Article

The International Journal of Robotics Research
1–15
© The Author(s) 2019
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/0278364919872545
journals.sagepub.com/home/ijr
SAGE

Learning task-oriented grasping for tool manipulation from simulated self-supervision

Kuan Fang[1], Yuke Zhu[1], Animesh Garg[1,2], Andrey Kurenkov[1], Viraj Mehta[1], Li Fei-Fei[1] and Silvio Savarese[1]

Abstract
Tool manipulation is vital for facilitating robots to complete challenging task goals. It requires reasoning about the desired effect of the task and, thus, properly grasping and manipulating the tool to achieve the task. Most work in robotics has focused on task-agnostic grasping, which optimizes for only grasp robustness without considering the subsequent manipulation tasks. In this article, we propose the Task-Oriented Grasping Network (TOG-Net) to jointly optimize both task-oriented grasping of a tool and the manipulation policy for that tool. The training process of the model is based on large-scale simulated self-supervision with procedurally generated tool objects. We perform both simulated and real-world experiments on two tool-based manipulation tasks: sweeping and hammering. Our model achieves overall 71.1% task success rate for sweeping and 80.0% task success rate for hammering.

Keywords
Grasping, manipulation, learning and adaptive systems

1. Introduction

The ability of using tools for problem solving is a common trait exhibited by intelligent animals. In robotics, the task of tool manipulation is the employment of a manipulable object, i.e., a tool, to fulfill a task. This invites a series of research challenges in understanding the semantic, geometric, and functional properties of objects. According to Brown and Sammut (2012), there are four key aspects to learning task-oriented tool usage: (a) understanding the desired effect; (b) identifying properties of an object that make it a suitable tool; (c) determining the correct orientation of the tool prior to usage; and (d) manipulating the tool. Therefore, a task-oriented grasp is a grasp that makes it possible to correctly orient the tool and then manipulate it to complete the task.

Consider a hammer as shown in Figure 1. The best grasp predicted by a *task-agnostic* grasp prediction model that optimizes for the robustness of lifting up, such as Dex-Net (Mahler et al., 2017), is likely to reside close to the center of mass. However, the hammering task can be best achieved by holding the hammer at the end of the handle, thus to generate a high moment at the point of impact on the head. And yet when the same object is used for sweeping trash off the table, it should be instead grasped by the head to spare the largest contact surface area with the target objects. Therefore, in order to select the best *task-oriented* grasp for a particular task, we need to take into account the synergy between grasping robustness and suitability for the manipulation objective.

The problem of understanding and using tools has been studied in robotics, computer vision, and psychology (Baber, 2003; Gibson, 1979; Osiurak et al., 2010; Zhu et al., 2015). Studies in robotics have primarily focused on reasoning about the geometric properties of tools (Dang and Allen, 2012; Song et al., 2010), and typically required prior knowledge of object shapes, predefined affordance labels, or semantic constraints. These requirements limit the usefulness of such techniques in realistic environments with uncertainty of perception and actuation. Some pioneering works have also grounded the tool grasping problem in an interactive environment (Mar et al., 2015, 2017).

[1]Stanford University, Stanford, CA, USA
[2]Nvidia, Santa Clara, CA, USA

Corresponding author:
Kuan Fang, Stanford University, 353 Jane Stanford Way, Stanford, CA 94305, USA.
Email: kuanfang@ai.stanford.edu

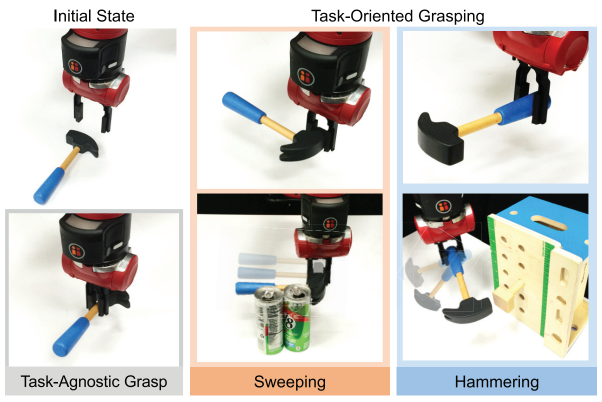


Fig. 1. The same object can be grasped by a robot in different ways. A task-agnostic grasp can lift up the hammer but it might not be suitable for specific manipulation tasks, such as sweeping or hammering. We aim to directly optimize the grasp selection for task success, by jointly choosing a task-oriented grasp and the subsequent manipulation actions.

Nonetheless, their approaches rely on hand-engineered features and simplified action spaces, allowing limited generalization to novel objects and new manipulation tasks.

In this work, we consider a tool manipulation task as a two-stage procedure. First, the robot picks up a tool resting on the tabletop. Second, it manipulates this grasped tool to complete a task. We propose the Task-Oriented Grasping Network (TOG-Net), a learning-based model for jointly predicting task-oriented grasps and subsequent manipulation actions based on the visual inputs. To accommodate the need for large amounts of training data for deep learning, we embrace the self-supervised learning paradigm (Levine et al., 2016; Mar et al., 2017; Pinto and Gupta, 2016), where training labels are collected through the robot performing grasping and manipulation attempts in a trial-and-error fashion. To scale up our self-supervised data collection, we leverage a real-time physics simulator (Coumans and Bai, 2016) that allows a simulated robot to execute the tasks with diverse procedurally generated 3D objects. We evaluate our method on a hammering task and a sweeping task, and show that our model learns robust policies that generalize well to novel objects both in simulation and the real world. In the real-world experiments, our model achieves 71.1% task success rate for the sweeping task and 80.0% for the hammering task using 9 unseen objects as tools.

Our primary contributions are three-fold.

1. We propose a learning-based model for jointly learning task-oriented grasping and tool manipulation that directly optimizes for task success.
2. To train this model, we develop a mechanism for generating large-scale simulated self-supervision with a large repository of procedurally generated 3D objects.
3. We demonstrate that our model can generalize to using novel objects as tools in both simulation and the real world.

In addition to providing an extended version of our conference paper (Fang et al., 2018b), this journal article provides two additional experiments. In the first experiment, we analyze how the model generalize across objects by training and testing on different types of shapes. In the second experiment, we apply the proposed method on the image containing multiple objects and select the best tool object for each task. Supplementary material is available at: bit.ly/task-oriented-grasp.

2. Related work

2.1. Task-agnostic grasping

Robotic grasping is a long-standing challenge that involves perception and control. Classical methods approach the grasping problem from a purely geometric perspective. They optimize grasp quality measures based on analytic models of constraints deriving from geometry and physics, such as force closure and form closure (Ferrari and Canny, 1992; Rodriguez et al., 2012; Weisz and Allen, 2012). While grasp quality measures offer a principled mathematical framework for grasp evaluation, their practical usage is limited by the large space of possible grasps. Several techniques have been proposed to restrict the search space of grasp candidates. This includes representing objects with shape primitives (Miller et al., 2003), simplifying the search of grasps in a subspace of reduced dimensionality (Ciocarlie and Allen, 2009), and leveraging a dataset of objects with known grasps to speed up grasp selection for novel objects (Goldfeder et al., 2009; Mahler et al., 2016).

Another limitation of these approaches is that they require the full 3D geometry of the object, which restricts their usage in unstructured real-world environments. The recent development of machine learning techniques, especially deep learning, has enabled a surge of research that applies data-driven methods to robotic grasping (Bohg et al., 2014; Kappler et al., 2015). These learning methods largely fall into two families depending on the source of supervision: (1) supervised learning approaches (Herzog et al., 2012; Kroemer et al., 2012; Lenz et al., 2015; Mahler et al., 2017; Morrison et al., 2018; Saxena et al., 2008; ten Pas, et al., 2017), where the models are trained with a dataset of objects with ground-truth grasp annotations or demonstrations; and (2) self-supervision approaches (Bousmalis et al., 2017; Fang et al., 2018a; Jang et al., 2017; Levine et al., 2016; Pinto and Gupta, 2016), where grasp labels are automatically generated by a robot's trial and error on large numbers of real-world or simulated grasp attempts. In the latter category, several works have also extended algorithms from reinforcement learning (Kalashnikov et al., 2018) for utilizing the self-supervision labels as rewards for learning a grasping policy.

To address the data-hungry nature of deep neural networks, several works (Mahler et al., 2017; Viereck et al., 2017) trained their models in simulation where 3D models of all objects are available and ground-truth grasps can be automatically annotated using analytic models of form and force closure. Using a large set of 3D models with ground-truth grasps derived from these analytic models, these works generated large datasets of images with annotations for either the sampled grasps on images and corresponding grasp qualities (Mahler et al., 2017) or the distance to the nearest good grasp (Viereck et al., 2017). In both cases, the images are generated by placing one or more objects in a simulated scene, rendering an image of the scene from a virtual depth camera, and calculating labels for multiple crops of the image representing different grasps or gripper positions, respectively. Only learning with rendered depth images as opposed to rendered RGB images enabled the trained models to transfer to execution on a real robot without further fine-tuning, because physical depth cameras produce images that are largely similar to rendered depth images. Other works have also leveraged simulation for learning models that use RGB images as input, with either no transfer to real robots (Bousmalis et al., 2018) or use domain adaptation to speed up real-world learning (Yan et al., 2018).

2.2. Task-oriented grasping

A major portion of research in grasping aims at holding the object in the robot gripper so as to not drop it despite external wrenches. In practice, however, the end goal of grasping is often to manipulate an object to fulfill a goal-directed task once it has been grasped. When the grasping problem is contextualized in manipulation tasks, a grasp planner that solely satisfies the stability constraints is no longer sufficient to satisfy the task-specific requirements. In the classical grasping literature, researchers have developed task-oriented grasp quality measures using a task wrench space (Haschke et al., 2005; Li and Sastry, 1988; Prats et al., 2007). Data-driven approaches have also been used to learn task-related constraints for grasp planning (Dang and Allen, 2012; Song et al., 2010). These studies incorporate semantic constraints, which specify which object regions to hold or avoid, based on a small dataset of grasp examples. Recently, several works further proposed to train a vision-based deep-learning model from a larger synthetic dataset for estimating the semantic constraints more robustly (Detry et al., 2017; Kokic et al., 2017). However, these grasping methods usually do not entail the success of the downstream manipulation tasks, and the hand-labeled semantic constraints cannot generalize across a large variety of objects.

In contrast, our work jointly learns the task-aware grasping model and the manipulation policy given a grasp. Thus, our grasping model is directly optimized to fulfill its downstream manipulation tasks. Furthermore, we employ deep neural networks to train our task-aware grasping models on a large repository of 3D objects, enabling it to generalize from this repository of objects to unseen objects as well as from simulation to the real world.

2.3. Affordance learning

Another line of related work centers around understanding the affordances of objects (Do et al., 2017; Koppula et al., 2013; Zhu et al., 2014, 2015). The notion of affordances introduced by Gibson (1979) characterizes the functional properties of objects and has been widely used in the robotics community as a framework of reasoning about objects (Katz et al., 2014; Varadarajan and Vincze, 2012).

Prior art has developed methods to learn different forms of object affordance such as semantic labels (Zhu et al., 2014), spatial maps (Jiang et al., 2012), and motion trajectories (Zhu et al., 2015). Our work follows a progression of previous work on behavior-grounded affordance learning (Fitzpatrick et al., 2003; Jain and Inamura, 2011; Mar et al., 2015, 2017; Stoytchev, 2005), where the robot learns object affordance by observing the effects of actions performed on the objects. Nonetheless, we do not explicitly supervise our model to learn and represent goal-directed object affordances. Instead, we demonstrate that our model's understanding of object affordance naturally emerges from training grasping and manipulation simultaneously. Recent work by Mar et al. (2017) has the closest resemblance to our problem formulation; however, their action space consists of a small set of discrete actions, whereas we employ a multi-dimensional continuous action space. Their method uses self-organizing maps with hand-designed tool pose and affordance descriptors, whereas we eschew feature engineering in favor of end-to-end deep learning.

3. Problem statement

Our goal is to control a robot arm to perform tool-based manipulation tasks using novel objects. Each task is a two-stage process. In the first stage, the robot grasps an object as a tool for a task. In the second stage, the robot manipulates the tool to interact with the environment and achieve the task goal. The visual appearance of the tool is provided for the robot to accomplish this.

3.1. Notation of grasping

The robot operates in a workspace based on camera observations, where $\mathcal{O}$ denotes the observation space. We consider the grasping problem in the 3D space, where $\mathcal{G}$ denotes the space of possible grasps. Given a pair of observation $\mathbf{o} \in \mathcal{O}$ and grasp $\mathbf{g} \in \mathcal{G}$, let $S_G(\mathbf{o}, \mathbf{g}) \in \{0, 1\}$ denote a binary-valued grasp success metric, where $S_G = 1$ indicates that the grasp is successful according to the predefined metric. In practice, the underlying sensing and motor noise introduce uncertainty to the execution of a grasp. We measure the robustness of a grasp $Q_G(\mathbf{o}, \mathbf{g})$ by the probability of grasp success under uncertainty, where $Q_G(\mathbf{o}, \mathbf{g}) = \Pr(S_G = 1 | \mathbf{o}, \mathbf{g})$. This grasp metric S_G is *task-agnostic*, which evaluates the quality of a grasp without grounding to a specific task. As we noted in the previous section, data-driven grasping methods (Mahler et al., 2017;

Viereck et al., 2017) have focused on optimizing *task-agnostic* grasps.

3.2. Problem setup

By contrast, we contextualize the grasping problem in tool manipulation tasks. In our setup, the grasping stage is followed by a manipulation stage, where a policy π produces actions to interact with the environment once the object is grasped. Intuitively, both the choice of grasps and the manipulation policy play an integral role in the success rate of a task.

Let $S_T(\mathbf{o}, \mathbf{g}) \in \{0, 1\}$ denote a binary-valued task-specific success metric, where $S_T = 1$ indicates that the task T was successfully done based on the goal specification. Clearly the grasp success is the premise of the task success, i.e., $S_T = 1$ entails $S_G = 1$. Given a manipulation policy π for the task, we measure the task quality Q_T^π of a *task-oriented* grasp by the probability of task success under policy π, where $Q_T^\pi(\mathbf{o}, \mathbf{g}) = \Pr(S_T = 1 | \mathbf{o}, \mathbf{g})$. Thereafter, the overall learning objective is to train both policies simultaneously such that

$$\mathbf{g}^*, \pi^* = \underset{\mathbf{g}, \pi}{\arg\max}\ Q_T^\pi(\mathbf{o}, \mathbf{g}) \tag{1}$$

We aim at selecting the optimal grasp $\mathbf{g}^*$ that is most likely to lead to the completion of the task, and at the same time finding the best policy π^* to perform the task conditioned on a grasp. In practice, we implement both the grasping policy and the manipulation policy using deep neural networks. We detail the design of the neural network models and their training procedures in the next section.

3.3. Assumptions

We consider the problem of task-oriented grasp planning with a parallel-jaw gripper based on point clouds from a depth camera. The training of our model uses simulated data generated from a real-time physics simulator (Coumans and Bai, 2016). Our design decision is inspired by the effective use of depth cameras in transferring the grasping model and manipulation policy trained in simulation to reality (Mahler et al., 2017; Viereck et al., 2017). Further, to reduce the search space of grasping candidates, we restrict the pose of the gripper to be perpendicular to the table plane. In this case, each grasp $\mathbf{g} = (g_x, g_y, g_z, g_\phi)$ has four degrees of freedom (DOFs), where $(g_x, g_y, g_z) \in \mathbb{R}^3$ denotes the position of the gripper center, and $g_\phi \in [0, 2\pi)$ denotes the rotation of the gripper in the table plane. Each observation $\mathbf{o} \in \mathbb{R}_+^{H \times W}$ is represented as a depth image from a fixed overhead RGB-D camera with known intrinsics.

4. Task-oriented grasping for tool manipulation

As shown in Figure 2, our TOG-Net consists of a task-oriented grasping model and a manipulation policy. The two modules are jointly trained to achieve task success together. The grasping model decouples the task success prediction using the chain rule to improve the data efficiency. In this section, we first present the design and implementation of the two modules, and then describe how they are jointly trained using simulated self-supervision.

The TOG-Net prediction is described in Algorithm 1. Given the input depth image $\mathbf{o}$, we first sample 200 antipodal grasp candidates as G based on the gradients (Mahler et al., 2017). We run the cross-entropy method (CEM) (Rubinstein and Kroese, 2004) for $K = 3$ iterations as in Mahler et al. (2017), in order to rank the grasp candidates and select the one that corresponds to the highest task quality. In each iteration, we first predict the task quality $\hat{Q}_T^\pi(\mathbf{o}, \mathbf{g})$ for each candidate grasp $\mathbf{g}$. If it is not the last iteration, we sort the grasps in the descending order of the computed $\hat{Q}_T^\pi(\mathbf{o}, \mathbf{g})$ and keep the top $\lambda = 0.25$ grasps. Then we fit a Gaussian mixture model (GMM) on these grasps and sample $N = 50$ grasps from the resultant GMM. In the last iteration, we just the choose the grasp with the highest predicted task quality and we compute the action for the chosen grasp using the stochastic policy $\pi(\mathbf{a}|\mathbf{o}, \mathbf{g})$.

4.1. Task-oriented grasp prediction

High-quality task-oriented grasps should simultaneously satisfy two types of constraints. First, the tool must be stably held in the robot gripper, which is the goal of task-agnostic grasping. Second, the grasp must satisfy a set of physical and semantic constraints that are specific to each task.

Algorithm 1. TOG-Net Prediction.

```
Require: $\mathbf{o}$: depth image
Require: $T$: task ID
Require: $K$: number of CEM iterations
Require: $N$: number of sampled grasps in each iteration
Require: $\lambda$: proportion of grasps used in the refitting
1: Sample a set of antipodal grasps $G$ from $\mathbf{o}$
2: for $k \leftarrow 1, \ldots, K$ do
3:     for each $\mathbf{g} \in G$ do
4:         Compute $\hat{Q}_T^\pi(\mathbf{o}, \mathbf{g}) \leftarrow \hat{Q}_{T|G}^\pi(\mathbf{o}, \mathbf{g}) \cdot \hat{Q}_G(\mathbf{o}, \mathbf{g})$
5:     end for
6:     Sort $G$ in the descending order of $Q_T^\pi(\mathbf{o}, \mathbf{g})$
7:     if $k < K$ then
8:         Keep the top $\lambda|G|$ grasps in $G$
9:         Fit the GMM on $G$
10:        Sample $N$ grasps from the GMM as $G$
11:    end if
12: end for
13: Take the first element in $G$ as $\mathbf{g}$
14: Sample action $\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{o}, \mathbf{g})$
15: return $\mathbf{g}$, $\mathbf{a}$
```

For a given observation $\mathbf{o} \in \mathcal{O}$, a small subset of grasps $\mathcal{G}_\alpha \subseteq \mathcal{G}$ can robustly lift up the object with a grasp quality higher than α, i.e., $Q_G(\mathbf{o}, \mathbf{g}) \geqslant \alpha$. Hence, the task agnostic grasp prediction problem involves finding the corresponding grasp $\mathbf{g}$ that maximizes the grasp quality

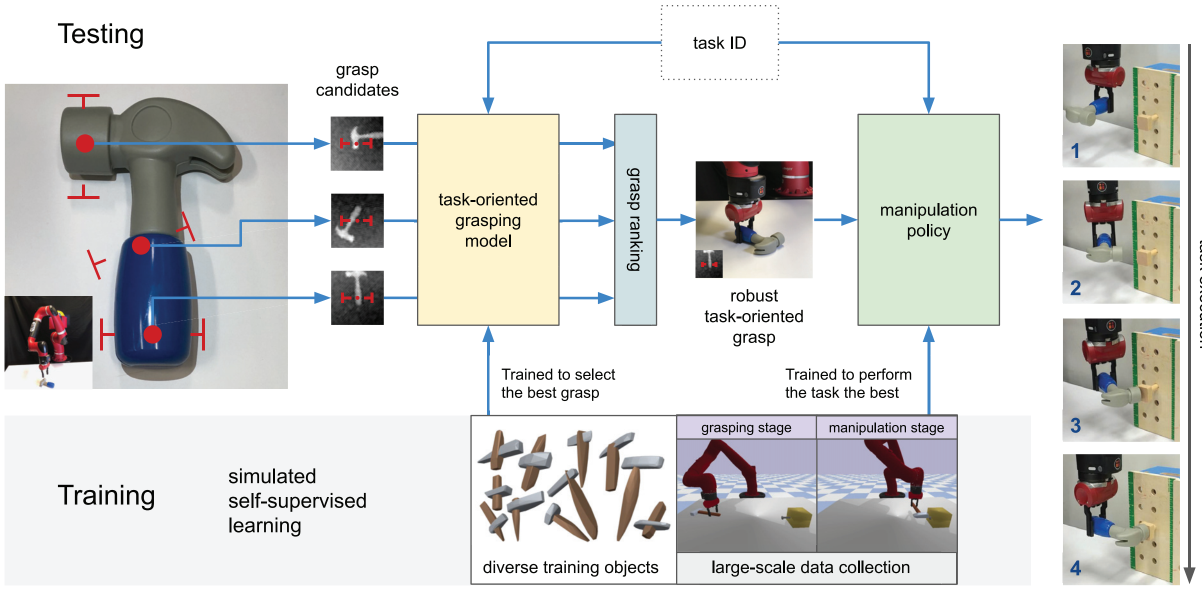

Testing
grasp candidates
task ID
task-oriented grasping model
grasp ranking
robust task-oriented grasp
manipulation policy
1
2
3
4
task execution
Trained to select the best grasp
Trained to perform the task the best
Training
simulated self-supervised learning
grasping stage
manipulation stage
diverse training objects
large-scale data collection

Fig. 2. Model overview. Our model consists of a task-oriented grasping model and a manipulation policy. Given the visual inputs of the object, we sample multiple grasp candidates. The task-oriented grasping model computes a grasp quality score for each candidate based on the planned task, and chooses the grasp with the highest score. Given the observation of the scene, the manipulation policy outputs actions conditioned on the selected grasp. The two modules are trained jointly using simulated self-supervision.

$Q_G(\mathbf{o},\mathbf{g}) = \Pr(S_G = 1|\mathbf{o},\mathbf{g})$. This prediction problem can be solved with a variety of methods, and we build upon the approach of Mahler et al. (2017) that uses quality function approximation with algorithmic supervision via analytic models.

As noted in the problem statement, the overall objective is maximizing the probability of task success $Q_T^\pi(\mathbf{o},\mathbf{g})$ under a policy π, grasp $\mathbf{g} \in \mathcal{G}$. However, directly solving this problem results in a discrete space search over a large space of grasps, and then a subsequent optimization problem to solve for a manipulation policy given each grasp. Furthermore, we note that tool manipulation task execution only succeeds if the grasp has succeeded. However, not all grasps $\mathbf{g} \in \mathcal{G}_\alpha$ result in a successful task. Specifically, we can define a conditional robustness metric $Q_{T|G}^\pi$ that measures the probability of task success (under policy π) conditioned on a successful grasp, where $Q_{T|G}^\pi(\mathbf{o},\mathbf{g}) = \Pr_\pi(S_T = 1|S_G = 1, \mathbf{o}, \mathbf{g})$. Then, task-oriented grasps form a subset of task-agnostic grasps: $\mathcal{G}_{\alpha,\delta} \subseteq \mathcal{G}_\alpha$, i.e., the grasps which lead to task success with a task quality conditioned on the grasp $Q_{T|G}^\pi(\mathbf{o},\mathbf{g}) \geqslant \delta$ where δ is a chosen threshold.

This formulation lets us effectively decouple the two problems as: (1) finding robust task-agnostic grasps; and (2) finding the best task-oriented grasp among these robust grasps. The key observation is that the task quality metric $Q_T^\pi(\mathbf{o},\mathbf{g})$ can be factorized into two independently computed terms: $Q_{T|G}(\mathbf{o},\mathbf{g})$ and $Q_G(\mathbf{o},\mathbf{g})$. Formally, the task robustness $Q_T(\mathbf{o},\mathbf{g})$ can be decomposed as follows:

$$\begin{aligned} Q_T^\pi(\mathbf{o},\mathbf{g}) &= \Pr_\pi(S_T = 1|\mathbf{o},\mathbf{g}) \\ &= \Pr_\pi(S_T = 1, S_G = 1|\mathbf{o},\mathbf{g}) \\ &= \Pr_\pi(S_T = 1|S_G = 1, \mathbf{o},\mathbf{g}) \cdot \Pr(S_G = 1|\mathbf{o},\mathbf{g}) \\ &= Q_{T|G}^\pi(\mathbf{o},\mathbf{g}) \cdot Q_G(\mathbf{o},\mathbf{g}) \end{aligned}$$

Our model learns to approximate the values of grasp quality $Q_G(\mathbf{o},\mathbf{g})$ and task quality conditioned on a grasp $Q_{T|G}^\pi(\mathbf{o},\mathbf{g})$ using deep neural networks given an object $\mathbf{o}$ and a grasp $\mathbf{g}$ as inputs. We denote the predicted values as $\hat{Q}_G(\mathbf{o},\mathbf{g};\theta_1)$ and $\hat{Q}_{T|G}^\pi(\mathbf{o},\mathbf{g};\theta_2)$, where θ_1 and θ_2 represent the neural network parameters.

4.2. Manipulation policy

To complete the task with different tools and different grasps, the manipulation policy should be conditioned on $\mathbf{o}$ and $\mathbf{g}$. The manipulation policy can be either an external motion planner or a learned policy. While our pipeline is not limited to a specific action space, we choose to control the robot in an open-loop manner using parameterized motion primitives. Each task has a single predefined type of motion primitive as a parameterized trajectory parallel to the planar table surface. After the motion primitive is chosen based on the task environment, the manipulation policy predicts the continuous manipulation action $\mathbf{a} = (a_x, a_y, a_z, a_\phi) \in \mathbb{R}^3$ as the parameters of the motion primitive, where (a_x, a_y, a_z) and a_ϕ are the translation and rotation of the motion primitive. We use a Gaussian policy $\pi(\mathbf{a}|\mathbf{o},\mathbf{g};\theta_3) = \mathcal{N}(f(\mathbf{o},\mathbf{g};\theta_3), \Sigma)$, where $f(\mathbf{o},\mathbf{g};\theta_3)$ is a

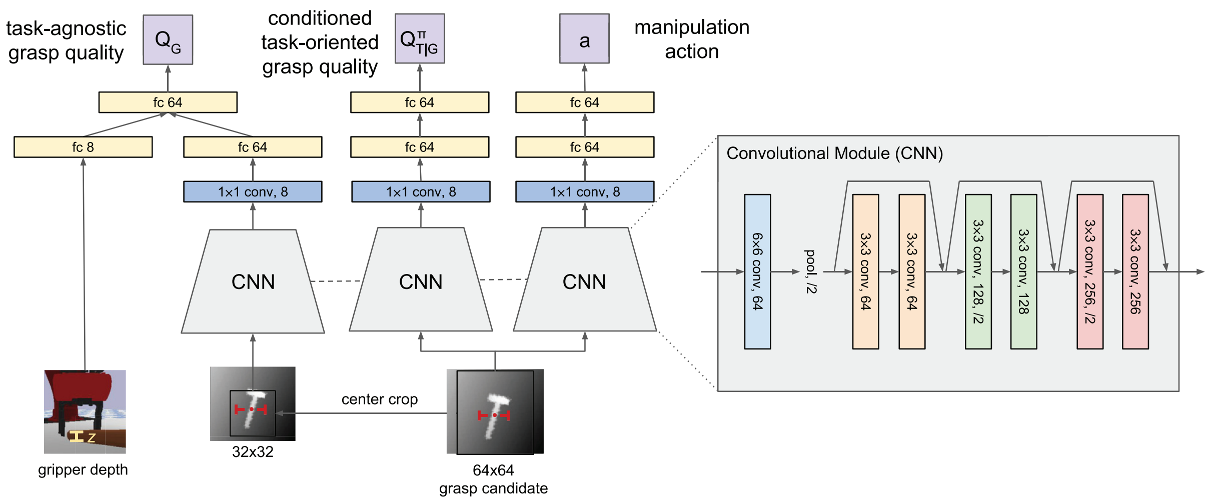


Fig. 3. TOG-Net. The inputs to the network are two depth image crops and the sampled gripper depth z. The network predicts task-agnostic grasp quality, conditioned task-oriented grasp quality, and manipulation actions. The convolutional neural network (CNN) modules share parameters as denoted by the dashed lines. Residual network layers and batch normalization are used in the CNN modules.

neural network for predicting the mean with parameters θ_3 and the covariance matrix Σ is a constant diagonal matrix.

4.3. Neural network architecture

In Figure 3, we propose a three-stream neural network architecture for jointly predicting $\hat{Q}_G$ and $\hat{Q}^{\pi}_{T|G}$ and $\mathbf{a}$. Following the practice of Mahler et al. (2017), we convert $\mathbf{o}$ and $\mathbf{g}$ into gripper depth z and image crops as inputs to the neural network. The gripper depth is defined as the distance from the center of the two fingertips to the object surface. The image crops are centered at the grasp center (g_x, g_y, g_z) and aligned with the grasp axis orientation ϕ. Mahler et al. (2017) used image crops of size 32×32 to focus on the contact between the gripper and the tool. To achieve the task success, our model is supposed to reason about the interactions between the tool and the task environment which requires a holistic understanding of the shape of the tool. Thus, our model predicts $\hat{Q}^{\pi}_{T|G}$ and $\mathbf{a}$ using larger image crops of size 64×64 which covers most of training and testing objects. Meanwhile the center crop of 32×32 is used to predict $\hat{Q}_G$. Our neural network is composed of three streams that share parameters in their low-level convolutional layers, extracting identical image features, denoted by dotted lines. Building atop the grasp quality convolutional neural network (GQCNN) of Mahler et al. (2017), we use residual network layers (He et al., 2016) to facilitate the learning process. On top of the convolutional layers with shared weights, we apply bottleneck layers of 1×1 convolutional filters for each stream to reduce the size of the network.

4.4. Learning objectives and optimization

We jointly train the task-oriented grasping model and the manipulation policy with simulated robot experiments of grasping and manipulation. Each simulated episode in our training dataset contains sampled grasp $\mathbf{g}$, action $\mathbf{a}$, the grasp success label S_G, and the task success label S_T. We use cross-entropy loss $\mathcal{L}$ for training the grasp prediction functions $\hat{Q}_G$ and $\hat{Q}^{\pi}_{T|G}$. For training the policy π, we use the policy gradient algorithm with gradients $\nabla \log \pi(\mathbf{a}|\mathbf{o}, \mathbf{g}; \theta_3)$. We use the task success label as the reward of the manipulation policy. As we are using a Gaussian policy as described above, this is equivalent to minimizing $\frac{1}{2} \| f(\mathbf{o}, \mathbf{g}; \theta_3) - \mathbf{a} \|^2_{\Sigma} \cdot S_G$ with respect to θ_3. Let the parameters of the neural network to be denoted as $\theta = \{\theta_1, \theta_2, \theta_3\}$, we jointly train our model by solving the following optimization problem:

$$\begin{aligned} \theta^* = \arg\min_{\theta} \sum_{i=1}^{N} & \mathcal{L}(S_G, \hat{Q}_G(\mathbf{o}, \mathbf{g}; \theta_1)) \\ & + \mathbb{1}[S_G = 1] \cdot \mathcal{L}(S_T, \hat{Q}^{\pi}_{T|G}(\mathbf{o}, \mathbf{g}; \theta_2)) \\ & + \mathbb{1}[S_T = 1] \cdot \tfrac{1}{2} \| f(\mathbf{o}, \mathbf{g}; \theta_3) - \mathbf{a} \|^2_{\Sigma} \end{aligned} \tag{2}$$

where $\mathbb{1}(\cdot)$ is the indicator function.

5. Self-supervision for grasping and manipulation

5.1. Procedural generation of tool objects

We train our model in simulation with a large repository of 3D models so as to generalize to unseen objects. However, existing 3D model datasets do not contain enough objects suitable for tool manipulation while exhibiting rich variations in terms of their geometric and physical properties. As shown in Figure 4, we leverage a common strategy of procedural generation (Bousmalis et al., 2017; Tobin et al., 2017) to produce a large set of diverse and realistic objects that can be used as tools for tasks we are interested in.

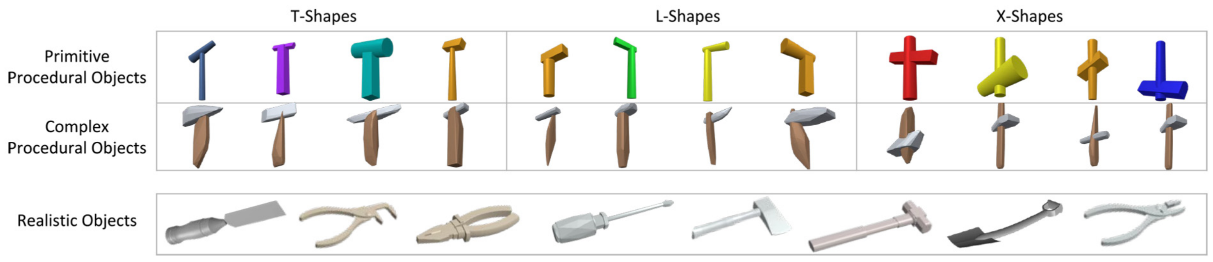


Fig. 4. Example of objects in simulation. The first two rows show the procedurally generated objects based on shape primitives as well as complex meshes. These objects are generated using three predefined composing rules to result in T-shapes, L-shapes, and X-shapes. The last row shows the realistic shapes from existing 3D model datasets.

While the generation process can be arbitrarily complex, we choose to generate objects composed of two convex parts. The two parts are connected by a fixed joint. We define three types of composed shapes: T-shapes, L-shapes, and X-shapes. For each object, two convex meshes are first sampled. Then the meshes are randomly scaled along the *x*-, *y*-, and *z*-axes. Depending on the type of object shape, the parts are shifted and rotated with respect to each other. We randomly sample physical dynamic properties such as density and friction coefficients.

We use two sets of meshes to generate two different set of objects: primitive and complex. We generate primitive meshes that are composed by a set of parameterized shape primitives including cuboids, cylinders, and polytopes. The dimensions and textures are randomly chosen from predefined ranges. The primitive meshes are generated by OpenSCAD. We also obtain convex object parts from a variety of realistic 3D object models as the complex meshes. This done by running convex decomposition (Mamou and Ghorbel, 2009) on each object from the dataset used by Kappler et al. (2015).

5.2. Data generation with simulated self-supervision

In order to collect large-scale datasets for training and evaluating our model, we develop a self-supervision framework to automatically generate training data. We leverage an open-source real-time physics simulator, PyBullet (Coumans and Bai, 2016), which allows a simulated robot to perform trial and error in millions of trials. We record grasp and task success labels in each trial and use them to train our models described previously. For training each task we collect the data in three rounds. After each round, we train the grasping model and the manipulation policy using the collected data to obtain an updated model. In each round we run the simulation for 500,000 trials.

In the first round, we perform a random policy using a GQCNN model trained on the Dex-Net 2.0 Dataset (Mahler et al., 2017). The original GQCNN model uses a CEM (Rubinstein and Kroese, 2004) to sample robust grasps corresponding to the highest task-agnostic grasp quality scores. However, the trained GQCNN usually leads to a collapsed mode of grasps that is most robust according to the ranking of the predicted scores. Ideally we want to collect data of diverse sampled grasps and evaluate how well they can be used in each task. An alternative is to uniformly sample grasps with grasp quality scores higher than a threshold. In practice, we found such sampling usually clusters on the long edge of a tool object because there are more antipodal grasps possible there. To encourage diverse exploration, we instead use non-maximum suppression (NMS) (Hartley and Zisserman, 2003), which is widely used in object detection algorithms. The NMS algorithm goes through the ranked antipodal grasps, and removes grasps which have short Euclidean distances with previous grasps with higher grasp quality scores. This guarantees all remaining grasps are separate from each other and usually produces 10–30 distinct modes. With these sampled grasps, the random policy uniformly samples manipulation actions from the action space for each task.

In the following rounds, we use the ϵ-greedy strategy with the updated grasping model. The grasping model uses the CEM with probability $1-\epsilon_1$, and uses the NMS method with GQCNN predictions as described above with ϵ_1 probability. The manipulation policy predicts and manipulation action parameters with probability $1-\epsilon_2$, and use random actions with probability ϵ_2. We set 0.2 for both ϵ_1 and ϵ_2.

6. Experiments

The goal of our experimental evaluation is to answer the following questions. (1) Does our method improve task performance compared with baseline methods? (2) Does the joint training qualitatively change the mode of grasping? (3) Can the model train with simulated self-supervision work in the real world?

We evaluate our method on two tabletop manipulation tasks: *sweeping* and *hammering*. We define our hammering task as a motion primitive that achieves fitting a peg in a hole with tight tolerance, which is prevalent in assembly tasks (Komizunai et al., 2008; Williamson, 1999). Sweeping, on the other hand, is a primitive in autonomous manipulation, such as in part positioning and reorientation

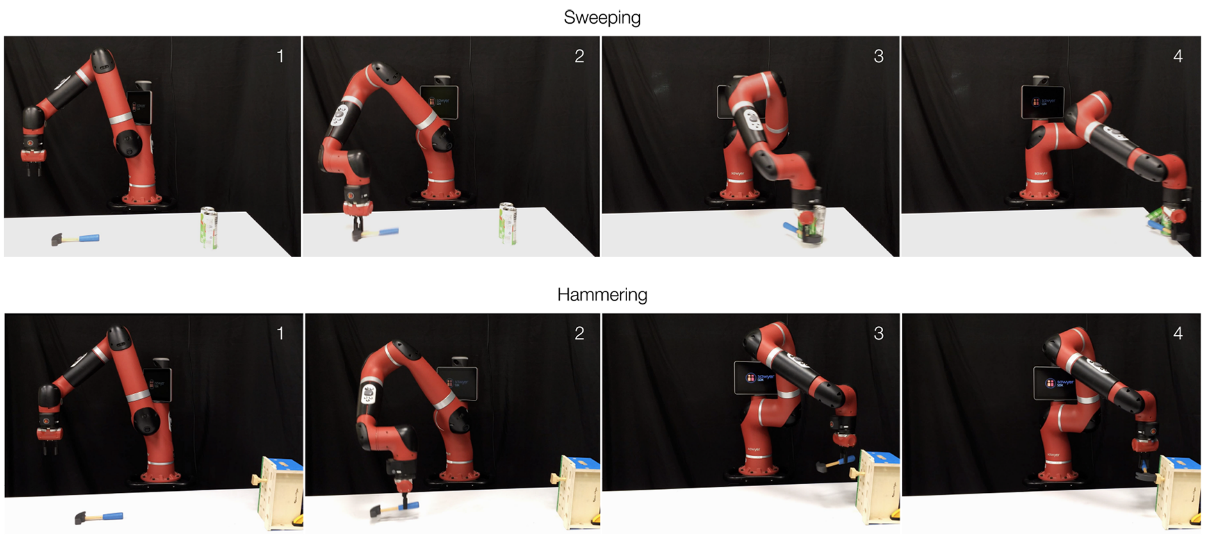


Fig. 5. Real-world environments. Sweeping and hammering are performed by a Sawyer robot arm in the real world. For sweeping, the robot grasps the tool object and use it to sweep the target objects (soda cans) off the table. For hammering, the robot hammers the peg into a slot using the tool object. A Kinect2 camera is mounted above the table to provide depth images as the input.

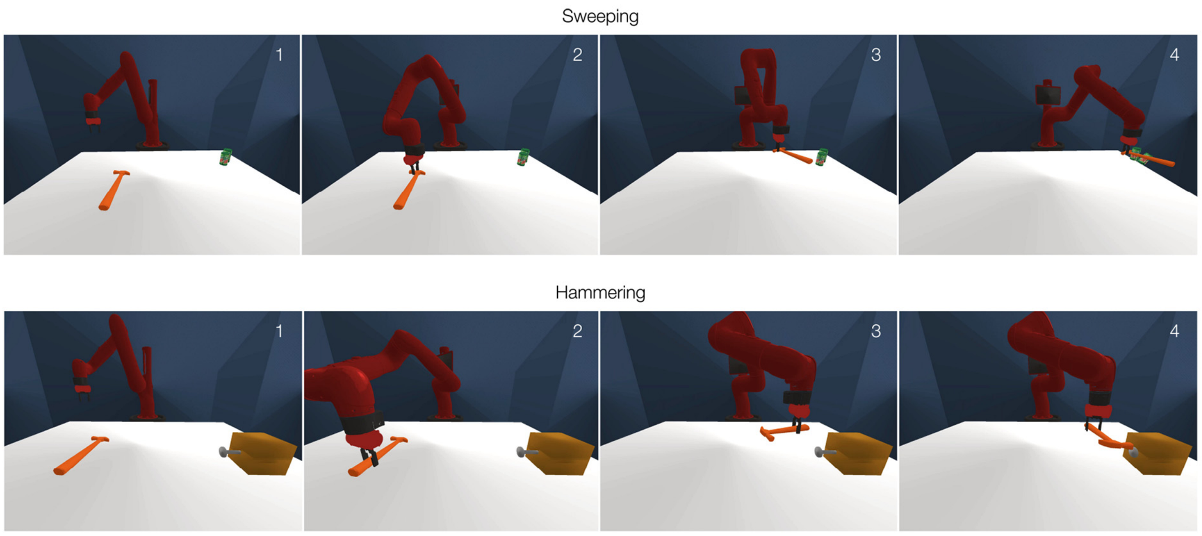


Fig. 6. Simulated environments. Simulated experiments are performed using a PyBullet physical simulation engine (Coumans and Bai, 2016). The simulation uses the CAD model of the Sawyer robot along with control application programming interfaces (APIs) that mimic the real-world setup. 3D shapes of the target soda cans, the peg, and the slot are loaded to the simulated scene. Realistic and procedurally generated objects are used as the tool objects.

(Lynch and Mason, 1996), grasping in clutter (Dogar and Srinivasa, 2011), and object manipulation without lifting (Meriçli et al., 2015). Sweeping with tools has been studied in the context of singulation and part retrieval (Eitel et al., 2017; Laskey et al., 2017). Each of these tasks requires grasping objects in specific modes which can often be different from the best stable grasp available, thereby resulting in competing objectives.

We evaluate our model in both simulation and the real world. The basic setup of both tasks includes a 7-DOF Rethink Robotics Sawyer Arm with a parallel jaw gripper, a $48'' \times 30''$ table surface, and an overhead Kinect2 camera. In simulation, the robot and camera are placed according to the real-world camera calibration results, in order to obtain consistent performance. For both experiments, we use models solely trained using simulated data.

6.1. Task design

The real-world and simulated environments are shown in Figures 5 and 6. Each environment consists of a robot arm, a table surface, and an overhead camera sensor. The robot in our environments is a 7-DOF Sawyer arm with an electric parallel gripper. Position control of the gripper is executed via the Intera 5 software platform. The simulated environments are performed using PyBullet physical simulation engine (Coumans and Bai, 2016). 3D models of the robot and the scene are loaded to the simulator with realistic physical parameters. In simulation, we implemented a control interface that mimics the Intera 5 control interface in the real world. As the robot uses open-loop position control in our task design, we do not need to simulate the control latency.

The table surface is a 1.22 m $\times$ 0.7 m rectangle, split into two halves: a grasping region and a manipulation region. Before each episode starts, an object is sampled and randomly dropped onto the grasping region to be used as the tool. A depth image is taken from the overhead Kinect2 camera as the input of the model. The model then predicts the 4-DOF grasp and the parameters of the motion primitive. The robot grasps the object from the grasping region and performs the task in the manipulation region. In our task design, the motion primitive is a predefined single step action. Our model predicts the starting gripper pose relative to a reference point predefined for each task.

6.1.1. Sweeping. Target objects are randomly placed in the manipulation region as the target objects. In the real world we use two soda cans as the target objects, and in simulation we randomly place one or two 3D models of cans. The task goal is to sweep all target objects off the table using the tool. The motion primitive of sweeping is a straight line trajectory of the gripper parallel to the table surface. The gripper trajectory starts from the pose (a_x, a_y, a_z, a_ϕ) and moves 40 cm along the y-axis of the world frame. Here (a_x, a_y, a_z, a_ϕ) is predicted relative to the mean position of the target objects. The task success is achieved when all target objects contact the ground. For robust sweeping, the tool ideally need to have a large flat surface in contact with the target object.

6.1.2 Hammering. A peg and a slot are randomly placed in the manipulation region, where the peg is horizontally halfway inserted into the slot. The task goal is to use the tool to hammer the peg fully into the slot. The motion primitive of hammering is a rotation of the gripper along the z-axis. The trajectory starts with the gripper pose (a_x, a_y, a_z, a_ϕ) and ends after the last arm joint rotates by 90° counterclockwise at full speed. Here (a_x, a_y, a_z, a_ϕ) is predicted relative to the position of the peg. The task success is achieved when the whole peg is inside the slot. This task requires a sufficient contact force between the tool and the peg to overcome the resistance. Meanwhile, the tool should avoid collisions with the peg before the hammering.

6.2. Experiment setup

Training used 18,000 procedurally generated objects including 9,000 PG-Primitive objects and 9,000 PG-Complex objects. In addition to randomizing physical properties, we randomly sample the camera pose and intrinsics by adding disturbances to the values obtained from the real-world setup.

During testing, we used 3,000 instances of each type of procedurally generated object. We also test on 55 realistic objects selected from Dex-Net 1.0 (Mahler et al., 2016) and the MPI Grasping Dataset (Bohg et al., 2014). These objects contain both tool-like and non-tool-like objects as shown in Figure 4. None of these test objects are seen during training.

We compare our method with four baselines.

1. *Antipodal + Random*: Use a sampled antipodal grasp with a random action uniformly sampled with x, y, z positions in $[-5, 5]$ in terms of centimeters and θ in $[-\frac{\pi}{20}, \frac{\pi}{20}]$.
2. *Task_Agn + Random*: A task-agnostic grasp from Dex-Net 2.0 (Mahler et al., 2017) with a random action.
3. *Task_Agn + Trained*: Same as above but with a manipulation policy trained with our method. This is akin to the current best solution.
4. *Task_Ori + Random*: An ablative version of our model with task-oriented grasps executed with a randomized action.

6.3 Simulated experiments

We evaluate our method on both tasks using the simulated setup described previously. For each algorithmic method, we run 100,000 episodes in simulation and report the task success rate. The task success analysis for each of the baselines and our method is presented in Figure 7.

Our model outperforms the four baselines in both tasks for all object categories. The contrast is more significant for hammering than sweeping. This is because hammering requires well-trained manipulation policy to direct the tool to hit the peg. A small deviance from the optimal hammering trajectory can let the tool miss the peg or collide with the slot. Meanwhile, for the sweeping task, when the robot uses a long edge of the tool to sweep, there is a high tolerance of manipulation action errors. Even random actions can often succeed. Among the three object categories, PG-Primitive is usually the easiest to manipulate with. Complex meshes cause more grasping failures and are harder for their geometric properties to reason about. Realistic objects are not usually good for sweeping because they are more roundish and very few have long edges. The hammering performances with realistic objects are much better, because many of these objects are cylinder objects with a bulky head and even actual hammers.

6.4. Real-world experiments

For our real-world experiments, we used nine unseen objects consisting of three categories as shown in Figure 8. T-shape and L-shape objects have similar geometric

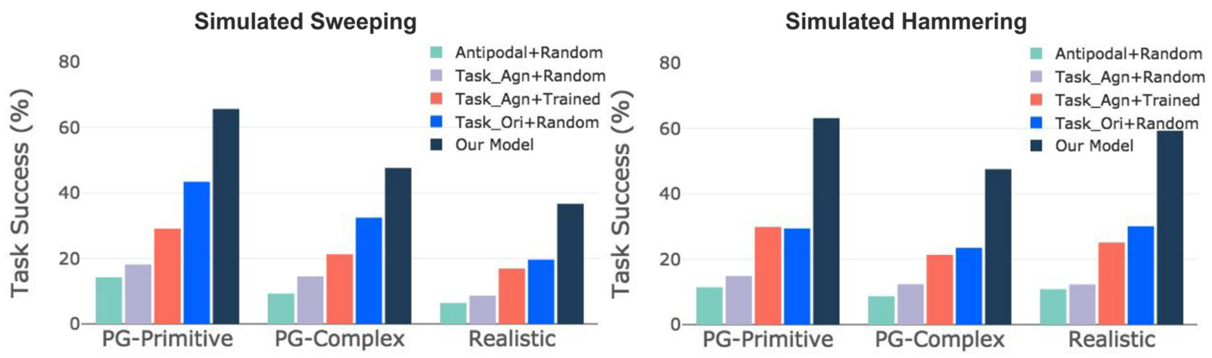


Fig. 7. Performance of simulated experiments. We perform an evaluation of our model for sweeping and hammering in simulation. We compare performance separately on three object categories as shown in Figure 4: procedurally generated objects with primitive meshes (PG-Primitive), procedurally generated objects with complex meshes (PG-Complex), and 3D objects from existing datasets (Realistic). Our model outperforms all baselines using the three object categories in both tasks.

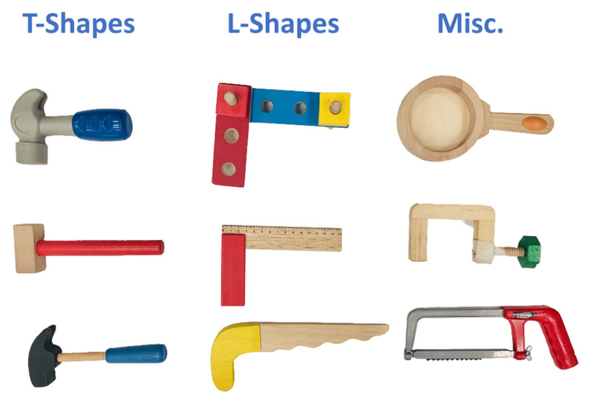


Fig. 8. Real-world test objects. We used nine unseen objects for our real-world experiments. These objects are grouped into three categories: T-shapes, L-shapes, and miscellaneous objects.

Table 1. Performance of real-world experiments. We compared our model with other grasping methods in terms of task success rates. We used nine real-world objects grouped into three categories. We performed five trials with each object for each method, for a total of 270 robot trials. The per-task and overall task success rates are reported in each cell.

Real-world sweeping	Grasping model		
	Antipodal + trained	Task-agnostic + trained	Our model
T-shapes	13.3	20.0	**73.3**
L-shapes	23.7	46.7	**80.0**
Miscellaneous	33.3	13.3	**60.0**
Overall	24.4	23.6	**71.1**
Real-world hammering	**Grasping model**		
	Antipodal + trained	Task-agnostic + trained	Our model
T-shapes	46.7	60.0	**86.7**
L-shapes	13.3	33.3	**86.7**
Miscellaneous	40.0	53.3	**66.7**
Overall	33.3	44.4	**80.0**

properties with our procedurally generated objects during training, whereas the miscellaneous objects have structures and curvatures totally unseen during training.

In the real world, we compared our model with two baseline methods: antipodal grasping with trained manipulation policy (antipodal + trained) and task-agnostic grasping with trained manipulation policy (task-agnostic + trained). We performed each task with each object for 5 robot trials for a total of 270 trials. The per-category and overall task success rates are listed in Table 1. For all object categories, our model achieved better performance compared with the baselines.

For sweeping, our model can successfully grasp the head of T-shapes or the short edge of L-shapes, and sweep with the longer part. For more complex miscellaneous objects, it is less obvious for the model to figure out which part should be grasped. However, for most trials, the grasp predicted by our model is intuitive to humans and leads to successful sweeping. For T-shapes, the difference between task-agnostic and task-oriented grasping is larger because the object usually only has one long handle. In contrast, for some L-shapes, the two edges are both long enough for the task, so grasping either edge does not make a significant difference. For miscellaneous objects, the model can have problems reasoning about novel object parts. For instance, it sometimes chooses to grasp the handle of the pan and sweep with the round part, which is unstable for sweeping roundish target objects.

For hammering, our model performs equally well for T-shapes and L-shapes. The failures are usually caused by occasional deviations during the execution of grasping or manipulation. As a comparison, baseline models often choose to grasp the head and hammer with the handle, which is sub-optimal. Compared with T-shapes and L-shapes, there might not be an obvious way to use

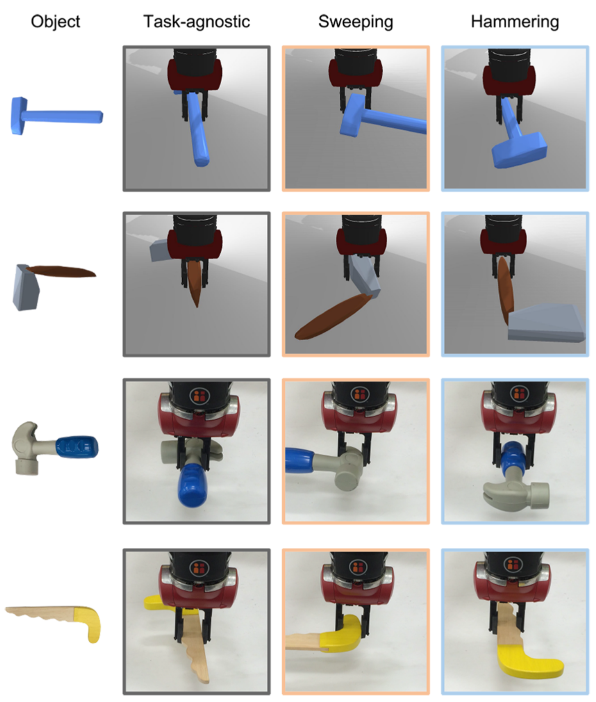


Fig. 9. Qualitative analysis of grasps. Column 1 shows RGB images of the tool objects. Column 2 shows example task-agnostic grasps. Columns 3 and 4 show task-oriented grasps chosen by our model for the sweeping and hammering tasks. Our model favors wide flat surfaces for sweeping and long moment arms for hammering.

miscellaneous objects as hammers. Among miscellaneous objects, the pan can be used as a hammer very well. The model tends to grasp the handle and hammer with the bulky roundish part.

6.5 Qualitative analysis of grasps

In Figure 9, we demonstrate that the same object can be grasped for different tasks by our trained model. Here we show the modes of task-agnostic grasping and task-oriented grasping for four example objects, two in simulation and two in the real world.

For the sweeping task, it is challenging to sweep all target objects off the table in one shot. It requires the tool to have a flat contact surface to facilitate the manipulation of roundish objects and to sweep across a large enough area to catch both of them. Our model learns to grasp the end of the tool object and spare as much surface area as possible for sweeping. This enables the robot to robustly sweep the cans most of the time.

For the hammering task, the main concerns are overcoming the resistance of the peg while avoiding collisions during hammering. We expect the robot to grasp the far end of the handle and hit the peg with the bulky part as the hammer head. Ideally, this could generate the largest torque on the hammer head when hitting the peg. In practice, we found the trained model tends to grasp a little closer to the hammer head on the handle. This is because we use a parallel jaw gripper and it is hard to balance the tool when the fingers are far away from the center of mass.

For robust task-agnostic grasping, the GQCNN model usually chooses the thin part near the center of mass. Although this sometimes still overlaps with the set of task-oriented grasps, it is not guaranteed if the selected grasp from the task-agnostic GQCNN is suitable for the task. In Figure 9, we show example task-agnostic grasps that are different from the task-oriented grasps mentioned previously.

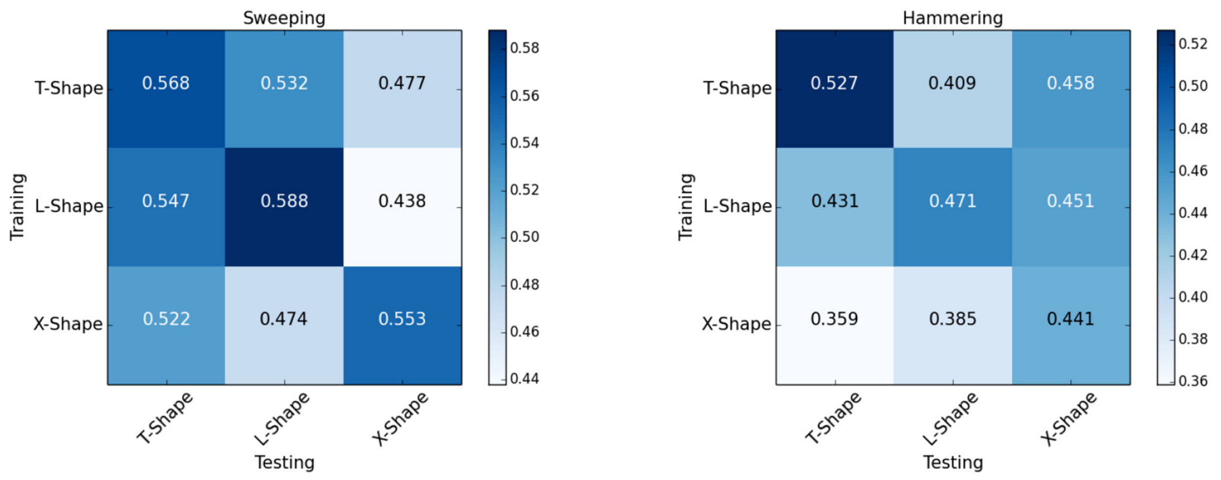
6.6 Analysis of generalization across objects

We performed experiments analyzing the generalizability of our model across different shapes as shown in Figure 10. We trained and tested models as described in the previous sections, except that the training objects and the testing objects only contain one type of shape in each experiment. More specifically, for each task, we trained three models for T-shapes, L-shapes, and X-shapes, respectively. Then each trained model is tested on each type of shape. Looking at the diagonal of each matrix, we concluded that L-shapes and T-shapes prove to be most useful for sweeping and hammering, respectively. Comparing elements on the diagonal with other elements, we found the model usually achieves the best performance when it is tested on the same type of shape as it was trained. When testing the model trained with X-shapes on X-shapes for hammering, the performance is slightly worse than models trained with T-shapes and L-shapes. This is because X-shapes are usually not very good for use as hammers, thus the model can get stuck in local optima during the learning process. When testing on different types of shapes, the task success rates decrease but are still reasonably good.

6.7 Tool selection

The TOG-Net trained for grasping a single object can also be used for tool selection. Given multiple objects placed on the table, we take the depth image that includes all candidate objects. We sample antipodal grasps over the depth image as candidate grasps and we forward the trained TOG-Net using the corresponding image crops as inputs to predict the grasps. As long as each image crop only contains the information of a single object, the predicted task quality scores will not be affected by other objects on the table. As in the original setup, we choose the grasp with the highest score and the corresponding object to be used as the tool.

We show heatmaps of the predicted grasps in Figure 11 to visualize the model’s preference over multiple objects for each task. We placed seven objects in random poses on the table, including four realistic shapes and three procedurally generated objects. Given the depth image of the table, we sampled 2,000 antipodal grasps on all objects and predicted the quality scores. We visualized the grasp-quality scores for task-agnostic grasping and task-quality scores



Sweeping (Training \ Testing)	T-Shape	L-Shape	X-Shape
T-Shape	0.568	0.532	0.477
L-Shape	0.547	0.588	0.438
X-Shape	0.522	0.474	0.553

Hammering (Training \ Testing)	T-Shape	L-Shape	X-Shape
T-Shape	0.527	0.409	0.458
L-Shape	0.431	0.471	0.451
X-Shape	0.359	0.385	0.441

Fig. 10. Generalization across shapes. We trained models on T-shapes, L-shapes, and X-shapes, respectively, as described in Figure 6. We evaluated the performance of each trained model on each type of shape. The resultant task success rates are reported in the two confusion matrices.

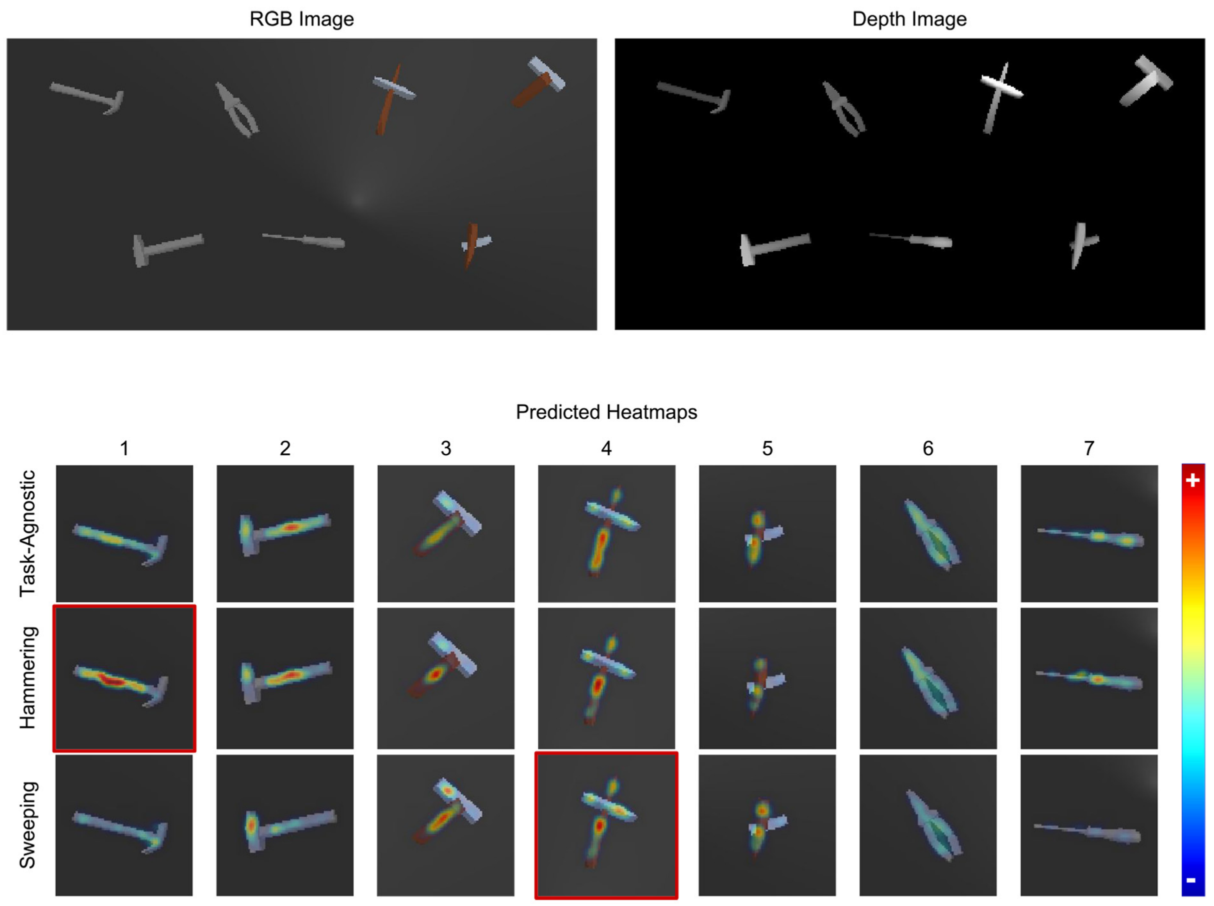


Fig. 11. Tool selection. With multiple objects resting on the table surface, our model can be used to select the optimal grasp on the suitable tools. We compared the task-oriented grasps for the two tasks and the task-agnostic grasps. The heatmap of the predicted task qualities are overlaid on the camera image. The red border indicates the object where the grasp with the highest predicted task quality is from.

for hammering and sweeping. To visualize the metrics in a comparable scale, we normalized the quality scores by the maximum predicted value of each task. We smoothed the heatmap using a Gaussian kernel with kernel size of 11 pixels. We overlayed the predicted heatmap on the RGB image crop of each object and we use a red border to indicate the best tool object chosen by the trained model.

The heatmaps highlight different objects and different regions for each task. For task-agnostic grasping, the highest grasp quality scores were on objects 1, 2, 3, 4, and 5. These objects have cylinder handles of suitable radius for the gripper to grasp, which makes them safe choices for task-agnostic grasping.

For hammering, the highest task quality was on object 1, which is a hammer with a long handle. The grasping region that is suitable for hammering is on the middle of the handle. Other objects including 2, 3, 4, and 7 were also chosen for hammering. Three of these objects are T-shapes or X-shapes that look like an ordinary hammer. Although object 7 (screwdriver) does not have a obvious hammer head, the model predicts that the robot can grasp the middle of the screwdriver and use the bulky handle as the hammer head.

For sweeping, the highest task quality score is predicted on object 4, which is a procedurally generated X-shaped object. The robot can grasp one of the two cylinder parts and uses the other to sweep the target objects. By inspecting objects 1, 2, 3, and 4, we note that the heatmaps highlight more on the short handle than hammering. In particular, for object 2, the head is too short to sweep both soda cans, so the robot tends to only grasp the head and use the long handle to contact with target objects. Object 7 has barely any predicted grasps for sweeping because either end is too thin for sweeping.

7. Discussion

Our method was proposed to solve tool-based manipulation as a two-stage problem composed of a task-oriented grasping stage and a manipulation stage. Under this problem formulation, we made several assumptions as described in previous sections. Clearly, there are more general and complex setups for each stage, which can lead to interesting technical challenges.

Following the practice of previous learning-based grasping works, our experiment setup chooses an action space using 4-DOF position control. To complete more challenging task goals, we would like the gripper to move and rotate freely in the 3D space. The action space would be much larger for such 6-DOF grasping and manipulation. However, the current sampling-based grasp prediction suffers from limited scalability. In the 6-DOF action space, the number of samples required by the CEM would increase exponentially. Instead of using sampling-based approaches, one can borrow ideas from object detection networks for 2D and 3D visual data. For example, one can use fully connected networks to predict all possible grasps directly from the whole image. Instead of sampling antipodal grasps using a hard-coded approach, one can design and train a neural network to efficiently predict the efficiently grasp proposals.

The motion primitives constrain how the manipulation tasks can be performed. It would be interesting to replace the motion primitives with manipulation policy using low-level control. Training a policy to perform high-frequency torque control of the robot arm would make the model more general and improve the task success rates. First, the motion primitives written by human might not be optimal. Directly optimizing a low-level control policy would likely to outperform the hard-coded motion primitives. Second, the mechanism of the robot arm is different from human bodies, so the robot does not need to complete the task in the same way as a human would. With more freedom in the manipulation policy, the robot might be able to learn interesting behaviors that are beyond human's expectation. Moving beyond the single-step motion primitives, the policy would need to be learned in a reinforcement learning setup or be learned from human experts' demonstrations. Such policies would be more complicated, so the data efficiency of the learning algorithm would be a major challenge.

One of the critical problems is to design feature representations that can be better generalized to unseen objects. In our experiments, we observed that models trained on one type of shape usually had inferior performance on different types of shapes. To generalize to unseen objects, we would need the training objects to contain objects of various shapes, sizes, and physical attributes in order to cover the distribution of testing objects. However, achieving a good coverage of real-world objects would be challenging and expensive. There could several directions to improve the ability of generalization from the representation perspective. One promising direction is to design a structured representation of object affordances in a way that it can be shared across different objects. For instance, knives and scissors have very different appearances but they both have sharp edges for cutting. In our current model design, it would be difficult to train on knives and test on scissors. With structured representations of object affordances, the knowledge learned for knives might be transferred to scissors more easily. Another direction is to resort to the meta-learning paradigm and learn feature presentations that can quickly adapt to unseen objects after training on a few trials on these objects.

8. Conclusion

We have developed a learning-based approach for task-oriented grasping for tool-based manipulation trained using simulated self-supervision. It jointly optimizes a task-oriented grasping model and its accompanying manipulation policy to maximize the task success rate. We have leveraged a physics simulator that allows a robot to autonomously perform millions of grasping and manipulation trials. The trial and error of the robot provides training data to supervise the deep neural network models. Our experimental results demonstrate that the task-oriented grasps selected by our model are more suitable for downstream manipulation tasks than the task-agnostic grasps.

In the future, our goal is to further improve the effectiveness and robustness of our model by training on a large dataset of realistic 3D models. In addition, we plan to scale up our model to complex manipulation tasks with end-to-end trained closed-loop manipulation policies.

Acknowledgements

We thank Ozan Sener for constructive discussions about the problem formulation. We thank Alex Fu, Julian Gao, and Danfei Xu for helping with the infrastructure of the real-world experiments. We thank Erwin Coumans for helping with the simulation.

Funding

This work was supported by Toyota (grant number 1186781-31-UDARO) and the Industrial Technology Research Institute (grant number 107-EC-17-A-21-1516).

ORCID iD

Kuan Fang https://orcid.org/0000-0002-7620-3117

References

Baber C (2003) *Cognition and Tool Use: Forms of Engagement in Human and Animal Use of Tools*. Boca Raton, FL: CRC Press.

Bohg J, Morales A, Asfour T and Kragic D (2014) Data-driven grasp synthesis – a survey. *IEEE Transactions on Robotics* 30(2): 289–309.

Bousmalis K, Irpan A, Wohlhart P, et al. (2017) Using simulation and domain adaptation to improve efficiency of deep robotic grasping. *arXiv preprint arXiv:1709.07857*.

Bousmalis K, Irpan A, Wohlhart P, et al. (2018) Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In: *2018 IEEE International Conference on Robotics and Automation*. IEEE, pp. 4243–4250.

Brown S and Sammut C (2012) Tool use and learning in robots. In: *Encyclopedia of the Sciences of Learning*. Berlin: Springer, pp. 3327–3330.

Ciocarlie MT and Allen PK (2009) Hand posture subspaces for dexterous robotic grasping. *The International Journal of Robotics Research* 28(7): 851–867.

Coumans E and Bai Y (2016) PyBullet, a Python module for physics simulation, games, robotics and machine learning. Available at: http://pybullet.org/ (accessed August 2019).

Dang H and Allen PK (2012) Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1311–1317.

Detry R, Papon J and Matthies LH (2017) Task-oriented grasping with semantic and geometric scene understanding. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3266–3273.

Do TT, Nguyen A, Reid I, Caldwell DG and Tsagarakis NG (2017) Affordancenet: An end-to-end deep learning approach for object affordance detection. *arXiv preprint arXiv:1709.07326*.

Dogar M and Srinivasa S (2011) A framework for push-grasping in clutter. In: *Robotics: Science and Systems*.

Eitel A, Hauff N and Burgard W (2017) Learning to singulate objects using a push proposal network. *arXiv preprint arXiv:1707.08101*.

Fang K, Bai Y, Hinterstoisser S, Savarese S and Kalakrishnan M (2018a) Multi-task domain adaptation for deep learning of instance grasping from simulation. In: *IEEE International Conference on Robotics and Automation*.

Fang K, Zhu Y, Garg A, et al. (2018b) Learning task-oriented grasping for tool manipulation from simulated self-supervision. In: *Robotics: Science and Systems*.

Ferrari C and Canny J (1992) Planning optimal grasps. In: *IEEE International Conference on Robotics and Automation*, pp. 2290–2295.

Fitzpatrick P, Metta G, Natale L, Rao S and Sandini G (2003) Learning about objects through action-initial steps towards artificial cognition. In: *IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 3140–3145.

Gibson JJ (1979) *The Ecological Approach to Visual Perception*. Houghton Mifflin.

Goldfeder C, Ciocarlie M, Dang H and Allen PK (2009) The Columbia grasp database. In: *IEEE International Conference on Robotics and Automation*, pp. 1710–1716.

Hartley R and Zisserman A (2003) *Multiple View Geometry in Computer Vision*. Cambridge: Cambridge University Press.

Haschke R, Steil JJ, Steuwer I and Ritter H (2005) Task-oriented quality measures for dextrous grasping. In: *Proceedings 2005 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2005)*, pp. 689–694.

He K, Zhang X, Ren S and Sun J (2016) Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778.

Herzog A, Pastor P, Kalakrishnan M, Righetti L, Asfour T and Schaal S (2012) Template-based learning of grasp selection. In: *2012 IEEE International Conference on Robotics and Automation*, pp. 2379–2384.

Ioffe S and Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: *International Conference on Machine Learning*, pp. 448–456.

Jain R and Inamura T (2011) Learning of tool affordances for autonomous tool manipulation. In: *2011 IEEE/SICE International Symposium on System Integration (SII)*, pp. 814–819.

Jang E, Vijaynarasimhan S, Pastor P, Ibarz J and Levine S (2017) End-to-end learning of semantic grasping. arXiv preprint arXiv:1707.01932.

Jiang Y, Lim M and Saxena A (2012) Learning object arrangements in 3D scenes using human context. arXiv preprint arXiv:1206.6462.

Kalashnikov D, Irpan A, Pastor P, et al. (2018) QT-OPT: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*.

Kappler D, Bohg J and Schaal S (2015) Leveraging big data for grasp planning. In: *IEEE International Conference on Robotics and Automation*. IEEE, pp. 4304–4311.

Katz D, Venkatraman A, Kazemi M, Bagnell JA and Stentz A (2014) Perceiving, learning, and exploiting object affordances for autonomous pile manipulation. *Autonomous Robots* 37(4): 369–382.

Kokic M, Stork JA, Haustein JA and Kragic D (2017) Affordance detection for task-specific grasping using big ol' neural nets. In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, pp. 91–98.

Komizunai S, Teppei T, Fumiya N, Nomura Y and Owa T (2008) Experiments on hammering a nail by a humanoid robot HRP-2. In: *Proceedings of the 17th CISM-IFToMM Symposium on Robot Design, Dynamics, and Control*.

Koppula HS, Gupta R and Saxena A (2013) Learning human activities and object affordances from RGB-D videos. *The International Journal of Robotics Research* 32(8): 951–970.

Kroemer O, Ugur E, Öztop E and Peters J (2012) A kernel-based approach to direct action perception. In: *2012 IEEE International Conference on Robotics and Automation*, pp. 2605–2610.

Laskey M, Chuck C, Lee J, et al. (2017) Comparing human-centric and robot-centric sampling for robot deep learning from demonstrations. In: *IEEE International Conference on Robotics and Automation*, pp. 358–365.

Lenz I, Lee H and Saxena A (2015) Deep learning for detecting robotic grasps. *The International Journal of Robotics Research* 34(4–5): 705–724.

Levine S, Pastor P, Krizhevsky A, Ibarz J and Quillen D (2016) 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.

Li Z and Sastry SS (1988) Task-oriented optimal grasping by multifingered robot hands. *IEEE Journal on Robotics and Automation* 4(1): 32–44.

Lynch KM and Mason MT (1996) Stable pushing: Mechanics, controllability, and planning. *The International Journal of Robotics Research* 15(6): 533–556.

Mahler J, Liang J, Niyaz S, et al. (2017) Dex-Net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*.

Mahler J, Pokorny FT, Hou B, et al. (2016) Dex-Net 1.0: A cloud-based network of 3D objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In: *IEEE International Conference on Robotics and Automation*, pp. 1957–1964.

Mamou K and Ghorbel F (2009) A simple and efficient approach for 3D mesh approximate convex decomposition. In: *2009 16th IEEE International Conference on Image Processing*.

Mar T, Tikhanoff V, Metta G and Natale L (2015) Self-supervised learning of grasp dependent tool affordances on the iCub humanoid robot. In: *IEEE International Conference on Robotics and Automation*, pp. 3200–3206.

Mar T, Tikhanoff V, Metta G and Natale L (2017) Self-supervised learning of tool affordances from 3D tool representation through parallel som mapping. In: *IEEE International Conference on Robotics and Automation*. IEEE, pp. 894–901.

Meriçli T, Veloso M and Akn HL (2015) Push-manipulation of complex passive mobile objects using experimentally acquired motion models. *Autonomous Robots* 38(3): 317–329.

Miller AT, Knoop S, Christensen HI and Allen PK (2003) Automatic grasp planning using shape primitives. In: *IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 1824–1829.

Morrison D, Tow AW, McTaggart M, et al. (2018) Cartman: The low-cost Cartesian manipulator that won the Amazon robotics challenge. In: *2018 IEEE International Conference on Robotics and Automation*, pp. 7757–7764.

Osiurak F, Jarry C and Le Gall D (2010) Grasping the affordances, understanding the reasoning: toward a dialectical theory of human tool use. *Psychological Review* 117(2): 517.

Pinto L and Gupta A (2016) Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In: *IEEE International Conference on Robotics and Automation*, pp. 3406–3413.

Prats M, Sanz PJ and Del Pobil AP (2007) Task-oriented grasping using hand preshapes and task frames. In: *IEEE International Conference on Robotics and Automation*. IEEE, pp. 1794–1799.

Rodriguez A, Mason MT and Ferry S (2012) From caging to grasping. *The International Journal of Robotics Research* 31(7): 886–900.

Rubinstein RY and Kroese DP (2004) The cross-entropy method: A unified approach to monte carlo simulation, randomized optimization and machine learning. In: *Information Science & Statistics*. New York: Springer Verlag.

Saxena A, Driemeyer J and Ng AY (2008) Robotic grasping of novel objects using vision. *The International Journal of Robotics Research* 27(2): 157–173.

Song D, Huebner K, Kyrki V and Kragic D (2010) Learning task constraints for robot grasping using graphical models. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1579–1585.

Stoytchev A (2005) Behavior-grounded representation of tool affordances. In: *IEEE International Conference on Robotics and Automation*, pp. 3060–3065.

ten Pas A, Gualtieri M, Saenko K and Platt R (2017) Grasp pose detection in point clouds. *The International Journal of Robotics Research* 36: 1455–1473.

Tobin J, Zaremba W and Abbeel P (2017) Domain randomization and generative models for robotic grasping. *CoRR* abs/1710.06425.

Varadarajan KM and Vincze M (2012) Afrob: The affordance network ontology for robots. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1343–1350.

Viereck U, Pas At, Saenko K and Platt R (2017) Learning a visuomotor controller for real world robotic grasping using easily simulated depth images. *arXiv preprint arXiv:1706.04652*.

Weisz J and Allen PK (2012) Pose error robust grasping from contact wrench space metrics. In: *IEEE International Conference on Robotics and Automation*, pp. 557–562.

Williamson MM (1999) *Robot arm control exploiting natural dynamics*. PhD Thesis, Massachusetts Institute of Technology.

Yan X, Hsu J, Khansari M, et al. (2018) Learning 6-DOF grasping interaction via deep geometry-aware 3D representations. In: *2018 IEEE International Conference on Robotics and Automation*. IEEE, pp. 1–9.

Zhu Y, Fathi A and Fei-Fei L (2014) Reasoning about object affordances in a knowledge base representation. In: *European Conference on Computer Vision*.

Zhu Y, Zhao Y and Chun Zhu S (2015) Understanding tools: Task-oriented object modeling, learning and recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2855–2864.