IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Self-supervised Learning for Precise Pick-and-place without Object Model

Lars Berscheid, Pascal Meißner, and Torsten Kröger

***Abstract*— Flexible pick-and-place is a fundamental yet challenging task within robotics, in particular due to the need of an object model for a simple target pose definition. In this work, the robot instead learns to pick-and-place objects using planar manipulation according to a single, demonstrated goal state. Our primary contribution lies within combining robot learning of primitives, commonly estimated by fully-convolutional neural networks, with one-shot imitation learning. Therefore, we define the place reward as a contrastive loss between real-world measurements and a task-specific noise distribution. Furthermore, we design our system to learn in a self-supervised manner, enabling real-world experiments with up to $25\,000$ pick-and-place actions. Then, our robot is able to place trained objects with an average placement error of $(2.7 \pm 0.2)\,\mathrm{mm}$ and $(2.6 \pm 0.8)\,^\circ$. As our approach does not require an object model, the robot is able to generalize to unknown objects while keeping a precision of $(5.9 \pm 1.1)\,\mathrm{mm}$ and $(4.1 \pm 1.2)\,^\circ$. We further show a range of emerging behaviors: The robot naturally learns to select the correct object in the presence of multiple object types, precisely inserts objects within a peg game, picks screws out of dense clutter, and infers multiple pick-and-place actions from a single goal state.**

I. Introduction

The task of grasping and placing an object with desired accuracy is essential for robotic object handling in general, and even more so for today's industrial and logistics automation [1]. The ideal solution to this so-called *pick-and-place* task needs to fulfill a wide range of requirements: First, for greatest flexibility the approach should even work for unknown objects without model. Second, it needs to ensure high reliability for picking objects out of dense clutter or an obstacle-rich environment like a randomly filled bin. Third and important for real-world applications, the computation needs to be as fast as possible, all while keeping the desired placing precision.

Robot learning has shown significant progress in recent years, enabling skills like grasping of unknown objects or pre-grasping manipulation. For many real-world use cases, these approaches have improved the flexibility and reliability of grasping significantly. However, transferring these approaches from grasping alone to pick-and-place yields two essential challenges: First, grasping and placing are interdependent; influencing and limiting each other in cluttered scenarios or for high-accuracy requirements. Second, how does the robot know where to place a never-seen object? We will address this question by using a single demonstration as a goal state, leading to an approach known as *one-shot imitation learning*.

The authors are with the Intelligent Process Automation and Robotics Lab (IPR), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany {lars.berscheid, pascal.meissner, torsten}@kit.edu

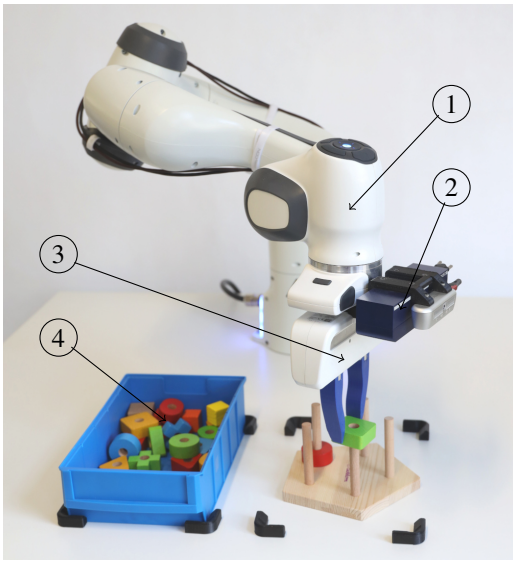


Fig. 1. Our robot is able to grasp objects out of clutter and place them with high precision, given a (demonstrated) goal state. The system has learned in a self-supervised manner with minimal human interaction in the real-world and does not depend on an object model, enabling the robot to generalize to unknown objects. We use a robot arm (1), depth cameras (2), planar manipulation with a two-finger gripper (3) and bins filled with various objects (4).

In this work, we emphasize the precision of pick-and-place and define our desired accuracy in the order of millimeters. In contrast, if the object sizes exceed the desired precision of the task, we found the term *pick-and-place* to be used as well [2], [3]. Then, the interdependence between grasping and placing can be neglected, simplifying the problem tremendously. Moreover, we restrict ourselves to the following constraints: First, we limit the robot to planar manipulation. Second, we separate the grasping and placing scene spatially, allowing for more industrial use-cases. Third, we consider multiple object types in the grasping scene. The robot then needs to select and grasp the demonstrated object.

We see our contributions as follows: First, we extend our approach of learning using manipulation primitives and fully-convolutional neural networks (FCNNs) for reward estimation to pick-and-place actions. Second and most importantly, we combine this approach with one-shot imitation learning for defining the goal pose flexibly. Third, we design a self-supervised learning strategy without human interaction, enabling the real-world training to scale to over 120 robot hours. Finally, we evaluate our system for pick-and-place precision and selection accuracy quantitatively, and demon-

strate its limits in a range of tasks qualitatively. This includes placing screws out of dense clutter in an industrial use case, as well as inferring multiple actions from a single demonstration.

II. Related Work

Object handling, and in particular grasping as a first interaction for further manipulation, are well-researched areas within robotics. Regarding grasping, Bohg et al. [4] differentiate between analytical and data-driven approaches. Historically, grasps were synthesized commonly based on analytical constructions of force-closure [5]. In comparison, data-driven approaches sample grasps using object recognition, pose estimation or specific feature extraction [4], [6]. For known objects, pick-and-place reduces to estimating the object's pose, a grasp point, *and* the following pose displacement during the grasping process. In the following however, we will focus on the case of manipulation without object model.

In recent years, manipulation has seen great progress as a key area in robot learning. From our point of view, two fundamental approaches have emerged: First, an *end-to-end* approach using a step-wise, velocity-like control of the end effector [7]–[9]. Second, with the usage of predefined *manipulation primitives* a controller needs to decide where to apply which primitive. This is often combined with a FCNN, estimating the reward for a discrete set of poses in the image space. Grasping can then be learned in simulation like DexNet [10], with real-world interaction [11], or including pre-grasping manipulation for dense clutter [12], [13].

Task-based grasping investigates the interdependence between grasps and the subsequent task [14]–[16]. Recently, the TossingBot [17] has learned to pick and throw objects into target bins, extending prior work of primitive-based grasping. However, future toss actions do not influence prior grasps.

Regarding pick-and-place itself, Zeng et al. [2] used classical image matching to pick-and-place in the broader sense of *semantic grasping*, without further requirements for a precise place pose. While the robot was able to grasp objects out of clutter reliably, grasping was trained in a human-annotated manner. Gualtieri et al. [18] learned pick-and-place of 6-degrees of freedom (DoF) in simulation and were able to transfer the results to real-world cluttered scenes. However, the robot is limited in generalizing to unknown object classes and novel place poses. Besides grasping, some work has focused on individual subproblems of pick-and-place. Jiang et al. [19] have focused on finding a stable place pose given a grasping configuration. Zhao et al. [20] calculated object displacements during the grasping action without an object model.

Pick-and-place is a popular task within imitation and reinforcement learning (RL). Finn et al. [3] used meta-learning on multiple pick-and-place tasks to achieve one-shot imitation learning for novel objects and scenarios. Their accuracy suffices for placing objects within a larger box or bowl, however their grasping approach seems restricted to non-cluttered scenarios. Duan et al. [7] used one-shot imitation learning for the task of block stacking, including training in simulation, virtual reality, and a final sim-to-real transfer. Singh et al. [21] has learned robot manipulation in the context of Inverse RL. While their bookshelf scenario is quite simplified in the context of pick-and-place, their method allows for an easy and flexible definition of a goal state without explicit reward definition.

III. Learning for Pick-and-place

We introduce our system using the notation of RL, however limited to a single action step. Then, the underlying Markov decision process (MDP) is defined by its tuple $(\mathcal{S}, \mathcal{A}, r)$ with the state space $\mathcal{S}$, the action space $\mathcal{A}$, and the reward function r. Furthermore, we will specify the action space (III-A), the state space observations (III-B), and the learned reward function (III-C) in detail. The solution to the MDP is a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ mapping the current state $s_t \in \mathcal{S}$ to an action $a_t \in \mathcal{A}$.

A. Manipulation Primitives

A pick-and-place action $a \in \mathcal{A}_g \times \mathcal{A}_p$ is a *grasp* $a_g \in \mathcal{A}_g$ following a *place* action $a_p \in \mathcal{A}_p$. We limit both action types to a set of manipulation primitives, given by predefined motions at a specified pose. Due to planar manipulation, each action space $\mathcal{A}_g$, $\mathcal{A}_p$ is given by $\mathrm{SE}(2) \times \mathbb{R} \times \mathbb{N}$ with actions parametrized by a tuple (x, y, θ, z, i) with the planar translation (x, y) parallel to the table surface, the 2D rotation θ around the z-axis and the index of the manipulation primitive i. The height z is calculated directly from the depth image. In general, the controller needs to decide where to apply which manipulation primitive i.

We define four *grasp* primitives, which differ in the gripper opening as a pre-shaped gripper width. The robot approaches the given manipulation pose (or grasp point in this case) from above along the z-axis. If a collision is detected by its internal force sensors, the robot retracts a few millimeters. Then, the robot closes its fingers via force-control. A grasp success is measured if the closed gripper width is larger than zero in a given height above the bin. A single *place* action opens the gripper at a given pose using the same approach trajectory.

While both learning to grasp from a dense reward and finding missing objects in the goal state is comparably easy, the challenge of precise pick-and-place is to find corresponding grasp and place actions. In this regard, we propose an approach to efficiently handle the Cartesian product $\mathcal{A}_g \times \mathcal{A}_p$ of both action spaces.

B. Visual State Space

Given the state space $\mathcal{S}$, let s be a set of images in orthographic projection. Each image is either a depth or a RGBD image, depending on the chosen camera configuration. Due to the orthographic projection, each translation or rotation in the image space corresponds to a planar transformation (x, y, θ) of the robot in the task space. To observe the entire bin without occlusion, we use top-down views of the scene.

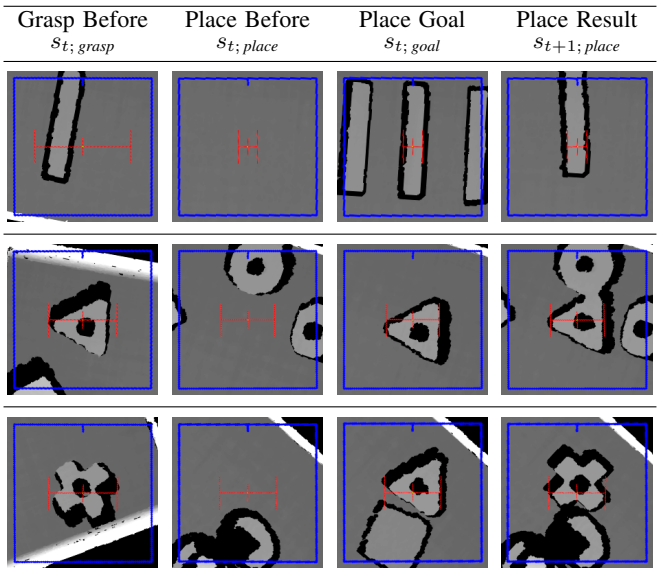


Fig. 2. Dataset samples from successful pick-and-place actions. Our approach uses four relevant observations, in particular the shown windows of the scene around the robot's tool center point (TCP). The robot grasps an object out of the *grasp before* image, and places it into the *place before* image with a given *place goal* in mind. The subsequent *place result* is only required for training. In a simplified manner, the robot should choose pick-and-place actions so that it cannot differentiate between the *place goal* and *place result* image.

Let $s' \subset s$ be an image window around the tool center point (TCP) of the robot, which is defined by the tip of the closed gripper. As common in robot learning for manipulation, we train a FCNN to estimate the action-value function Q_a from an action given the corresponding image window s'. During inference, we use the same FCNN to estimate Q_a at a grid of poses efficiently. This corresponds to a pixel-wise sliding-window approach for (x, y), while θ is calculated by pre-rotating the image inputs. The window's side length is roughly equal to the maximal object size. In terms of RL, the policy $\pi(s) = \omega \circ Q_a(s, a)$ can be defined by a so-called selection function ω composed with the action-value function Q_a. We introduce different selection strategies ω for the training and inference phase in (III-D).

We consider the grasp and place actions to be in different scenes. Importantly, all images of the same scene are taken using the same camera pose. Then, we define the state space $\mathcal{S}$ by the set s of following three observations:

1) $s_{\texttt{grasp}}$, state of the grasp scene before the grasp a_g,
2) $s_{\texttt{place}}$, state of the place scene before the place a_p, and
3) $s_{\texttt{goal}}$, goal state of the place scene. Usually, this is an image of a scene configured by a human demonstrator.

Furthermore, the state of the place scene after the place a_p is a fourth important observation. It is denoted as $s_{t+1;\,\texttt{place}}$ and can be understood as the *result* of a pick-and-place action. It is only available and required for training. Examples of the four relevant observations are shown in Fig. 2.

C. Learning from Goal States

Since the robot should imitate the given goal state, the reward r needs to capture the similarity between $s_{\texttt{goal}}$ and the expected $s_{t+1;\,\texttt{place}}$, conditioned on both before states $s_{\texttt{grasp}}$ and $s_{\texttt{place}}$. In general, the robot should choose those actions that most probably result in the *place goal* image as the real *place result* image. We interpret this as a task of *density estimation* and apply noise contrastive estimation (NCE) [22] to this unsupervised learning problem. Given a dataset D of pick-and-place actions a, a model is trained to discriminate between the real data D (with label $C = 1$) and a noise distribution Q (with $C = 0$). Then, a classifier is able to measure the probability ξ of an image s' being a realistic outcome of a pick-and-place action:

$$\xi(s') := p(C = 1 | s', s'_{\texttt{grasp}}, s'_{\texttt{place}})$$

We refer to this probability as the *place reward* $\xi \in [0, 1]$. Besides, the estimated *grasp reward* $\psi \in [0, 1]$ of the binary grasp success can be interpreted like a grasp probability.

For NCE, the design of the noise distribution Q is crucial for estimating the density of the data D. For each sample in D with a given grasping image $s'_{\texttt{grasp}}$, we augment the remaining pair of place images. For the data distribution ($C = 1$) in a pick-and-place task, we generate two positive samples:

Hindsight Foremost, the real measured pair of $\left(s'_{\texttt{place}}, s'_{t+1;\,\texttt{place}}\right)$ is the basic positive sample in the data distribution.

Further Hindsight If further objects are placed into the same bin for $t_{\texttt{bin}}$ steps after the current pick-and-place action, the images $\left(s'_{\texttt{place}}, s'_{t+n;\,\texttt{place}}\right)$ for $1 \leq n \leq t_{\texttt{bin}}$ are used as positive samples as well. This sample enables the robot to consider future actions when multiple pick-and-place actions are needed to achieve a goal state, e.g. for box stacking.

For the noise distribution ($C = 0$), we generate a range of negative samples as follows:

Negative Foresight We create negative pairs using either identical images (for both $s'_{\texttt{place}}$ and $s'_{t+1;\,\texttt{place}}$) or positive pairs in the wrong temporal order $\left(s'_{t+1;\,\texttt{place}}, s'_{\texttt{place}}\right)$.

Augmented Hindsight Moreover, we generate additional negative samples by jiggling the pose of the place images s'. In particular, we jiggle real hindsight images with a minimum displacement of $1\,\text{mm}$ or $3\,°$.

Goal The goal image pair $\left(s'_{\texttt{place}}, s'_{\texttt{goal}}\right)$ is used as a negative sample. As the training progresses and accuracy improves, both images should converge and lead to a median contrastive loss. To circumvent this effect, we apply methods of *confident learning*: If a trained classifier predicts a goal image sample to be from the real data distribution with given certainty, we remove the goal image from the noise distribution furthermore.

Other Hindsight Finally, place images of independent actions a_g and a_p are used. In particular, this results in mismatching object types as negative samples.

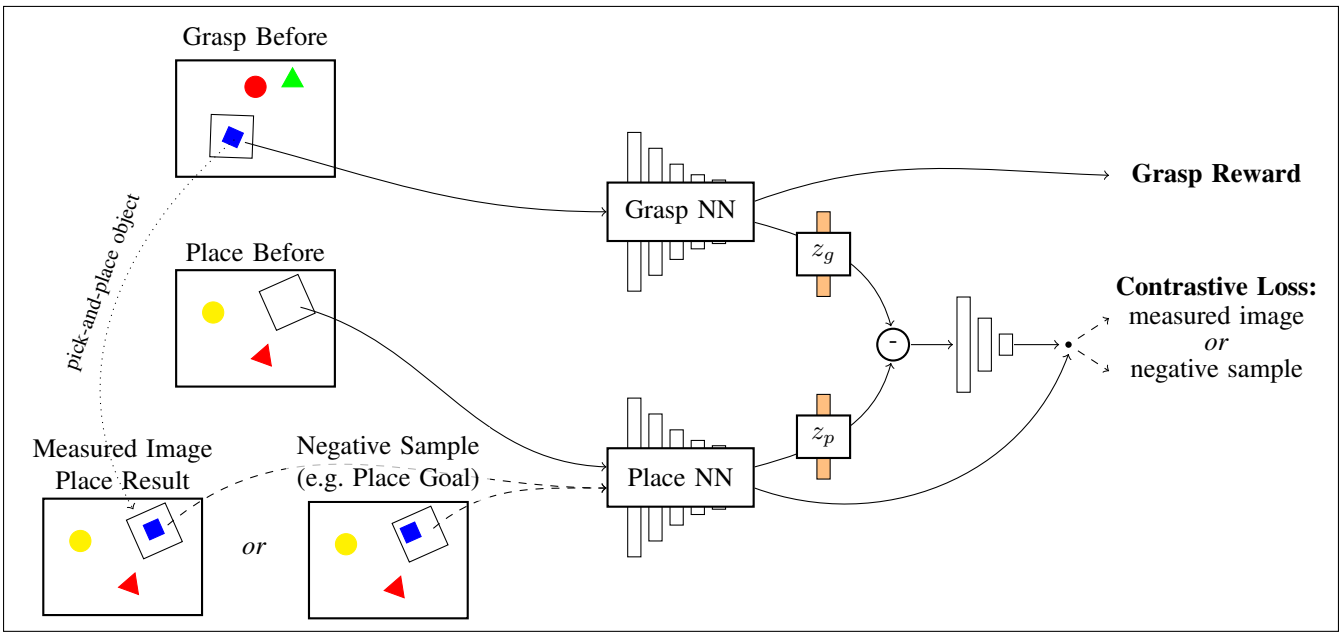


Fig. 3. Our neural network (NN) architecture during training: Given the images of both the grasping and placing scene, a system of three NNs predicts whether a given third image is a real measured image based on an executed action or a negative sample such as the goal image. While both the *Grasp NN* and the *Place NN* are limited to their corresponding scene, a third NN predicts a combined contrastive loss via the difference between the z_g and z_p embeddings. We interpret this contrastive loss as a *place reward*. Additionally, the fully-convolutional Grasp and Place NN predict the grasp success and the place reward as an initial estimation, respectively.

Using NCE, the imitation learning problem simplifies to ordinary supervised learning. The contrastive loss is defined using the binary crossentropy (BCE)

$$y_i = z_i - \log p(i)$$
$$\text{BCE} = \sum_i^N C_i \log \sigma(y_i) - (1 - C_i) \log(1 - \sigma(y_i)) \quad (1)$$

with the logits z_i adapted by the probability p of the sample i, and the sigmoid function σ. We train a system of three NNs by minimizing the BCE of the contrastive loss (eq. 1) as well as the binary grasp reward. However, three constraints are applied to our NN architecture (Fig. 3): First, we define two separate Grasp and Place NNs as FCNNs and limit their input to their corresponding scene. Second, the Grasp NN predicts the grasp reward as an additional output. Similarly, the Place NN pre-estimates the place reward, however without information about the grasp action. We denote this prediction by the Place NN as ξ_p. Third, both NNs calculate action embeddings z_g or z_p respectively. They are combined in the (non-convolutional) Merge NN using their element-wise difference. Since information from both the grasp and the place scene are joined here, the place reward ξ can then be predicted with significantly improved accuracy.

D. Pick-and-place Inference

During the inference phase, we first rotate the images s of the grasp and place scene given a discrete set of rotations (Fig. 4). The image batches are fed into their corresponding FCNN, resulting in action rewards ψ and ξ_p with their corresponding embeddings z_g and z_p for a discrete set of action poses. The final action space $\mathcal{A}$ for pick-and-place scales by the number of combinations $\mathcal{A}_g \times \mathcal{A}_p$. For performance reasons, not all combinations can be evaluated in the Merge NN. Therefore, a small fraction of grasp and place actions are pre-selected using the probability distribution

$$p(a_g|\psi) = \frac{\psi^\alpha}{\sum_{\mathcal{A}_g} \psi^\alpha}, \quad p(a_p|\xi_p) = \frac{\xi_p^\alpha}{\sum_{\mathcal{A}_p} \xi_p^\alpha}, \quad \alpha > 1.$$

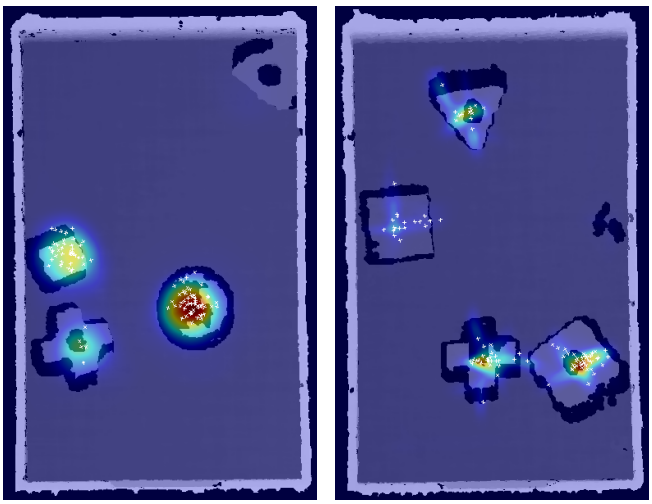
Furthermore, we set $\alpha = 6$ and sample $N := N_g = N_p = 200$ action proposals without replacement, respectively. Fig. 5 illustrates the grasp and place reward ψ and ξ_p as heatmaps, moreover indicating the position of sampled action proposals. For each combination of proposed grasp a_g and place a_p, the Merge NN predicts the refined place reward ξ from $z_g - z_p$. Since we only train the Place and Merge NN on pick-and-place actions with successful grasps, the place reward ξ is conditioned on $\psi = 1$. This way, we extend common approaches for learning *grasping*, and are able to learn the grasp confidence with additional grasp data independently. Finally, the pick-and-place reward $r = \psi \cdot \xi$ is defined by the product of grasp and place reward.

One of three selection functions ω is applied to the set of N^2 proposed pick-and-place actions: First, a uniform *random* selection for initial exploration. Second, a *sampling* based approach using $a_g, a_p \sim \psi \cdot \xi$. Third and most important, a *greedy* selection method $a_g, a_p = \arg\max \psi \cdot \xi$ is used for application. Then, given the scene images, the robot chooses the grasp and place combination that makes the scene after the executed action look most like the goal image.

Data: Images $s_{\texttt{grasp}}$, $s_{\texttt{place}}$, $s_{\texttt{goal}}$
Result: Grasp a_g, Place a_p

Create set of rotated grasp images S_g
Create set of rotated place and goal images S_p
Calculate grasp rewards ψ and embeddings z_g for each pose in S_g using Grasp FCNN
Calculate place rewards ξ_p and embeddings z_p for each pose in S_p using Place FCNN
Sample N_g grasps z_{gi} with probability $(\psi_i)^\alpha$
Sample N_p places z_{pj} with probability $(\xi_{p;j})^\alpha$
for each combination (z_{gi}, z_{pj}) **do**
    Subtract z_{gi} and z_{pj} element-wise
    Calculate final place reward ξ using Merge NN
end for
Select actions greedily via $a_g, a_p = \arg\max_{ij} \psi \cdot \xi$

Fig. 4. Algorithm of inferring a pick-and-place action using FCNNs and reward pre-estimation.



(a) Grasp reward ψ (b) Place reward ξ_p

Fig. 5. Example heatmaps of the estimated grasp and place reward, ranging from low (blue) to high (red). The estimations of each FCNN are averaged over rotations. For visualization, we reduce the number of sampled actions proposals to $N = 80$ (white dots).

E. Self-supervised Learning

In order to scale the real-world training, the learning process needs to work with minimal human interaction. Therefore, two bins are used for continuous and diverse training. We apply a simple curriculum learning strategy, starting with single objects and increasing the number and complexity of object types over time. In the beginning, we sample grasps from a given grasping policy and place objects in the second bin randomly. Later on, we sample goal states from a range of previously seen before states, denoted as the goal database. As our approach is off-policy, we train the NN using the most current dataset in parallel to the real-world data collection. Then, we sample the pick-and-place action with an *ε-sample* strategy, reducing ε over time. In the end of the training, we switch to the *ε-greedy* approach.

Optionally, the training of the grasp reward can be improved using single grasps without place action. As pick-and-place is an extension to more common bin picking pipelines, the training can then be bootstrapped by reusing prior data.

IV. Experimental Results

For our real-world experiments, a Franka Emika Panda robot including the default gripper with custom made jaws (Fig. 1) were used. Both an Ensenso N10 depth- and a RealSense D435 RGBD-camera are mounted on the flange. The system uses an Intel Core i7-8700K processor and a NVIDIA GTX 1070 Ti for computing. In front of the robot, two bins with a variety of objects for interaction are placed during training.

We crop and scale the camera images to 32×32 pixels during training and 110×110 during inference, resulting in an effective translational resolution of around $3\,\mathrm{mm}$ for place and $6\,\mathrm{mm}$ for grasp actions. However, the final pick-and-place precision may fall below this value if matching grasp and place actions are inferred. We use 37 image rotations, leading to an overall action space size of both $236\,800$ grasps (enlarged by four grasping primitives) and places, as well as 5.6×10^{10} pick-and-place actions. The embedding size of z_g and z_p is set to 48. Calculating the next action takes around $400\,\mathrm{ms}$.

Further details, the source code, more dataset samples, and supplementary videos showing our experimental results are published at `https://pantor.github.io/learning-pick-and-place/`.

A. Data Collection and Training

We evaluate our approach with two distinct models: First, a *specialized* model for pick-and-placing screws (M10×60) using RGBD images. This model was trained with around 3500 pick-and-place actions. We reuse $12\,000$ sole grasps from prior experiments for improving the grasp reward estimation of the screw model. Second, a *general* model was trained for all remaining object types and experiments. Due to reliability issues of the RealSense camera, this model uses only depth-images of the Ensenso N10. It was trained on wooden primitive shapes with side lengths of $\approx 4\,\mathrm{cm}$ (Fig. 1) for around $25\,000$ pick-and-place actions, corresponding to around $120\,\mathrm{h}$.

Both Grasp and Place NNs are fully-convolutional and share the first few layers between the reward and embedding outputs, respectively. The merge NN is a three-layer dense neural network. We double the loss weight of the final place reward and optimize the NNs using Adam with a learning rate of 2×10^{-4}. After 100 epochs, we remove goal images with a predicted contrastive loss of above 0.7 from the training set. For further details, we refer to our open-sourced implementation linked above.

B. Object Placement Error

The precision of the pick-and-place task is evaluated using the placement error of a single object. We define this error as

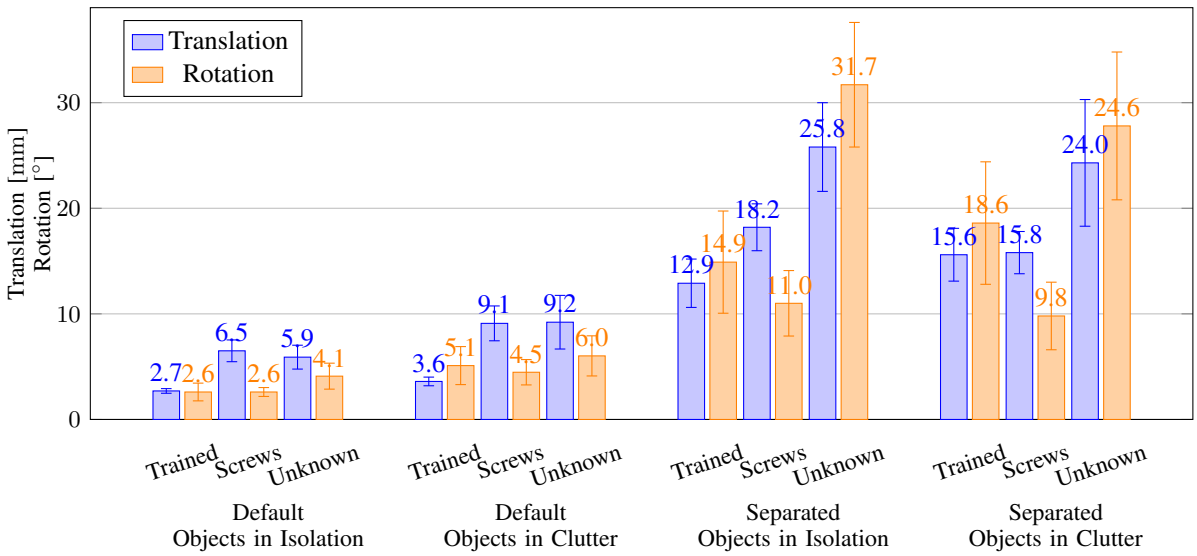


Fig. 6. The translational and rotational mean placement error in different settings. We compare the *default* case (N=200) with the *separated* case where the best grasp and best place actions are chosen independently (N=1). Here, N is the number of proposed action embeddings for further combination. Moreover, we differentiate between experimental results for grasping isolated objects and out of clutter.

the distance between the object's pose within the goal state and the pose after the executed pick-and-place action. Given a high repetition accuracy of the robot and a well-calibrated depth-camera, the goal and result images are taken from the same camera pose. Then, we measure the placement error by determining the 2D transformation bringing $s'_{\texttt{goal}}$ and $s'_{t+1;\texttt{place}}$ in alignment.

Fig. 6 shows the translational and rotational placement error in various settings. The placement error is investigated for isolated objects as well as objects in clutter. For the latter, we fill the grasping bin with 25 trained objects, 80 screws, or 10 unknown objects respectively and measure only places of the correct object type. Additionally, we compare the results of our proposed default system with a separated approach: Then, we set $N = 1$, leading to a system that chooses the best grasp action and the best place action independently of each other, and does not make use of the Merge NN.

For trained objects, our robot achieves an average placement error below of $3\,\mathrm{mm}$ and $3\,°$. Interestingly, the translational error falls just below the placing resolution. We assume the worse results for screws to be caused by less training data and a more complex, e.g. reflective, visual appearance. For unknown objects in our test set (Fig. 7), the robot is still able to achieve an average precision of around $6\,\mathrm{mm}$ and $4\,°$. In clutter, the precision decreases in particular for screws and unknown objects. Here, typical grasp rates lie around $95\,\%$ for trained objects, and around $85\,\%$ for unknown objects. Over all experiments, we find an average error of $1.6\,\mathrm{mm}$ in the direction of the gripper jaws constraining the object, and $3.9\,\mathrm{mm}$ orthogonal thereto. These findings suggest that the initial object displacement caused by the closing gripper might influence the placing precision significantly.

Fig. 7. Our evaluation set of 50 unknown objects.

C. Insertion Task

Although the robot did not learn to insert objects directly, we investigate this capability despite tolerances of around $1\,\mathrm{mm}$ using the peg game (Fig. 1), pushing the robot to its pick-and-place precision limits. On average, the robot achieves a success rate of $72\,\%$ for the insertion of isolated objects, however depending heavily on the object type (Table I). For example, we observed that predicting the displace-

TABLE I

SUCCESS RATES OF INSERTING A GIVEN OBJECT ONTO A PEG

	Default		Separated	
Object	Isolation	Clutter	Isolation	Clutter
Circle	9 / 10	7 / 10	5 / 10	1 / 10
Triangle	8 / 10	4 / 10	0 / 10	0 / 10
Square	9 / 10	7 / 10	1 / 10	0 / 10
Oval	6 / 10	6 / 10	3 / 10	0 / 10
Cross	4 / 10	2 / 10	1 / 10	0 / 10
	72 %	52 %	20 %	2 %

ment of the cross shape during clamping is difficult to do - again suggesting that the grasp displacement is a primary source of imprecision. The system is able to increase success rates by around 50 % in comparison to a separated approach. Moreover, the peg game also represents a perturbation of the environment. Success rates of up to 90 % suggest that the system is able to generalize to unknown environments. We classify a wrong object type while grasping in clutter as a failure.

D. Selection Task

Given multiple object types in the grasping scene, the task of selecting the correct object to place arises naturally. We evaluate this task by choosing five objects randomly from the set of either all known or unknown (Fig. 7) objects. Then, the robot needs to select the solely shown object from the goal state. For this 1-out-of-5 selection task, the robot achieves a success rate of (86 ± 5) % for trained, and (60 ± 7) % for unknown objects, in comparison to a random success rate of 20 % (Fig. 8). Note however, that the evaluated model

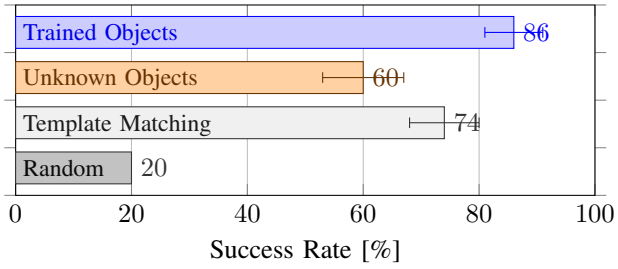


Fig. 8. Success rates of grasping the demonstrated objects out of a set of five distinct objects (1-out-of-5 selection task), independent of the final place precision. This comparison uses depth images only.

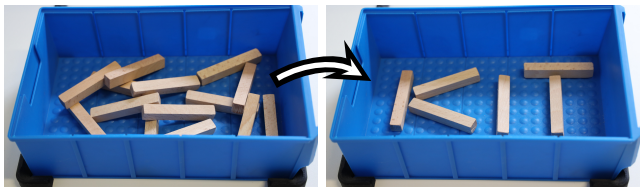
does not make use of color images, which we assume to be very helpful for this task. We additionally compare our model against a simple *template matching* baseline. First, the target object is detected using the difference between the goal and place image. Second, we apply this template to match a corresponding object within the grasp image. Although we find that this baseline is sensitive to depth shadows, it still outperforms our approach for unknown objects.

E. Multiple-step Tasks

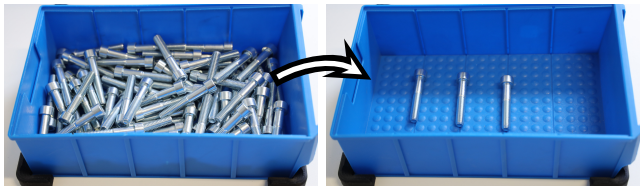
Our approach does also allow to infer multiple actions from a single goal image, while updating the grasp and place images after each action. Given a single demonstration, the robot is able to place multiple objects out of clutter, with examples shown in (Fig. 9a and 9b). Moreover, we can take a sequence of goal images as an instruction list. This allows a wide range of easy-programmed pick-and-place tasks (Fig. 9c). Videos of all three examples are included in our supplementary material linked above.

V. Discussion and Outlook

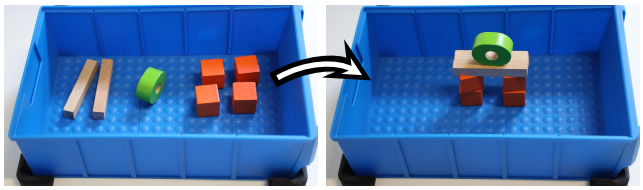
We have presented an approach for learning pick-and-place tasks in a self-supervised manner. Since our approach does not depend on an object model and instead takes a



(a) Placing the logo of our Alma Mater (KIT) (6 actions, 1 goal).



(b) Placing screws in an industrial scenario (3 actions, 1 goal).



(c) Building a house with wooden blocks (6 actions, 4 goals).

Fig. 9. Given a single goal state, our robot is able to infer *multiple* pick-and-place actions from the (cluttered) grasp scene (left) to the place scene (right). Using an instruction list of multiple goal states, the robot is able to reproduce more complex examples (c).

demonstrated goal state as an input, it allows for the flexible yet precise placing of unknown objects. The evaluated system was trained in the real-world with up to 25 000 pick-and-place actions, resulting in average placement errors of (2.7 ± 0.2) mm and (2.6 ± 0.8) ° for trained objects. A separated approach, where grasp and place actions are calculated independently, results in four to five times lower precision. For unknown objects, the translational placement error increases to around 6 mm and 9 mm for grasping in clutter. The robot learns to select the demonstrated objects out of five alternatives with up to 86 % accuracy. A second model was trained specially for screws in an industrial use case. Moreover, we demonstrated the robot's capability to pick-and-place multiple objects from a single goal state.

We see our system as a combination of two common approaches in robot learning: First, we build upon learning of manipulation primitives, estimating their reward for planar poses using FCNNs. So far, we found that this was mostly done for grasping or pre-grasping [2], [10], [17], as the reward can here be defined and measured more easily. Second, we integrate methods of one-shot imitation learning. In particular, the work of Singh et al. [21] used a contrastive loss approach to classify goal states from policy-generated, self-executed states. Similarly, we use a demonstrated goal state to define a precise object pose for placing.

Regarding other approaches to pick-and-place, we see both advantages as well as shortcomings. In comparison to Finn et al. [3], we limit our policy to a single time step and a

discrete action space. While this diminishes the generality and ignores the trajectory in between, our approach increases the final object precision significantly. Moreover, we extend a common, state-of-the-art approach of learning for grasping, i.a. resulting in the capability of pick-and-place out of clutter. Due to the use of manipulation primitives, our trained models depend only on the gripper and generalize in principle to other robotic arms.

Gualtieri et al. [18] learn a policy for pick-and-place in full 6-DoF. Despite this impressive advantage over our 4-DoF planar manipulation, we see shortcomings in the restricted generalization capability as well as a reward-based placing objective. In contrast, our approach allows for wider generalization, easy training of additional object types and flexible place poses. Additionally, our robot is limited by its data consumption of real-world training in comparison to learning in simulation.

As a part of a pick-and-place pipeline, Zhao et al. [20] predicted the object displacement during the gripper's closing action. We found that this displacement is a major source of imprecision in our experiments. Still, our overall pick-and-place precision is similar to their displacement prediction error. In comparison, our contributions allow to learn the entire pick-and-place pipeline at once. Their approach would still require a pose estimation of the object model and a grasp point detection for targeted placing. Furthermore, our approach was also evaluated in clutter.

Regarding this grasp displacement, we assume that our approach is limited by its open-loop nature: Both grasping and placing actions are planned ahead. In future work, we will explore ways to increase the precision of pick-and-place actions, for example by observing the grasped object within the gripper. Then, we can close the loop and refine the place action after picking up the object.

Moreover, we would like to extend our work to pick-and-place in the same scene, e.g. for correcting placed objects. Additional manipulation primitives might augment the robot's capability, in particular for placing objects using discrete rotations in full 6-DoF. With these ideas in mind, we hope that our research will pave the way to greater flexibility in production and industrial automation.

Acknowledgement

We would like to thank Tamim Asfour for his helpful suggestions and discussions.

References

[1] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*, ser. Springer Handbooks. Springer International Publishing, 2016, ch. 54. Industrial Robots.

[2] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo *et al.*, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in *2018 IEEE international conference on robotics and automation (ICRA)*, 2018.

[3] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, "One-shot visual imitation learning via meta-learning," in *Conference on Robot Learning*, 2017, pp. 357–368.

[4] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-Driven Grasp Synthesis - A Survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, Apr. 2014.

[5] C. Ferrari and J. Canny, "Planning optimal grasps," in *Proceedings 1992 IEEE International Conference on Robotics and Automation*. IEEE, 1992, pp. 2290–2295.

[6] A. T. Miller and P. K. Allen, "Graspit! A Versatile Simulator for Robotic Grasping," *IEEE Robotics & Automation Magazine*, vol. 11, no. 4, pp. 110–122, 2004.

[7] Y. Duan, M. Andrychowicz, B. Stadie, O. J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba, "One-shot imitation learning," in *Advances in neural information processing systems*, 2017, pp. 1087–1098.

[8] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, "Scalable deep reinforcement learning for vision-based robotic manipulation," in *Proceedings of The 2nd Conference on Robot Learning*, 2018, pp. 651–673.

[9] D. Quillen, E. Jang, O. Nachum, C. Finn, J. Ibarz, and S. Levine, "Deep reinforcement learning for vision-based robotic grasping: A simulated comparative evaluation of off-policy methods," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6284–6291.

[10] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," in *Robotics: Science and Systems (RSS)*, 2017.

[11] L. Berscheid, T. Rühr, and T. Kröger, "Improving Data Efficiency of Self-supervised Learning for Robotic Grasping," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[12] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4238–4245.

[13] L. Berscheid, P. Meißner, and T. Kröger, "Robot Learning of Shifting Objects for Grasping in Cluttered Environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2019)*, 2019.

[14] Y. Li, J. L. Fu, and N. S. Pollard, "Data-driven grasp synthesis using shape matching and task-based pruning," *IEEE Transactions on visualization and computer graphics*, vol. 13, no. 4, pp. 732–747, 2007.

[15] J. Bohg, K. Welke, B. León, M. Do, D. Song, W. Wohlkinger, M. Madry, A. Aldóma, M. Przybylski, T. Asfour *et al.*, "Task-based grasp adaptation on a humanoid robot," *IFAC Proceedings Volumes*, vol. 45, no. 22, pp. 779–786, 2012.

[16] T. Pardi, R. Stolkin, and A. M. Ghalamzan E, "Choosing grasps to enable collision-free post-grasp manipulations," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, Nov 2018, pp. 299–305.

[17] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, "TossingBot: Learning to throw arbitrary objects with residual physics," in *Robotics: Science and Systems (RSS)*, 2019.

[18] M. Gualtieri, A. t. Pas, and R. Platt, "Pick and place without geometric object models," in *Proceedings of 2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[19] Y. Jiang, M. Lim, C. Zheng, and A. Saxena, "Learning to place new objects in a scene," *International Journal of Robotics Research*, vol. 31, no. 9, pp. 1021–1043, 2012.

[20] J. Zhao, J. Liang, and O. Kroemer, "Towards precise robotic grasping by probabilistic post-grasp displacement estimation," in *Proceedings of Field and Service Robotics (FSR) 2019*, 2019.

[21] A. Singh, L. Yang, K. Hartikainen, C. Finn, and S. Levine, "End-to-end robotic reinforcement learning without reward engineering," in *Robotics: Science and Systems (RSS)*, 2019.

[22] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 297–304.