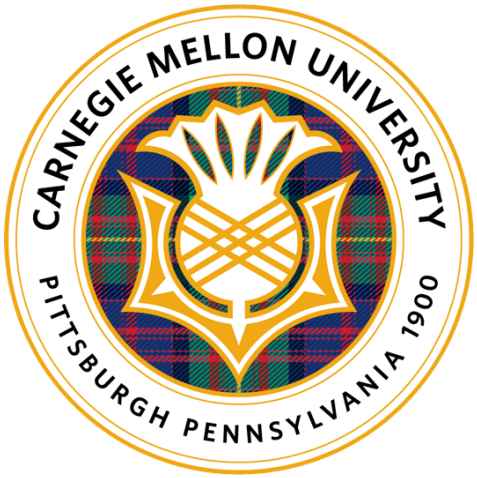
Learning to Plan Precise and Task-oriented Grasps for Autonomous Robotic Assembly

Jialiang Zhao

CMU-RI-TR-20-06
May 2020



The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania

Thesis Committee:
Oliver Kroemer, *Chair*
Wenzhen Yuan
Mohit Sharma

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

Abstract

Robust, precise, and task-oriented grasp planning is vital for autonomous robotic assembly. It requires reasoning about the object geometry and preconditions of a task so as to properly grasp an object and complete the downstream tasks. However, achieving such grasps is challenging due to the difficulty in understanding constraints and dynamics during objects interaction, as well as noise in control and unknown object properties. To tackle this problem, we proposed two data-driven, learning-based frameworks to plan precise and task-oriented grasps.

Our first experiment focuses on planning robotic grasps that are both robust and precise by training two convolutional neural networks - one to predict the robustness of a grasp and another to predict a distribution of post-grasp object displacements. Our networks are trained with depth images in simulation on a dataset of over 1000 industrial parts and were successfully deployed on a real robot without having to be further fine-tuned. The proposed displacement estimator achieves a mean prediction errors of 0.68cm and 3.42deg on novel objects in real world experiments.

Our second experiment further investigates whether a grasp is appropriate to a given downstream task. We propose a method that optimizes for grasp robustness, precision, and task performance all together by learning three cascaded networks. We collect training data based on large-scale self-supervised grasp simulation with procedurally generated objects. We form the training process as a curriculum learning problem, and perform both simulated and real world experiments on two common assembly tasks: inserting gears onto pegs and aligning brackets into corners. Our model achieves 4.28mm precision for bracket insertion and 1.44mm precision for gear insertion in real world experiments.

Acknowledgements

I would like to first thank my advisor, Professor Oliver Kroemer, for his extraordinary support during my two wonderful years in Carnegie Mellon University. I'm extremely grateful for the opportunity and the freedom he gave me to study the topic I'm most excited about, which has played an incredibly large part shaping my research path. Not only did he help me to formulate this research, but also he backed me up with great guidance and mentorship in future plan making and personal character development.

My lab mates in IAM Lab, who have been along with me though this journey, especially Jacky Liang, Kevin Zhang and Mohit Sharma, are always there to help me when I was stuck and push me to keep up with their pace.

Working with all the diligent and brilliant colleagues I met in the Robotics Institute has been a great pleasure. They made RI such a vibrant environment for research, coursework and life. I never regretted the decision of joining RI. What I learned here has been invaluable.

I'm also grateful for the funding received from the National Robotic Engineering Center and Epson America. This Master's study would not have been made possible if it were not for their generous support.

The crisis of the COVID-19 Pandemic has brought unprecedented challenges and uncertainties in deciding future paths to people who are facing transitions. I would like to thank Yun Wang, who has been standing along with me during these tough times, for always supporting me when I needed advice, encouraging me when I was under pressure, and believing in me even when I did not myself.

Finally, I would like to thank my parents, for their constant love and support since I was born. Thanks for encouraging me, believing in me, pushing me to do great things, giving me the freedom to do what I think is correct, and standing with me through all these ups and downs.

Contents

List of Figures

1 Introduction

Grasping is a fundamental skill required by robots in autonomous manufacturing. In robotic assembly, an agent needs to first grasp the object in some specific manner, and then assemble it with other objects. Precise and appropriate grasps are crucial to the success of assembly tasks. For example, to insert a bracket into a corner, a good grasp should be aware of the possible post-grasp object displacement, and avoid being too close to the insertion point (see Fig.1). The robot needs to have an estimate of how the object will be displaced as a result of the grasp, and how suitable this grasp is to the down-stream tasks. A suitable grasp could constrain the object in-hand pose more, but at the same time it should not interfere in the assembly task. Such a grasp increases the chance of successful task completion.

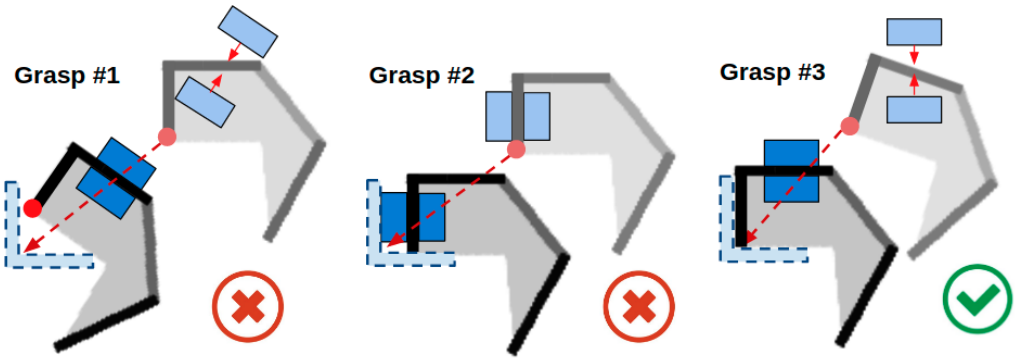


Figure 1: **Grasps for a bracket insertion task** The red point denotes the insertion frame that should be inserted to the corner. Attempt 1 doesn't consider the post-grasp object displacement and results in large insertion error (left), attempt 2 doesn't consider the insertion parameters and the gripper interferes between the object and the corner (middle), attempt 3 takes into considerations of both displacement and task parameters and achieves higher precision during insertion (right).

However, achieving such grasps is challenging due to the uncertainties in sensing, actuation, and object properties during interaction. Recent works have largely focused on efficiently picking up objects without constraining object in-hand poses [32, 28, 22, 31, 14, 4, 13], or grasping object in a specific location that is appropriate for manipulation tasks without considering precision [14, 38, 29]. Such grasps are generally unaware of post-grasp object displacement caused by the grasping motion or the task-specific contact limitation.

In this work, we address the challenge of predicting precise and task-oriented grasps for three example tasks: palletizing general industrial parts, inserting gears onto pegs and aligning brackets into corners.

In Section 2, we investigate the problem of predicting object displacements during grasping. We propose a method that estimates the post-grasp object displacement as a probabilistic distribution. We use one convolutional neural network to predict the robustness of a grasp, and another one to predict the mean and the variance of a displacement. The robot then selects grasps with a high success probability and low object displacement variance. The predicted mean object displacement can then be used to estimate the in-hand pose for downstream tasks such as assembling and palletizing objects.

In Section 3, we investigate the problem of planning grasps that are both precise and

appropriate to the downstream tasks. We form this problem as a curriculum learning problem and propose a method that decomposes it into three sub-problems - choosing robust grasps, estimating grasp precision, and choosing a suitable grasp for the task. For each of the sub-problems we propose one neural network. They are then cascaded together to predict the best grasp. This grasp can then be used to achieve better precision in down-stream tasks such as gear insertion or bracket insertion.

2 Learning to plan precise grasps by estimating the post-grasp object displacement as probability distributions

2.1 Introduction

In this chapter, we investigate the problem of predicting object displacements during grasping. We propose a method that uses two neural networks - the first predicts whether a grasping action will result in a successful grasp that allows an object to be lifted, and the second predicts a distribution over post-grasp object displacements. The robot then selects grasps with a high success probability and low object displacement variance. The predicted mean object displacement can then be used to estimate the in-hand pose for downstream tasks such as assembling and palletizing objects (see Figure 2).

Although our system is trained only in simulation, we were able to successfully deploy the networks on a real Franka Panda robot for precise grasping of 3D printed industrial parts without further fine tuning. Videos, datasets, and supplementary material are available at: https://precise-grasping.jialiangz.me.

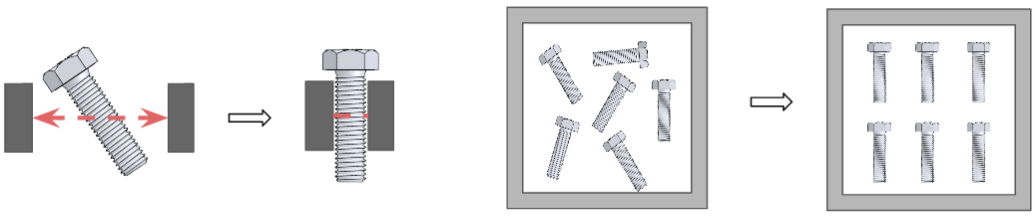


Figure 2: Example of post-grasp object displacement that our method predicts (left) and palletizing as an example application that requires precise grasp planning (right).

2.2 Related Works

To generalize grasps between objects, many recent works have used data-driven grasp synthesis [2] techniques to predict high quality grasps from vision observations. While early works used hand-tuned visual features [36], recent methods have focused on using Convolutional Neural Networks (CNNs) to learn grasp quality functions from a large amount of training data [30, 31, 40].

Robots can collect training data for grasping in a self-supervised manner by attempting thousands of random grasps and observing whether or not they result in successful lifts [25, 28, 32]. By contrast, collecting grasping data in simulation can be much faster and less costly. However, learning on simulation data often suffers from the simulation-to-reality (sim2real) gap, where the visual appearance and object dynamics of simulated data deviate from their real world counterparts. Methods for overcoming the sim2real gap include domain randomization [39] and domain adaptation [3, 20]. For grasping, the sim2real gap can be reduced by using only depth images for the visual input. Depth-only grasping methods used

in [11, 17, 30, 31] do not require further fine-tuning or domain adaptation to perform well in the real world.

Recent works have extended data-driven grasp synthesis for task-oriented grasping, where the system plans grasps that optimize for the success of downstream tasks. Detry et al. [11] had human experts label parts of objects that are suitable for tasks like handover and pouring. They trained a CNN to segment depth images based on task suitability to guide the grasp selection. Other works forgo human labels and use simulations and self-supervision to jointly optimize grasp selection and policies for downstream tasks such as sweeping and hammering [14] or tossing [41].

Due to noise in sensing and actuation, as well as unknown object properties, the grasp that the robot intends to execute is often not the grasp that is actually achieved. Jentoft et al. [23] analyzed sources of grasp variations and quantified basins of attraction for multi-fingered grasps. Dogar and Srinivasa [12] learned from human strategy and used pushing to funnel the clutter of objects before grasping to reduce uncertainty. Gupta et al. [18] explicitly learned a noise model to compensate for actuation noise of low-cost robot arms; while this method improves grasp success, it does not explicitly optimize for precise grasps. Chen et al. [7] combine a probabilistic signed distance function representation of object surfaces from depth images with analytical grasp metrics to plan grasps that optimize for small post-grasp object displacements.

Instead of choosing precise grasps that minimize object displacement, other works have explored estimating the in-hand object poses using additional sensory signals, e.g., vision and tactile [5]. To address the challenge of in-hand occlusions, Choi et al. [8] trained a CNN to segment out robot grippers such that localization can be done on only the object-relevant pixels, and Izatt et al. [19] used tactile sensing to provide point cloud data on the occluded parts of grasped objects.

In this work, we address the challenges of optimizing grasps for tasks that require precise post-grasp object poses. Given this context, we note that it is acceptable for a grasp to result in a significant object displacement as long as the robot can reliably predict the displacement and adapt the task execution accordingly. Our approach stands in contrast with previous works as we train a CNN to predict the expected post-grasp object displacement and the variance of the displacement. In this manner, the grasp planner can choose grasps that are robust and have the lowest displacement variance.

2.3 Problem Statement

Our proposed approach addresses the problem of estimating a distribution of post-grasp object displacements of top-down, parallel-jaw grasps of singulated objects lying on a flat surface. The method needs to generalize over novel objects unseen during training. We are motivated by the palletizing application and therefore focus our experiments on rigid objects commonly found in industrial and manufacturing settings, such as gears, brackets, and screws.

Let the initial pose of an object on the work surface be $\mathbf{p}$. We assume that during manipulation, the object can only undergo translational movements $(\Delta x, \Delta y, \Delta z)$ and planar rotation $\Delta\theta$, and we define the post-grasp object displacement as $\Delta\mathbf{p} = [\Delta x, \Delta y, \Delta z, \Delta\theta]^T$. As the robot will not be able to use additional sensors to perform in-hand localization, it needs to predict the displacement $\Delta\tilde{\mathbf{p}}$ based on the object pose $\mathbf{p}$ and observation $\mathbf{o}$. The observation $\mathbf{o} \in \mathbb{R}^{64\times64}$ is a depth image of the object. We use either object-centric full images or grasp-centric image patches of the object for the observations. In both cases the

image size is 64×64 pixels.

A grasp $\mathbf{g}$ has 4 degrees of freedom $\mathbf{g} = [g_x, g_y, g_z, g_\theta]^T$. The position parameters $(g_x, g_y, g_z) \in \mathbb{R}^3$ denote the location of the grasp relative to the object's geometric center $\mathbf{p}$. The orientation parameter $g_\theta \in [-\pi, \pi)$ denotes the planar rotation of the gripper about an axis orthogonal to the table surface.

Rather than predicting the post-grasp object displacement $\Delta\mathbf{p}$ directly, our networks instead predict the post-grasp *grasp* displacement $\Delta\mathbf{g}$, i.e., the difference between the grasp parameters and the realized grasp pose relative to the object coordinate frame, which is located at the object's geometric center. This grasp displacement is then converted back to the object displacement $\Delta\mathbf{p}$ for reporting the results.

2.4 Method

In this section we explain our approach for addressing this problem.

2.4.1 Overview of Approach

The proposed approach consists of three parts: 1) predicting the grasp quality, 2) predicting the distribution of post-grasp displacements, and 3) combining these predictions to choose robust and precise grasps.

Grasp Quality Prediction Following the notation in [14], we define the grasp quality Q of grasp $\mathbf{g}$ with observation $\mathbf{o}$ as the probability of a successful lift S using the grasp $Q(\mathbf{g}, \mathbf{o}) = P(S = 1|\mathbf{g}, \mathbf{o})$. We learn this mapping $Q(\mathbf{g}, \mathbf{o})$ using a neural network that we refer to as the Grasp Quality Network (GQN).

Grasp Displacement Prediction Let $\Delta\mathbf{g}$ denote the distribution of the post-grasp displacement in the object frame. We assume that the displacement follows a Gaussian distribution:

$$\Delta\mathbf{g} \sim \mathcal{N}(\mu(\mathbf{g}, \mathbf{o}), \sigma^2(\mathbf{g}, \mathbf{o}))$$

with mean $\mu(\mathbf{g}, \mathbf{o}) = \{\mu_x, \mu_y, \mu_z, \mu_\theta\}$ and variance $\sigma^2(\mathbf{g}, \mathbf{o}) = \{\sigma_x^2, \sigma_y^2, \sigma_z^2, \sigma_\theta^2\}$. We learn a neural network to predict the mean and variance of $\Delta\mathbf{g}$, which we refer to as the Grasp Displacement Network (GDN). This network allows the robot to reason about the stochasticity of the grasps and select grasps that minimize the variance over the resulting object poses.

Precise Grasp Planning Given the two learned networks, we can form a grasp planner that chooses grasps with high probability of successfully lifting the object as well as low variance over the resulting object pose.

2.5 Simulation Data

To generate grasping data for training the networks and generalizing between different objects, we collected 1011 CAD models of industrial parts such as gears, screws, bolts, and hinges from an online hardware shop[1]. Then, in simulation, 1000 random grasp attempts per object were generated and simulated. We gathered simulation data using the robotic simulation framework V-REP [34] with Bullet ver 2.78 [2] as the physics engine. We assume each object has uniform density.

As shown in Figure 3, a depth image is captured at the start of each grasp attempt. The robot robot executes a grasp $\mathbf{g}$, where the grasp center (g_x, g_y, g_z) is uniformly sampled

[1] McMaster-Carr, https://www.mcmaster.com

[2] https://pybullet.org/

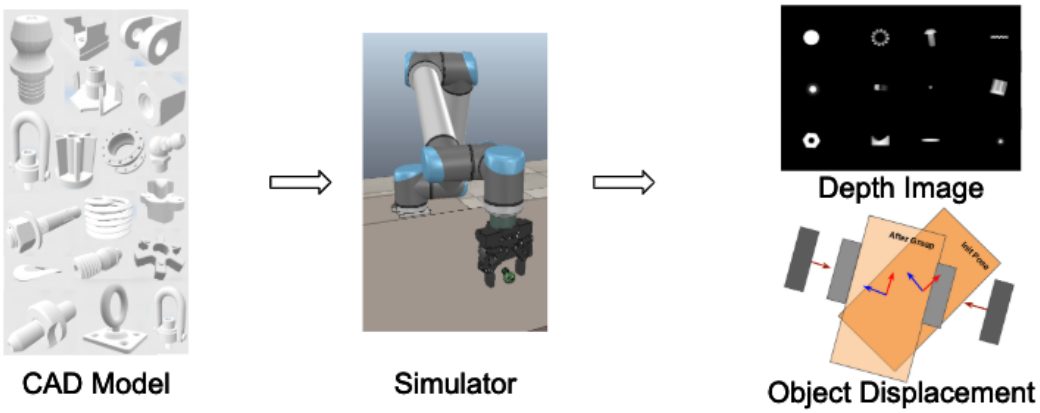


Figure 3: **Simulation Data Collection.** We collected a dataset of 1011 industrial parts (left). During simulation all objects are set to have uniform density and the same coefficient of friction. We uniformly sampled top-down grasps across the object and evaluate whether or not the grasps resulted in successful lifts (middle). In addition to lift success, we also record the top-down depth image that are used as inputs to our grasp quality and grasp displacement networks (top right). We also recorded the post-grasp object displacement (bottom right).

from the bounding box of the object, and the grasp orientation g_θ is uniformly sampled from $[-\frac{\pi}{2}, \frac{\pi}{2})$. The bounding box's coordinate frame is always parallel to the camera coordinate frame. The bounding box is calculated as the minimum area that encloses the entire object in the depth image. In contrast to previous works that sample antipodal grasps [30, 31], we use uniformly sampled grasps as using antipodal grasps may introduce a bias in the dataset when estimating post-grasp displacements. For each grasp attempt $\mathbf{g}$, we record the lift success of the grasp S, the overhead depth image $\mathbf{o}$, and the post-grasp object displacement $\Delta\mathbf{p}$.

For training the networks we collected a simulation dataset with 1.011 million grasps. After removing objects that are either 1) too hard to grasp (random grasp success rate $< 5\%$), 2) too easy to grasp (random grasp success rate $> 40\%$), 3) too big (longest axis longer than 15cm), or 4) too small (longest axis smaller than 2cm), we have 773k grasp attempts for 773 objects. Data is split object-wise, with 660 objects used for training and 113 for validation.

2.6 Grasp Quality and Post-grasp Displacement Estimation

We train two types of CNNs - the GQN and the GDN. While the GQN and the GDN share the same convolution architecture, they do not share weights. Rather, the GQN is trained first, and we use its learned convolution filters to initialize the filter weights of the GDN. See Figure 4 for details.

The **Grasp Quality Network** $Q(\mathbf{g}, \mathbf{o})$ was trained using grasp-centric image patches for the observations, and the translation between the grasp and the object's center (g_x, g_y, g_z). We train the GQN using a binary cross entropy loss.

The **Grasp Displacement Networks** are trained to predict a Gaussian displacement

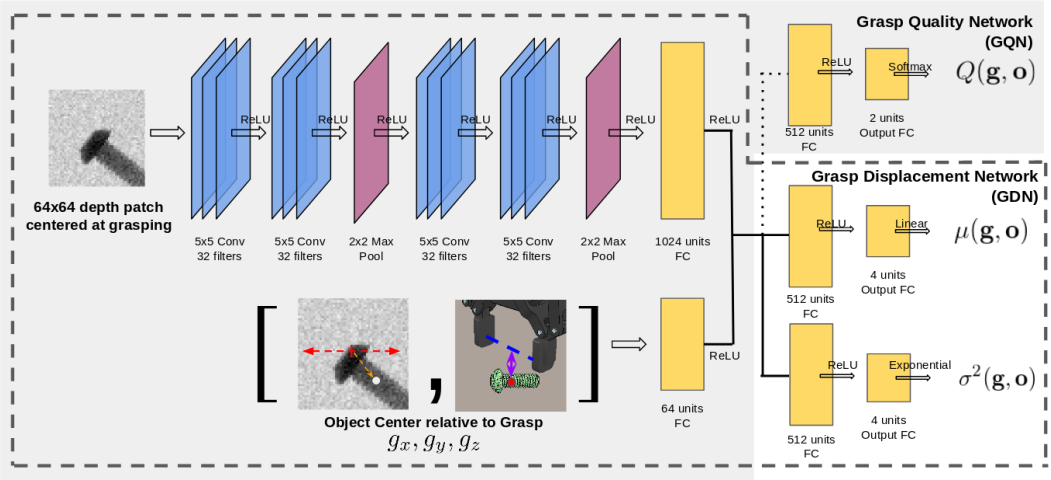


Figure 4: **Grasp Quality Network (GQN) and Grasp Displacement Network (GDN)**. Input to both networks contains a depth image $\mathbf{o}$ cropped at the desired grasp center $\{\mathbf{g_x}, \mathbf{g_y}\}$ aligned to then grasp rotation $\mathbf{g}_\theta$, and the relative translation of the object's geometric center with respect to the grasp center (g_x, g_y, g_z). The GQN predicts grasp quality $Q(\mathbf{g}, \mathbf{o})$, and the GDN jointly predicts the mean $\mu(\mathbf{g}, \mathbf{o})$ and variance $\sigma^2(\mathbf{g}, \mathbf{o})$ of post-grasp displacements. The GQN is trained first, and its learned convolution weights are used to initialize the convolution filters of the GDN. The two networks share the same convolution architecture but not the same weights. We use dropout of 0.5 for the fully connected layers.

distribution with mean $\mu(\mathbf{g}, \mathbf{o})$ and variance $\sigma^2(\mathbf{g}, \mathbf{o})$. For the observations $\mathbf{o}$ we have GDN variants that use the object-centric full images or the cropped grasp-centric image patches. The grasp-centric image variants are initialized using the weights from the GQN, while the object-centric image variants are initialized randomly. We also have GDN variants that predict only the mean $\mu(\mathbf{g}, \mathbf{o})$ or both the mean $\mu(\mathbf{g}, \mathbf{o})$ and the variance $\sigma^2(\mathbf{g}, \mathbf{o})$. These four GDN variants are all trained only on successful $S = 1$ grasps and we do not try to predict how failed grasps will displace the objects.

To train the GDN, we form a loss to maximize the log likelihood on the predicted distribution of post-grasp displacement:

$$P(\Delta\mathbf{g}|\mathbf{g}, \mathbf{o}) = \frac{\exp\left(-\frac{1}{2}(\Delta\mathbf{g} - \mu(\mathbf{g}, \mathbf{o}))^\top \Sigma(\mathbf{g}, \mathbf{o})^{-1}(\Delta\mathbf{g} - \mu(\mathbf{g}, \mathbf{o}))\right)}{\sqrt{(2\pi)^4 det(\Sigma(\mathbf{g}, \mathbf{o}))}} \tag{1}$$

$$\mu^*, \sigma^{2^*} = \arg\max_{\mu, \sigma^2} \ \log(P(\Delta\mathbf{g}|\mathbf{g}, \mathbf{o})) \tag{2}$$

$$= \arg\min_{\mu_j, \sigma_j^2} \left(\sum_{j=1}^{4} \frac{1}{2} \log(\sigma_j^2) + \frac{(\Delta\mathbf{g}_j - \mu_j)^2}{2\sigma_j^2} \right) \tag{3}$$

Where $\Sigma(\mathbf{g}, \mathbf{o}) = diag(\sigma^2(\mathbf{g}, \mathbf{o}))$. Thus the loss function to train the GDN is:

$$L = \sum_{i=1}^{N} \left(\sum_{j=1}^{4} \log(\sigma^2(\mathbf{g}_j^{(i)}, \mathbf{o}_j^{(i)})) + \frac{(\Delta \mathbf{g}_j^{(i)} - \mu(\mathbf{g}_j^{(i)}, \mathbf{o}_j^{(i)}))^2}{\sigma^2(\mathbf{g}_j^{(i)}, \mathbf{o}_j^{(i)})} \right) \tag{4}$$

The **Grasp Planner** needs to select a grasp based on the learned networks. An ideal grasp for industrial applications needs to satisfy two requirements: (1) the grasp should be able to lift an object up and (2) the uncertainty over the object's post-grasp pose should be small in order for a confident in-hand object pose. We fulfill these two requirements by first running the GQN to select the top **3%** of the scored randomly generated grasps, denoted as $\mathbf{G}$. This step makes sure the planned grasp satisfies requirement (1). We then choose the optimal grasp $\mathbf{g}^*$ as the one that has the lowest displacement variance predicted by the GDN: $\mathbf{g}^* = \arg\min_{\mathbf{g} \in \mathbf{G}} \sigma^2(\mathbf{g}, \mathbf{o})$. In other words, among all the top successful grasps $\mathbf{G}$, we select the one with the smallest displacement variance. The expected displacement $\mu(\mathbf{g}^*, \mathbf{o})$ is then used as a correction of the object pose to parameterize downstream tasks such as assembly and palletizing.

2.7 Experiments and Evaluations

We perform experiments in simulation and the real world to evaluate the performance of both the GQN and the GDNs. We also evaluate how selecting low-variance grasps affects performance. The GQN and GDN networks are always trained with simulation data only.

2.7.1 Evaluated Grasp Displacement Models

We compare 5 models for post-grasp displacement estimation:

- **LOWESS** - Locally Weighted Regression
- **OCFI-M** - GDN with Object-Centric Full Image Input that predicts only the mean grasp displacement.
- **OCFI-M+V** - GDN with Object-Centric Full Image Input that predicts both the mean and the variance of grasp displacement.
- **GCIP-M** - GDN with Grasp-Centric Image Patches that predicts only the mean grasp displacement.
- **GCIP-M+V** - GDN with Grasp-Centric Image Patches that predicts both the mean and the variance of grasp displacement.

LOWESS [9] is a non-parametric regression method that is similar to nearest neighbors except the weight of the neighbors is computed from a Gaussian kernel. Because this model doesn't generalize to novel objects, we only use **LOWESS** as a baseline to predict grasp displacements for *known* objects. Given N grasps on the known training object, and a new query grasp $\mathbf{g}_j$, the predicted grasp displacement is computed by:

$$\Delta \hat{\mathbf{g}_j} = \frac{\sum_i^N w(\mathbf{g}_i, \mathbf{g}_j) \Delta \mathbf{g}_i}{\sum_i^N w(\mathbf{g}_i, \mathbf{g}_j)}$$

where $w(\mathbf{g}_i, \mathbf{g}_j) = \mathcal{N}(\mathbf{g}_j | \mathbf{g}_i, \Sigma)$ is the probability density function of the isotropic multivariate Gaussian distribution with mean $\mathbf{g}_i$ and variance Σ evaluated at $\mathbf{g}_j$. We choose the variance terms to be $\Sigma = diag([0.02, 0.02, 0.05, 1.00])$.

The full image GDN variants take as input the uncropped image centered around the object geometric center, instead of the grasp center as is the case with image patches. These models help us understand how important it is for the GDN to focus on local features around the grasp vs. global features that describe overall object geometry. Although the GDN variants without variance prediction cannot be used to select grasps by variance, we evaluate against them to see whether or not training to predict this variance helps improve the prediction accuracy of the mean displacements.

2.7.2 Training GQN and GDN

All the CNN models we used have the same convolutional layers structure, the only differences are the input action sizes and output layer activations. Before training, all action network inputs are normalized to range $[-1, 1]$, and all depth images are reshaped to 64×64. To simulate noise from real depth cameras we add uncorrelated pixel-wise Gaussian noise of zero mean and 3mm standard deviation to the depth image. We preprocess all depth images by subtracting their mean and dividing by their standard deviation.

We trained the GQN with balanced positive and negative data for 100 epochs with the RMSProp optimizer using a learning rate of 10^{-5} and decay of 10^{-6}. The final accuracy of the GQN is 86.7% on the training set and 85.3% on the validation set.

We trained all four variants of GDN in a similar fashion. Root mean square error (RMSE) between predicted mean displacement and actual displacement of each model on the validation set is shown in Figure 5.

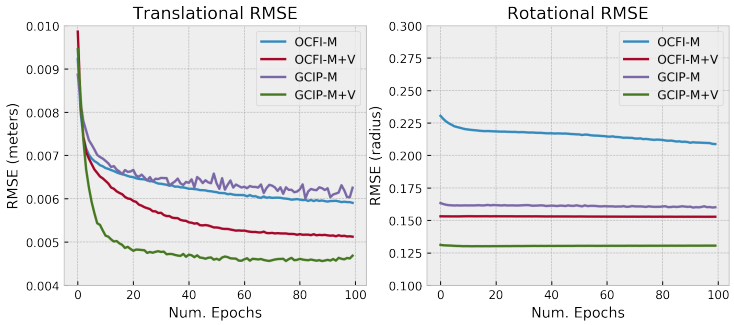


Figure 5: **Validation RMSE of Mean Post-Grasp Object Displacement Predictions for Variants of GDN** We observe that models that incorporate displacement variance prediction consistently outperform ones that do not, and GDN on image patches outperforms operating on the full image.

We observe that including the displacement variance prediction improves the RMSE for the predicted displacement means. This improvement is because, with variance prediction, the network is allowed to increase the variance term on data that have high variance instead of fitting a mean which may incur high loss.

2.7.3 Evaluation in Simulation

To evaluate the performance of GQN for planning robust grasps, we formed a grasp planning policy that uniformly samples 3200 grasps across the object and picks the grasp with the highest predicted quality. We ran this policy on a set of 130 novel objects not found in the training or the validation set, and performed 30 grasp trials for each object. In simulation the trained GQN achieved a grasp success rate of **94.9**%.

To evaluate the performance of the GDNs, we ran two sets of experiments using different grasp selection policies: (1) select grasps with high predicted qualities according to the GQN and (2) select grasps that have both high quality and low variance. The latter is only applicable for **LOWESS**, **OCFI-M+V**, and **GCIP-M+V** which predict the variances.

All of the models in both experiments were evaluated on a set of 85 objects, of which 50 are from the training dataset, and 35 are from the validation dataset. We perform 35 grasping trials for each object, and only successful grasps were used to evaluate the performance of displacement prediction models.

Results are shown in Fig. 6. For the high-quality grasps, experiment (1), the average translational error and rotational error are 0.43cm and 8.29deg with **GCIP-M+V**. For the high-quality low-variance grasps, experiment (2), the errors are further reduced to 0.24cm and 7.01deg respectively. **GCIP-M+V** has the best performance in both experiments. We also observe that choosing high-quality, low-variance grasps generally improves the post-grasp displacement prediction accuracy.

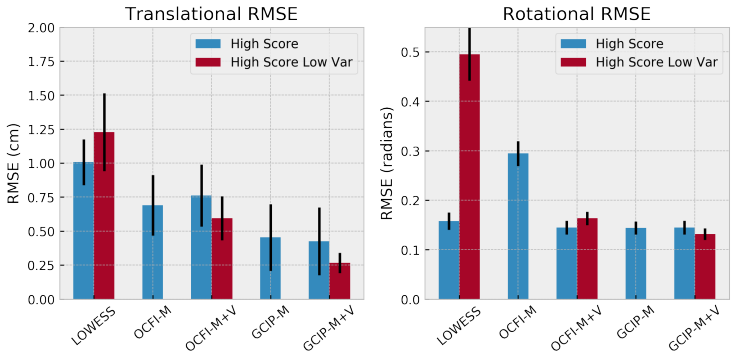


Figure 6: **Translational and Rotational RMSE of Grasp Displacement Predictions in Simulation.** Blue bars show results of GDN estimated displacement for the grasps with highest qualities predicted by the GQN. Red bars show results of choosing grasps that have both high quality and low variance as predicted by a GDN. Grasp displacement models that do not have variance output are not reported for the second set of experiments.

We observe that in both experiments the proposed **GCIP-M+V** GDN has a good performance in terms of translational displacement estimation, but it does not reduce the error for rotation in comparison to other baselines. This might be because predicting translational displacement is easier than rotational displacement, as the latter needs more information regarding the object's overall geometry.

2.7.4 Evaluation with Real World Robot

We use a 7-DoF Franka Emika Panda robot arm for grasping and a Kinect v2 time-of-flight sensor for depth sensing. For our grasping experiments, we 3D printed 7 novel objects from our dataset of industrial parts that were not used during training. At the start of each grasp experiment trial, we place one object in a 24cm by 24cm square region in a bin in front of the robot and directly below the depth camera. Then, the human operator puts a cardboard box over the object before shaking the box for a few seconds. This helps to reduce human bias and randomize the location and orientation of the object.

During each trial, we use the GQN trained with simulation data to predict the qualities of randomly sampled grasps from the depth image. A grasp is considered a success if the object is still grasped by the robot after the lift. If a grasp is successful, the robot proceeds to lowering the end-effector back to the grasp pose before releasing the grippers.

By placing the object at the same gripper pose as the grasping pose, we can compute the relative pre-grasp and post-grasp translation and rotations to estimate object displacement. One way to do this is via object pose registration with their known 3D CAD models, but this approach is not robust due to rotational symmetries and the low-resolution of the depth sensor. Instead we opted for a marker-based approach by making the assumption that object displacements during real robot experiments only occur in a plane. This is done by placing two small, square pieces of white masking tape on top of the object such that the line that connects them crosses the object's geometric center. Their relative translation and rotation after grasping can be robustly determined from registered color images. See Figure 17 for robot setup and an illustration of our objects with these markers.

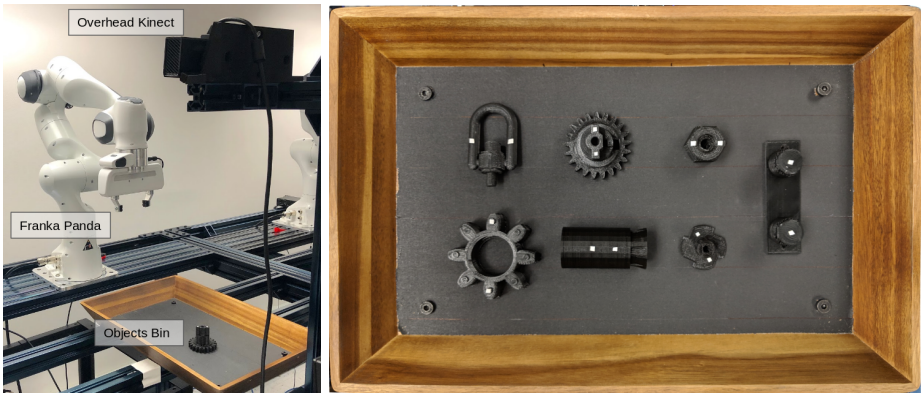


Figure 7: **Real World Robot Setup.** We use a 7-DoF Franka Panda robot arm and a Kinect v2 RGB-D camera (left) and 3D printed test objects used for real world experiments (right). Two white markers are placed on each object to estimate the post-grasp object displacements.

2.7.5 Real World Experiment Results

We carried out the same two displacement estimation experiments with real world robot: (1) evaluate on high quality grasps and (2) evaluate on high quality and low-variance grasps. Because we do not have a set of successful training grasps for these objects, **LOWESS** could

not be applied. Results are shown in Fig.8. In the first experiments, by choosing highest GQN scored grasp in each trial, the robot achieved an **86.3%** lifting success rate.

The **GCIP-M+V** GDN has the best displacement estimation in both experiments. In the first experiment it achieved a translational RMSE of **0.72cm** and rotational RMSE of **3.79deg**. These errors are further reduced to **0.68cm** and **3.42deg** respectively by choosing high quality grasps with low predicted displacement variance. Although the mean RMSE values are similar across the two experiments, the error bars are greatly decreased when using the high-score low-variance grasps, which indicates that these models are more consistent and robust across different objects.

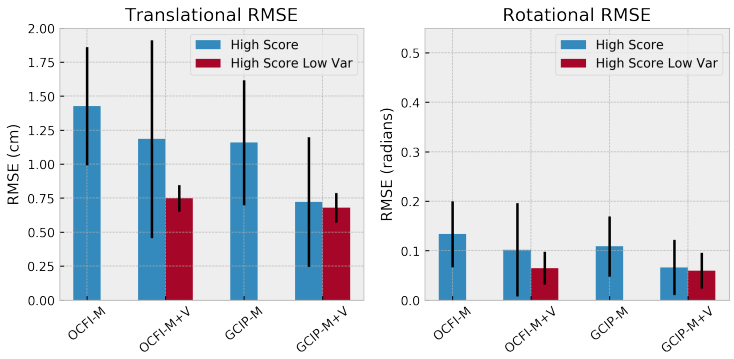


Figure 8: **Real World Translational and Rotational RMSE of Grasp Displacement Predictions.**

In Figure 8 we show an instance of the palletizing application, wherein we use **GCIP-M+V** to estimate the post-grasp displacement and compensate for the placing action accordingly.

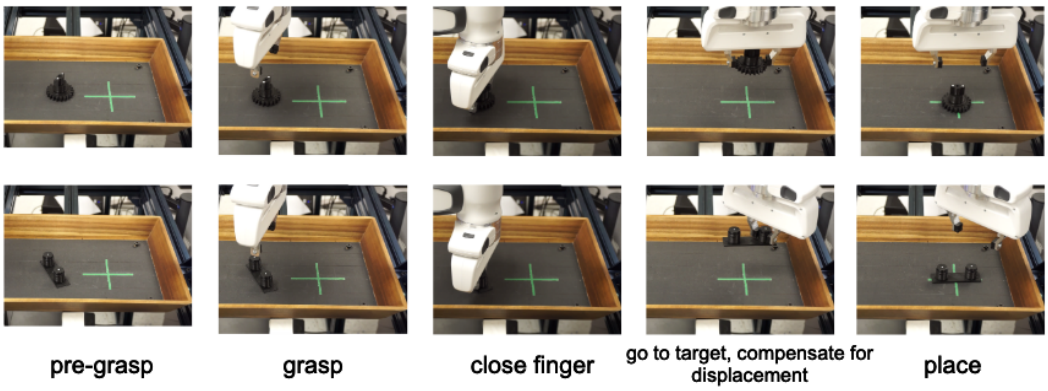


Figure 9: **Example Palletizing Application.** We use the post-grasp displacement prediction from our **GCIP-M+V** model to accurately place test objects at a target pose. The target location is marked by the center of the green cross and the target orientation is the horizontal axis of the green cross. By compensating the placing target pose with the estimated post-grasp displacement, the objects can be placed more accurately.

2.8 Conclusion

In this work we propose a method to plan robust and precise grasps by training two networks - one to predict grasp robustness and the other to predict the distribution of post-grasp object displacements. We trained the networks in simulation and deployed them in the real world without further fine-tuning. Experiments in simulation and the real world show our method can effectively predict post-grasp displacements, and choosing grasps that have low predicted variance result in lower displacement prediction errors.

3 Learning to plan precise and task-oriented grasps for robotic assembly

3.1 Introduction

In this work, we address the challenge of predicting task-oriented grasps for two example tasks: inserting gears onto pegs and aligning brackets into corners. We form this problem as a curriculum learning problem and propose a method that decomposes it into three sub-problems - choosing robust grasps, estimating grasp precision, and choosing a suitable grasp for the task. For each of the sub-problems we propose one neural network. They are then cascaded together to predict the best grasp. This grasp can then be used to achieve better precision in down-stream tasks such as gear insertion or bracket insertion. We train the policy with simulated grasps and we show the efficiency of our proposed approach both in simulation and on a real robot.

3.2 Related Works

Researchers have made many contributions on planning robust robotic grasps. Many works have focused on using analytical approaches according to physical models, such as the ability to resist external wrenches [15], to constrain the object's movement [33, 12], or to combine multiple analytical metrics [35, 27]. However, they usually assume the object is static and the pose is known. Recent works have used data-driven grasp synthesis to predict grasps directly from sensory inputs. While early works used human-labeled grasps [24, 26], an increasing number of works have started to collect data in a self-supervised manner by attempting thousands of grasps physically in real world [32, 28, 22] or in simulation [31, 14, 4, 13]. However, most of those grasp planners only focus on efficiently picking objects up. They do not take into account of down-stream tasks that require objects to be grasped within some precision or in a specific manner.

Due to imperfect approximated dynamics as well as noise in sensing and actuation, a grasp that a robot executes often results in a state that is not intended. Recent works have also studied how to avoid or decrease such error. Chen et al explored a closed-loop active planning framework to improve precision for objects with known model [6]. They further generalized it to unseen objects by proposing a probabilistic signed distance function representation of object surfaces from depth images and analytical metrics [7]. Dogar et al used pushing to funnel the clutter of objects in front of grasps to reduce object pose uncertainty [12]. Jentoft et al analyzed the source of uncertainties in grasping tasks and predicted grasps that were robust to local variations based on the range of variations a grasp can tolerate [23]. Our previous work reasoned about grasping precision to improve accuracy of unconstrained tasks by approximating post-grasp object displacement as a probabilistic distribution [42]. Gupta et al explicitly modeled the latent noise in perception and actuation [18]. Those works constrain the errors introduced by task-agnostic grasps.

Recent works have extended grasp planning for task-oriented grasping, where the planner also optimizes for the success of downstream tasks. Dang et al manually labeled grasps with semantics and learned a *semantic affordance map* to relate object geometry to predefined grasps that are appropriate to different tasks [10]. Gao and Tedrake transferred the semantic grasping problem into a key point detection and object reconstruction problem by learning an object representation based on semantic key points [16]. Fang et al proposed a method that jointly optimized for grasping and manipulation skills with simulated grasps

and generated objects [14]. There have also been works that use reinforcement learning (RL) methods with object geometry features extracted from CAD [38], auxiliary sensing [29], or classical controllers and demonstrations [37]. Those works take the constraints introduced by down-stream tasks into consideration.

In this work, we address the challenges of precision assembly where a grasp should be accurate and well-suited to the task. Our work stands in contrast with previous works by decomposing this problem into three sub problems: planning robust grasps, estimating post-grasp displacement, and choosing suitable grasps for tasks. We train one CNN for each, and cascade them to find the most appropriate grasp. We form it as a curriculum learning [1] problem to improve training performance. In this manner, the grasp planner can choose grasps that are robust, precise, and appropriate to the task.

3.3 Problem Statement

Our proposed approach enables robots to perform industrial insertion tasks with top-down, parallel-jaw grasps of singulated objects lying on a flat surface. Two categories of insertion tasks are discussed: inserting gears onto pegs and aligning brackets into corners. Our method generalizes over novel objects unseen during training.

Let the initial pose of an object on the work surface be $\mathbf{p}$. We assume that during manipulation, an object can only undergo translational movements $(\Delta x, \Delta y, \Delta z)$ and planar rotation $\Delta\theta$. We define the post-grasp object displacement as $\Delta\mathbf{p} = [\Delta x, \Delta y, \Delta z, \Delta\theta]^T$. We assume the robot will not be able to use additional sensors to perform in-hand localization. The grasp planner needs to predict the displacement $\Delta\mathbf{p}$ based on the object pose $\mathbf{p}$, the candidate grasp $\mathbf{g}$, and the observation $\mathbf{o}$. The observation $\mathbf{o}$ is a depth image of the object captured with a top-down stereo camera.

A grasp $\mathbf{g}$ also has 4 degrees of freedom $\mathbf{g} = [g_x, g_y, g_z, g_\theta]^T$. where the position $(g_x, g_y, g_z) \in \mathbb{R}^3$ is the location of the center of gripper relative to the object's geometric center $\mathbf{p}$. The orientation parameter $g_\theta \in [-\frac{\pi}{2}, \frac{\pi}{2})$ denotes the planar rotation of the gripper.

In this work we focus on two commonly seen tasks in manufacturing sectors: inserting objects that have shaft holes (such as gears) onto pegs and inserting brackets into corners. For the gear insertion problem, an optimal grasp should be robust in picking up, and also be aware of the object pose after grasping. This task requires the planning system to learn a robust grasp planner, and accurately estimate the post-grasp displacement of the object. For bracket insertion tasks, besides robustness and precision, an optimal grasp planner should also take the task parameters into consideration. As shown in Fig.1, a grasp that is robust in picking up may fail the insertion task because of the post-grasp object displacement (left), or the gripper interfering between the object and the corner (middle), whereas an optimal grasp should take into consideration the post-grasp displacement and avoid the insertion frame (right).

3.4 Method

3.4.1 Overview of Approach

The proposed method divides the grasp planning problem into three sub-problems: (1) predicting the task-agnostic grasp quality in terms of lifting up the object, (2) predicting the post-grasp object displacement, and (3) predicting the insertion quality of a grasp based

on the task parameters. One convolutional neural network was trained for each of these tasks. They are cascaded to gradually funnel down candidate grasps to the best ones.

Grasp Quality Prediction: Defined similarly to our previous work in [42], the grasp quality Q_G of a grasp $\mathbf{g}$ with observation $\mathbf{o}$ is defined as the probability of a successful lift S using the grasp, $Q_G(\mathbf{g}, \mathbf{o}) = P(S = 1|\mathbf{g}, \mathbf{o})$. We learn this mapping $Q_G(\mathbf{g}, \mathbf{o})$ using a CNN that we refer to as the Grasp Quality Network (GQN).

Grasp Displacement Prediction: Let $\Delta\mathbf{p} = [\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{z}, \Delta\theta]^T$ denote the post-grasp displacement of an object. We learn a CNN to predict $\Delta\mathbf{p}(\mathbf{g}, \mathbf{o})$ from grasp $\mathbf{g}$ and observation $\mathbf{o}$, which we refer to as the Grasp Displacement Network (GDN).

Task Quality Prediction: We define the insertion quality Q_I of a grasp $\mathbf{g}$ with observation $\mathbf{o}$ and GDN estimated displacement $\Delta\mathbf{p}$ as the probability of a successful insertion with insertion error ϵ smaller than a threshold σ, under task parameter $\mathbf{k}$, $Q_I(\mathbf{g}, \mathbf{o}, \Delta\mathbf{p}, \sigma, \mathbf{k}) = P(\epsilon < \sigma|\mathbf{g}, \mathbf{o}, \Delta\mathbf{p}, \mathbf{k}, \sigma)$. Here the task paramter $\mathbf{k}$ encodes the insertion type (gear or bracket) and the transformation from the insertion frame to the object geometric center T_o^i. For bracket insertion, ϵ is the Euclidean distance between the insertion frame and the corner, while for gear insertion it is the negative peg ID that could be successfully inserted onto. Pegs with highers IDs are thicker than the ones with lower IDs. We define six pegs with diameters $\{3, 4, 4.5, 5, 5.5, 6\}$mm as Peg #1 to Peg #6. If a grasp doesn't successfully insert the gear onto any peg, we have $\epsilon = 0$. The mapping of $Q_I(\mathbf{g}, \mathbf{o}, \Delta\mathbf{p}, \sigma, \mathbf{k})$ is learned by a CNN that we refer to as the Insertion Quality Network (IQN). Curriculum learning [1] has been proven to be able to outperform learning from randomly presented training data by gradually increasing the complexity of presented data during training. We form IQN as a curriculum learning problem by gradually decreasing the insertion error tolerance σ.

Given the three learned networks, we can form a grasp planner that first chooses grasps with high probability of successfully picking up the object, then estimates the post-grasp object displacement for each grasp, and finally filters them with the probability of completing the insertion with low errors. After grasping an object and having its in-hand object pose by adding the post-grasp displacement to its initial pose, we use a fixed sequence of motion calculated based on this in-hand object pose to do the insertion. We remove the prediction of object displacement variance and variance-based grasps ranking as we did in [42], because of the introduction of IQN. IQN should implicitly learns whether an estimated displacement is reliable and we use IQN to rank grasps.

We do not assume knowledge of other object geometrical information, thus this method generalizes to unseen objects.

3.4.2 Self-Supervised Simulated Data Collection

To generate grasps for training the networks, we first procedurally generate $5,000$ bracket models and $5,000$ gear models. In simulation, we collect a total of $495,000$ data points for training the task-agnostic grasp quality network (GQN) and the displacement estimation network (GDN), as well as $170,000$ data points for training the insertion quality network (IQN).

Procedural Generation of Brackets and Gears We train our models in simulation with a large number of objects so as to generalize to unseen objects. However, the existing 3D model datasets do not contain enough parts suitable for the insertion tasks while exhibiting rich variations in terms of their geometric and physical properties. We generate a large set of diverse and realistic objects that are close to the actual parts with primitive shapes for insertion tasks. Some examples are shown in Fig.10.

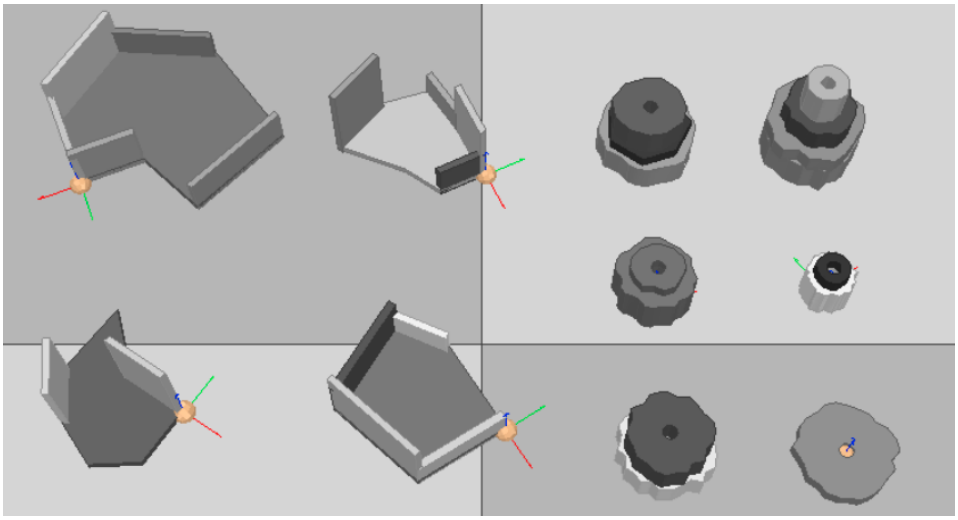


Figure 10: **Procedurally generated brackets and gears** Brackets with polygon base and walls (left), and multi-layer gears with 6.3mm shaft holes (right). The coordinate frames illustrated on the brackets denote the insertion frames. The center of an insertion frame should be inserted along the red axis to the corner center.

- *Bracket Generation* The generated brackets are constructed with two components - a flat polygon base plate with 4 to 8 vertices and 1 to 5 walls with random heights. The vertices of the polygon base plate are sampled from a circle of radius from 30mm to 70mm with some irregularity. The heights of walls are randomly picked from 10mm to 50mm. The base plate can be convex or concave, and we make sure that there is always at least one right angle corner, which we mark as the insertion frame.

- *Gear Generation* A generated gear contains up to 5 layers, and the radius of one layer is always smaller than the layer under it. All the gears have a shaft hole of 6.3mm such that all of them are able to be inserted onto the largest peg (6mm) that we test with. Each layer is approximated with a polygon of 4 to 50 vertices. More vertices make a layer smoother and rounder.

Simulated Grasps Generation We gathered grasps using the robotic simulation framework V-REP [34] and PyRep [21] with Bullet 2.78 [3] as the physics engine. We assume each object has uniform density.

- *Data Generation for GQN and GDN* A depth image is captured at the start of each grasp attempt. The robot executes a grasp $\mathbf{g}$, where the grasp center (g_x, g_y, g_z) is uniformly sampled from the bounding box of the object, and the grasp orientation g_θ is uniformly sampled from $[-\frac{\pi}{2}, \frac{\pi}{2})$. The bounding box's coordinate frame is always parallel to the camera coordinate frame. The bounding box is calculated as the minimum area that encloses the entire object in the depth image. In contrast to previous works that sample antipodal grasps [30, 31], we use uniformly sampled grasps as using antipodal grasps may introduce a bias in the dataset when estimating post-grasp

[3] https://pybullet.org/

displacements. For each grasp attempt $\mathbf{g}$, we record the lift success S of the grasp, the overhead depth image $\mathbf{o}$, and the post-grasp object displacement in world frame $\Delta\mathbf{p}$.

- *Data Generation for IQN* After GQN and GDN are trained, we collected another $170,000$ data for training the insertion quality network, IQN. The robot first samples and executes a random grasp $\mathbf{g}$ inside the bounding box area. Besides capturing a depth image, we also calculate a binary contact area mask and a binary no-go area mask. The contact area mask is the object area that is $2cm$ away from the insertion frame, while the no-go area is the object area within $2cm$ to the insertion frame, as illustrated in Fig.11. We use the two masks to encode task parameters $\mathbf{k}$.

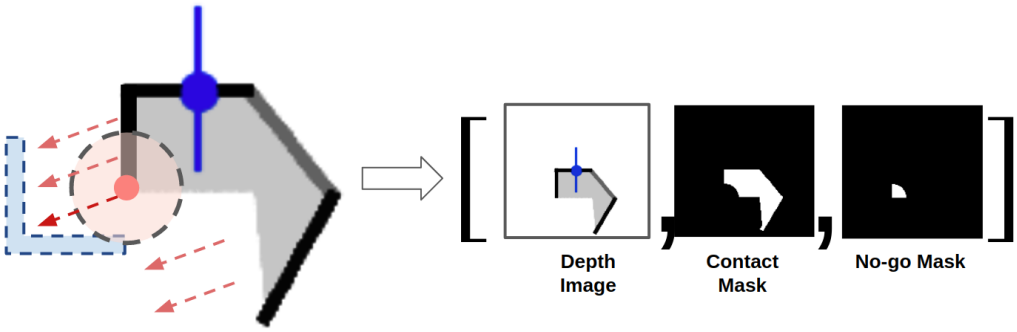


Figure 11: **Contact and no-go masks generation** Both masks are binary, indicating the object areas $2cm$ away from or within $2cm$ to the insertion frame. We choose $2cm$ because that's half of the gripper's width. Those two masks indicate whether the gripper will interfere with insertion if grasped from such area. We use those two binary masks to incorporate task parameters.

If a grasp successfully picks the object up, we use the GDN estimated object displacement to correct the current object in-hand pose. The robot then uses this corrected pose to insert the object to the target position. For brackets, we record an insertion error calculated as the Euclidean distance between the target corner and the object insertion frame in the world frame. For gears, the robot tries to insert it onto 6 pegs with diameter $\{3, 4, 4.5, 5, 5.5, 6\}$mm starting from the smallest one until completion or failure. The negative id for the best peg that could be successfully inserted is recorded as ϵ. For example, if the best peg that can be successfully inserted is the 3^{rd} peg (4.5mm), we will have $\epsilon = -3$.

If a grasp doesn't lift the object up, this trial will be aborted and it won't be used as training data to learn IQN.

3.4.3 Network Training

We train three types of CNNs - GQN, GDN, and IQN. While they all share a similar convolution architecture, they do not share weights. Rather, the GQN is trained first, and we use its learned convolution filters to initialize the filter weights of the GDN. IQN has a different input dimension and is trained separately after GDN and GQN are trained. See Fig.12 for details on network architecture.

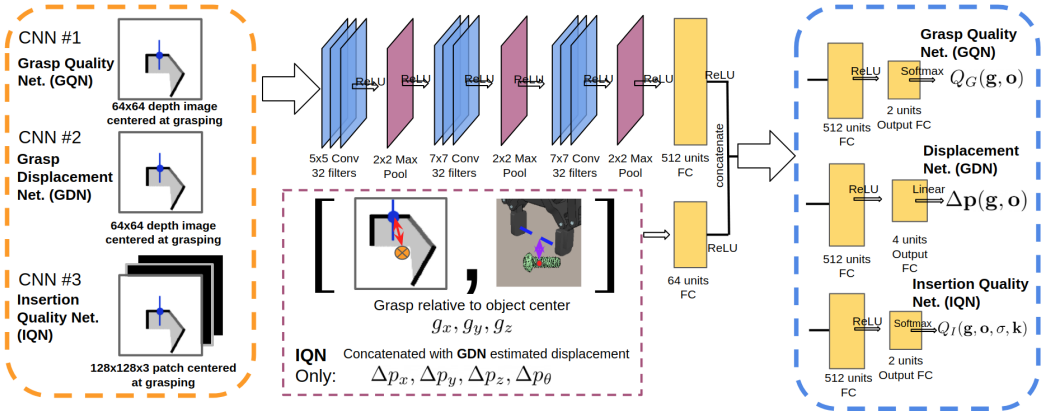


Figure 12: **Network structure** We trained three separate networks: the Grasp Quality Network (**GQN**), the Grasp Displacement Network (**GDN**), and the Insertion Quality Network (**IQN**). All the three networks have the same convolution layer architecture, but have different inputs and outputs. All three networks have the same grasp relative to object center input $\mathbf{g}$. Note that although they share the same structure, they don't share weights.

The **Grasp Quality Network**, $Q_G(\mathbf{g}, \mathbf{o})$, and the **Grasp Displacement Network**, $\Delta\mathbf{p}(\mathbf{g}, \mathbf{o})$ were trained using grasp-centric image patches, and the translation (g_x, g_y, g_z) between the grasp and the object's geometric center. We trained the GQN using a binary cross entropy loss and the GDN using a square error loss.

The **Insertion Quality Network** $Q_I(\mathbf{g}, \mathbf{o}, \sigma, \mathbf{k})$ was trained using grasp-centric image patches stacked with contact area masks and no-go area masks, as well as a 7 DOF vector composed of the grasp to object center translation (g_x, g_y, g_z) and the GDN estimated displacement $(\Delta p_x, \Delta p_y, \Delta p_z, \Delta p_\theta)$. IQN was trained using a binary cross entropy loss.

The training process of IQN was formed as a curriculum learning problem. Based on the distribution of collected data, we choose the insertion error thresholds for brackets to be $\sigma_\mathbf{b} = \{16.65, 8.32, 5.55, 4.16\}$mm and form it as a four-stage curriculum. Thresholds for pegs insertion are chosen as the negative IDs of the six pegs from the smallest one to the largest one, $\sigma_\mathbf{g} = \{-1, -2, -3, -4, -5, -6\}$, which is formed as a six-stage curriculum.

3.5 Experiments and Evaluations

3.5.1 Evaluated Models

We compare 3 models to evaluate the insertion accuracy:

- **GQN-only** - Select grasps only based on predicted grasp quality. Assume the grasped object has zero post-grasp displacement.
- **GQN+GDN** - Select grasps only based on predicted grasp quality. Adapt the insertion motion primitives based on post-grasp object displacement.
- **GQN+GDN+IQN** - Select grasps that have both high predicted grasp quality and insertion quality. Adapt the insertion motion primitives based on post-grasp object displacement.

3.5.2 Evaluations in Simulation

With the same object generation procedure, we generate another 500 novel brackets and 500 novel gears that are isolated from the training data to perform evaluations on. We perform 2000 grasp and insertion simulated trials for each of the model.

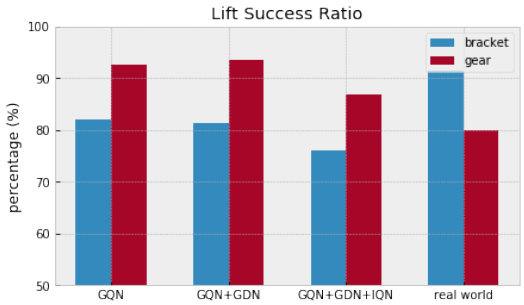


Figure 13: **Lift success ratio** The proposed method **GQN+GDN+IQN** has a slight decrease in terms of lift success. The lift success rate for real-world experiments are 91.4% for brackets and 80% for gears.

Lift success rate is shown in Fig.13. **GQN+GDN+IQN** has a slightly decreased lift success rate compared to **GQN** and **GQN+GDN**. We believe this is because the grasps that are suitable to tasks are sometimes less stable.

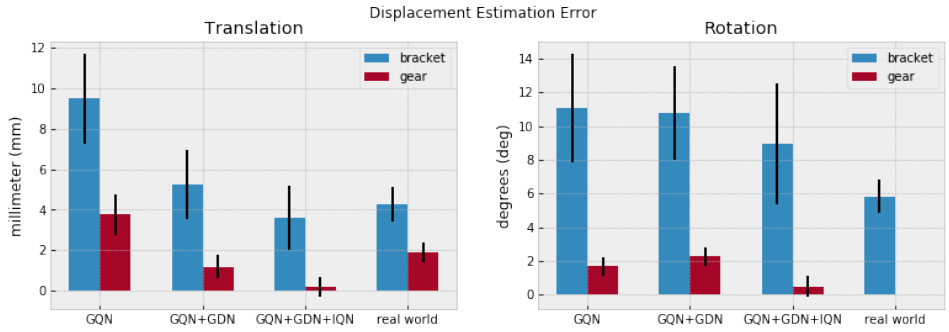


Figure 14: **RMSE of estimated object displacement** The proposed method **GQN+GDN+IQN** achieves the lowest displacement estimation error for both brackets and gears. For real-world experiments, the translational RMSE's are 4.28mm for brackets and 1.89mm for gears; the rotational RMSE's are 5.83deg for brackets. Rotation for gears are not available in real-world tests because of symmetricity.

Displacement Estimation RMSE is shown in Fig.14. **GQN+GDN+IQN** has a higher accuracy than **GQN** and **GQN+GDN**. It achieves 3.55mm and 8.92deg precision for brackets, 0.18mm and 0.46deg precision for gears. All tests with gears have significant lower RMSE than tests with brackets, because most gears are symmetrical, and a good grasp is most likely to be at its center. Such grasps does not displace the gear by much.

Learning insertion as a curriculum As shown in Fig.15, in both insertion tasks,

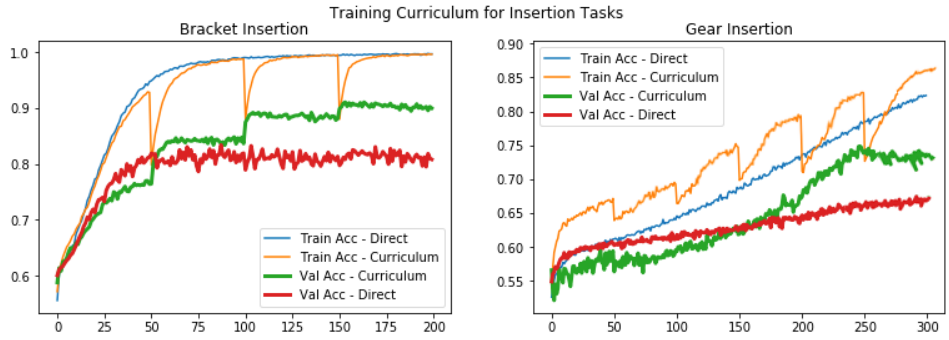


Figure 15: **Training curve for insertion** The training process for IQN is formed as a 4-step curriculum for brackets and a 6-step curriculum for gears. In both cases training with a curriculum achieves better validation accuracy than training directly does. Training as curriculum and training directly are validated with the same validation dataset.

training IQN as a curriculum results in better validation accuracy and faster convergence than training it directly.

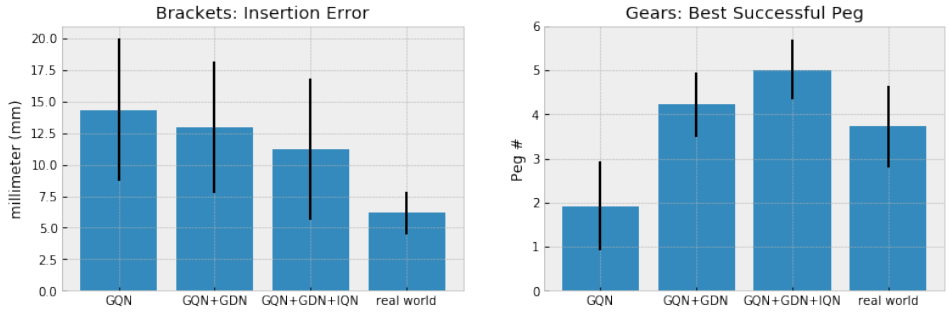


Figure 16: **Insertion performance** Mean and std for bracket insertion error (left) and for the best peg that a gear can be successfully inserted onto (right). For real-world experiments, the average bracket insertion error is 6.17mm, the average best successful peg for gear insertion is 3.71, or 1.44mm insertion precision (calculated as shaft hole diameter subtracted by gear diameter).

Insertion Performance is shown in Fig.16. **GQN+GDN+IQN** has the best performance, which is 11.21mm for brackets and 5.02 for gears. We noticed that the simulator sometimes became unstable after making contact during bracket insertion, which is probably the cause for larger errors in bracket insertion.

3.5.3 Evaluations with Real Robot

For real-world evaluations we use a 7-DOF Epson C4L robot arm, with an over-head ENSENSO N35 stereo camera for depth sensing. We use 6 brackets (metallic and plastic) and 4 gears (resin and plastic) for evaluation. We perform 5 trials with each object. We use Epson Object Detection and Pose Estimation (ODPE) software to get the ground truth

object pose. ODPE has a rated accuracy of 96% that an object can be successfully detected and an object pose can be estimated within 5mm/5deg error. The setup and the parts we used for experiments are shown in Fig.17.

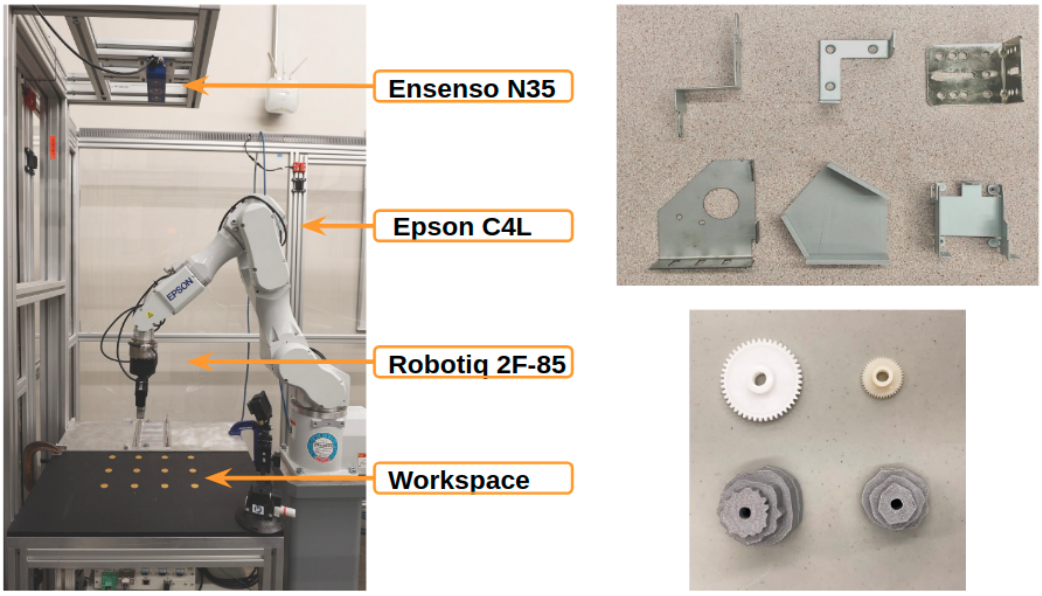


Figure 17: **Real-world experiment setup** We use a 7-DOF Epson C4L robot arm, a Robotiq 2F-85 parallel jaw gripper, a top-mounted Ensenso N35 depth camera (left) as well as 6 brackets and 4 gears (right) for real-world experiments.

The workspace is divided into three areas: a bin area, a palletize area, and an insert area, as shown in Fig.18. The insert area is further divided into a bracket insert area and a gear insert area. There are four metallic corners built in the bracket insert area. During experiments the robot picks one corner based on object's orientation. There are six pegs built in the gear insert area. They have the same $\{3, 4, 4.5, 5, 5.5, 6\}$mm diameters as in simulation.

The evaluation workflow (see Fig.19) is: (1) Human places one object in the bin area in a random pose. (2) The planner runs **GQN+GDN+IQN** networks to predict a best grasp and its corresponding displacement. The robot executes this grasp and compensates the displacement. (3) The robot places the object in the palletize area for pose estimation. The result is compared against the estimated displacement later during analysis. The robot then re-grasps the object using the same parameter as in placing. (4) Brackets: the robot goes to a corner and prepare to insert. Gears: the robot goes to the smallest peg and tries to insert. (5) Bracket: insert it to the corner and record the insertion error. Gears: try all the pegs until completion or failure and record the best successful peg. After insertion, we run pose estimation again. Finally the robot moves the object back to the bin area.

Note that the palletizing step is needed for estimating the in-hand object pose. We assume the pick and place in the palletize area do not change the in-hand object pose.

To reduce the gap between simulation and reality we use the simulator to render depth image with ODPE estimated object pose , instead of directly using real-world captured

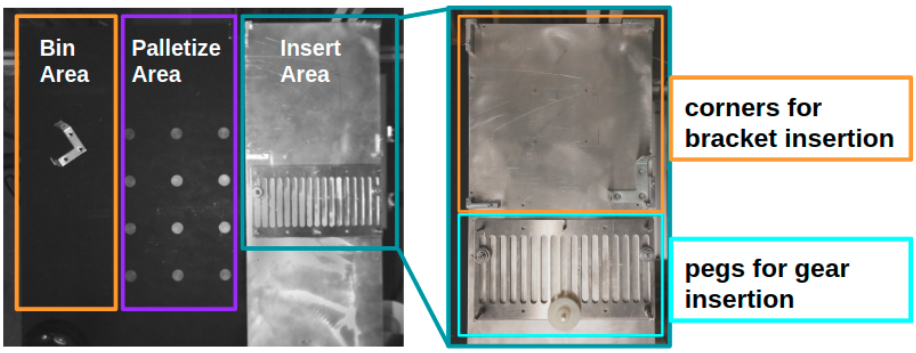


Figure 18: **Workspace configuration** The workspace is divided into a bin area, a palletize area and an insert area. The upper part of the insert area has 4 corners for bracket insertion; the bottom part has 6 pegs of diameters $\{3, 4, 4.5, 5, 5.5, 6\}$mm for learning gear insertion curriculum.

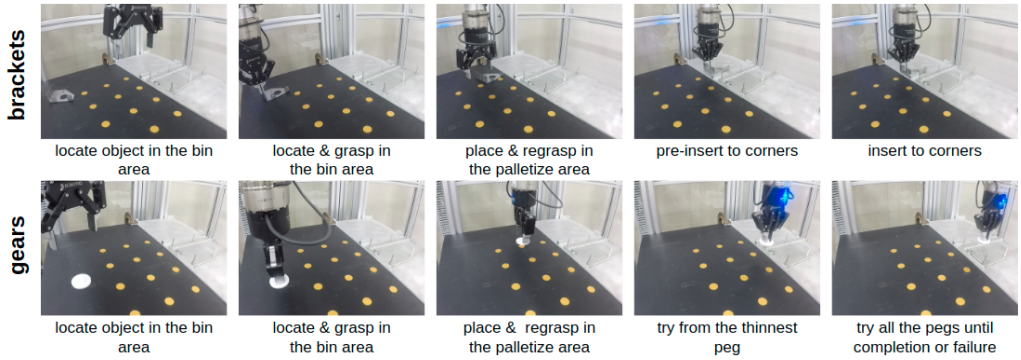


Figure 19: **Real-world experiment pipeline** Brackets insertion (up) and gears insertion (down).

depth image. Appearance variations due to materials and lighting are thus removed. The pipeline is illustrated in Fig.20.

In Fig.21 we visualize some grasps during real world evaluations of bracket insertion. A good grasp should grasp away from the insertion frame, and compensate for the post-grasp object displacement. In most of the time, our model could successfully achieve both. Without GDN, the grasp in (2) will not be correctly compensated. Without IQN, a planner may choose a grasp that is robust in picking but too close to the insertion frame, resulting in the gripper interfering between the object and the corner. However, as in (5) and (6), because of the discrepancy between simulated and real-world dynamics, our model fails to predict reliable grasps in some cases. Additional fine tuning of grasps using real world data from specific objects could potentially alleviate some of these issues.

In the real world evaluations, our model achieved a lift success rate of 91.4% for brackets and 80% for gears. The displacement estimation RMSE is $4.28mm/5.83deg$ for brackets and $1.89mm$ for gears. Rotational error is not accessible for gears because of symmetricity. The

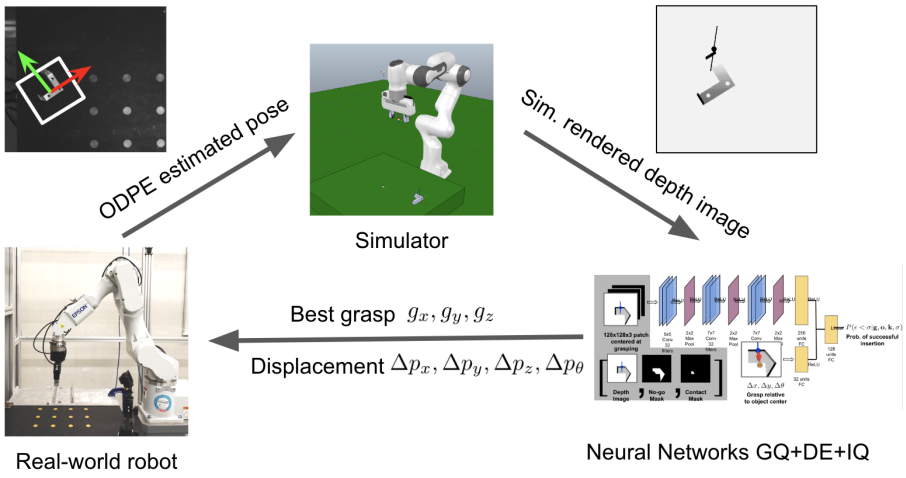


Figure 20: **Planning pipeline in real-world experiments** ODPE first estimates the pose of the object in the bin area, then it sends the pose to the simulator. Simulator sends the rendered depth image based on the estimated pose to the networks. The networks predict the best grasp g_x, g_y, g_z and its corresponding displacement $\Delta p_x, \Delta p_y, \Delta p_z, \Delta p_\theta$, then sends the policy back to the robot.

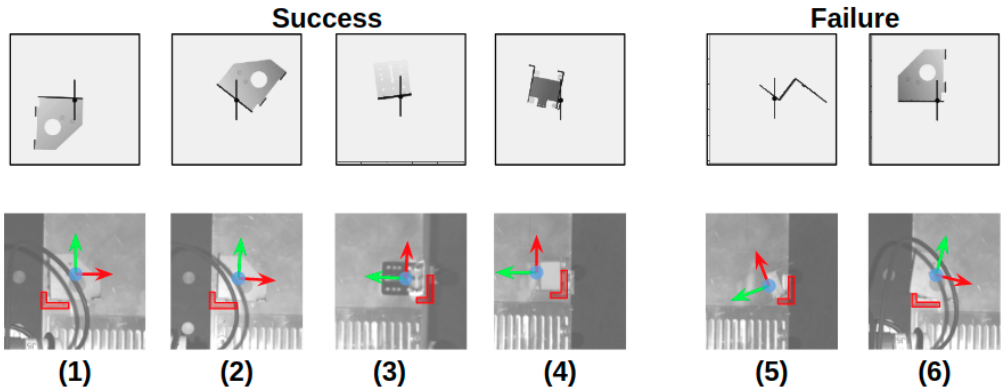


Figure 21: **Visualization of example grasps** The top row shows depth images that are aligned with candidate grasps. The bottom row shows the results of insertion. The L insertion bracket is marked with red outlines for better visibility. Successful grasps should avoid the insertion frame and take into account of post-grasp object displacement (1-4), while (5) fails to estimate the correct post-grasp object displacement and (6) grasps too close to the insertion frame.

average task error for bracket insertion is 4.28mm. The average best peg that the gears can be successfully inserted onto is 3.71, or 1.44mm insertion precision.

3.6 Conclusion

We proposed a method to plan robust and precise task-oriented grasps by dividing the process into three sub-problems: grasp robustly, grasp precisely, and find grasps that are suitable to the task. We solve them by training one neural network for each, and chain them together to gradually funnel down candidate grasps. Estimation of task robustness of a grasp is formed as a curriculum learning problem. We train the policy in simulation and transfer it to real robots. Experimental results show that the proposed method efficiently improves the insertion performance, reducing the insertion error by 21% for bracket insertion and 37.5% for gear insertion, compared with task-agnostic grasps without precision estimation. We also showed that the learned policy successfully transferred to real robots.

In the future, our goal is to further improve and generalize our model by (1) introducing more realistic objects with diverse object properties such as mass density and friction coefficient, and (2) introducing other task constraints as well as experimenting with other tasks such as tool manipulation.

References

[1] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[2] Jeannette Bohg, Antonio Morales, Tamim Asfour, and Danica Kragic. Data-driven grasp synthesis—a survey. *IEEE Transactions on Robotics*, 30(2):289–309, 2014.

[3] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[4] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, Yunfei Bai, Matthew Kelcey, Mrinal Kalakrishnan, Laura Downs, Julian Ibarz, Peter Pastor, Kurt Konolige, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4243–4250. IEEE, 2018.

[5] Yevgen Chebotar, Oliver Kroemer, and Jan Peters. Learning robot tactile sensing for object manipulation. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3368–3375. IEEE, 2014.

[6] Dong Chen and Georg von Wichert. An uncertainty-aware precision grasping process for objects with unknown dimensions. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4312–4317. IEEE, 2015.

[7] Dong Chen, Vincent Dietrich, Ziyuan Liu, and Georg Von Wichert. A probabilistic framework for uncertainty-aware high-accuracy precision grasping of unknown objects. *Journal of Intelligent & Robotic Systems*, 90(1-2):19–43, 2018.

[8] Changhyun Choi, Joseph Del Preto, and Daniela Rus. Using vision for pre-and post-grasping object localization for soft hands. In *International Symposium on Experimental Robotics*, pages 601–612. Springer, 2016.

[9] William S Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836, 1979.

[10] Hao Dang and Peter K Allen. Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1311–1317. IEEE, 2012.

[11] Renaud Detry, Jeremie Papon, and Larry Matthies. Task-oriented grasping with semantic and geometric scene understanding. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3266–3273. IEEE, 2017.

[12] Mehmet Dogar and Siddhartha Srinivasa. A framework for push-grasping in clutter. *Robotics: Science and systems VII*, 1, 2011.

[13] Kuan Fang, Yunfei Bai, Stefan Hinterstoisser, Silvio Savarese, and Mrinal Kalakrishnan. Multi-task domain adaptation for deep learning of instance grasping from simulation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3516–3523. IEEE, 2018.

[14] Kuan Fang, Yuke Zhu, Animesh Garg, Andrey Kurenkov, Viraj Mehta, Li Fei-Fei, and Silvio Savarese. Learning task-oriented grasping for tool manipulation from simulated self-supervision. *Robotics: Science and Systems (RSS)*, 2018.

[15] Carlo Ferrari and John F Canny. Planning optimal grasps. In *ICRA*, volume 3, pages 2290–2295, 1992.

[16] Wei Gao and Russ Tedrake. kpam-sc: Generalizable manipulation planning using keypoint affordance and shape completion. *arXiv preprint arXiv:1909.06980*, 2019.

[17] Marcus Gualtieri and Robert Platt. Learning 6-dof grasping and pick-place using attention focus. *Conference on Robot Learning (CoRL)*, 2018.

[18] Abhinav Gupta, Adithyavairavan Murali, Dhiraj Prakashchand Gandhi, and Lerrel Pinto. Robot learning in homes: Improving generalization and reducing dataset bias. In *Advances in Neural Information Processing Systems*, pages 9094–9104, 2018.

[19] Gregory Izatt, Geronimo Mirano, Edward Adelson, and Russ Tedrake. Tracking objects with point clouds from vision and touch. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4000–4007. IEEE, 2017.

[20] Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. *arXiv preprint arXiv:1812.07252*, 2018.

[21] Stephen James, Marc Freese, and Andrew J. Davison. Pyrep: Bringing v-rep to deep robot learning. *arXiv preprint arXiv:1906.11176*, 2019.

[22] Eric Jang, Sudheendra Vijayanarasimhan, Peter Pastor, Julian Ibarz, and Sergey Levine. End-to-end learning of semantic grasping. *arXiv preprint arXiv:1707.01932*, 2017.

[23] Leif P Jentoft, Qian Wan, and Robert D Howe. How to think about grasping systems-basis grasps and variation budgets. In *Robotics Research*, pages 359–372. Springer, 2018.

[24] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. Efficient grasping from rgbd images: Learning using a new rectangle representation. In *2011 IEEE International conference on robotics and automation*, pages 3304–3311. IEEE, 2011.

[25] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *Conference on Robot Learning (CoRL)*, 2018.

[26] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5):705–724, 2015.

[27] Beatriz León, Carlos Rubert, Joaquín Sancho-Bru, and Antonio Morales. Characterization of grasp quality measures for evaluating robotic hands prehension. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3688–3693. IEEE, 2014.

[28] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436, 2018.

[29] Jianlan Luo, Eugen Solowjow, Chengtao Wen, Juan Aparicio Ojea, Alice M Agogino, Aviv Tamar, and Pieter Abbeel. Reinforcement learning on variable impedance controller for high-precision robotic assembly. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3080–3087. IEEE, 2019.

[30] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *Robotics: Science and Systems (RSS)*, 2017.

[31] Jeffrey Mahler, Matthew Matl, Vishal Satish, Michael Danielczuk, Bill DeRose, Stephen McKinley, and Ken Goldberg. Learning ambidextrous robot grasping policies. *Science Robotics*, 4(26), 2019.

[32] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016.

[33] Alberto Rodriguez, Matthew T Mason, and Steve Ferry. From caging to grasping. *The International Journal of Robotics Research*, 31(7):886–900, 2012.

[34] Eric Rohmer, Surya PN Singh, and Marc Freese. V-rep: A versatile and scalable robot simulation framework. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1321–1326. IEEE, 2013.

[35] Carlos Rubert, Beatriz León, Antonio Morales, and Joaquín Sancho-Bru. Characterisation of grasp quality metrics. *Journal of Intelligent & Robotic Systems*, 89(3-4): 319–342, 2018.

[36] Ashutosh Saxena, Justin Driemeyer, and Andrew Y Ng. Robotic grasping of novel objects using vision. *The International Journal of Robotics Research*, 27(2):157–173, 2008.

[37] Gerrit Schoettler, Ashvin Nair, Jianlan Luo, Shikhar Bahl, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Deep reinforcement learning for industrial insertion tasks with visual inputs and natural rewards. *arXiv preprint arXiv:1906.05841*, 2019.

[38] Garrett Thomas, Melissa Chien, Aviv Tamar, Juan Aparicio Ojea, and Pieter Abbeel. Learning robotic assembly from cad. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9. IEEE, 2018.

[39] Josh Tobin, Lukas Biewald, Rocky Duan, Marcin Andrychowicz, Ankur Handa, Vikash Kumar, Bob McGrew, Alex Ray, Jonas Schneider, Peter Welinder, et al. Domain randomization and generative models for robotic grasping. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3482–3489. IEEE, 2018.

[40] Andy Zeng, Shuran Song, Kuan-Ting Yu, Elliott Donlon, Francois R Hogan, Maria Bauza, Daolin Ma, Orion Taylor, Melody Liu, Eudald Romo, et al. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[41] Andy Zeng, Shuran Song, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *arXiv preprint arXiv:1903.11239*, 2019.

[42] Jialiang Zhao, Jacky Liang, and Oliver Kroemer. Towards precise robotic grasping by probabilistic post-grasp displacement estimation. *arXiv preprint arXiv:1909.02129*, 2019.