Robot Learning of Shifting Objects
for Grasping in Cluttered Environments

Lars Berscheid, Pascal Meißner, and Torsten Kröger

Abstract— Robotic grasping in cluttered environments is often infeasible due to obstacles preventing possible grasps. Then, pre-grasping manipulation like shifting or pushing an object becomes necessary. We developed an algorithm that can learn, in addition to grasping, to shift objects in such a way that their grasp probability increases. Our research contribution is threefold: First, we present an algorithm for learning the optimal pose of manipulation primitives like clamping or shifting. Second, we learn non-prehensible actions that explicitly increase the grasping probability. Making one skill (shifting) directly dependent on another (grasping) removes the need of sparse rewards, leading to more data-efficient learning. Third, we apply a real-world solution to the industrial task of bin picking, resulting in the ability to empty bins completely. The system is trained in a self-supervised manner with around $25\,000$ **grasp** and 2500 **shift actions**. Our robot is able to grasp and file objects with 274 ± 3 **picks per hour**. Furthermore, we demonstrate the system's ability to generalize to novel objects.

I. INTRODUCTION

Grasping is an essential task in robotics, as it is the key to successfully interact with the robot's environment and enables further object manipulation. The fundamental challenges of grasping are particularly visible in bin picking, the task of grasping objects out of unsystematic environments like a randomly filled bin. It emphasizes challenges as partially hidden objects and an obstacle-rich environment. Furthermore, bin picking is of enormous significance in today's industrial and logistic automation, enabling pick and place applications or automatic assembly. To enable future robotic trends like service or domestic robotics, bin picking and therewith robotic grasping needs to be solved robustly.

In general, grasping is more complex than a single clamping action. For example, surrounding obstacles might prevent all possible grasps of a specific object. In this case, it needs to be moved first so that it can be grasped afterwards. While pre-grasping manipulations are trivial for humans, they require interactions like sliding, pushing or rolling, which are complex for robots. In the context of bin picking,
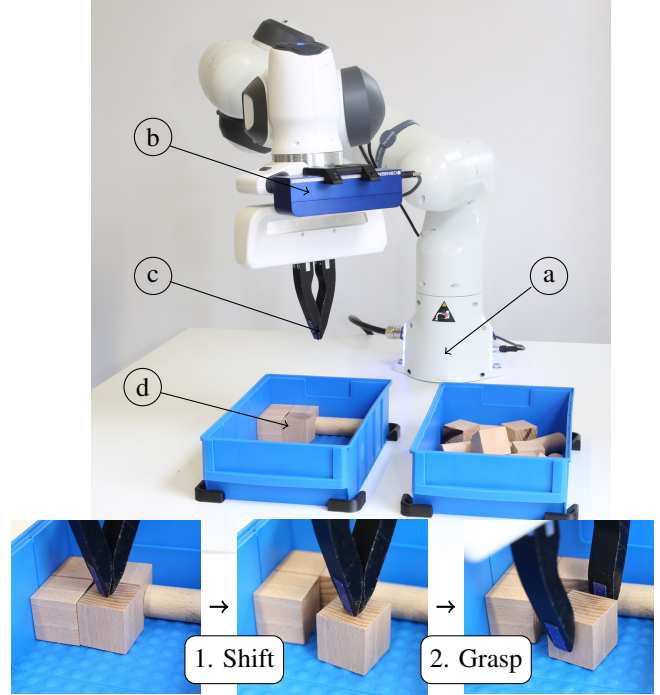
Intelligent Process Automation and Robotics Lab (IPR), Karlsruhe Institute of Technology (KIT) {lars.berscheid, pascal.meissner, torsten}@kit.edu

Fig. 1: Our setup of a Franka robotic arm including the standard force-feedback gripper (a), an Ensenso depth camera (b), custom 3D-printed gripper jaws with anti-slip tape (c), and two industrial bins with objects (d). The robot learns first grasping (2) and then shifting objects in order to explicitly increase grasp success (1).

pre-grasping is essential to empty a bin completely, since the bin itself might block grasps in its corners. Additionally, when items are stored as space-efficiently as possible, objects often prevent each other from being grasped in densely filled bins.

Our work is structured as follows: First, we present a vision-based algorithm for learning the most rewarding pose for applying object manipulation primitives. In our case, we define five primitives: Three for grasping at different gripper widths and two for shifting. Second, we use that approach to learn grasping by estimating the grasp probability at a given pose. Third, we derive both a grasping-dependent reward function as well as a training procedure for shifting. This way, sparse rewards of the grasp success can be bypassed for more data-efficient learning of shifting. Fourth, we present a robotic system (Fig. 1) which learns the industrial task of bin picking. Beyond the capabilities of the first two steps,

the system is able to empty bins completely and achieves arbitrary grasp rates at the expense of picks per hour (PPH). Furthermore, we evaluate the system's ability of grasping novel objects from non-graspable positions.

II. RELATED WORK

Object manipulation and in particular grasping are well-researched fields within robotics. Bohg et al. [1] differentiate between analytical and data-driven approaches to grasping. Historically, grasp synthesis was based on analytical constructions of force-closure grasps [2]. In comparison, data-driven approaches are defined by sampling and ranking possible grasps. Popular ranking functions include classical mechanics and model-based grasp metrics [3], [2]. As modeling grasps itself is challenging, even more complex interactions like motion planning of pre-grasping actions were studied less frequently. Within this scope, Dogar et Srinivasa [4] combined pushing and grasping into a single action, enabling them to grasp more cluttered objects from a table. Chang et al. [5] presented a method for rotating objects to find more robust grasps for transport tasks.

In recent years, the progress of machine learning in computer vision enabled robot learning based on visual input [6]. As most approaches, in particular deep learning, are limited by its data consumption, data generation becomes a fundamental challenge. Possible solutions were supposed: First, training in simulation with subsequent sim-to-real transfer showed great results for grasp quality estimation [7], [8]. However, as contact forces are difficult to simulate, training of more complex object interactions for pre-grasping manipulation might be challenging. Second, imitation learning deals with integrating expert knowledge by observing demonstrations [9]. Third, training of a robot using real-world object manipulation in a self-supervised manner showed great results for generalizing to novel objects [10]. Levine et al. [11] improved the grasp rate to 82.5%, at the cost of upscaling the training to a multitude of robots for 2 months. Data consumption of learning for robotic grasping can be minimized by utilizing space invariances and improving the data exploration strategy [12].

More recently, robot learning of manipulation skills are formulated as reinforcement learning (RL) problems. For differentiation, we find the action space either to be discrete, defined by a few motion primitives [12], [13], or continuous as a low-level control [14], [15]. While the latter allows for an end-to-end approach for general grasping strategies, it comes with the cost of very sparse rewards and a high data consumption. Kalashnikov et al. [14] trained an expensive multi-robot setup for $580\,000$ grasp attempts, resulting in an impressive grasp rate of 96% for unknown objects. Furthermore, the robots implicitly learned pre-grasping manipulation like singularization, pushing and reactive grasping. In contrast, Zeng et al. [13] introduced a pushing motion primitive and learned grasping and pushing in synergy by rewarding the sparse grasp success. Using significant less training data than [14], their robot was able to clear a table from tightly packed objects.
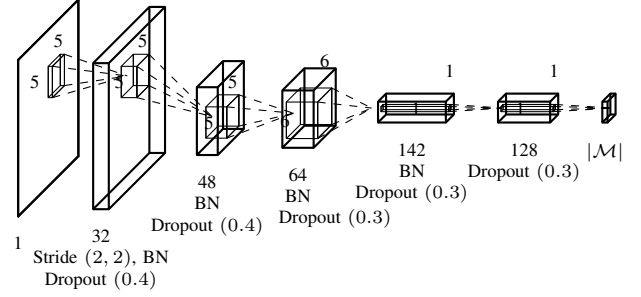

Fig. 2: Our fully-convolutional neural network (NN) architecture, making use of both batch normalization (BN) and dropout for a given motion primitive set $\mathcal{M}$.

III. SHIFT OBJECTS FOR GRASPING

Reinforcement learning (RL) provides a powerful framework for robot learning. We introduce a Markov decision process (MDP) $(\mathcal{S}, \mathcal{A}, T, r, p_0)$ with the state space $\mathcal{S}$, the action space $\mathcal{A}$, the transition distribution T, the reward function r and the initial configuration p_0. Similar to other data-driven approaches, RL is limited by its challenging data consumption, and even more so for time-dependent tasks including sparse rewards. For this reason, we reduce our process to a single time step. Then, a solution to this MDP is a policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ mapping the current state $s \in \mathcal{S}$ to an action $a \in \mathcal{A}$.

A. Spatial Learning of Object Manipulation

Given the visual state space $\mathcal{S}$, let s denote the orthographic depth image of the scene. We simplify the action space to four parameters $(x, y, a, d) \in \mathcal{A} = \mathbb{R}^3 \times \mathbb{N}$ in the planar subspace. The spatial coordinates (x, y, a) are given in the image frame, using the usual x- and y-coordinate and the rotation a around the axis z orthogonal to the image frame. While the relative transformation between the camera frame and tool center point (TCP) needs to be known, the absolute extrinsic calibration is learned. To get a full overview image of the object bin, we set the remaining angles $b = c = 0$ resulting in planar object manipulation. The fourth parameter d corresponds to the index within the discrete set of motion primitives $\mathcal{M}$.

The policy $\pi(s) = \sigma \circ Q(s, a)$ is split into an action-value function Q and a selection function σ. $Q(s, a)$ estimates the reward r for an action a given an orthographic depth image s. We introduce a sliding window $s' \subset s$ that crops the orthographic image at the given translation (x, y) and rotation (a). The x-y-translation is implemented efficiently as a fully convolutional NN, the rotation parameters by applying the NN on multiple pre-rotated images. The motion primitive set $\mathcal{M}$ is calculated by the number of output channels of the last convolutional layer. Fig. 2 shows the detailed architecture of the used NN. The training output has a size of $(1 \times 1 \times |\mathcal{M}|)$, corresponding to the size of the motion primitive set $\mathcal{M}$. During inference, the NN calculates an output of size $(40 \times 40 \times |\mathcal{M}|)$, corresponding to the (x, y, d) parameters. For the rotation a, the input image is

(a) $\hat{\psi}_w = 0.046\,\%,\ \hat{\psi}_w' = 93.2\,\%$ (b) $\hat{\psi}_w = 14.7\,\%,\ \hat{\psi}_w' = 75.9\,\%$ (c) $\hat{\psi}_w = 40.2\,\%,\ \hat{\psi}_w' = 97.7\,\%$

(d) $\hat{\psi}_w = 92.1\,\%,\ \hat{\psi}_w' = 43.9\,\%$ (e) $\hat{\psi}_w = 90.5\,\%,\ \hat{\psi}_w' = 95.8\,\%$ (f) $\hat{\psi}_w = 90.8\,\%,\ \hat{\psi}_w' = 45.8\,\%$
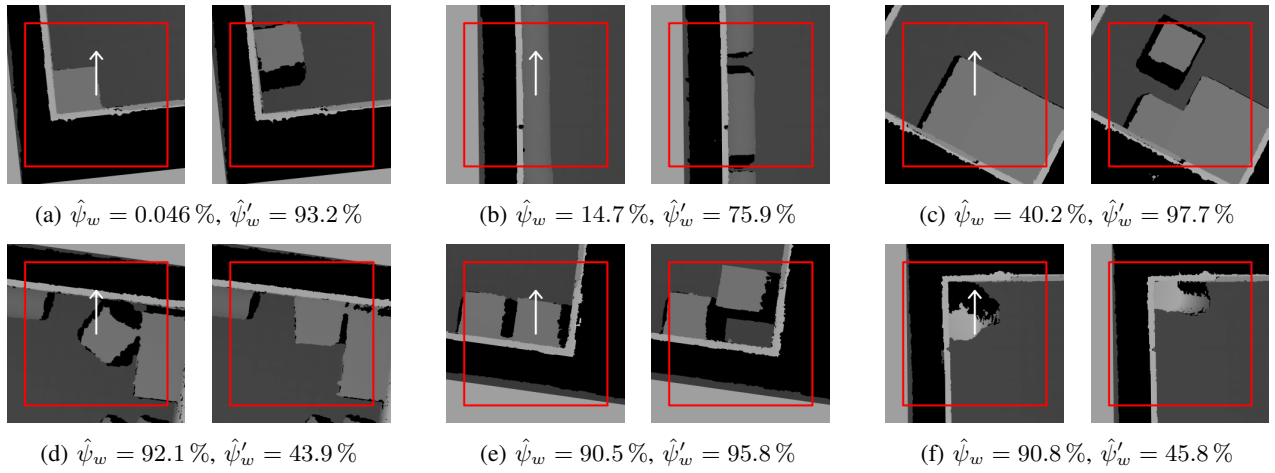
Fig. 3: Examples of depth images before (left) and after (right) an applied motion primitive. The maximal grasp probability within the red window $\hat{\psi}_w$ before and $\hat{\psi}_w'$ after are given below; their difference is then estimated by a fully-convolutional neural network.

pre-transformed and the NN is recalculated for 20 angles. This way, 32 000 reward estimations are calculated for each motion primitive. Overall, the NN approximates the action-value function Q for a discrete set of actions a within four dimensions.

The selection function σ maps the four dimensional tensor of reward predictions r to an action a. In RL, the greedy strategy using $\arg\max_a Q(s,a)$ is commonly used for sufficiently trained systems. By returning one of the N maximum elements uniformly, we integrate a stochastic component so that possible manipulation failures are not repeated. The indices of the selected element at (x,y,a) are then transformed into the camera frame using the intrinsic calibration. The height z is read trivially from the depth image at position (x,y), adding an fixed height offset for each specific motion primitive d. Finally, the algorithm outputs the pose $(x,y,z,a,b=\mathrm{const},c=\mathrm{const})$ of a defined motion primitive d with an estimated reward Q.

B. Action Space Exploration

As the design of an exploration strategy is a key to fast and data-efficient learning for robotic grasping [12], we introduce high-level strategies generalized for further object manipulation. We strictly divide between exploration (training) and exploitation (application) phase. Without prior information, the system explores the environment by sampling continuously and uniformly random poses within the hull of the action space $\mathcal{A}$. Let ε define the fraction of random samples. Due to our off-policy algorithm, we are able to combine an ε-based strategy with the following set of high-level strategies:

1) **Maximize self-information** corresponding to $\max_x -\log \tilde{P}(x)$ with the estimated probability mass function $\tilde{P}(x)$. For manipulation tasks, actions with high absolute rewards are usually more seldom. Therefore, the action with the maximum reward estimation $\max_a |Q(s,a)|$ should be chosen. This conforms with the common ε-greedy strategy. In comparison, we find that sampling corresponding to $p(a) \sim |Q(s,a)|$ yields a more extensive exploration.

2) **Minimize uncertainty of prediction** given by $\max_a \mathrm{Var}\,[Q(s,a)]$. In RL, this is usually added to the action-value function itself (for exploitation), leading to the common Upper confidence bound (UCB) algorithm. We approximate the Bayesian uncertainty of our NN using Monte-Carlo dropout for variance sampling [16].

3) **Minimize uncertainty of outcome** for binary rewards $r = \{0,1\}$ by choosing $\min_a |Q(s,a)-\frac{1}{2}|$. The system is not able to predict the outcome for those actions reliably, e.g. due to missing information or stochastic physics.

C. Learning for Grasping

A major contribution of our work is making one skill (shifting) explicitly dependent on the other (grasping). This way, we bypass the problem of sparse-rewards in time-dependent MDPs. Besides faster and more data-efficient training, this also allows to learn the skills successively.

Therefore, we can reuse successful approaches of learning for grasping [12] and focus on pre-grasping manipulation. Briefly, we define the set of motion primitives $\mathcal{M}$ as gripper clamping actions starting from three different pre-shaped gripper widths. The robot's trajectory is given by the grasp pose (x,y,z,a,b,c) and its approach vector parallel to the gripper jaws. If the robot detects a collision with its internal force-sensor, the robot retracts a few millimeters and closes the gripper. Then, the object is lifted and the robot moves to a random pose above the filing bin. Then, the grasp success is measured using the force-feedback of the gripper. We define the binary reward function

$$r_g(s) = \begin{cases} 1 & \text{if grasp and filing successful,} \\ 0 & \text{else.} \end{cases} \tag{1}$$

For binary rewards, the grasping action-value function $Q_g(s,a)$ can be interpreted as a grasp probability ψ. We train a NN mapping the image s to ψ and use it for: (1) estimating the grasp probability at a given position (x,y,a), (2) calculating the best grasp (x,y,a,d), (3) calculating the maximum grasp probability in the entire bin $\hat{\psi}$, and (4) calculating the maximum grasp probability $\hat{\psi}_w(s,x,y,a)$ in a window with a given side length centered around a given pose (x,y,a).

D. Learning for Shifting

We integrate prior knowledge about the relationship between shifting and grasping by making the reward function for shifting explicitly dependent on the grasping probability. More precisely, the system predicts the influence of a motion primitive on the maximum grasp probability $\hat{\psi}$. We train a second NN using the reward function

$$r_s(s) = \frac{1}{2}\left(\hat{\psi}_w(s',x,y,a) - \hat{\psi}_w(s,x,y,a) + 1\right) \quad (2)$$

mapping the image s to the difference of the maximum grasping probability in a window $\hat{\psi}_w(s,x,y,a)$ before s and after s' the manipulation primitive. Therefore, depth images before and after the shifting attempt are recorded and applied to the grasp probability NN. Additionally, the reward is re-normalized to $r_s \in [0,1]$. The window w is approximately $50\,\%$ larger than for the grasping action, the latter corresponding roughly to the maximum gripper width. In contrast to the grasping reward function $r_g \in \{0,1\}$, estimating shift rewards is a regression task. We further denote the estimated reward for shifting $Q_s(s,a)$ as ρ. The NN is trained optimizing the mean squared loss between the predicted and actual reward ρ and r_s.

We define two motion primitives for pre-grasping object manipulation. In both cases, the gripper closes completely and approaches the shift pose parallel to its gripper jaws. If a collision occurs, the approach motion stops. Then, the robot moves either $30\,\mathrm{mm}$ in positive x-direction or positive y-direction of the gripper frame. Those two motion primitives are distinct as the gripper is asymmetric.

Since data generation is the limiting factor in deep RL, it is important that training is self-supervised and requires as little human intervention as possible. To guarantee a continuous training, the system first estimates the overall maximum grasping probability $\hat{\psi}$. If $\hat{\psi} < 0.2$, the system tries to increase the grasping probability until $\hat{\psi} \geq 0.8$ is reached. Then, the system tries to decrease the maximum grasping probability until $\hat{\psi} < 0.2$ again. This is done by exploring the negative action-value-function $-Q_s(s,a)$ while keeping the selection function σ constant. Training started with a single object in the bin, further ones were added over time.

E. Combined Learning and Inference

For the task of bin picking, grasping and shifting needs to be combined into a single controller. Beside inference itself, combined learning also enables matching data distributions for training and application. Firstly, let ψ_g be a threshold

Fig. 4: State diagram of the combined grasping and shifting controller; common threshold parameters are $\psi_g \approx 0.75$ and $\rho_s \approx 0.6$.

probability deciding between a grasping and a shifting attempt. Secondly, let ρ_s denote a threshold between a shift attempt and the assumption of an empty bin. As shown in Fig. 4, the system first infers the maximum grasping probability $\hat{\psi}$. If $\hat{\psi}$ is higher than ψ_g the robot grasps, else it evaluates the shifting NN and estimates the maximum shifting reward $\hat{\rho}$. If $\hat{\rho}$ is larger than ρ_s the robot shifts and restart the grasp attempt, else it assumes the bin to be empty. ψ_g can be interpreted as a high-level parameter corresponding to the system's cautiousness for grasp attempts as well as its readiness to rather increase the grasp probability by shifting.

IV. EXPERIMENTAL RESULTS

Our experiments are performed on the Franka Panda robotic arm with the standard gripper as seen in Fig. 1. The Ensenso N10 stereo camera is mounted on the robot's flange. For all state observations, the camera is positioned above the bin looking down vertically, keeping transformations between robot, camera and world coordinates fixed. We designed and printed custom gripper jaws and attached common household anti-slip silicone roll to the robot's fingertips. Otherwise, the friction between the fingers and most objects would not be sufficient for shifting. The robot uses an Intel Core i7-8700K processor and two NVIDIA GeForce GTX 1070 Ti for computing. Inferring the NN takes around $10\,\mathrm{ms}$ on a single GPU, calculating the next action including image capturing takes less than $100\,\mathrm{ms}$. The source code, trained models and a supplementary video showing our experimental results are published at `https://github.com/pantor/learning-shifting-for-grasping`.

A. Data Recording

In bin picking, shifting objects becomes either necessary because of static obstacles like the bin or dynamic obstacles like other objects. To enforce the first case, we use a small bin with the size of $18\,\mathrm{cm} \times 28\,\mathrm{cm}$. The latter case is emphasized by using cubes as the most suitable shape for blocking grasps among themselves. Additionally, we train with wooden cylinders as second object primitives. For our final data set, we trained $25\,000$ grasps in around $100\,\mathrm{h}$. For learning to shift, we recorded 2500 attempts in around $9\,\mathrm{h}$.

TABLE I: Picks per hour (PPH), grasp rate, and shifts per grasp in different bin picking scenarios. In the experiment, the robot grasped n objects out of a bin with m objects without replacement. The random grasp rate was $\approx 3\,\%$.

n out of m	Picks per hour (PPH)	Grasp Rate	Shifts per Grasp	Grasp Attempts
1 out of 1	323 ± 3	$100\,\%$	$0\,02 \pm 0\,01$	100
1 out of 1 (non-graspable position)	170 ± 3	$(98.2 \pm 1.8)\,\%$	1.02 ± 0.02	56
10 out of 10	272 ± 6	$(98.4 \pm 1.1)\,\%$	0.10 ± 0.02	122
10 out of 20	299.6 ± 1.4	$100\,\%$	0	120
20 out of 20	274 ± 3	$(98.4 \pm 1.0)\,\%$	0.07 ± 0.01	122

We find that grasping and shifting need different amounts of training time; separate training allows for easy stopping at an appropriate success measure and an overall data-efficient recording. Regarding exploration strategies, we maximized *self-information* around $60\,\%$, minimized the *uncertainty of prediction* around $20\,\%$, and minimized the *uncertainty of outcome* around $5\,\%$ with decreasing ε, as well as using random actions during the first $15\,\%$ of recording. Furthermore, we generated data using the combined training approach for the last $10\,\%$ of the training time. For robust and low-noise calculations of ρ, the grasping probability ψ needs to be trained reliably for at least $15\,000$ grasp attempts.

B. Shifting Objects

The recorded shifting data set has a mean reward of 0.521 ± 0.112. The trained NN achieves a cross validated mean squared loss of 0.053, corresponding to a mean error of the grasp probability difference $|\hat{\psi}'_w - \hat{\psi}_w|$ of around $14\,\%$. Setting $\rho_s = 0.6$ is a robust threshold for perceiving empty

bins. The output of the NN can be interpreted as a heat map over the original input image. Fig. 5 shows qualitatively that the system learned to shift objects either apart or away from the bin's edge for improved grasping. We denote non-graspable positions as object poses, where at least one shift is required before grasping. As given in Table I, on average $2\,\%$ of shifts are not sufficient for grasping single objects from those positions.

C. Picks Per Hour

Shifting objects enabled the robot to empty the bin completely throughout the evaluation. For realistic scenarios like grasping 20 objects out of a bin with 20 objects without replacement, our robotic setup achieved 274 ± 3 PPH. The fewer shifts per grasp are necessary for the bin picking scenario, the higher the PPH. Using the grasp threshold parameter ψ_g, we can adapt the grasp strategy to be more cautious. Let the grasp rate be the number of grasp success over the total number of grasp attempts. As expected, Fig. 6

Fig. 5: Examples of heat maps for shifting. The NN predicts the maximum positive reward ρ at a given pose (x, y) for all rotations a and motion primitives d. The rewards are shown from low (blue, $\rho = 0.5$) to high (red, $\rho = 1$), the direction is shown (white) for the ten highest rewards.

Fig. 6: Grasp rate and picks per hour (PPH) depending on the minimum grasp probability ψ_g deciding between a grasp or shift. While the robot grasped 10 objects out of a bin with 10 objects without replacement, an optimal threshold $\psi_g \approx 0.75$ regarding PPH was measured.

shows that a higher ψ_g results in an improved grasp rate. In particular, the system is able to achieve a $100\,\%$ grasp rate already for $\psi_g > 0.8$. For this reason, the grasp rate looses its significance as the major evaluation metric. Instead, we can directly optimize the industrially important metric of picks per hour (PPH). At low ψ_g, frequent failed grasps take a lot of time. For high ψ_g, the improved grasp rate comes with the

Fig. 7: The object set for testing the system's ability to generalize to unknown objects. All objects can be placed within a bin so that they can not be grasped directly.

cost of more shifts. Since some shifts might be superfluous and worsen the PPH, an optimal threshold has to exist. Fig. 6 confirms our expectation, resulting in an optimal threshold $\psi_g \approx 0.75$. Interestingly, the corresponding grasp rate is less than 1.

D. Generalization

The ability to generalize to novel (unknown) objects is important for extending the range of applications. While training only with cylinders and cubes, we further evaluate the system on the object test set shown in Fig. 7. On average, the robot was able to grasp novel objects from non-graspable positions after 1.2 ± 0.3 shifts (Table II). Then, the robot achieved an average grasp rate of $(92.1 \pm 7.8)\,\%$. As expected, we find a qualitative relation between the similarity to the training objects and the success rate of each test object. However, the most common cause of failure is missing depth information from the stereo camera and the resulting confusion by large black regions (e.g. shown in Fig. 3). Furthermore, the shifting motion primitives are not equally suitable for each object. For example, the marker was usually shifted twice, as it did not roll far enough else. The brush did sometimes rebound back to its original position, which could be prevented by a larger shifting distance.

TABLE II: Grasp rate and shifts per grasp for novel (unknown) objects from non-graspable positions.

Object	Grasp Rate	Shifts	Grasp Attempts
Brush	77 %	1.3	10
Cardboard box	94 %	1	15
Duplo bricks	91 %	1.3	10
Folding rule	83 %	1.1	10
Marker	100 %	1.9	10
Pliers	83 %	1.2	10
Screw driver	83 %	1.2	10
Shape primitives	98 %	1	25
Table tennis balls	100 %	1.1	10
Tape	92 %	1.5	10
Tissue wrap	100 %	1	10
Toothpaste	100 %	1.1	10
	$(92 \pm 7)\,\%$	1.2 ± 0.3	140

V. DISCUSSION AND OUTLOOK

We presented a real-world solution for self-supervised learning of grasping and manipulating to improve expected grasp success. We find two implications of our work particularly interesting: First, emptying a bin completely is important for industrial applications. Second, we integrated prior knowledge into the learning algorithm by making the reward of one skill (shifting) dependent on the other (grasping). This way, we were able to bypass sparse rewards for data-efficient learning.

In contrast, both Kalashnikov et al. [14] and Zeng et al. [13] rewarded grasp success in a time-dependent manner. Additionally, we focused on bin picking scenarios of densely filled, industrial storage bins. For this task, we find multiple gripper openings, neglected by both [14], [13], inevitable. While our approach is more similar to [13], we highlight four concrete improvements: First, we integrated multiple motion primitives into a single NN. Second, our controller is able to change its readiness to assume risk. This way, our robot achieves arbitrary grasp rates, so that we can directly optimize in relation to picks per hour (PPH). Third, while we find their contribution of training grasping and pushing in synergy necessary, it is not ideal on its own. By splitting both the rewards and training procedures for grasping and shifting, we incorporate different training complexities for different skills. Fourth, our algorithm is around an order of magnitude faster, resulting in increased PPH. Regarding [14], the PPH are fundamentally restricted by their repeating inference and motion steps.

As all data-driven methods benefit from a similar training and test data distribution, generalization can always be improved by training on more diverse object and scenario sets. While we showed that our approach is able to generalize to novel objects, possible limits of this ability should be further investigated. Moreover, our algorithm requires orthographic images for NN input and therefore depth information. However, as viewing shadows or reflective surfaces limit depth availability for stereo cameras, we find the robustness against missing depth information to be a key part for further improving the robotic bin picking system.

REFERENCES

[1] 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, 2014.

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

[3] 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.

[4] M. R. Dogar and S. S. Srinivasa, "Push-grasping with dexterous hands: Mechanics and a method," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 2123–2130.

[5] L. Y. Chang, S. S. Srinivasa, and N. S. Pollard, "Planning pre-grasp manipulation for transport tasks," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 2697–2704.

[6] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel, "Deep Spatial Autoencoders for Visuomotor Learning," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 512–519.

[7] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kroger, J. Kuffner, and K. Goldberg, "Dex-Net 1.0: A cloud-based network of 3d objects for robust grasp planning using a Multi-Armed Bandit model with correlated rewards." IEEE, May 2016, pp. 1957–1964.

[8] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige *et al.*, "Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4243–4250.

[9] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal, "Online movement adaptation based on previous sensor experiences," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 365–371.

[10] L. Pinto and A. Gupta, "Supersizing Self-supervision: Learning to Grasp from 50k Tries and 700 Robot Hours," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 3406–3413.

[11] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning Hand-Eye Coordination for Robotic Grasping with Large-Scale Data Collection," in *International Symposium on Experimental Robotics*. Springer, 2016, pp. 173–184.

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

[13] 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.

[14] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke *et al.*, "Scalable deep reinforcement learning for vision-based robotic manipulation," in *Conference on Robot Learning*, 2018, pp. 651–673.

[15] 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 *International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6284–6291.

[16] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Insights and applications," in *Deep Learning Workshop, ICML*, vol. 1, 2015, p. 2.