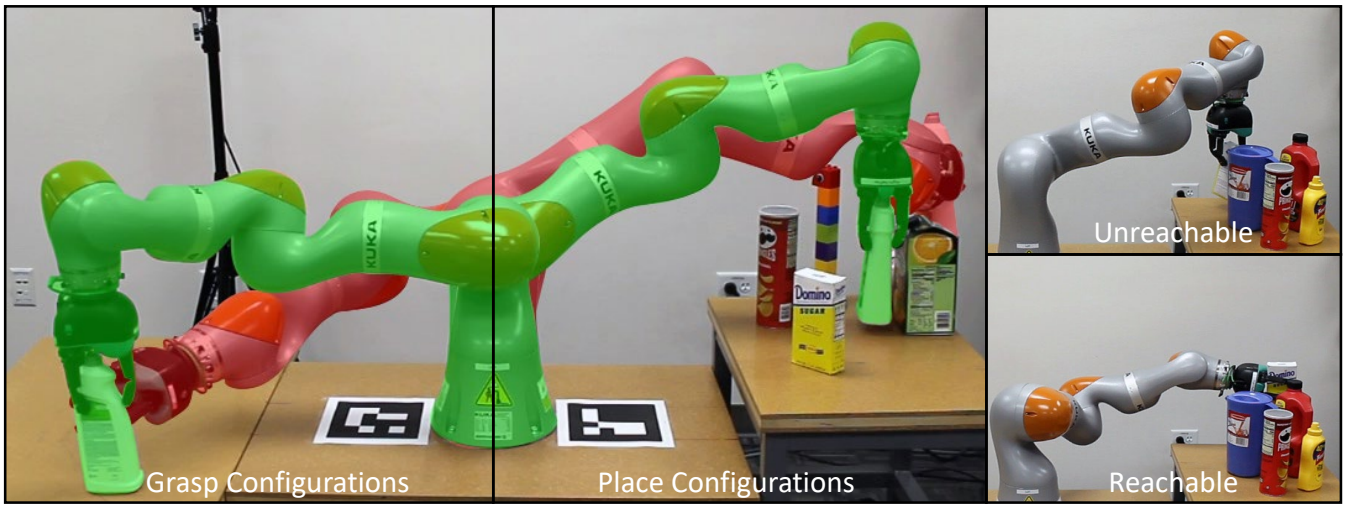


Fig. 1: Examples illustrating effects of grasp and placement configurations for placing into clutter. **Left & Middle:** Multiple grasp and place configurations for the same pick and place task. **Left:** Different successful grasp solutions for the same object (top grasp: green, side grasp: red). **Middle:** Corresponding place configurations: place for side grasp configuration (red) is infeasible due to the robot in collision, place configuration for top grasp (green) is feasible. **Right:** Reachability effects of grasp choice: **Right-Top:** Unable to reach target behind red can due to limited reachability with top grasp. **Right-Bottom** Target reached with side grasp.

***Abstract*— Robotic pick and place stands at the heart of autonomous manipulation. When conducted in cluttered or complex environments robots must jointly reason about the selected grasp and desired placement locations to ensure success. While several works have examined this joint pick-and-place problem, none have fully leveraged recent learning-based approaches for multi-fingered grasp planning. We present a modular algorithm for joint pick and place planning that can make use of state of the art grasp classifiers for planning multi-fingered grasps for novel objects from partial view point clouds. We demonstrate our joint pick and place formulation with several costs associated with different placement tasks. Experiments on pick and place tasks with cluttered scenes using a physical robot show that our joint inference method is more successful than a sequential pick then place approach, while also achieving better placement configurations.**

I. Introduction

Pick and place operations, where a robot grasps, lifts, and then safely deposits an object at a desired location, define the quintessential problem in robotic manipulation. The research literature reflects this key importance with considerable work examining grasping objects [1,2], with contemporary methods capable of grasping novel objects with high success [3–5]. Research focused on object placement, though not as extensive as grasping, investigates various aspects including stability of placements [6,7], semantic placement [8,9], and multi-object rearrangement [10,11]. Though pick and place naturally go hand-in-hand, most research investigates the two highly related sub-tasks individually.

Treating the problems independently ignores a number of important issues. In particular, while grasp success is necessary for successful placement, it is not sufficient to guarantee it. In fact, a grasp configuration might succeed in lifting an object, but could end up contributing to placement failure if the robot collides with other objects in the scene during placement as shown in Fig. (1)(Left & Middle). Likewise, if one ensures that a previously planned grasp does not collide with objects during placement, it might do so at the expense of object instability or reachability by the robot at the placement location Fig. (1)(Right). Thus causing either placement planning or execution failure.

Works that have tackled pick and place jointly, restrict themselves in some way, making simplifying assumptions not needed by modern grasp planners. These simplifications include requiring full geometry of the object and environment as meshes [12], a restricted class of known object categories [13], restricting the planner to use a fixed subset of grasps (e.g. overhead) [14], or simplified grippers [15].

In contrast, we examine the problem of joint pick and place planning given only partial view point clouds of the object and environment. This includes the case of grasping and placing previously unseen objects. Further, we plan over arbitrary grasps from the full continuous space of feasible robot configurations, as done in recent grasping work [3].

We formalize the joint pick and place task as a constrained

[1]Robotics Center, University of Utah, Salt Lake City, UT, USA; [2]NVIDIA, Seattle, WA, USA. {mohanraj.devendranshanthi, tucker.hermans}@utah.edu This work is supported by DARPA under grant N66001-19-2-4035 and by NSF Award #1846341.

optimization problem (Sec. III). Our framework enables us to jointly solve for both the optimal placement location of the given object in clutter and a corresponding grasp configuration suitable with the placement. We do so using only sensor information of the scene enabling our approach to work with novel objects. Jointly solving for the grasp and placement configurations ensures compatibility between pick and place by means of propagating gradients. We use a state of the art, grasp learning approach to encode the grasp success likelihood [3]. Like other works using neural networks for learning [2,16–19] the ability to compute gradients through the model enables efficient gradient-based planning. We detail our proposed solution in Sec. IV.

We evaluate our planner in various placement tasks including placing items in a line, tight packing of objects, and object stacking. We define associated costs for each of these tasks, highlighting the modularity of our approach. We validate our approach on a physical robot with a multi-fingered hand. Our results in Sec. V show our approach has higher placement success rate than the baselines that treat the individual pick and place planning as sequential, non-interacting problems. Along with improved pick and place success rate, our method is able to handle harder placement configurations with clutter in both pick and place scenes.

A primary limitation to our work as implemented is the assumption that the object maintains the same rigid offset to the hand as that decided in the initial plan. This could easily be incorrect as the object might move during the pick or transit phase. Our work also does not examine any visual or tactile feedback during placement to ensure gentle contact [20,21] with the environment or correct for inaccuracies in planning. Other areas for improvement including placing on sloped or non-planar surfaces. We discuss further ideas for improving on our work in Sec. VI.

We make the following contributions.

1) Present a framework for reasoning about pick and place planning jointly. The components of which could be easily swapped to achieve different tasks.
2) Provide a concrete implementation using a learned multi-fingered grasp classifier to encode grasp cost in the objective.
3) Empirically validate the ability of our framework to also pick from clutter.
4) A fast, GPU-accelerated 3D signed distance generator based on partial view point clouds, that can be easily reused and updated as the placement scene changes during sequential pick and place executions.
5) We replicate the grasp learning method of [3] on a different hand, providing further support for its effectiveness.

II. Related Work

We now describe related work in placement and joint pick and place planning. We note that the joint grasp and placement planning problem can be seen as a special case of the more general task-oriented grasping problem [22]. We only examine those task-oriented papers that specifically examine placement. The robotic object placement problem typically focuses on finding a placement pose for an object, such that it will be stable when released and potentially meet some semantic requirements [6,7,9,23].

Jiang et al were the first to examine learning for stable object placement of novel objects from partial-view point clouds [9]. Their proposed method employs hand-modeled features to learn stable and semantic placement locations for multiple objects in complex scenes and solves the associated inference problem for planning as an integer linear program. Though they have real robot demonstrations, they assume given and feasible grasps. [7] discusses a GPU based method to generate orientation and contact points for object and environment models constructed from sensors, using local optimization to validate stability. Only placement poses of the object are generated ignoring any robot constraints.

A few works have examined arbitrary reorientation of objects. Furrer et al. [6] show impressive results of stable placement configurations for stacking stones by optimizing over costs generated using a physics simulation. Newbury et al. [23] learn stable, human-preferred orientations for placing objects observed as point clouds onto flat surfaces.

The pick and place method described in [24] learns to pick novel objects in clutter using grasping primitives and drops the objects into bins according to the measured appearance. Zeng et al, then proposed Transporter Networks in [25] using spatially consistent visual representation to learn pick-conditioned placing. Gualtieri et al. [13] propose a reinforcement learning approach to learn joint pick and place policies trained separately for different object classes.

Haustein et al. propose an anytime algorithm in [12] to solve for optimal and stable placement locations given the object and the environment meshes on user provided heuristic. Though this method is not scalable to novel objects, due to the need for meshes, they propose a similar constrained optimization approach to the one we present.

Zhao et al. propose a task-oriented grasping approach in [26] solving for highly precise task-oriented grasps in $SE(2)$, by filtering sampled grasps through separate networks for predicting grasp quality, post-grasp displacement and task quality, trained using a curriculum learning approach.

Paxton et al. [8] propose a method to generate stable placement configurations satisfying semantic relationships specified as logical predicates for novel objects. They learn discriminators to predict stability as well as logical predicates and plan for placement locations through a gradient-free optimization. While this approach uses a learned grasp planner [27], grasps are selected after the placement location via rejection sampling, making it inefficient in clutter.

Berscheid et al. [14] learn embeddings from top views of 2 fingered grasps and placement images such that feasible grasp and place pairs are close to each other in the latent space. They use this learned space to jointly select grasps and placements; experiments show joint inference of grasp and placement outperforms separate inference in clutter.

Mitash et al. [28] propose a pick and place pipeline with pick, place, handoff (regrasp), and sense actions. They estimate the object geometry in order to generate plans for constrained placement of novel objects. Using a combination

of suction and two fingered grasps with simple placement scenes their results show that task oriented grasping and perception perform better than the pick then place methods.

He et al. [15] extend the object reconstruction and grasp planning approach for parallel-jaw grippers from [29], to placement-aware grasping by learning to generate affordance maps using a NeRF representation of the scene with a reconstructed object SDF. They use a sampling based approach for determining an optimal grasp and placement pair.

Our work builds on the findings that joint pick and place outperforms pick then place planning. In contrast to existing work, we propose a modular pick and place framework for use with multi-fingered grasps on novel objects, that plans over continuous grasps and placement configuration.

III. Pick and Place as Constrained Optimization

Let O be an object to be placed in a cluttered environment E, with partial-view depth images Z_O and Z_E respectively. The grasp configuration $\boldsymbol{\theta}_g = [\boldsymbol{x}_g, \boldsymbol{q}_g^h]$ is a vector including the robot palm pose $\boldsymbol{x}_g \in SE(3)$ and preshape joint angles of the gripper's fingers $\boldsymbol{q}_g^h \in \mathcal{Q}_h$. The placement configuration $\boldsymbol{x}_p \in SE(3)$ defines the 6-DOF pose of where the centroid of object point cloud Z_O should be once placed. We can then define the probability of successfully grasping the object as:

$$P\left(r_g{=}1|\boldsymbol{\theta}_g, Z_O\right) = F\left(\boldsymbol{\theta}_g; Z_O\right) \tag{1}$$

and the probability of the place configuration x_p being successful for object O in environment E as:

$$P\left(r_p{=}1|\boldsymbol{x}_p, \boldsymbol{\theta}_g, Z_O, Z_E\right) = G\left(\boldsymbol{x}_p; \boldsymbol{\theta}_g, Z_O, Z_E\right) \tag{2}$$

The joint probability for pick and place success is then:

$$\begin{aligned} P\left(r_g{=}1, r_p{=}1|\boldsymbol{\theta}_g, \boldsymbol{x}_p, Z_O, Z_E\right) \\ = F\left(\boldsymbol{\theta}_g; Z_O\right) G\left(\boldsymbol{x}_p; \boldsymbol{\theta}_g, Z_O, Z_E\right) \end{aligned} \tag{3}$$

Which we visualize as a factor graph in Fig. (2). We see that while the success probabilities are conditionally independent given the planning parameters, they do not fully decoupled, requiring joint inference over pick and place parameters.

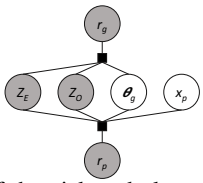


Fig. 2: Factor graph of the pick and place probability distribution. We see that while the success probabilities are conditionally independent given the planning parameters, they can not fully decouple, requiring joint inference over pick and place parameters.

We define the pick and place inference problem as finding a tuple of grasp configuration and place configurations $(\boldsymbol{\theta}_g, \boldsymbol{x}_p)$, that maximizes the joint probability defined in Eq. (3). Taking the negative log on Eq. (3), we formalize this as a constrained optimization in Eq. (4)

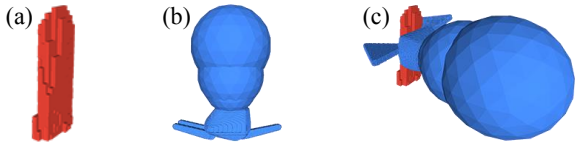


Fig. 3: Robot-object geometry for pick-and-place collision checking: a object; b robot; c union of object and robot.

$$\underset{\boldsymbol{x}_p, \boldsymbol{\theta}_g, \boldsymbol{q}_g^a, \boldsymbol{q}_p^a, \tau}{\arg\min} \; -\ln\left(G\left(\boldsymbol{x}_p; \boldsymbol{\theta}_g, Z_O^+, Z_E\right)\right) - \ln\left(F\left(\boldsymbol{\theta}_g; Z_O\right)\right) \tag{4a}$$

$$\text{subject to} \quad \boldsymbol{x}_p \in \mathcal{P} \tag{4b}$$

$$\boldsymbol{x}_g = \phi_h(\boldsymbol{q}_g^a); \quad \boldsymbol{x}_p = \phi_O(\boldsymbol{q}_p^a) \tag{4c}$$

$$\boldsymbol{q}_i^- \leq \boldsymbol{q}_i \leq \boldsymbol{q}_i^+ \quad \forall\, i \in \{g, p\} \tag{4d}$$

$$Z_O^+ = Z_O \cup R_G(\boldsymbol{\theta}_g) \tag{4e}$$

$$\epsilon \leq \texttt{SDF}\left(\boldsymbol{x}_p, Z_O^+\left(\boldsymbol{\theta}_g\right), Z_E\right) \tag{4f}$$

$$\tau(\boldsymbol{x}_p, \boldsymbol{x}_g) \in \Omega \tag{4g}$$

Equation (4a) defines the objective of the optimization as a log-linear combination of the placement success probability $G\left(\boldsymbol{x}_p; \boldsymbol{\theta}_g, Z_O, Z_E\right)$ and grasp success probability $F\left(\boldsymbol{\theta}_g; Z_O\right)$. The remaining constraints ensure physical validity for successful execution, i.e., the grasp and placement must be reachable by the robot and the objects and robot should not interpenetrate. Eq. (4b) constraints the placement configurations $\boldsymbol{x}_p$ to be within the footprint of the placement surface and above it. Depending on the task the placement configuration $\boldsymbol{x}_p$ is in $SE(3)$ or $SE(2)$. Eq. (4c) encodes the arm forward kinematics for the grasp and placement, while Eq. (4d) defines the joint limits, where the superscript, i, denotes the joints associated with the robot arm and hand.

Eq. (4e) augments the object cloud with the geometry of the hand and spheres approximating the geometry of the wrist (last 2 arm links) defined by $R_G(\boldsymbol{\theta}_g)$, at the current grasp pose. We note this is a similar procedure to that in [30]. We visualize the gripper geometry augmentation in Fig. (3). Using this we define the placement collision constraint in Eq. (4f). Finally, Eq. (4g) defines that there must be a feasible, collision-free trajectory from grasp to placement.

IV. Solving the Joint Pick and Place Problem

In this section we discuss our approach to instantiating and solving the problem defined by Eq. (4). We first discuss the details of different placement probabilities, $G(\cdot)$, which we examine in our experiments. We then briefly review the learning-based grasp method from [3] and its use as our grasp probability $F(\cdot)$. Following that we present an efficient algorithm for SDF-based collision checking built specifically for repeated placement into clutter. We conclude this section by discussing the choice of solver used and generation of grasp and place priors to perform MAP inference.

A. Placement Probabilities

Given any placement cost $H(\boldsymbol{x}_p)$ that accepts an object placement configuration $\boldsymbol{x}_p \in SE(3)$ as input and outputs a scalar value that quantifies the suitability of $\boldsymbol{x}_p$ for the task considered, with lower values being more desirable for the task and higher values being less desirable.

We can convert this $H(\boldsymbol{x}_p)$ to a probability likelil

$$G(\boldsymbol{x}_p) \propto \exp\left(-\alpha H(\boldsymbol{x}_p)\right)$$

Higher values of the α parameter make the solve placements more desirable to the task at the expense success, lower values of α prefer more confident gr

We now define the four placement likelihoods this paper. We use the notation $G(\boldsymbol{x}_p)$ to denote pl likelihood and $H(\boldsymbol{x}_p)$ to denote placement cost. A we define can be converted to a likelihood using Ec

1) Target Pose: The simplest of costs, the encodes the object to a target pose. We can define this as minimizing the squared-Euclidean distance between and target pose, giving the likelihood:

$$G_{\text{target}}(\boldsymbol{x}_p; \boldsymbol{x}_t) \propto \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_t - \boldsymbol{x}_p\right)^T \left(\boldsymbol{x}_t - \boldsymbol{x}_p\right)\right) \quad (6)$$

2) Tight Packing: The cost defined in Eq. (7) aims to place objects as close to each other as possible. This is relevant for organizing objects into shelves or boxes. We encode this cost as the area of the bounding box enclosing all objects in the scene plus the area of the bounding box enclosing the newly placed object and a reference point.

$$H_{\text{pack}}(\boldsymbol{x}_p; Z_O, Z_E) = L_E(\boldsymbol{x}_p, Z_O, Z_e) \cdot W_E(\boldsymbol{x}_p, Z_O, Z_e) \\ (1,1,0) \cdot T_2(\boldsymbol{x}_p) \cdot (L_O, W_O, 1)^T \quad (7)$$

where (L_E, W_E) define the length and width of the bounding box enclosing all objects in Z_E, including the newly placed object, (L_O, W_O) define the length and width of the object point cloud Z_O, and $T_2(\boldsymbol{x}_p)$ defines the homogeneous $SE(2)$ transformation matrix associated with pose $\boldsymbol{x}_p$, the place probability for this cost is obtained using Eq. (5). Fig. (10) shows results of planning with the tight packing cost.

3) Stacking: Allows objects to be placed on top of each other. Given the centroid x_c of the point cloud of an existing object or stack Z_E, the cost defined in Eq. (8) penalizes the height H_O and width L_O of the object at place configuration x_p. The position of the object is constrained to be close to x_c while the orientations are not constrained. This task allows for placement configurations $\boldsymbol{x}_p$ to be in $SE(3)$. Fig. (14) shows the robot stacking five blocks using this cost.

$$H_{\text{stack}}(\boldsymbol{x}_p; Z_O, Z_E) = (1,0,1) \cdot \hat{R}(\boldsymbol{x}_p) \cdot (L_O, 0, H_O)^T \quad (8)$$

where $\hat{R}(\boldsymbol{x}_p) = \hat{R}_x(\theta)\hat{R}_y(\psi)\hat{R}_z(\phi)$ and $\hat{R}_k$ defines the absolute values of the rotation matrix about object-axes k.

4) Place Inline: Places a sequence of objects in a straight line given a point on the line x_t and its angle of slope θ_l. We model this as a Gaussian about the x component of the placement configuration in Eq. (9).

$$G_{\text{inline}}(\boldsymbol{x}_p; x_t, \theta_l) = -\exp\left(-(\boldsymbol{x}_p - x_t)^T K_{\theta_l} (\boldsymbol{x}_p - x_t)\right) \quad (9)$$

where $K_{\theta_l} = R_z(\theta_l)^T K_x R_z(\theta_l)$ with $R_z(\theta_l) \in SO(2)$ and $K_x = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$, aligns the x-coordinates. Fig. (13) shows the real robot rearranging as set of cups in a straight line.

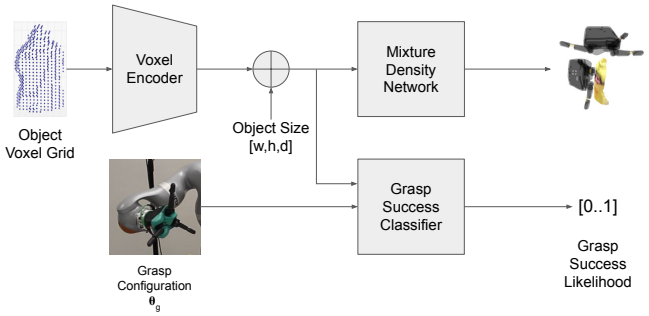


Fig. 4: Overview of our grasp prediction pipeline based on [3]. A grasp classifier predicts grasp success given an object voxel grid and grasp configuration. A mixture density network models a distribution over grasp configurations given an input voxel grid.

B. Grasp Prediction

Following the success of recent learning-based grasp planning approaches [3,31] we define our grasp cost as the negative log probability of grasp success $-\log\left(F(\boldsymbol{\theta}_g; Z_O)\right)$. However, we note that our framework could handle any differentiable grasp cost encoding.

Fig. (4) shows an overview of the grasp prediction pipeline used here. A neural network classifier defines the core of the grasp prediction model $F(\boldsymbol{\theta}_g; Z_O)$. This outputs a scalar value between 0 and 1 that represent the grasp success probability for the given grasp $\boldsymbol{\theta}_g$ on the observed object Z_O. We learn this model as a 3D convolutional neural network classifier using the approach proposed in [3]. This takes a voxel representation of the object, converted from the point cloud, as input and passes it through several 3D convolutional layers to predict grasp success. The only modification we make to the neural network structure is changing the grasp input model to accept the one-dimensional preshape configuration (the finger spread) instead of the higher-dimensional vector used for the dexterous hand in [3].

A mixture density network (MDN) comprises an additional part of the model. It takes the same voxel grid as input and generates a distribution over grasp configurations as output (c.f. Fig. (6)). We use this to initialize the solver described later in this section. For further details see [3].

C. SDF Collision Constraint Computation

To account for the collision constraints in Eq (4f), we require signed distances from the partial view points of the objects in the environment. For efficient computation we compute a discrete approximation of the SDF that we can quickly update as more objects are placed into the scene. Fig. (5) shows the steps in generating the SDF queries for the collision constraint.

Given a point cloud of the environment we convert it to a 3D voxel grid encoding the point cloud occupancy. We then interpret this occupancy grid as a discretization of the zero level set of the environment's SDF. To compute the positive signed distances associated with the surface we use a brushfire algorithm to iteratively march outward till a truncation distance from the zero level set to obtain discrete positive distances at uniform increments. Similarly the negative signed distances are obtained by marching

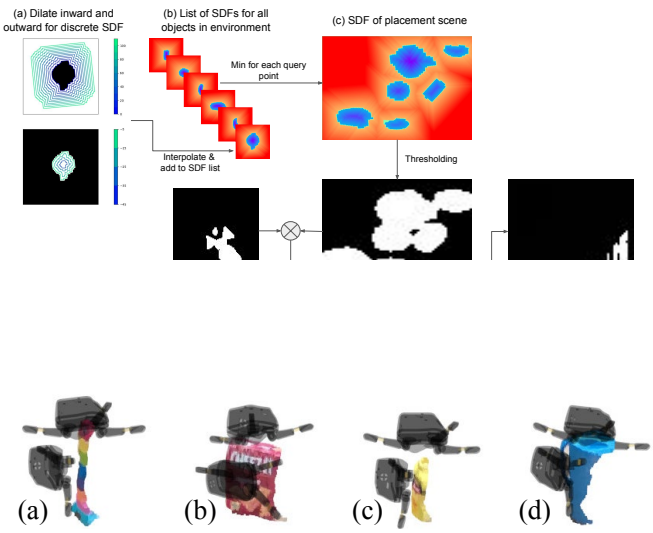


Fig. 6: Mean configurations of the MDN top and side grasp modes, visualized with the partial view point clouds of the objects (a) lego blocks, (b) cracker box, (c) mustard bottle and (d) pitcher.

inward, resulting in a truncated discretized signed distance field (DSDF). By treating the DSDF as a 3D image we apply 1D finite differentiation filters along the x, y and z-axes to compute the gradients.

Given the discrete SDF, we can query it with any continuous point with trilinear interpolation of neighboring points. To handle the collision constraints in the optimization problem Eq. (4), we obtain a uniform discretized set of points associated with the object being placed, Z_O, and the known robot geometry, $R(\boldsymbol{\theta}_g)$ denoted as $\mathbb{Z}_o$ and $\mathbb{Z}_R(\boldsymbol{\theta}_g)$ respectively, where we keep the dependence on the grasp configuration explicit. For use in the optimization we transform these points $\mathbb{Z} = \mathbb{Z}_o \cup \mathbb{Z}_R(\boldsymbol{\theta}_g)$ according to the placement pose, $\boldsymbol{x}_p$, this transformation and querying is done efficiently in parallel as defined below:

$$\texttt{SDF}\left(\boldsymbol{x}_p, \mathbb{Z}, Z_E\right) = \min_{\boldsymbol{x}\in\mathbb{Z}, \mathbb{Z}\in\{\mathbb{Z}_O, \mathbb{Z}_R\}} \texttt{DSDF}_E\left(T(\boldsymbol{x}_p)\mathbb{Z}\right) \quad (10)$$

where the min over both object and robot geometry accounts for the union operation Eq. (4e). In practice we can treat the collision constraints for the object O and the robot R separately for the same grasp and place configurations to have more informative gradients.

D. Optimization and Motion Planning

We perform MAP inference by solving the optimization problem from Eq. (4) without constraint Eq. (4g). We relax the forward kinematics Eq. (4c) and the collision SDF constraints Eq. (4f) into the objective using an Augmented Lagrangian method. We convert the constraint, $\boldsymbol{x} \in \mathcal{P}$, to bound constrain the placement configuration, $\boldsymbol{x}_p$, within the table edges. We solve the resulting bound constrained problem using BFGS [32] with projections to handle the bounds on the joint angles and placement pose. We ensure Eq. (4g) when motion planning for the arm after solving for the pick and place configurations.

The solver is initialized with grasp configurations ${\boldsymbol{\theta}_g}^0$ sampled from the MDN prior described in IV-B and shown in Fig. (6). The place prior is then obtained by convolving the 2D binary occupancy of the augmented object-robot geometry (Fig. (5e)) over the coarse 2D binary occupancy of the place scene (Fig. (5d)), which outputs collision-free place configurations (Fig. (5f)), these are then ranked by predicted place probabilities and filtered by kinematic feasibility, to obtain the initial place configuration ${\boldsymbol{x}_p}^0$.

Given the solutions of the grasp and place configurations in joint space, the full mesh models of the robot, the grasp and placement surfaces, and meshes of all objects in the scene generated from the computed SDFs , we use MoveIt!'s constrained RRT planner [33] for planning trajectories to pre-grasp pose, transfer from pick to place configurations, and pre-place pose. We constrain the task space orientation of the hand to restrict rotation of the object being during transfer to the place pose. The pre-grasp and pre-place poses are obtained by offsetting the end-effector by a small distance along the negative direction of the end-effector's x-axis. We use a task space velocity controller during grasp approach, lift, place approach, and post-placement retraction to soften interaction with the environment.

V. Experiments

We now validate the benefits of our proposed pick and place framework (i.) by benchmarking against sequential pick and place and a sampling based baseline, and (ii.) qualitative experiments demonstrating the applicability of the approach to a variety of scenarios. We experiment using a KUKA iiwa 7-DOF arm with a Reflex Takktile 2 gripper and a Realsense D455 camera for sensing, with an 8-core Ryzen 5800X CPU and a 12GB Nvidia RTX 3060 for computation.

A. Benchmarks

We benchmark the success rate of pick and place executions and the optimality of our joint optimization approach against the baselines in 2 different tasks, the following baselines are considered:

1) **The pick then place approach:** Where we solve for the best grasp configuration subject to all the constraints in Eq. 4 that apply during grasping. Then keeping this grasp configuration fixed, the placement configuration that suits the grasp is solved for. This is essentially done by solving the optimization problem described in III and IV twice with the grasp and place costs individually. This is similar to the pick-conditioned placing in [25] with grasps in $SE(3)$.

2) **Sampling:** Similar to the approach in [8,15], We develop a baseline that generates compatible grasp and placement configurations using Monte Carlo sampling. First we sample a set of grasp configurations with high success rate from the trained grasp MDN network, and a set of placement configurations not in collision with the environment for both object and robot using the generated SDF, then the generated grasp and placement configurations are refined locally for feasibility with other constraints.

1) Place to an unreachable target: The robot is tasked with picking an object in isolation and placing it on a corner of a table (placement surface) that is not reachable by the

robot. We recorded 30 executions for this task with 10 different objects in varied levels of clutter ranging from 4 – 7 objects in the placement scene. We use the target pose cost defined in Eq. (6). Since, the target pose is unreachable and may be infeasible due to other objects being in the way, the solver must find a feasible placement solution as close to the target as possible while accounting for the grasp.

We report the success rate, and predicted placement probabilities for each method. Fig. (7) shows the grasp and place success rates. We call a grasp successful if the robot lifts the object without dropping it. We call a placement successful if the robot places it on the placement surface and all objects remain upright. Cases where the solver fails to find a feasible solution are failures. For this task the joint method has an average run time of 58.21s with a standard deviation of 26.83s for varied levels of clutter, the sequential method takes on average 71.48s (37.27s std dev); sampling takes on average 39.93s (5.63s std dev) for 450 samples.

We see that the joint method significantly outperforms the baselines in terms of place success with 80% of the executions being successful, While unsurprisingly the joint and sequential methods achieve comparable grasp success rates. The sampling based baseline is not able to reliably generate safe and stable placement configurations. It was often in violation of constraints, due to the initial discretization of the placement configuration samples and lack of gradient information for refinement. The sequential baseline primarily fails due to the lack of feasible placement initializations for the fixed grasp generated. Another interesting cause of failure was the object slipping and falling during the placement trajectory more than the joint optimization method, we hypothesize this could be due to the joint method preferring more tighter grasps to avoid collisions with other objects.

Fig. (8) shows the placement likelihoods from Eq. (6) for each of the 30 executions for all 3 methods. The likelihood encodes the closeness of the solved placement configuration to the target configuration with a likelihood of 1 when the placement configuration aligns with the target and the value approaching 0 as the distance increases. We set the placement likelihoods of unsuccessful executions to 0. We see from the plot that the joint method generally produces solutions closer to the target than the baselines.. Most failures of the joint method also fail for the baselines. We observe the leading cause of failure being the object shifting during grasp and transfer. Fig. (9) shows example placement executions for each method considered in each place scene.

2) Pick from clutter: If Z_E in Eq. (4) includes the point cloud of nearby objects in the grasp scene, our framework can enable grasping in clutter. We experimentally validate this by having the robot pick up an object from clutter and place it as close as possible to another cluster of objects in the place scene as shown in Fig. (10). We use the packing cost from Eq. (7) for placement. We drop the sampling-based inference for this benchmark, as the grasp sampler failed to reliably find grasps not in contact with the grasp scene clutter.

Similar to the previous task, we report the grasp and place success rates in Fig. (11) 16 executions each for the joint and

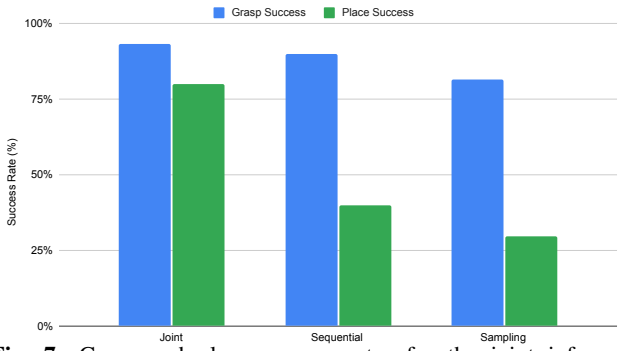


Fig. 7: Grasp and place success rates for the joint inference, sequential inference and sampling methods across 30 executions, for placing into clutter benchmark.

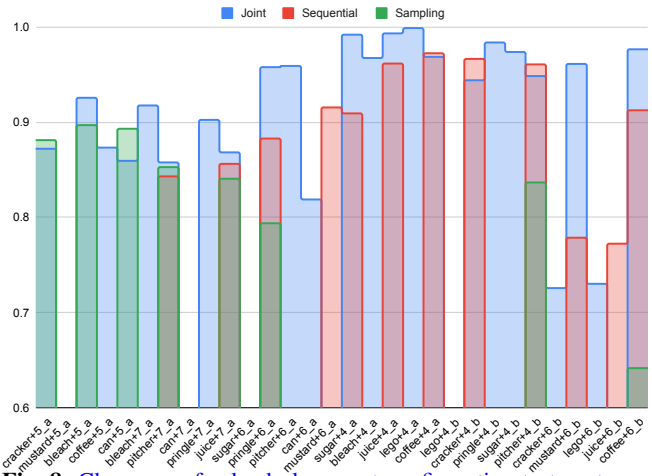
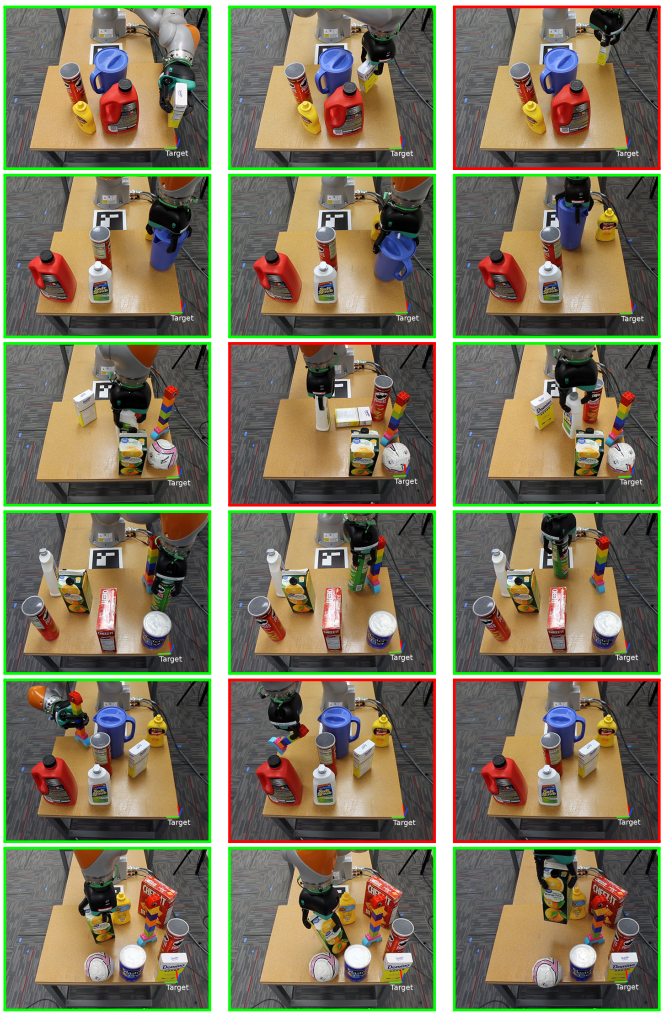


Fig. 8: Closeness of solved placement configuration to target pose as likelihood $[0, 1]$. (1 being the closest and 0 being farthest away). Likelihood of failed executions are set to zero.

sequential methods. In addition to the previous requirements for grasp success, a successful grasp must not knock over any objects in the scene during grasp and lift. Fig. (11) shows that though the grasp and place success rates drop relative to the previous experiments, joint inference still significantly outperforms the sequential baseline, with 69% of placements being successful. The drop in success rate compared to the previous benchmark can be explained by the added difficulty of grasping from clutter for both methods.

We also report the packing likelihood from Eq. (7) in Fig. (12). Here the likelihood encodes the growth in area of the bounding box enclosing all objects after placement. A likelihood of 1 denotes no growth and a likelihood of 0 denotes the bounding box has grown infinitely. We report the likelihood as 0 in cases of failed placement. Fig. (12) shows that sequential inference performs close to joint in cases with low levels of clutter in the placement scene, but outright fails with denser clutter. Hence the joint method is more capable of handling clutter in both grasp and place scenes.

For this task the joint method had an average runtime of 82.01s (28.94 std dev). The sequential method took on average 73.59s (39.74s. std dev). We attribute the lower average to the placement optimization failing in many cases.



(a) (b) (c)

Fig. 9: Example executions for each scene in the unreachable target task: **(a)** joint inference, **(b)** sequential, **(c)** sampling. The robot is shown at the placement configuration before opening its fingers. Successful executions are outlined in green and failures in red.

Fig. 10: Example of picking from clutter (left) and packing (right).

B. Qualitative Demonstrations

We demonstrate our joint pick and place inference in two sequential objects placement tasks using costs from IV-A:

1) **Place objects in line:** Using the cost from Eq. (9). We execute the solutions from our framework for the robot to rearrange a set of cups in a straight line shown in Fig. (13).
2) **Stacking in 6 DOF:** Fig. (14) show the robot stacking a sequence of blocks on top of each other, by allowing rotation in all 3-axes, each block is placed on top of another to have the minimum height possible. Thus showing our framework is capable of solving for placement configurations in $SE(3)$

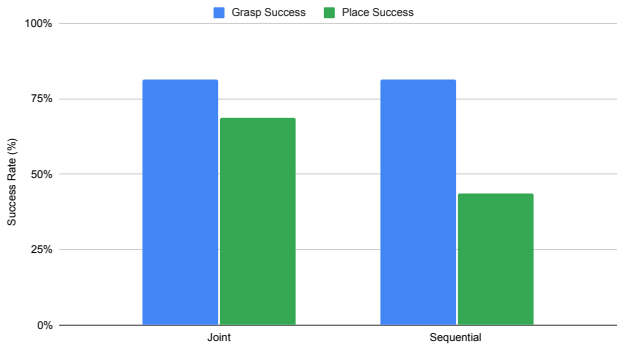


Fig. 11: Grasp and place success rates for pick from clutter and packing task for both joint and sequential inference methods.

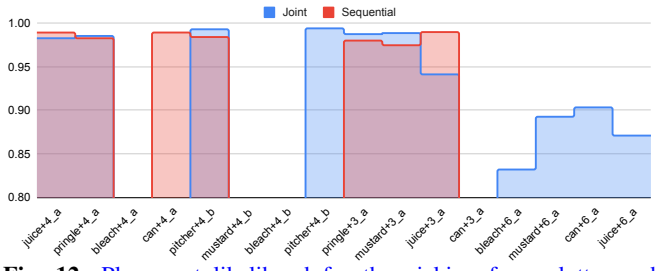


Fig. 12: Placement likelihood for the picking from clutter and packing task, for joint and sequential inference methods.

VI. Conclusion

We presented an approach for planning a grasp for picking an unknown object jointly with a downstream placement task. By formalizing this problem as a joint inference we were able to leverage both model-based geometric and learning-based costs and constraints into a single framework.

Many opportunities exist for future work. One could learn a post-grasp classifier, akin to the grasp classifier, in order to handle placement on non-planar surfaces or other downstream tasks (e.g. handover). Using visual or tactile feedback during placement could account for shifts of object pose relative to the gripper during transport.

In conclusion, our work is the first to show unified planning of a multi-fingered grasp for pick and place operations. Our results show the benefit of taking the placement location into account when planning grasps. In particular we enable higher success for placement in cluttered scenes relative to planning placements sequentially after a successful grasp. We also show that our method applies to grasping in clutter scenarios without much loss in performance.

References

[1] R. Grupen, "Planning grasp strategies for multifingered robot hands," in *IEEE Intl. Conf. on Robotics and Automation*, 1991.

[2] D. Kappler, J. Bohg, and S. Schaal, "Leveraging big data for grasp planning," in *IEEE Intl. Conf. on Robotics and Automation*, 2015.

[3] Q. Lu, M. Van der Merwe, B. Sundaralingam, and T. Hermans, "Multifingered grasp planning via inference in deep neural networks: Outperforming sampling by learning differentiable models," *IEEE Robotics Automation Magazine*, vol. 27, no. 2, 2020.

[4] D. Morrison, J. Leitner, and P. Corke, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," in *Robotics: Science and Systems*, 2018.

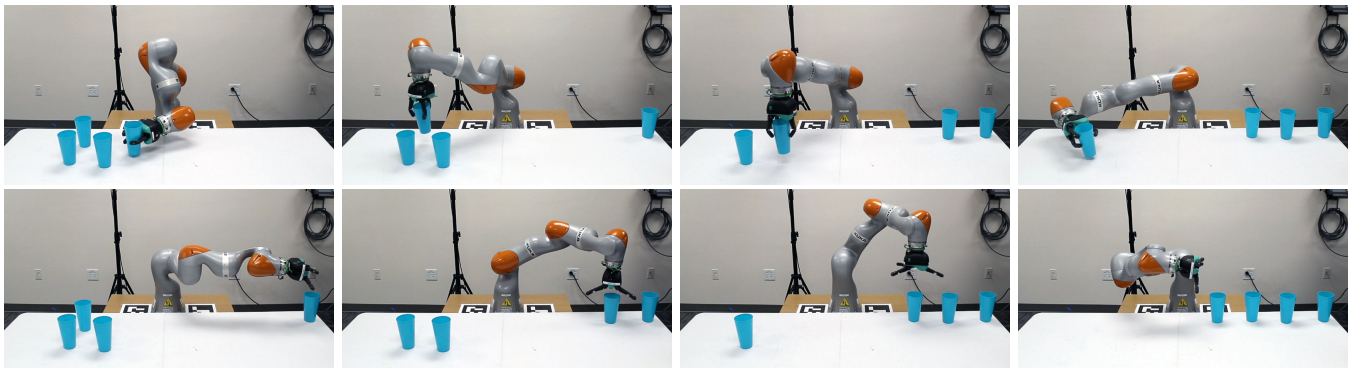


Fig. 13: Sequential pick (top row) and place (bottom row) of objects inline, using the cost defined in Eq. (9).

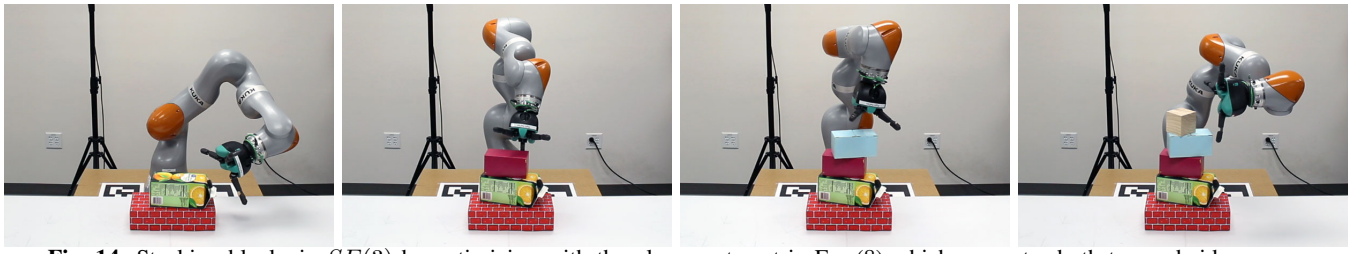


Fig. 14: Stacking blocks in $SE(3)$ by optimizing with the placement cost in Eq. (8) which generates both top and side grasps.

[5] 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*, 2017.

[6] F. Furrer, M. Wermelinger, H. Yoshida, F. Gramazio, M. Kohler, R. Siegwart, and M. Hutter, "Autonomous robotic stone stacking with online next best object target pose planning," in *IEEE Intl. Conf. on Robotics and Automation*, 2017.

[7] J. Baumgartl, T. Werner, P. Kaminsky, and D. Henrich, "A fast, gpu-based geometrical placement planner for unknown sensor-modelled objects and placement areas," in *IEEE Intl. Conf. on Robotics and Automation*, 2014.

[8] C. Paxton, C. Xie, T. Hermans, and D. Fox, "Predicting Stable Configurations for Semantic Placement of Novel Objects," in *Conference on Robot Learning*, 2021.

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

[10] S. Han, N. Stiffler, A. Krontiris, K. Bekris, and J. Yu, "High-quality tabletop rearrangement with overhand grasps: Hardness results and fast methods," in *Robotics: Science and Systems*, 2017.

[11] A. Cosgun, T. Hermans, V. Emeli, and M. Stilman, "Push Planning for Object Placement on Cluttered Table Surfaces," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2011.

[12] J. A. Haustein, K. Hang, J. Stork, and D. Kragic, "Object placement planning and optimization for robot manipulators," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2019.

[13] M. Gualtieri and R. W. Platt, "Learning 6-dof grasping and pick-place using attention focus," in *Conference on Robot Learning*, 2018.

[14] L. Berscheid, P. Meibner, and T. Kroeger, "Self-supervised learning for precise pick-and-place without object model," *IEEE Robotics and Automation Letters*, 2020.

[15] Z. He, N. Chavan-Dafle, J. Huh, S. Song, and V. Isler, "Pick2place: Task-aware 6dof grasp estimation via object-centric perspective affordance," in *IEEE Intl. Conf. on Robotics and Automation*, 05 2023.

[16] B. Wu, I. Akinola, and P. K. Allen, "Pixel-attentive policy gradient for multi-fingered grasping in cluttered scenes," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2019.

[17] M. Liu, Z. Pan, K. Xu, K. Ganguly, and D. Manocha, "Generating grasp poses for a high-dof gripper using neural networks," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2019.

[18] J. Varley, J. Weisz, J. Weiss, and P. Allen, "Generating multi-fingered robotic grasps via deep learning," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2015.

[19] M. Veres, M. Moussa, and G. W. Taylor, "Modeling grasp motor imagery through deep conditional generative models," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, 2017.

[20] J. M. Romano, K. Hsiao, G. Niemeyer, S. Chitta, and K. J. Kuchenbecker, "Human-inspired robotic grasp control with tactile sensing," *IEEE Transactions on Robotics*, vol. 27, no. 6, 2011.

[21] B. Sundaralingam, A. Lambert, A. Handa, B. Boots, T. Hermans, S. Birchfield, N. Ratliff, and D. Fox, "Robust Learning of Tactile Force Estimation through Robot Interaction," in *IEEE Intl. Conf. on Robotics and Automation*, 2019.

[22] Z. Li and S. Sastry, "Task-oriented optimal grasping by multifingered robot hands," *IEEE Journal on Robotics and Automation*, vol. 4, no. 1, 1988.

[23] R. Newbury, K. He, A. Cosgun, and T. Drummond, "Learning to place objects onto flat surfaces in upright orientations," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, 2021.

[24] A. Zeng, S. Song, K. Yu, E. Donlon, F. R. Hogan, M. Bauzá, D. Ma, O. Taylor, M. Liu, E. Romo, N. Fazeli, F. Alet, N. C. Dafle, R. Holladay, I. Morona, P. Q. Nair, D. Green, I. J. Taylor, W. Liu, T. A. Funkhouser, and A. Rodriguez, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in *IEEE Intl. Conf. on Robotics and Automation*, 2018.

[25] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee, "Transporter networks: Rearranging the visual world for robotic manipulation," in *Conference on Robot Learning*, 2020.

[26] J. Zhao, D. Troniak, and O. Kroemer, "Towards robotic assembly by predicting robust, precise and task-oriented grasps," in *Conference on Robot Learning*, 2020.

[27] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox, "Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes," in *IEEE Intl. Conf. on Robotics and Automation*, 2021.

[28] C. Mitash, R. Shome, B. Wen, A. Boularias, and K. Bekris, "Task-driven perception and manipulation for constrained placement of unknown objects," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, 2020.

[29] N. C. Dafle, S. Popovych, S. Agrawal, D. D. Lee, and V. Isler, "Object shell reconstruction: Camera-centric object representation for robotic grasping," *arXiv preprint*, 2021.

[30] F. Wang and K. Hauser, "Robot packing with known items and nondeterministic arrival order," in *Robotics: Science and Systems*, 2019.

[31] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[32] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

[33] D. Coleman, I. Sucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: a moveit! case study," 2014.