasp parameters in two stages with low efficiency.

Pixel-level grasping is similar in nature to human grasping. Before grasping, humans first identify the graspable regions on the surface of objects based on observation information, and then select a position within the area for grasping. Similarly, in order to identify graspable regions in an image, the grasping algorithm needs to predict the probability that each pixel belongs to a graspable region (i.e., grasp quality) and the corresponding grasp pose, namely, pixel-level grasping.

### III. ORTHOGRAPHIC-DEPTH GRASP GENERATION

#### A. Overview

Our ODG-Generation method has the following four features.

1) Our method can fully automatically generate datasets based on object models, and simulation models are readily available on the Internet.

2) All object states are readily available and fully accurate, such as pose, point cloud and shape. Real-world datasets require human-labeled object poses, which inevitably introduce errors.

3) The grasp labels obtained by our method are on-policy and pixel-level. On-policy means that the distributions of training and testing grasps are approximated. Pixel-level means there are no omission grasp labels.

4) Our method can be easily extended to the real-world, which is described in detail in Section VI.

Based on the ODG-Generation method, we construct a dataset in cluttered scenario for pixel-level grasp pose detection named PLGP-Dataset. The dataset contains 9110 simulated cluttered scenes, and each contributes five RGB-D images from different views, bringing 45 550 images in total. We also provide noise-added depth images to simulate real camera. The grasp poses for each image varies from 100 to 7000, and in total our dataset contains over 58 million grasp poses. Besides, we also provide accurate object 6-D pose annotations, object masks, and bounding boxes. Fig. 2 illustrates the key components of our dataset. A summary of the properties of public grasp datasets can be found in Table I.

#### B. Data Collection

Our dataset contains 4555 3-D object models from three model datasets including: 2204 synthetic models from ShapeNet [36], 70 laser scans from the YCB object database [37], and 2281 synthetic models from EGAD [38]. All objects are diverse in shape, and scaled to fit the gripper. To collect images, a simulated camera with a vertical field of view of 60° and output image size of 480 × 640 is constructed.

Two sets of stacked scenes containing 5 and 15 random objects, respectively, are constructed. The objects are loaded into the simulation environment in random poses and fall freely into the tray to form cluttered scenes. For each scene, we randomly sample 5 view points (i.e., camera poses) on a hemisphere with the scene center as the origin and generate synchronized RGB-D images. The scene center is located on the optical axis of the camera. The radius of the hemisphere is set to 0.7 m to make the camera’s field of view encompass all objects in the scene. The mask and pose of objects are exported by PyBullet [9].

Omitting fine-tune the network on real data [39], [40], we partially overcome the gap between simulation and reality by making the synthetic depth images realistic, as with Dex-Net [1]. We follow the following steps below to generate a more realistic depth image.

1) The values in the depth image are probabilistically set to 0, and the probability is proportional to the local gradient of the point to simulate the missing regions of the depth image where the depth varies greatly.

2) Gaussian noise is added to the image to simulate the depth error of real cameras.

---

**References**

[1] **Yu et al.** [23], proposed SE-ResUNet, which introduced an attention mechanism to improve the utilization of important features.

[2] **Ni et al.** [28], used PointNet++ to generate and evaluate grasps.

[3] **Zhao et al.** [5], proposed an end-to-end network REGNet, which generated grasp poses from observed point cloud, and its performance was superior to several methods such as GPD [29], PointnetGPD [30], and S4G [27].

[4] **Fang et al.** [4], proposed learning the approach direction and grasping parameters in a decoupled manner, which achieved good performance.

[5] **Omitting fine-tune the network on real data** [39], [40], we partially overcome the gap between simulation and reality by making the synthetic depth images realistic, as with Dex-Net [1].
Fig. 2. Key components of PLGP-Dataset. RGB-D images are rendered in pybullet simulation environment from different views. The grasp poses, the 6-D pose of each object, the bounding boxes and the instance masks are annotated.

TABLE I

| Dataset                  | Graps per image | Objects per image | Grasp label | Modality | Total objects | Total grasps | Total images | Data source | Automated generation | Pixel level | On-policy |
|--------------------------|-----------------|-------------------|-------------|----------|---------------|--------------|--------------|-------------|----------------------|-------------|-----------|
| Cornell [10]             | ~8              | 1                 | Rect.       | RGB-D    | 240           | 8,019        | 1,035        | Cam.        | No                   | No          | No        |
| Pinto et al. [11]        | 1               | —                 | Rect.       | RGB-D    | 150           | 50K          | 50K          | Cam.        | Yes                  | No          | No        |
| Levine et al. [12]       | 1               | —                 | Rect.       | RGB-D    | —             | 800K         | 800K         | Cam.        | Yes                  | No          | No        |
| Dex-Net 2.0 [2]          | 1               | 1                 | 4-DOF       | Depth    | 1,500         | 6.7M         | 6.7M         | Sim.        | Yes                  | No          | Yes       |
| Dex-Net 2.1 [3]          | 1               | ~5                | 4-DOF       | Depth    | 1,500         | 102.6K       | 102.6K       | Sim.        | Yes                  | No          | Yes       |
| Jacquard [13]            | ~20             | 1                 | Rect.       | RGB-D    | 11k           | 1.1M         | 54K          | Sim.        | Yes                  | No          | Yes       |
| VMRD [14]                | ~20             | ~3                | Rect.       | RGB      | —             | 100K         | 4.683        | Cam.        | No                   | No          | No        |
| MultiObject [15]         | ~30             | ~4                | Rect.       | RGB-D    | —             | 2904         | 96           | Cam.        | No                   | No          | No        |
| VR-Grasping-101 [16]     | 100             | 1                 | 6-DOF       | RGB-D    | 101           | 4.8M         | 10K          | Sim.        | Yes                  | No          | No        |
| GraspNet-1 Billion [4]   | 5K~18K          | ~10               | 6-DOF       | RGB-D    | 88            | 1.2B (incl. negative) | 97K | Cam. | Half | No | No |
| PLGP-Dataset (Ours)      | ~1280           | S/15              | 4-DOF       | RGB-D    | 4.5K          | 58M          | 45.5K        | Sim.        | Yes                  | Yes         | Yes       |

Rect.: Grasp rectangle [10]. Cam.: Camera. Sim.: Simulation.

3) The image is sequentially scaled down by $n$ times and scaled up to original size to remove small depth variations which are often not captured by real depth camera.

C. Grasp Definition

We simplify the 6-DOF grasp pose $g$ to a 4-DOF grasp $\tilde{g}$ parallel to the camera’s $z$-axis: $\tilde{g} = [p, d, \theta, w]$, where $p \in \mathbb{R}^{2 \times 1}$ is the grasp point in the image, $d \in \mathbb{R}$ denotes the gripper approaching distance at $p$, $\theta \in \mathbb{R}$ denotes the angle between the gripper closing direction and the image horizontal axis, and $w \in \mathbb{R}$ denotes the gripper opening width. The corresponding grasp pose $\tilde{g}$ in camera coordinate frame can be calculated given $\tilde{g}$ and camera intrinsics.

D. Orthographic Depth Image-Based Grasp Annotation

Most previous grasp datasets have sparse labels, and different distributions for training and testing, which reduces the performance. We compute dense grasp labels based on depth images. The usual depth image is generated based on TOF or structured light [41] which are actually pinhole imaging models, so some information in the scene cannot be collected due to occlusion [Fig. 4(a)]. To solve the above problems, we propose an ODG-Generation method. The method first generates an orthographic depth image of the scene, then aligns the pixels in training images $I_p$ with the points in image $T_c$, and finally computes robust grasp poses using the orthographic depth image. The automated annotation pipeline is illustrated in Fig. 3, and the detailed process is described as follows.
For the convenience of description, let $I_c$ and $I_p$ denote the usual camera depth image and orthographic depth image, respectively, let $O$, $X$, and $C$ denote the coordinate frame of the 3-D object model, the simulation environment, and the simulated camera, respectively.

1) Orthographic Depth Image Generation: Different from image $I_c$, projection lines of points in the scene that generate image $I_p$ are parallel to each other instead of meeting at the optical center, which is illustrated in Fig. 4. In our experiments, the imaging plane of the image $I_p$ coincides with the $x$-$y$ plane of frame $C$, so the image $I_p$ can be generated using the simulated camera intrinsics.

For a point $p_O$ on the 3-D object surface, the projected point $p_C = [x_C, y_C, z_C]$ in frame $C$ is formulated as $p_C = T_C^X \cdot T_X^O \cdot p_O$, where $T_C^X \in \mathbb{R}^{4 \times 4}$ denotes the transformation matrix of frame $X$ relative to $C$, and $T_X^O \in \mathbb{R}^{4 \times 4}$ denotes the transformation matrix of frame $O$ relative to $X$. Then, the projected point $\hat{p} = [\hat{x}, \hat{y}]^T$ of the image $I_p$ of size $H \times W$ is formulated as follows:
\[
\begin{bmatrix}
\hat{c} \\
\hat{r}
\end{bmatrix} =
\begin{bmatrix}
\eta & 0 & (W - 1)/2 \\
0 & \eta & (H - 1)/2 \\
0 & 1 & 1
\end{bmatrix} \cdot
\begin{bmatrix}
x_C \\
y_C
\end{bmatrix}
\]
\[
(1)
\]

where $\hat{c}$ and $\hat{r}$, respectively, represent the column and row coordinates of $\hat{p}$ with the upper left corner of the image $I_p$ as the origin. $\eta$ is the projection ratio of the actual length in the image (5000 in this study). The depth value of the image $I_p$ is $z_C$. Fig. 4 shows the simplified principle of generating camera depth image and orthographic depth image, the gap between two objects is lost in image $I_c$, and preserved in image $I_p$.

2) Grasp Pose Calculation: In order to calculate the pixel-level grasp pose, we traverse each pixel on $I_c$ and calculate the aligned point $\hat{p}$ on image $I_p$, and use the $\hat{p}$ as grasp center to calculate the robust grasp pose based on the image $I_p$.

For the point $\hat{p}$, grasp angle $\theta$ and depth $d$ are searched in a 2-D grid $\Theta \times D$, where $\Theta = \{x_k \mid 0 \leq k < K, k \in \mathbb{N}\}$ and $D = \{g \mid 0 \leq k \cdot g < L_1, k \in \mathbb{N}\}$. $K$ is the number of grasp angle samples, we experimentally set $K = 18$. $g$ is the grasp depth sampling interval, we set $g = 2 \text{ mm}$. $L_1$ represents the maximum grasp depth of the gripper. Gripper width $w$ is set to the opening width when the gripper just collides with the object plus tolerance (5 mm in our experiments). After these steps, we generate densely candidate grasp poses.

Then, a series of checks are performed to get positive grasps. Flatness measures the stability of the grasp, and the force-closure metric measures the slipperiness.

Flatness detection measures the stability of grasp. Under the assumption that the bottom of object surface is solid, the actual contact point $p$ between the gripper and object is used as the boundary, the upper area has no contact with the gripper, and the lower area has the same shape as the horizontal line near the contact point $p$. Therefore, the stability of grasp is mainly related to the shape of the horizontal line near the contact point $p$. Two points with a depth of $d$ in the closing direction of the gripper in the image $I_p$ are selected as contact points. Let $\Omega$ denote points that are adjacent to each contact point and have the same depth. We fit a straight line using $\Omega$ and use variance to judge flatness. The grasps with variance greater than threshold are filtered out.

Force-closure metric measures the slipperiness of the grasp. We compute the 2-D force-closure metric [17] at each contact point. Force-closure metric is met if the grasp axis lie within the friction cone at contact point, so that the object will not slide relative to the gripper. Note that 3-D object model is not taken into account. Friction coefficient is set to 0.2 to ensure that most objects can be grasped stably.

Collision detection is performed to prevent the gripper from colliding with objects. We use larger-than-true coefficients of friction, geometry of objects and gripper to build force-closure metric and collision detection. Therefore, the generated grasps can tolerate small motion errors. Because grasp detection algorithms usually do not have object recognition capabilities, they cannot identify the center of mass of objects during testing. Correspondingly, we assume that when the grasps fit the flatness and force-closure metrics which we set large threshold, they are stable regardless of the radius of grasp wrench. After these steps, densely positive grasps are generated. There may be multiple positive grasps at one grasp angle of a point, and we choose the grasp with largest depth as the label. Exhaustive computations approximate the distribution of training grasps to the distribution of testing, which avoids covariate shift.

Model-based methods, such as Dex-Net [1], cannot generate positive grasps for the object in Fig. 5, because the sampled grasps cannot fit the force-closure metric due to the excessively large angle between the two sides of the object. In contrast, ODG-Generation assumes that the area below the surface of object is solid, so the angle between the two sides of object is zero, and the sampled grasps fit the force-closure metric, resulting in dense positive grasps.

E. Evaluation

We design a bin-picking benchmark for testing where 320 cluttered scenes are constructed using 77 seen objects and 77 novel objects. We further divide our test scenes into four
categories according to the number of objects \( m = \{5, 15\} \) and whether objects have been seen or novel.

To evaluate the performance, previous methods rely on the available information to calculate the grasp stability. The rectangle metric [10] considers a grasp as correct if the rotation error is less than \( 30^\circ \) and the rectangle IOU is larger than 0.25. Fang et al. [4] believed that the correct grasp fits force-closure metric and collision detection. However, slight collisions and tilts sometimes do not cause the grasp to fail.

In this work, we adopt simulation grasp to evaluate the grasp performance. For each predicted grasp pose and friction coefficient, a simulated gripper is driven to the grasp pose to close and then lift, the prediction is correct if the object is lifted to the specified location. We compare performance on this benchmark with the following metrics.

1) **Success Rate (SR):** Objects in the scenes are grasped one by one with the first-ranked grasp until either a) no objects remain or b) the robot has five consecutive failed grasps. SR measures the percentage of grasp attempts that lift an object.

2) **Percent Cleared (PC):** The fraction of objects that are lifted when the aforementioned grasp ends [3].

3) **Average Precision (AP):** The average precision of top-50 ranked grasps after NMS processing. The scene is reset to its initial state after each grasp process.

We report all metrics as the average of the results at different \( \mu \) ranging from 0.2 to 1.0, with \( \Delta \mu = 0.2 \) as interval.

IV. NETWORK

A. Network Architecture

Fully convolutional network (FCN) [42] is selected as the basic structure of our network since the grasp label is pixel-level. FCN is widely used for semantic segmentation of images [43], [44], [45], [46]. Our previous work [6] proved that FCN is effective for predicting dense grasp poses. We constructed a pixel-level grasp detection network based on our previous network, AFFGA-Net [6]. AFFGA-Net first extracts features of different receptive fields and scales through the ResNet-101 network and the hybrid atrous spatial pyramid module, and then fuses the features through an adaptive decoder, and finally outputs the grasping parameters through three task heads.

We retain the encoder-decoder structure of AFFGA-Net as backbone and only modify the final task head to output grasp set \( \hat{G} = \{Q_p, Q_\theta, W_g\} \). We use \( Q_p \) to denote an output map of size \( H \times W \) that represents the quality of a grasp executed at each point \( p \). \( Q_\theta \) is an output map of size \( K \times H \times W \), which represents the quality of the grasp at point \( p \) with angle category \( k \), from which we calculate the grasp angle by \( \theta = \frac{\pi}{K} k \). \( W_g \) is an output map of size \( K \times H \times W \), which represents the required gripper width \( w \) to execute a grasp at point \( p \) with angle \( \theta = \frac{\pi}{L} k \). Given the highest quality point \( p \), highest quality angle \( \theta \) and gripper width \( w \), grasp depth \( d \) is calculated from the depth image so that the gripper does not collide with the scene, and \( d \) does not exceed \( L_3 \). Although our dataset includes grasp depth labels, a preliminary experiment demonstrated that predicting additional grasp depths reduces learning efficiency, while subsequent experiments demonstrated that heuristically computing grasp depths can still maintain high grasps performance. The network structure is shown in Fig. 6. The input to AFFGA-Net is an optional single RGB, depth, or RGB-D image.

B. Augmentation and Normalization

We take all annotated grasp poses as positive samples and other grasp poses in the parameter space as negative samples. In PLGP-Dataset, the ratio of positive and negative samples is 1:1439. Significant imbalances in the number of samples degrade performance. In addition to using the focal loss function [47] to reduce the difference in loss of different samples, we also perform data augmentation. Specifically, we copy the labeled positive grasps to other pixels in its eight-neighborhood. Larger extensions may cause negative grasps to be incorrectly labeled as positive, confusing network learning and degrading performance. The ratio is 1:317 after data augmentation.

To facilitate the training of our network, we normalize the learning target as follows.

**Grasp Quality:** We treat grasp quality of each point as a scalar in the range \([0,1]\). We set the already labeled grasp points a value of 1, and set the enhanced points a value of 0.9. All other points are 0.
Grasp Angle: We convert the angle \( \theta \) into category \( k = \lfloor \frac{\theta}{K} \rfloor \), where \( K = 18 \). We treat the quality of the grasps run at point \( p \) with angle category \( k \) as a scalar in the range \([0, 1]\). The already labeled grasp angles are set to 1, and the enhanced grasp angles are set to 0.9. All other angles are 0.

Gripper Width: We scale \( w \) by \( \frac{1}{\sqrt{2}} \) to the range \([0, 1]\).

C. Loss Function

We use the sigmoid function to activate the output value, and use the binary focal loss function to calculate the loss to avoid gradient reduction and reduce the loss difference between positive and negative samples

\[
L(P, P^*, K) = -\frac{1}{N} \sum_{i=0}^{H} \sum_{j=0}^{W} \sum_{k=0}^{K} \left( (P_{ijk} \log(P_{ijk}^*) + (1-P_{ijk}^*) \log(1-P_{ijk})) \cdot |P_{ijk} - P_{ijk}^*| \cdot \alpha \right),
\]

where \( \alpha = (1 - P_{ijk} > 0) \cdot \alpha + (1 - (1 - (P_{ijk} > 0)) \cdot (1 - \alpha) \)

\( (2) \)

where \( P \) is the output map of size \( K \times H \times W \), \( P^* \) is the target, \( \gamma \) is fixed as 2 as recommended by [47]. Our network has the highest success rate in test scenarios when \( \alpha = 0.95 \).

During training, the whole network is updated by minimizing the follow objective function:

\[
L_{\text{total}} = \beta_1 L(Q_p, Q_p^* , 1) + \beta_2 L(Q_g , Q_g^* , K) + \beta_3 L(W_g , W_g^* , K)
\]

\( (3) \)

where \( \beta_1, \beta_2, \) and \( \beta_3 \) are weight coefficients of the loss. In our study, we experimentally set \( \beta_1 = 10, \beta_2 = 10, \) and \( \beta_3 = 20. \)

V. EXPERIMENTS

In this section, we experimentally demonstrate the advantages of our method under different experiments and comparisons. From extensive experiments, we aim to investigate the following.

1) Do binary focal loss function and data augmentation improve neural network performance?
2) Is network performance sensitive to sparse labels?
3) Does our on-policy method perform better than off-policy methods using the same network and dataset scale?
4) How well does the grasp method that is trained entirely on simulated datasets perform in real scenarios?
5) Does our 4-DOF grasping method perform better than 6-DOF methods in real-world bin-picking tasks?

A. Augmentation and Loss

In order to verify the effectiveness of binary focal loss function and data augmentation, we conduct ablation studies on the test scenes containing 15 novel objects, and the network input is a depth image. We follow the following steps below to improve the AFFGA-Net and test its performance.

1) Modify the final task head to output grasp set \( G \).
2) Perform data augmentation on labeled grasps.
3) Use the binary focal loss function (BFL) instead of the binary cross-entropy function to calculate the loss.

TABLE II

| ABLATION STUDIES |
|------------------|
| Baseline | BFL |
| SR (%) | PC (%) | AP (%) |
| 24.41 | 18.02 | 39.81 |
| 55.87 | 78.15 | 84.14 |
| 63.31 | 84.75 | 82.85 |
| 65.61 | 84.79 | 87.37 |

4) Combination of 2) and 3).

All networks are trained on the PLGP-Dataset and results are reported in Table II. Baseline achieves only 24.41%, 18.02%, and 39.81% on the SR, PC, and AP, respectively. Data augmentation improves accuracies by 31.46%, 60.13%, and 44.33%, and binary focal loss function improves accuracies by 38.90%, 66.73%, and 43.04% compared to the baseline, respectively. The great improvement is due to data augmentation and binary focal loss significantly alleviating the imbalance of positive and negative samples. Combining data augmentation and binary focal loss function gives the highest performance.

B. Experiments of Sparse Label

To investigate how sensitive network performance is to sparse labels, we test the performance of AFFGA-Net under different pixel strides and grasp angle bins. Using different pixel strides \( s \), we only compute the loss at coordinates that are integer multiples of \( s \). The loss function is defined as follows:

\[
L_{ij} = \begin{cases} 
L_{ij}, & \text{if } i = s \cdot n \text{ and } j = s \cdot m \\
0, & \text{otherwise}
\end{cases}
\]

\( (4) \)

where \( n \in \mathbb{N} \) and \( m \in \mathbb{N} \), \( L_{ij} \) is the original loss function. Using different grasp angle bins \( b \), we convert the grasp angle category \( k \) in the dataset to a new category \( k' \) by \( \frac{kb}{N} \).

All networks are trained on the PLGP-Dataset and tested on the test scenes containing 15 novel objects, and the network input is a depth image. Data augmentation for grasps is not used. Results are reported in Fig. 7. As the pixel stride increases, the AP value gradually decreases, and the difference exceeds 10% when stride is equal to 16, while SR and PC remain stable. The results show that as the training grasps decrease, the deviation of the network’s fit to extreme points increases gradually. As the grasp angle bin decreases, both SR and AP gradually decrease, and PC decreases rapidly when the angle bin is less than 6. Grasping
TABLE III
EVALUATION OF AFFGA-NET BASED ON DIFFERENT GRASP LABEL GENERATION METHODS

| Method       | Number of labels | Seen (%)  |
|--------------|------------------|-----------|
|              |                  | 5 objects | 15 objects | 5 objects | 15 objects |
|              |                  | SR  PC  AP | SR  PC  AP | SR  PC  AP |
| Dex-Net-1000 [1] | 537.2K           | 69.12  96.45  67.39  61.07  81.80  60.69 | 70.58  94.25  81.20  60.78  80.23  69.44 |
| Dex-Net-2000 [1] | 1,053.3K         | 73.33  97.98  72.11  63.01  83.21  64.23 | 71.50  94.33  82.76  62.31  82.05  72.81 |
| Manual [6]     | 2,294.1K         | 71.89  96.15  82.78  61.71  82.42  66.43 | 68.44  95.85  74.13  59.66  81.82  72.72 |
| ODG-Generation | 1,148.0K         | 79.77  98.10  85.24  71.33  89.82  74.01 | 71.87  97.85  85.29  63.73  89.48  75.00 |

Values in bold represent the results of the best performing method under different metrics.

C. On-Policy versus Off-Policy

We take the following off-policy methods as baseline.
1) Dex-Net-1000: Sample 1000 antipodal grasps on the surface of each 3-D object model based on Force-closure metric and then project into image space [1].
2) Dex-Net-2000: Same as Dex-Net-1000, except sampling 2000 grasps.
3) Manual: The easily identifiable grasp regions on each object are manually labeled, and the grasp labels of each pixel are decoded from each grasp region [6].

To ensure that the experimental results are not disturbed by differences in training images, we construct a dataset using 77 seen object models from the PLGP-Dataset test set where 770 simulated train scenes are constructed. Using the same images, we adopt the ODG-Generation, Dex-Net-1000, Dex-Net-2000, and Manual methods to generate grasp labels, and train AFFGA-Net, respectively. The number of labels for different methods are shown in Table III. To compare the impact of the original labels generated by different methods on the network performance, data augmentation was not used. All networks are tested on the test set of PLGP-Dataset, and the results are shown in Table III.

Both Dex-Net and ODG-Generation employ force closure to generate labels, but the distribution difference between training and testing of Dex-Net (i.e., off-policy) degrades performance. Compared with Dex-Net-2000, Dex-Net-1000 with fewer training samples reduces the performance of network. Compared with other methods, our ODG-Generation method achieves the highest performance under all metrics.

D. Experiments of Networks

To test the performance of different backbone networks, we replace the backbone module with the previous grasp detection networks GGCNN2 [31], GRCNN [48], and semantic segmentation networks UNet [49], DeepLabv3+ [50], and DANet [51]. The results are reported in Table IV.

Comparing the results of different inputs, the network with depth image as input has better performance than RGB image, because the depth image contains the shape information of the objects. The difference in accuracy is especially noticeable in lightweight networks, because it is difficult for lightweight networks to extract shape features related to grasping using color information.

Compared with heavy networks, lightweight networks with RGB and RGB-D images as input have a larger accuracy gap between seen objects and novel objects, and the gap in AP is more obvious in the scene containing 15 objects. This indicates that lightweight networks with RGB image as input are more prone to overfitting, and the actual quality of predicted grasp pose drops drastically as the predicted quality decreases.

Besides, networks taking depth images as inputs are less affected by scene complexity than networks taking RGB images as input, because color information cannot reflect the positional relationship of stacked objects.

Comparing the results of different networks under the same input, our AFFGA-Net performs best. The reasons that contribute to this achievement are the hybrid atrous spatial pyramid module extracts features of different receptive fields and scales, and the adaptive decoder provides the most relevant features for different prediction tasks.

Qualitative results of our method on public datasets are shown in Fig. 8. For objects with different data distributions, shapes, and sizes, our method shows amazing detection performance.

E. Real-World Grasping

To perform robotic grasping experiments, we use a Kinova Gen2 7-DOF robot fitted with a robotiq 2f-140 gripper. Our camera is an Intel RealSense D435i RGB-D camera and is...
Table IV: Evaluation for Networks With Different Backbone on Benchmark

| Backbone  | Input | Seen (%) | Novel (%) |
|-----------|-------|----------|-----------|
|           |       | 5 objects | 15 objects | 5 objects | 15 objects |
|           | SR    | PC       | AP        | SR    | PC       | AP        |
| GGCNN2    | Depth | 70.49    | 94.47     | 77.87 | 59.96    | 72.47     | 66.99     | 68.40    | 90.06    | 86.15     | 62.59     | 81.63     | 75.18     | 0.079     | 1.36   |
|           | RGB   | 59.88    | 82.54     | 77.97 | 47.80    | 47.46     | 57.65     | 27.46    | 43.02    | 28.43     | 27.00     | 20.48     | 7.49      | 0.083     | 1.56   |
|           | RGB-D | 65.34    | 89.80     | 75.68 | 50.85    | 57.54     | 62.86     | 29.26    | 52.80    | 30.33     | 32.16     | 30.20     | 6.47      | 0.085     | 1.66   |
| GRCNN     | Depth | 73.44    | 95.74     | 82.92 | 60.87    | 77.18     | 72.05     | 75.49    | 93.13    | 89.47     | 68.05     | 84.63     | 82.70     | 1.90      | 13.38  |
|           | RGB   | 60.17    | 83.67     | 72.76 | 48.84    | 51.32     | 55.46     | 27.85    | 46.38    | 28.25     | 30.68     | 25.66     | 6.36      | 1.90      | 13.64  |
|           | RGB-D | 64.43    | 88.01     | 77.61 | 51.27    | 58.76     | 59.25     | 30.47    | 55.71    | 29.84     | 33.97     | 35.30     | 7.23      | 1.91      | 13.77  |
| UNet      | Depth | 73.74    | 93.92     | 83.89 | 61.74    | 72.58     | 76.98     | 76.95    | 93.43    | 88.39     | 67.20     | 84.20     | 79.90     | 34.53     | 50.18   |
|           | RGB   | 67.65    | 89.82     | 74.96 | 50.44    | 51.25     | 55.14     | 63.14    | 82.33    | 81.74     | 55.51     | 75.98     | 66.92     | 34.53     | 50.24   |
|           | RGB-D | 63.75    | 86.17     | 75.67 | 52.20    | 55.47     | 60.90     | 61.91    | 88.19    | 82.98     | 56.48     | 65.35     | 68.30     | 34.53     | 50.27   |
| DeepLabv3+| Depth | 55.30    | 79.29     | 73.23 | 50.52    | 55.15     | 71.12     | 66.17    | 87.65    | 83.54     | 62.21     | 77.75     | 78.14     | 62.02     | 83.45   |
|           | RGB   | 54.26    | 77.47     | 66.69 | 45.62    | 43.95     | 58.94     | 59.96    | 83.27    | 79.75     | 55.18     | 60.44     | 71.41     | 62.02     | 83.53   |
|           | RGB-D | 56.20    | 80.27     | 69.20 | 47.02    | 48.64     | 64.63     | 58.81    | 84.93    | 82.17     | 52.77     | 60.88     | 66.55     | 62.03     | 83.57   |
| DANet     | Depth | 81.31    | 96.22     | 87.79 | 69.43    | 66.55     | 76.59     | 80.11    | 94.28    | 89.59     | 69.25     | 85.64     | 78.14     | 26.18     | 25.33   |
|           | RGB   | 69.87    | 92.84     | 82.86 | 55.78    | 64.15     | 66.14     | 66.74    | 87.67    | 83.16     | 60.67     | 69.16     | 68.64     | 26.19     | 25.36   |
|           | RGB-D | 79.29    | 96.60     | 87.97 | 68.37    | 79.83     | 76.73     | 74.42    | 91.40    | 89.97     | 67.24     | 79.49     | 77.06     | 26.19     | 25.37   |
| AFFGA-Net (Ours) | Depth | 83.14    | 96.39     | 84.67 | 75.30    | 85.84     | 75.25     | 80.41    | 92.02    | 91.13     | 65.61     | 84.79     | 87.37     | 67.59     | 29.58   |
|           | RGB   | 74.40    | 86.71     | 77.66 | 71.02    | 75.65     | 74.65     | 82.92    | 94.07    | 90.68     | 63.24     | 67.42     | 75.71     | 67.60     | 29.67   |
|           | RGB-D | 85.59    | 96.20     | 89.47 | 78.81    | 85.78     | 78.33     | 84.48    | 96.80    | 94.07     | 70.86     | 84.87     | 85.49     | 67.60     | 29.70   |

Values in bold represent the results of the best performing method under different metrics.

Fig. 9. (a) Set-up for robotic grasping experiments. (b) 40 test objects for real-world grasping. (c) 11 test objects for comparing 4-DOF and 6-DOF grasping methods.

We conducted three sets of experiments to separately test the performance of AFFGA-Net in scenes containing \( M = \{1, 5, 10\} \) objects. In the initial state of each scene, we select \( M \) objects and place them randomly in the tray. On each timestep, with the AFFGA-Net, we input a depth image and camera intrinsics, and output the grasp pose with highest predicted quality. The robot then approaches the target and closes the jaws. Grasp success is defined by whether or not the grasp transport the target object to the receptacle. If there are no objects in the tray or the robot has five consecutive failures, the next scene will be tested. Each object is placed in the tray a total of ten times. Table V shows the performance. We conduct a total of 2428 grasping trials, and the total number of failed grasps is 28. Grasping success rate decreases as the test scenes contains more objects, because stacked objects increase the error of the depth images and squeeze the positive grasping parameter space. However, the success rate in the scenes consisting of ten objects still reaches 95.92%. Meanwhile, the percent cleared are 100% in all experiments. As shown in Fig. 10, pixels with high grasp quality predicted by our network are adjacent and clustered to form regions, and the region center points have the locally highest

Table V: Robotic Grasp Results in Real-World

| Objects | SR (%) | PC (%) | Attempts | Failures |
|---------|--------|--------|----------|----------|
| 1       | 99.01  | 404    | 4        | 7        |
| 5       | 98.28  | 407    | 7        | 17       |
| 10      | 95.92  | 417    | 17       |          |

Fig. 10. Experiments for real-world robotic grasping.
TABLE VI
COMPARISON OF 4-DOF AND 6-DOF METHODS IN SINGLE-OBJECT SCENES

| Object       | GraspNet-1Billion | AFFGA-Net |
|--------------|------------------|-----------|
|              | NIP | NFG | NSG | SR(%) | NIP | NFG | NSG | SR(%) |
| Screwdriver  | 0   | 1   | 8   | 88.89 | 0   | 0   | 9   | 1     |
| Toy (blue)   | 0   | 3   | 6   | 66.67 | 0   | 0   | 9   | 1     |
| Toy (Yellow) | 0   | 2   | 7   | 77.78 | 0   | 0   | 9   | 1     |
| Stone (White)| 0   | 1   | 8   | 88.89 | 0   | 0   | 9   | 1     |
| Stone (Black)| 7   | 2   | 0   | 0     | 1   | 0   | 8   | 88.89 |
| U-disk       | 9   | 0   | 0   | 0     | 0   | 0   | 9   | 1     |
| Tape         | 3   | 0   | 6   | 66.67 | 0   | 0   | 9   | 1     |
| Fastenings   | 1   | 1   | 7   | 77.78 | 0   | 2   | 7   | 77.78 |
| Conduit      | 0   | 3   | 6   | 66.67 | 0   | 0   | 9   | 1     |
| Stapler      | 1   | 1   | 7   | 77.78 | 0   | 0   | 9   | 1     |
| Wire         | 2   | 0   | 7   | 77.78 | 0   | 0   | 9   | 1     |
| Sum          | 23  | 14  | 62  | 62.63 | 1   | 2   | 96  | 96.97 |

NIP: The number of invalid predictions. NFG: The number of failed grasps. NSG: The number of successful grasps.

TABLE VII
COMPARISON OF 4-DOF AND 6-DOF METHODS IN CLUTTER SCENES

| Method       | NIP | NUG | NHG | NOD | NSG | SR(%) |
|--------------|-----|-----|-----|-----|-----|-------|
| GraspNet-1Billion | 45  | 20  | 7   | 6   | 78  | 50.00 |
| AFFGA-Net     | 4   | 0   | 11  | 5   | 97  | 82.91 |

NUG: The number of unreachable grasps. NHG: The number of grasps that cannot hold object. NOD: The number of objects dropped while moving.

Grasp quality. When the robot performs the highest quality grasp pose, the position of the gripper end is still within the graspable region even with errors introduced by camera calibration and motor play, leading to a successful grasp.

There are three situations that can easily cause failed grasping as follows.

1) Adjacent objects are grasped at the same time, and drop during lift. The reason is that our method does not have the function of object recognition.
2) Large cylindrical objects are placed vertically. This is due to fewer training scenes with upright cylindrical objects in the PLGP-Dataset.
3) Robot movement error. The control error of the Kinova Gen2 robot we use is 5–10 mm, which causes grasping to fail easily in some space-tight locations. The experimental procedure is shown in the supplementary video. In future work, we will introduce object recognition method to filter grasp poses that contain multiple objects, and gradually enrich the PLGP-Dataset to include more scenes, and use the UR5 robot with higher control precision for manipulation tasks.

F. Compared With 6-DOF Grasping

We compare the performance of GraspNet [4] with our AFFGA-Net in real-world bin-picking tasks. GraspNet is a representative 6-DOF grasping method, and we use network weights trained on the GraspNet-1Billion dataset [4]. The test objects are shown in Fig. 9(c). Among them, stones with high hardness and smooth surface require the high precision of the grasping method, and the small U-disk requires high generalization. The results in the single object and clutter scenes are shown in the Tables VI and VII. In the single-object experiment, each object is randomly placed in the bin and then grasped, repeating nine times. In the clutter scene experiments, all objects are randomly placed in the bin and then grasped sequentially, repeating nine times. Invalid prediction means the predicted grasps do not lie on objects. Failed grasps means the following: 1) gripper cannot hold object; 2) robot arm collides with itself or the scene; 3) grasps are unreachable; 4) object dropped while moving.

In single-object scenarios, GraspNet fails to predict a valid grasp for 23.23% of prediction attempts and fails to grasp object for 14.14% of grasp attempts, most of which are due to unreachable grasp poses. Invalid predictions are mainly for U-disk and stone, which is due to the insufficient diversity of the GraspNet-1Billion dataset. Some grasps are unreachable because the angle between the grasp approach vector and the table normal is too large. Because the approach vectors for 4-DOF grasping coincide with the table normally, the grasps predicted by our method are all reachable. Meanwhile, our PLGP-Dataset uses uniformly distributed and pixel-level training samples to make the generalization of AFFGA-Net better.

VI. DISCUSSION

In this section, we discuss the limitations and expandability of our method.

The limitation is that our method cannot handle scenarios where the graspable position is occluded. Like many researchers, we represent the grasping process as three subtasks of approaching, gripping, and lifting. If the graspable position on the object is occluded, such as when the object is against a wall, the robot arm must first push the object away from the wall before grasping. We will explore more general manipulation methods to improve robot dexterity in the future.

For now, large-scale real datasets may be sufficient. But with the development of metaverse and large models, larger datasets may be required. Our virtual dataset can be used as a supplement to the real dataset. Also, our method can be extended to real world datasets to avoid extra sim-to-real. Although ODG-Generation method proposed in this article built a synthetic dataset, in fact it can be easily extended to real-world. For a stacked scene, one first estimates the pose of each object, then places the object models in the corresponding position in the simulation scene to form a data twin system, and finally renders the orthographic depth image at each camera view and uses the ODG-Generation method to generate 4-DOF grasp labels.

To extend to different types of grippers, the grasp angle sampling and negative grasps filtering process of the proposed
method need to be adapted: 1) grasp angle sampling: For each grasp point, that is, the position of palm center, grasp angle of each finger is sampled according to the relative position of the finger and palm center; 2) negative grasps filtering: Flatness detection, force-closure metric, and collision detection are performed for each finger.

VII. CONCLUSION

In this article, we proposed an on-policy grasp detection method. First, the ODG-Generation method was introduced to compute pixel-level grasp labels based on orthographic depth images, which trained and tested the grasping network on the same distribution, and improved the generalization ability of the network. Then, the first on-policy pixel-level grasp detection dataset PLGP-Dataset was constructed and released based on ODG-Generation method. The PLGP-Dataset consisted of images captured by a simulated sensor, with rich and dense annotations. At last, we constructed a grasp detection network equipped with a novel data augmentation mode and loss function. Extensive experiments demonstrated the superiority of our method in both bin-picking benchmark and real grasping. In future work, we aim to extend the proposed ODG-Generation method to generate large-scale real-world dataset, and improve robot dexterity.

REFERENCES

[1] J. Mahler et al., “Learning ambidextrous robot grasping policies,” Sci. Robot., vol. 4, no. 26, 2019, Art. no. eaau4984.
[2] J. Mahler et al., “Dex-Net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics,” in Proc. Robot.: Sci. Syst., 2017.
[3] M. Jeffrey and G. Ken, “Learning deep policies for robot bin picking by simulating robust grasping sequences,” in Proc. Conf. Robot. Learn., 2017, pp. 515–524.
[4] H.-S. Fang, C. Wang, M. Gou, and C. Lu, “GraspNet-illhilion: A large-scale benchmark for general object grasping,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. 2020, pp. 11444–11453.
[5] Z. Binglei, Z. Hanbo, L. Xuguang, W. Hanbo, T. Zhiquian, and Z. Nanning, “RegNet: Region-based grasp network for end-to-end grasp detection in point clouds,” in Proc. IEEE Int. Conf. Robot. Automat., 2021, pp. 13474–13480.
[6] D. Wang, C. Liu, F. Chang, N. Li, and G. Li, “High-performance pixel-level grasp detection based on adaptive grasping and grasp-aware network,” IEEE Trans. Ind. Electron., vol. 69, no. 11, pp. 11611–11621, Nov. 2022.
[7] V. Satish, J. Mahler, and K. Goldberg, “On-policy dataset synthesis for learning robot grasping policies using fully convolutional deep networks,” IEEE Robot. Automat. Lett., vol. 4, no. 2, pp. 1357–1364, Apr. 2019.
[8] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, “Learning synergies between pushing and grasping with self-supervised deep reinforcement learning,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2018, pp. 4238–4245.
[9] E. Coumans and Y. Bai, “Pybullet, a python module for physics simulation for games, robotics and machine learning,” 2016–2021. [Online]. Available: http://pybullet.org
[10] Y. Jiang, S. Moseon, and A. Saxena, “Efficient grasping from RGBD images: Learning using a new rectangle representation,” in Proc. IEEE Int. Conf. Robot. Automat., 2011, pp. 3304–3311.
[11] L. Pinto and A. Gupta, “Supersizing self-supervision: Learning to grasp from 50 k trials and 700 robot hours,” in Proc. IEEE Int. Conf. Robot. Automat., 2016, pp. 3406–3413.
[12] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection,” Int. J. Robot. Res., vol. 37, no. 4-5, pp. 421–436, 2018.
[13] A. Depiere, E. Dellandréa, and L. Chen, “Jacquard: A large scale dataset for robotic grasp detection,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2018, pp. 3511–3516.
[14] H. Zhang, X. Lan, S. Bai, X. Zhou, Z. Tian, and N. Zheng, “ROI-based robotic grasp detection for object overlapping scenes,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2019, pp. 4768–4775.
[15] F.-J. Chu, R. Xu, and P. A. Vela, “Real-world multijobbject, multigrasp detection,” IEEE Robot. Automat. Lett., vol. 3, no. 4, pp. 3355–3362, Oct. 2018.
[16] X. Yan et al., “Learning 6-DoF grasping interaction via deep geometry-aware 3D representations,” in Proc. IEEE Int. Conf. Robot. Automat., 2018, pp. 3766–3773.
[17] V.-D. Nguyen, “Constructing force-closure grasps,” Int. J. Robot. Res., vol. 7, no. 3, pp. 3–16, 1988.
[18] Q. Yu, W. Shang, Z. Zhao, S. Cong, and Z. Li, “Robotic grasping of unknown objects using novel multilevel convolutional neural networks: From parallel gripper to dexterous hand,” IEEE Trans. Automat. Sci. Eng., vol. 18, no. 4, pp. 1730–1741, Oct. 2021.
[19] I. Akinola et al., “Visionary: Vision architecture discovery for robot learning,” in Proc. IEEE Int. Conf. Robot. Automat., 2021, pp. 10779–10785.
[20] S. Ainetter and F. Fraundorfer, “End-to-end trainable deep neural network for robotic grasp detection and semantic segmentation from RGB,” in Proc. IEEE Int. Conf. Robot. Automat., 2021, pp. 13452–13458.
[21] X. Zhu, D. Wang, O. Biza, G. Su, R. Walters, and R. Platt, “Sample efficient grasp learning using equivariant models,” in Proc. Robot.: Sci., 2022. [Online]. Available: https://www.robotsproceedings.org/rss18/p071.html
[22] D. Morrison, P. Corke, and J. Leitner, “Learning robust, real-time, reactive robotic grasping,” Int. J. Robot. Res., vol. 39, no. 2-3, pp. 183–201, 2020.
[23] S. Yu, D.-H. Zhai, Y. Xia, H. Wu, and J. Liao, “SE-ResUNet: A novel robotic grasp detection method,” IEEE Robot. Automat. Lett., vol. 7, no. 2, pp. 5238–5245, Apr. 2022.
[24] F. Alladkani, J. Akil, and B. Calli, “ECNNs: Ensemble learning methods for improving planar grasp quality estimation,” in Proc. IEEE Int. Conf. Robot. Automat., 2021, pp. 4769–4775.
[25] X. Zhu, L. Sun, Y. Fan, and M. Tomizuka, “6-DoF contrastive grasp proposal network,” in Proc. IEEE Int. Conf. Robot. Automat., 2022, pp. 6371–6377.
[26] K. S. A. Ten Pas, M. Gualtieri, and R. P. Jr., “Grapse pose point in clouds,” Int. J. Robot. Res., vol. 36, no. 13–14, pp. 1455–1473, 2017.
[27] Y. Qin, R. Chen, H. Zhu, M. Song, J. Xu, and H. Su, “S4G: Amodal single-view single-shot 6E3Grasp detection in cluttered scenes,” in Proc. Conf. Robot. Learn., 2020, pp. 53–65.
[28] 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, 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/4b8b4e8c3b00d1247bd5d05e9b9839f6-Abstract.html
[29] G. Ren, W. Geng, P. Guan, Z. Cao, and J. Yu, “Pixel-wise grasp detection via twin deconvolution and multi-dimensional attention,” IEEE Trans. Circuits Syst. Video Technol., vol. 33, no. 8, pp. 4002–4010, 2023, doi: 10.1109/TCSVT.2023.3237866.
[30] H. Liang et al., “PointNetGPD: Detecting grasp configurations from point sets,” in Proc. IEEE Int. Conf. Robot. Automat., 2019, pp. 3629–3635.
[31] D. Morrison, P. Corke, and J. Leitner, “Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach,” in Proc. Robot.: Sci., Syst., Pittsburgh, Pennsylvania, 2018, doi: 10.15607/RSS.2018.XIV.021.
[32] H. Tachikake and W. Watanabe, “A learning-based robotic bin-picking with flexibly customizable grasping conditions,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2020, pp. 9040–9047.
[33] J. Zhang, W. Wang, R. Song, L. Ma, and Y. Li, “Grasp for stacking via deep reinforcement learning,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2543–2549.
[34] P.-K. Chang, J.-T. Huang, Y.-Y. Huang, and H.-C. Wang, “Learning end-to-end 6DoF grasp choice of human-to-robot handover using affordance prediction and deep reinforcement learning,” to be published. Accessed: Jan. 2, 2023. [Online]. Available: https://huangjuite.github.io/socially_aware_handover.pdf
[35] K. Wada, S. James, and A. J. Davison, “Reorientbot: Learning object reorientation for specific-posed placement,” in Proc. Int. Conf. Robot. Automat., 2022, pp. 8252–8258.
[36] A. X. Chang et al., “Shapenet: An information-rich 3D model repository,” 2015, arXiv:1512.03012.
Benchmarking in manipulation research: Using the Yale-CMU-Berkeley object and model set,” IEEE Robot. Automat. Mag., vol. 22, no. 3, pp. 36–52, Sep. 2015.

D. Morrison, P. Corke, and J. Leitner, “EGAD! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation,” IEEE Robot. Automat. Lett., vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

S. James et al., “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12627–12637.

R. Jeong et al., “Self-supervised sim-to-real adaptation for visual robotic manipulation,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2718–2724.

S. Zennaro et al., “Performance evaluation of the 1st and 2nd generation object and model set,” IEEE Robot. Automat. Lett., 2021, pp. 1622–1631.

B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Multicamera tracking methodology.”

D. Morrison, P. Corke, and J. Leitner, “EGAD! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation,” IEEE Robot. Automat. Lett., vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

S. James et al., “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12627–12637.

R. Jeong et al., “Self-supervised sim-to-real adaptation for visual robotic manipulation,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2718–2724.

S. Zennaro et al., “Performance evaluation of the 1st and 2nd generation object and model set,” IEEE Robot. Automat. Lett., 2021, pp. 1622–1631.

B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Multicamera tracking methodology.”

D. Morrison, P. Corke, and J. Leitner, “EGAD! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation,” IEEE Robot. Automat. Lett., vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

S. James et al., “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12627–12637.

R. Jeong et al., “Self-supervised sim-to-real adaptation for visual robotic manipulation,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2718–2724.

S. Zennaro et al., “Performance evaluation of the 1st and 2nd generation object and model set,” IEEE Robot. Automat. Lett., 2021, pp. 1622–1631.

B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Multicamera tracking methodology.”

D. Morrison, P. Corke, and J. Leitner, “EGAD! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation,” IEEE Robot. Automat. Lett., vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

S. James et al., “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12627–12637.

R. Jeong et al., “Self-supervised sim-to-real adaptation for visual robotic manipulation,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2718–2724.

S. Zennaro et al., “Performance evaluation of the 1st and 2nd generation object and model set,” IEEE Robot. Automat. Lett., 2021, pp. 1622–1631.

B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Multicamera tracking methodology.”

D. Morrison, P. Corke, and J. Leitner, “EGAD! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation,” IEEE Robot. Automat. Lett., vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

S. James et al., “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12627–12637.

R. Jeong et al., “Self-supervised sim-to-real adaptation for visual robotic manipulation,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2718–2724.

S. Zennaro et al., “Performance evaluation of the 1st and 2nd generation object and model set,” IEEE Robot. Automat. Lett., 2021, pp. 1622–1631.

B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Multicamera tracking methodology.”

D. Morrison, P. Corke, and J. Leitner, “EGAD! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation,” IEEE Robot. Automat. Lett., vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

S. James et al., “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12627–12637.

R. Jeong et al., “Self-supervised sim-to-real adaptation for visual robotic manipulation,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2718–2724.

S. Zennaro et al., “Performance evaluation of the 1st and 2nd generation object and model set,” IEEE Robot. Automat. Lett., 2021, pp. 1622–1631.

B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Multicamera tracking methodology.”

D. Morrison, P. Corke, and J. Leitner, “EGAD! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation,” IEEE Robot. Automat. Lett., vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

S. James et al., “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12627–12637.

R. Jeong et al., “Self-supervised sim-to-real adaptation for visual robotic manipulation,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2718–2724.

S. Zennaro et al., “Performance evaluation of the 1st and 2nd generation object and model set,” IEEE Robot. Automat. Lett., 2021, pp. 1622–1631.

B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Multicamera tracking methodology.”

D. Morrison, P. Corke, and J. Leitner, “EGAD! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation,” IEEE Robot. Automat. Lett., vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

S. James et al., “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12627–12637.

R. Jeong et al., “Self-supervised sim-to-real adaptation for visual robotic manipulation,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2718–2724.

S. Zennaro et al., “Performance evaluation of the 1st and 2nd generation object and model set,” IEEE Robot. Automat. Lett., 2021, pp. 1622–1631.

B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Multicamera tracking methodology.”

D. Morrison, P. Corke, and J. Leitner, “EGAD! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation,” IEEE Robot. Automat. Lett., vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

S. James et al., “Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12627–12637.

R. Jeong et al., “Self-supervised sim-to-real adaptation for visual robotic manipulation,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 2718–2724.

S. Zennaro et al., “Performance evaluation of the 1st and 2nd generation object and model set,” IEEE Robot. Automat. Lett., 2021, pp. 1622–1631.