Linear Push Policies to Increase Grasp Access for Robot Bin Picking

Michael Danielczuk[1], Jeffrey Mahler[1], Chris Correa[1], Ken Goldberg[1,2]

***Abstract*— To facilitate automated bin picking when parts cannot be grasped, pushing actions have the potential to separate objects and move them away from bin walls and corners. In the context of the Dexterity Network (Dex-Net) robot grasping framework, we present two novel push policies based on targeting free space and diffusing clusters, and compare them to three earlier policies using four metrics. We evaluate these in simulation using Bullet Physics on a dataset of over 1,000 synthetic pushing scenarios. Pushing outcomes are evaluated by comparing the quality of the best available grasp action before and after each push using analytic grasp metrics. Experiments conducted on scenarios in which Dex-Net could not successfully grasp objects suggest that pushing can increase the probability of executing a successful grasp by more than 15%. Furthermore, in cases where grasp quality can be improved, the new policies outperform a quasi-random baseline by nearly 2 times. In physical experiments on an ABB YuMi, the highest performing push policy increases grasp quality by 24%.**

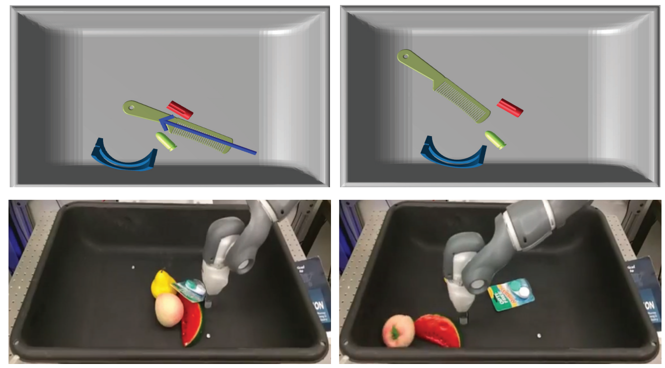

Fig. 1: Before (left) and after (right) images of successful pushes in simulation (top) and in physical experiments with the ABB YuMi (bottom).

I. Introduction

E-commerce warehousing applications often involve grasping and extracting objects with varying geometries and material properties from densely cluttered heaps or bins. Industry has shown interest in automating these tasks with robots. Deep learning methods can enable robots to grasp objects in isolation on a flat workspace, and these methods have recently been extended to grasping in clutter [19], [28], [34], [35]. Perception in cluttered bin environments remains a difficult problem, and scenarios may arise where the robot is unable to execute a collision-free grasp due to object proximity to the bin walls or to other objects [16], [37]. Pushing can change the position or orientation of parts so that a grasp with a higher probability of success can be executed [7], [14], [20].

It may not be necessary to completely separate objects for robust grasps to become available, and a series of pushes may be inefficient. Previous work measures the success of pushes as the degree of separation between objects and attempts to minimize the number of push actions to achieve separation [7], [12]. We explore directly using grasp confidence metrics instead of an indirect metric such as object separation for comparison of push policies, and attempt to maximize grasp confidence over a single push action.

In this paper, we explore pushing policies and analyze performance in bin picking environments. We simulate grasping and pushing policies over a set of 3D CAD models as sample objects using robust quasi-static analysis and the Bullet physics engine. By dropping objects into a virtual bin and executing sequences of grasping actions, we generate a dataset of over 1,000 simulated scenarios in which pushing could potentially be useful. For each of these scenarios, we evaluate five pushing policies against a baseline policy using metrics that quantify changes in grasp availability.

This paper makes three contributions:

1) Metrics to measure effectiveness of pushing actions,
2) Two novel push policies based on targeting free space and diffusing clusters
3) Experimental data from 1,000 simulated and physical bin scenarios where all initial state grasps generated by Dex-Net 2.0 and 3.0 have low confidence [35], [36].

II. Related Work

Mason pioneered research on analytic models of push mechanics [38]. Lynch and Akella described the mechanics of stable pushing and described how to create a plan for controlled pushing of an object through a series of obstacles [3], [31], [32], [33]. Recently, both analytic and learning techniques have been applied to learn physical pushing models for objects so they can be guided to a target location [1], [2], [4], [18], [25]. Goldberg and Brost showed that pushing objects to a location is desirable because it allows for objects to be grasped [6], [15]. Dogar and Srinivasa built on the idea of "push-grasping" to reduce uncertainty in grasping objects in clutter, using predefined 3D models of the objects to be grasped and attempting to estimate pose of each object in the scene before planning an action [10], [11]. Kehoe *et al.* extended these ideas to model uncertainty in the shape of the object [22]. In the Amazon Picking Challenge, the Technische Universitat Berlin team used pushing to enhance suction [13]. Team MIT used "topple" and "push-rotate" primitives to aid in grasping and bin-clearing [45]. These primitives were designed for a specific scenario in the competition, where some objects were initially ungraspable.

Another class of related work focuses on pushing heaps of objects for interactive segmentation. Hermans *et al.* [17]

[1]Department of Electrical Engineering and Computer Science
[2]Department of Industrial Engineering and Operations Research
[1−2]The AUTOLAB at UC Berkeley; Berkeley, CA 94720, USA
{mdanielczuk, jmahler, chris.correa, goldberg}@berkeley.edu

reason about boundaries between objects and keep a "push history" to estimate likelihoods that each cluster in an image is a separate object. They plan pushes along the object boundaries to attempt to force two nearby objects apart. While they score potential pushes on workspace boundaries and likelihood of collision with other clusters, their analysis does not account for scenarios where objects may be laying on top of one another. Chang *et al.* also consider interactive singulation of objects in clutter, pushing clusters of objects away from other clusters so that they can be grasped [7]. They track clusters between actions to gain evidence of singulation and adapt their pushing strategy based on the results of the previous pushes. However, their approach relies on object tracking, which may be sensitive to sensor noise. Several other groups study pushing as a means of interactive perception or segmentation, where actions are used to inform segmentation of objects from the background [21], [24], [30]. These use fixed-length pushes to distinguish objects from each other or the background, but do not use pushing to aid in future grasping attempts. In all of these cases, multiple pushing actions are necessary to confirm singulation of the object. We seek to extend these previous pushing methods for bin picking applications by introducing a bounded workspace and new push metrics.

"Singulation" applies pushing mechanics and planning to the task of separating or extracting objects that lie close together, and it is often required for successful object recognition or grasping. Model-based approaches such as the one proposed by Cosgun *et al*. [8] planned a series of robot actions to clear space for a target object in two-dimensional, tabletop pushing scenarios. Without prior knowledge of the objects, it can be challenging to estimate object geometries, poses, and other physical properties of the objects and environment [43], which can affect the efficiency of pushing actions [44]. For this reason, recent work has focused on learning strategies to push objects apart. Laskey *et al.* have used learning from demonstrations to successfully singulate and grasp one object in clutter with continuous push trajectories [26], [27]. Omrcen *et al.* [39] used many examples of pushing objects to train a fully-connected two-layer neural network that predicted 3D object motion for proposed actions. They then used these predictions to push objects to the edge of a table where they could be grasped. Boularias *et al.* [5] have also applied reinforcement learning to the task of pushing and grasping in clutter, extending their application to two objects. Eitel *et al.* [12] explore singulation in clutter using a push proposal convolutional neural network, showing that they can separate up to 8 objects with at least a 40% success rate in an average of 11 push actions. In contrast to their work, which seeks to minimize the number of push actions to separate all objects, we find one push at each opportunity, take into account bin walls and corners, and analyze push success based on new metrics in addition to object separation distance.

III. Problem Statement

Given a depth image of one or more objects in a bin as input, find the push action that maximizes the probability of a successful grasp of an object from the bin. A push action is defined by a line segment at a constant height parallel to the workspace.

A. Assumptions

We assume a robot with a parallel-jaw gripper or suction end effector, and rigid objects in clutter resting in a bin. We assume quasi-static physics and soft finger point contacts. All objects considered are able to be grasped in at least one stable pose. We also assume known gripper and bin geometries and a single overhead depth camera with known intrinsics. For purposes of finding free regions in the bin and boundaries between objects, we approximate distances using object center of mass. This approximation allows for reasonable computational efficiency in simulation, as finding minimum distances between two meshes is an expensive operation. In addition, we assumed object distances to be the distance between each object's center of mass when determining which objects to push. In physical experiments we approximate the center of mass of each object to be the centroid of each objects points from the point cloud.

B. Definitions

1) States: Let $\mathbf{x} = (\mathcal{O}, T_o) \in \mathcal{X}$ be the ground truth state of the heap, where $\mathcal{O}$ represents the geometries and inertial and material properties of all objects in the heap, and T_o represents the poses of the objects with respect to the camera.

2) Observations: Let $\mathbf{y} \in \mathcal{Y}$ be an observation of the state $\mathbf{x}$ which consists of a depth image of height H and width W taken by the overhead camera.

3) Actions: Let:

- $\mathbf{u}_j = (\mathbf{p}, \phi) \in \mathbb{R}^3 \times \mathcal{S}^1$ be a parallel jaw grasp defined by a center point $\mathbf{p} = (x, y, z) \in \mathbb{R}^3$ between the jaws and an angle in the plane of the table $\phi \in \mathcal{S}^1$ representing the grasp axis,
- $\mathbf{u}_s = (\mathbf{p}, \phi, \theta) \in \mathbb{R}^3 \times \mathcal{S}^2$ be a suction grasp defined by a target point $\mathbf{p} = (x, y, z) \in \mathbb{R}^3$ and spherical coordinates $(\phi, \theta) \in \mathcal{S}^2$ representing the axis of approach,
- $\mathbf{u}_g \in \mathcal{U}_j \bigcup \mathcal{U}_s$ be a generic grasp action, either a suction grasp or parallel jaw grasp, and
- $\mathbf{u}_p = (\mathbf{q}, \mathbf{r}) \in \mathbb{R}^3 \times \mathbb{R}^3$ be a linear pushing action in 3D space defined by a start point $\mathbf{q} = (x, y, z)$ and an end point $\mathbf{r} = (x', y', z')$, with respect to the camera

4) Reward: Let the reward function $\mathcal{R} \in \{0, 1\}$ be a binary function that indicates whether an object has been successfully grasped and removed from the bin. Pushing actions receive a reward of 0.

5) Grasp Success Distribution: Let $p(\mathcal{R}|\mathbf{u}, \mathbf{x})$ be a grasp success distribution that models the ability of a suction or parallel jaw grasp to resist gravitational wrenches under uncertainty in sensing, control, and disturbing wrenches [36], [40].

6) Grasp Quality Function: Let the grasp quality function $Q(\mathbf{u}, \mathbf{x}) = \mathbb{E}[\mathcal{R}|\mathbf{u}, \mathbf{x}]$ be a function that takes as input a parallel jaw grasp $\mathbf{u}_j$ or suction grasp $\mathbf{u}_s$ and the state $\mathbf{x}$ and evaluates the probability of success for that action given the current state. The grasp quality function Q is a continuous function on the interval $[0, 1]$.

7) Composite Policy: A composite policy $\pi(\mathbf{x})$ receives as input a state $\mathbf{x}$ and returns either a grasp action $\mathbf{u}_g$ or a push action $\mathbf{u}_p$.

C. Objective

The high-level goal is to maximize mean picks per hour (MPPH), defined as:

$$\mathbb{E}[\rho] = \mathbb{E}\left[\frac{\sum_{i=0}^{N} \mathcal{R}(\pi(\mathbf{x}_i), \mathbf{x}_i)}{\sum_{i=0}^{N} \Delta(\pi(\mathbf{x}_i))}\right] \quad (1)$$

where $\Delta(\pi(\mathbf{x}_i))$ is the time spent sensing, planning, and executing the action $\pi(\mathbf{x}_i)$. If we assume a constant execution time for each of N push or grasp actions, Equation 1 can be simplified to:

$$\mathbb{E}[\rho] = \frac{\sum_{i=0}^{N} Q(\pi(\mathbf{x}_i), \mathbf{x}_i)}{N\bar{\Delta}} \quad (2)$$

where $\bar{\Delta}$ is the average time needed to execute an action.

Multiple consecutive unsuccessful grasps increase the total number of actions N without contributing a reward $\mathcal{R}$, decreasing $\mathbb{E}[\rho]$. When grasp quality is low, successful pushing actions lead to successful grasps, contributing a higher reward over the same number of actions. We focus on the subtask of choosing a push action to maximize probability of a robust grasp being available in the next timestep in scenarios where the initial grasp quality is low.

Let the function $f : \mathcal{X} \times \mathcal{U}_p \rightarrow \mathcal{X}$ define the state transition such that $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_{p,t})$ for a pushing action $\mathbf{u}_{p,t}$ at time t. Then we wish to find:

$$\mathbf{u}_{p,t}^* = \underset{\mathbf{u}_{p,t} \in \mathcal{U}_p}{\arg\max}\, Q(\mathbf{u}_{g,t+1}^*, \mathbf{x}_{t+1})$$

where $\mathbf{u}_{g,t+1}^*$ is the grasp that maximizes Q for the next state $\mathbf{x}_{t+1}$.

To consider when a push action would be more efficient than a grasp action at the current timestep $t = 0$, we analyze which action $\mathbf{u}_0$ maximizes the sum of expected rewards Ψ over a two-timestep period $t = \{0, 1\}$. At time $t = 0$, we measure the binary reward for grasp actions and assign a reward of 0 for push actions. At time $t = 1$ we measure the probability of success for the best grasp action given the new state of the heap, $Q(\mathbf{u}_{g,1}^*, \mathbf{x}_1)$:

$$\begin{aligned}\Psi(\mathbf{u}_0, \mathbf{x}_0) &= \mathbb{E}[\mathcal{R}(\mathbf{u}_0, \mathbf{x}_0) + \mathcal{R}(\mathbf{u}_{g,1}^*, \mathbf{x}_1)] \\ &\approx \mathcal{R}(\mathbf{u}_0, \mathbf{x}_0) + Q(\mathbf{u}_{g,1}^*, \mathbf{x}_1)\end{aligned}$$

By comparing the sum of expected rewards for both actions at times $t = 0$ and $t = 1$, we determine which action maximizes the total reward over the given period. We formalize this as follows:

$$\begin{aligned}\Psi_p &= \Psi(\mathbf{u}_p, \mathbf{x}_0) \approx Q(\mathbf{u}_{g,1}^*, \mathbf{x}_{p,1}) \\ \Psi_g &= \Psi(\mathbf{u}_{g,0}, \mathbf{x}_0) \approx \mathcal{R}(\mathbf{u}_{g,0}, \mathbf{x}_0) + Q(\mathbf{u}_{g,1}^*, \mathbf{x}_{g,1})\end{aligned}$$

We prefer the push action when $\Psi_p > \Psi_g$.

IV. Push-Action Metrics

We define four metrics to evaluate pushing policies on the dataset of generated heaps in simulation, where we have ground truth object poses. The first, *mean object separation gain*, is a metric based on previous work [7], [12] and rewards pushing policies that create as much separation between objects as possible. The other three metrics reward pushing policies that lead to available high-quality grasps in the next timestep, measured using robust gravitational wrench resistance analysis as in [36]. These metrics quantify the change in grasp quality (probability of a successful grasp) due to a pushing action. In all cases, we approximate inter-object distance and bin-object distance using centers of mass for computational efficiency.

A. Mean Object Separation Gain

Mean Object Separation Gain measures the average degree of separation between objects in the heap [12]. Prior work has focused on minimizing the number of pushes necessary to achieve a *minimum* separation between all objects in the heap; however our goal is to maximize the expected reward at the next timestep. Highly-successful pushing actions such as totally separating a single object from a heap might not increase the minimum separation at all, as other objects in the heap may still be touching. Therefore, we use a normalized change in the *mean* object separation for our first metric, and we call this adapted version of prior object separation metrics *mean object separation gain*. For n objects, mean object separation is defined as:

$$D = \frac{\sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \|o_i - o_j\|_2 + \sum_{i=0}^{n-1} \min_{b_j \in \mathcal{B}} \|o_i - b_j\|_2}{\binom{n}{2} + n}$$

where o is a vector of object centroids, and $\mathcal{B}$ is the set of bin edges. D is an average of the pairwise distances between objects and each object's distance to the bin. Thus, mean object separation gain is defined as:

$$\Delta D = \frac{D_1 - D_0}{\max(D_0, D_1)}$$

where D_0 is the initial mean object separation and D_1 is the mean object separation after the push action. We normalize this quantity by the larger mean object separation of the initial and final states to get a value between -1 and 1.

B. Parallel Jaw Grasp Quality Gain

Parallel jaw grasp quality gain measures the change in best available parallel jaw grasp, allowing us to differentiate grasp probabilities of success by end-effector. The parallel jaw grasp quality is defined as follows:

$$Q_j^* = \max_{\mathbf{u}_j \in \mathcal{U}_j} Q_j(\mathbf{u}_j, \mathbf{x})$$

where $\mathcal{U}_j$ is the set of all parallel-jaw grasps available on objects in the heap. We approximate the grasp set with a finite set of 100 sampled grasps per object. Parallel-jaw grasps are sampled from approximately antipodal pairs of points on the surface of each object. Parallel jaw grasp quality gain is then defined as:

$$\Delta Q_j^* = Q_{j,1}^* - Q_{j,0}^*$$

where $Q_{j,0}^*$ is the initial parallel jaw grasp quality and $Q_{j,1}^*$ is the parallel jaw grasp quality after the push action.

C. Suction Grasp Quality Gain

Suction grasp quality gain measures the change in best available suction grasp, again allowing us to differentiate grasp probabilities of success by end-effector. The suction grasp quality is defined as follows:

$$Q_s^* = \max_{\mathbf{u}_s \in \mathcal{U}_s} Q_s(\mathbf{u}_s, \mathbf{x})$$

where $\mathcal{U}_s$ is the set of all suction grasps available on objects in the heap. The grasp set is approximated in the same way as the parallel-jaw grasps, and suction grasps are sampled uniformly across the surface of each object with approach axes coinciding with surface normals. Suction grasp quality gain is then defined as:

$$\Delta Q_s^* = Q_{s,1}^* - Q_{s,0}^*$$

where $Q_{s,0}^*$ is the initial suction grasp quality and $Q_{s,1}^*$ is the suction grasp quality after the push action.

D. Overall Grasp Quality Gain

Overall grasp quality gain measures the change in best available grasp. Overall grasp confidence is defined as:

$$Q_o^* = \max\left(Q_j^*, Q_s^*\right)$$

Overall grasp quality gain is then defined as:

$$\Delta Q_o^* = Q_{o,1}^* - Q_{o,0}^*$$

where $Q_{o,0}^*$ is the initial overall grasp quality and $Q_{o,1}^*$ is the overall grasp quality after the push action.

V. Push Policies

We compare five policies, two from prior work, two novel and a baseline method. The baseline method is a quasi-random baseline policy similar to the one used by Hermans *et al.* [17]. All policies operate on both ground truth state information in simulation or the depth maps observations described in Section III. For a full state input, we performed collision checking using the Flexible Collision Library and use the known object centers of mass and poses. For a depth image input, we transform the depth image to a point cloud, segment the bin and the objects using the Euclidean Clustering method described in the PointCloud Library [41], and execute an Image Clearance Checker that checks for object and gripper collisions. The object center of mass is approximated to be the centroid of the segmented cluster of points from the point cloud [29]. For the following section, references to "object" can be replaced with segmented cluster in the case of point cloud inputs. The "free space point" p_i is defined as the point that maximizes the minimum Euclidean distance from object i to the other objects in the bin and the bin walls, penalizing distance from the starting location with an L2 regularization term.

A. Quasi-Random Policy

The Quasi-Random Policy generates a linear push action using the following three steps:

1) Choose one object in the heap at random,
2) Choose a direction at random, and
3) Push for a fixed length at the center of mass toward the chosen object in the chosen direction.

The push action is clipped to the bounds of the bin so that the gripper will not collide when executing the action.

B. Boundary Shear Policy

The boundary shear policy is adapted from the pushing policy introduced in Hermans et al. in [17]. It aims to separate the two closest objects in the heap by pushing one of them along the boundary between the two objects.

1) Find the two closest objects in the heap with centers of mass c_i and c_j,
2) Construct a line $\overline{c_i c_j}$ connecting the centers of mass of the two closest objects projected to the plane of the bin bottom, and a line $\overline{c_i c_j}_\perp$ perpendicular to $\overline{c_i c_j}$ that defines the vector approximating the boundary of the two objects,
3) Generate four possible push vectors, two for each object, that extend through the centers of mass of the objects in the direction $\overline{c_i c_j}_\perp$, and
4) Choose the push direction which is closest to the direction of free space and is collision free.

C. Free Space Policy

The free space policy aims to separate the two objects in the heap with closest centers of mass by pushing one of them along a direction toward the most free space, taking into account bin walls and other objects. It generates the push action using the following steps:

1) Find the two objects in the heap with closest centers of mass c_i and c_j,
2) For each object, find the free space point p_i defined above,
3) Draw lines $\overline{c_i p_i}, \overline{c_j p_j}$ from each of the centers of mass of the two closest objects to the points p_1 and p_2, respectively, with each point projected to the plane of the bottom of the bin,
4) Generate two possible push vectors, one for each object, that extend through the centers of mass of the objects in the collision-free directions closest to $\overline{c_i p_i}$ and $\overline{c_j p_j}$, and
5) Choose from the two possible collision-free push actions based on the minimum distance from the current center of mass of object i to p_i.

D. Maximum Clearance Ratio Policy

The maximum clearance policy, defined by Chang, Smith, and Fox [7], analyzes the available space for an object to be pushed into and the cluttered area it is being pushed from.

1) Calculate clearance in front of and behind each object for 16 uniform directions spanning angles between 0 and 2π by moving the objects footprint in the given direction and checking for collisions with other objects or the bin, and

2) Choose push action that maximizes ratio of space in the forward direction to space in the backward direction and is collision free.

E. Cluster Diffusion Policy

The cluster diffusion policy groups objects into clusters based on their position. It considers pushes of objects away from their corresponding cluster centers, along the vector originating from the cluster center to the object center of mass.

1) Separate objects into clusters of one to three objects and find the centroid of each cluster m_i,
2) Define pushing vectors $\overline{m_i c_i}$ that connect center of cluster to center of mass c_i of each object in its cluster, and
3) Score each of the potential push actions as their cosine similarity with the direction of most free space for the given object, and execute the push action with the highest score.

VI. Simulation Experiments

We generate heaps of 3D object meshes from the Thingiverse, and the KIT and 3DNet datasets in a bin. We initialize simulations by sampling over distributions of heap size, 3D object models, camera pose, and friction coefficients to get an initial state $\mathbf{x}_0$. We randomly drop objects into the bin, and repeatedly execute parallel jaw and suction grasps until the bin is cleared or the grasping policy described in [34], [35], [36] fails n times in a row or has confidence below a threshold. If the bin is not cleared, we record the heap. We then roll out each push policy on the same set of heaps, and measure the performance of each policy using the metrics in Section IV. Algorithm 1 describes the process of grasping objects out of the bin and marking scenarios during the bin picking simulation when no high-quality grasp actions are available.

With the dataset of over 1000 pushing scenarios collected, each of the policies described in Section V were rolled out on the set of heaps, using pybullet [9] for simulating gripper-object and object-object interactions in the bin, and the metrics in Section IV were measured for each pushing action. We reject heaps where none of the policies (including the quasi-random policy), were able to generate positive values for Overall Grasp Confidence Gain either due to object geometry or extreme cases of object positioning in the bin (e.g. object lying directly in a corner of the bin). These heaps reduce the performance across all policies, and rejection sampling allows us to focus on cases that highlight differences between the pushing policies. The remaining 481 pushing scenarios, termed *improvable heaps* were used to compare policies to the baseline policy.

Figure 2 shows that all five policies studied had positive Overall Grasp Confidence Gain over the set of improvable heaps by 5 percent, even in cases where they performed poorly in terms of Mean Object Separation Gain. The Free Space and Boundary Shear policies performed the best on the improvable heaps, with average Overall Grasp Quality Gains of 17% and 18%, which outperformed the baseline by an absolute 7% and 8%, respectively. We note that the maximal clearance ratio policy performed best on the mean

Algorithm 1: Bin Dataset Generation

```
Sample over distributions of heap size, 3D object models,
  camera pose, and friction coefficients to create heap;
t = 0;
consecutive_failures = 0;
while consecutive_failures < max_failures &&
  objects_in_heap > 0 do
    Randomly sample grasps over all objects in heap to
      create grasp set U;
    Prune U using collision checking and proximity to
      other grasps;
    Find best grasps u*_j and u*_s;
    Find Q*_o;
    if Q*_o > threshold then
        Execute best grasp u*;
        if grasp_succeeded then
            consecutive_failures = 0;
        else
            increment consecutive_failures;
        Allow objects to settle;
        increment t;
    else
        Add heap to bin dataset;
```

object separation gain metric, but the boundary shear and free space policies outperformed it by three times in overall grasp confidence gain. This result suggests grasp confidence gain is not always correlated with object separation and could better measure push action success in bin picking.

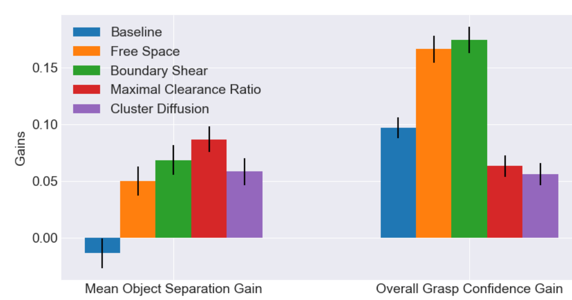

Fig. 2: Means and standard errors of the mean for each policy and each metric. All policies have Overall Grasp Confidence Gain values above 0.1, but Mean Object Separation Gain values do not correspond to Overall Grasp Confidence Gain values, suggesting objects do not need to be separated to be grasped.

To analyze the distribution of Overall Grasp Quality Gain, we separately recorded the Suction Grasp Quality Gain and Parallel Jaw Quality Gain for each pushing action. These results can be seen in Figure 3 and imply that pushing affects the parallel jaw grasps more than it affects suction grasps. Pushing actions typically move objects around the bin, but rarely topple them onto a new face or side. Suction relies on sampling grasps on the top faces of the objects; if the face does not change, then it is unlikely that the suction grasp confidence will change significantly. However, for the parallel jaws, grasp confidence depends strongly on available space around the object. Thus, pushing an object to a freer location can shift the parallel jaw grasp confidence more dramatically.

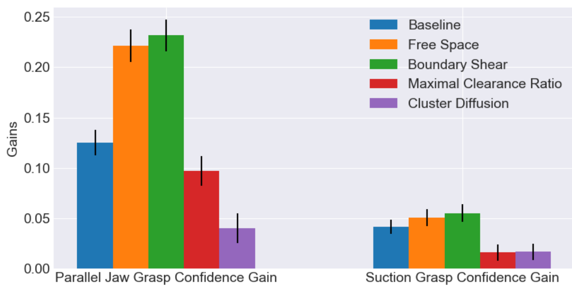

Fig. 3: Means and standard errors of the mean for each policy and each type of end effector. These results suggest pushing has a larger effect on the parallel jaws. We speculate that this effect occurs due to suction grasps relying on faces of objects being available, and are thus less likely to be affected by pushing, whereas parallel jaw grasps are heavily affected by space around the object.

Next, we hypothesized that the policies would outperform the baseline policy on average and make higher confidence grasps available to the grasping policies at the next timestep. We used analysis of variables (ANOVA) and linear regressions to quantify the differences between policies for each metric. A one-way repeated measures ANOVA was run for each metric, and at least one policy was determined to be statistically different from the baseline policy for Mean Object Separation Gain (F(3, 1907) = 19.01, $p < 0.001$), Overall Grasp Confidence Gain (F(3, 1205) = 24.81, $p < 0.001$), and Parallel Jaw Grasp Confidence Gain (F(3, 1205) = 21.38, $p < 0.001$). However, none of the policies were determined to be statistically different from the baseline for the Suction Grasp Confidence Gain metric.

To further analyze the differences between policies statistically, we ran robust linear regressions over 1907 observations for each metric, controlling for differences between heaps. The results showed that the free space and boundary shear policies are statistically different from the baseline in both Overall Grasp Confidence Gain and Parallel Jaw Confidence Gain ($p < 0.001$). Although they are statistically different from the baseline, we notice that the coefficient is about 0.1 units of grasp confidence, meaning that the baseline policy actually performs very well under many conditions. We hypothesize that the baseline policy performs well on average because it always contacts an object. Thus, it always changes the state of the heap, which can often generate grasps. Additionally, 337 of the 481 improvable heaps had fewer than four objects, so the baseline policy often planned similar actions to the other policies since it had fewer options to choose from. The heap sizes are often small since the original grasping policy which generated the heaps rarely fails consecutively on heaps with many objects to choose from.

We examined many individual heaps to understand the magnitude and variance of the policies impact. Figure 4 shows an example where each policy outperformed the baseline, while Figure 5 depicts an example where the baseline performance exceeds that of each policy. In Figure 4, we can see that the non-baseline policies choose to push one of the two objects that overlap, and they all achieve a large increase in the parallel jaw metric by uncovering the red object initially lying underneath another object. The Boundary Shear and Free Space policies perform especially well, separating all of the objects. Note that the object does not need to be completely uncovered for the grasp to be available. This reflects the difference between measuring grasping metrics and object separations because in this case, the objects are still touching but a parallel jaw grasp becomes available.

In contrast, in Figure 5, we can see that the non-baseline policies fail to find a collision-free push that can move one of the objects away from the corner of the bin. The baseline policy's action is clipped so that it does not collide with the bin, and results in it slightly increasing the parallel jaw grasp confidence by nudging the green object further from the other object. The non-baseline policies have no effect on the grasp confidence metrics. This figure illustrates one of the current failure modes with the pushing policies that we have implemented. By taking a conservative approach and avoiding collisions at all costs, we are sometimes unable to plan a push that moves the objects away from the bin edges.

VII. Physical Experiments

We planned pushes for bin picking on an ABB YuMi using the Boundary Shear policy, the best performing policy in simulation, over 35 heaps of objects with varying geometry such as tools, toys, produce, and industrial parts. Each heap contained between two and ten objects in configurations with few accessible grasps, such as two bottles side-by-side. For each heap, the robot acquired a point cloud of the bin with a Photoneo PhoXi depth sensor, segmented the point cloud using Euclidean Cluster Extraction implemented in the Point Cloud Library [41], and planned a push using point clusters as objects and the cluster centroid as an estimate of the center of mass of each object. The robot then executed the push by closing the parallel jaw gripper and following the linear push trajectory. Each push took approximately 1.0 seconds to plan.

For each push, we measured the Overall Grasp Quality Gain of parallel-jaw and suction grasps planned by a Grasp Quality Neural Network policy [35], [36]. We categorized performance based on the grasp quality for the best grasp in the initial heap (pre-pushing). Heaps with $Q_o < 0.25$ had an Overall Grasp Quality Gain of 0.24 ± 0.07, while heaps with $Q_o < 0.5$ had an Overall Grasp Quality Gain of 0.12 ± 0.06.

VIII. Discussion and Future Work

Analytic pushing methods can generalize well to heaps of different sizes and geometries on a tabletop environment, but with the bin as a workspace boundary, the action space for pushing is much more limited. Thus, in many cases, pushing any of the available objects in the bin in any direction will yield a change in the state of the heap large enough to change grasp access and affect grasp metrics. However, we have shown that several policies are statistically different from a baseline when controlling for the difficulty of each heap, based on new grasp-centric metrics to measure effectiveness of pushing and given the assumptions in Section III.

In future work, we will further explore how the approximations and assumptions made affect the results presented.

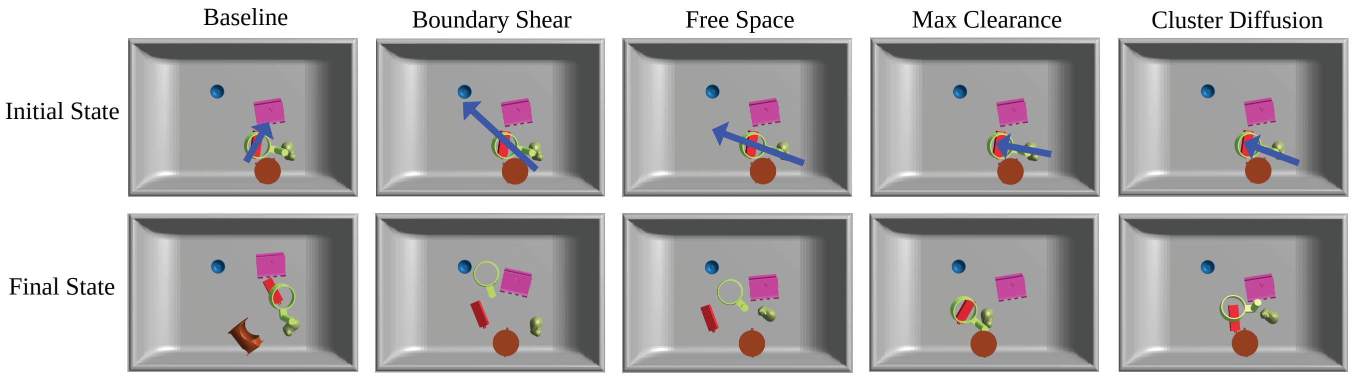

Fig. 4: For this heap, each policy outperforms the baseline with respect to overall grasp confidence gain. The initial state with the planned action (top row) and final state after executing the planned action (bottom row) are shown for each policy. The blue arrow represents the planned push and the the initial gripper position is represented by the tail of the arrow, while the final position is represented by the head.

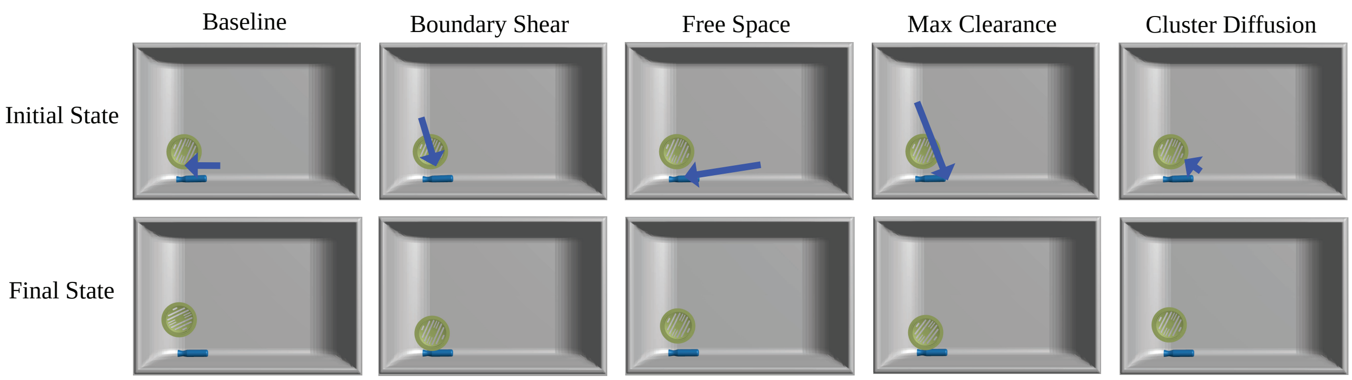

Fig. 5: The baseline marginally outperforms all the other policies with respect to overall grasp confidence gain due to object placement. The initial state with the planned action (top row) and final state after executing the planned action (bottom row) are shown for each policy. The blue arrow represents the planned push and the the initial gripper position is represented by the tail of the arrow, while the final position is represented by the head.

For example, we will continue to benchmark the policies presented here, as well as other policies, on the physical robot in order to determine the effectiveness of each policy on the physical system and how closely our simulator can represent the physical system. Further experiments will also provide a larger sample size for statistical analysis on the physical system. We will also consider different metrics, such as *maximum* separation distance, which could better inform the state of each heap. For this analysis, we chose a two-step time horizon as an approximation of the normalized total expected reward over all time shown in Equation 2 because greedy grasping policies have been shown to perform very well in clutter, suggesting long-term rewards are not strongly correlated to grasp actions [34], [42]. In our next experiments, we will test this approximation by measuring the effect of pushing over the course of longer time horizons. Additionally, we made strong assumptions about the boundaries, geometries, and poses of the objects that were analyzed by representing them as points at their center of mass for finding free space in the bin. We seek to modify our simulations to calculate minimum distances between meshes more efficiently while still accounting for the entirety of the objects. We also will look to exploit quicker free space computation in image space as an alternative to our current object assumptions.

Furthermore, in this work, we assume that if none of the five policies were able to prove grasp quality, then the heap is not improvable. Some heaps may be improvable by some policy not tested in this work. In the future, we will determine why some heaps are not able to be improved and seek a method for determining when heaps can be improved without testing several policies on them. For example, when objects are entangled, or cannot easily be pushed due to object pose or shape, we could attempt a different push or grasp action.

As extensions to this work, we will identify and explore more complex push policies that include multiple linear segments and continuous motions in the plane and out of plane (e.g., to topple or flip objects). We will also explore how push policies can be learned from human demonstrations [27] and from automated execution data shared via the Cloud [23].

IX. Acknowledgments

This research was performed in the AUTOLab at UC Berkeley in affiliation with the Berkeley AI Research (BAIR) Lab, the Real-Time Intelligent Secure Execution (RISE) Lab, and the CITRIS "People and Robots" (CPAR) Initiative.The authors were supported in part by donations from Siemens, Google, Honda, Intel, Comcast, Cisco, Autodesk, Amazon Robotics, Toyota Research Institute, ABB, Samsung, Knapp, and Loccioni. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Sponsors.

We thank our colleagues who provided helpful feedback and suggestions, in particular Matt Matl, Michael Laskey, and Bill DeRose. We also thank Katie Glockner, who provided assistance with statistical analysis.

REFERENCES

[1] W. C. Agboh and M. R. Dogar, "Pushing fast and slow: Task-adaptive mpc for pushing manipulation under uncertainty," *arXiv preprint arXiv:1805.03005*, 2018.

[2] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to poke by poking: Experiential learning of intuitive physics," in *Proc. Advances in Neural Information Processing Systems*, 2016, pp. 5074–5082.

[3] S. Akella and M. T. Mason, "Posing polygonal objects in the plane by pushing," *Int. Journal of Robotics Research (IJRR)*, vol. 17, no. 1, pp. 70–88, 1998.

[4] W. Bejjani, R. Papallas, M. Leonetti, and M. R. Dogar, "Learning deep policies for physics-based manipulation in clutter," *arXiv preprint arXiv:1803.08100*, 2018.

[5] A. Boularias, J. A. Bagnell, and A. Stentz, "Learning to manipulate unknown objects in clutter by reinforcement." in *AAAI*, 2015, pp. 1336–1342.

[6] R. C. Brost, "Automatic grasp planning in the presence of uncertainty," *The International Journal of Robotics Research*, vol. 7, no. 1, pp. 3–17, 1988.

[7] L. Chang, J. R. Smith, and D. Fox, "Interactive singulation of objects from a pile," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2012, pp. 3875–3882.

[8] A. Cosgun, T. Hermans, V. Emeli, and M. Stilman, "Push planning for object placement on cluttered table surfaces," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2011, pp. 4627–4632.

[9] E. Coumans, Y. Bai, and J. Hsu. Pybullet physics simulator. [Online]. Available: http://pybullet.org

[10] M. Dogar, K. Hsiao, M. Ciocarlie, and S. Srinivasa, "Physics-based grasp planning through clutter," in *Proc. Robotics: Science and Systems (RSS)*, 2012.

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

[12] A. Eitel, N. Hauff, and W. Burgard, "Learning to singulate objects using a push proposal network," in *Int. S. Robotics Research (ISRR)*, 2017.

[13] C. Eppner, S. Höfer, R. Jonschkowski, R. M. Martin, A. Sieverling, V. Wall, and O. Brock, "Lessons from the amazon picking challenge: Four aspects of building robotic systems." in *Proc. Robotics: Science and Systems (RSS)*, 2016.

[14] K. Y. Goldberg, "Orienting polygonal parts without sensors," *Algorithmica*, vol. 10, no. 2-4, pp. 201–225, 1993.

[15] ——, "Stochastic plans for robotic manipulation," Ph.D. dissertation, Carnegie Mellon University, 1991.

[16] M. Gualtieri, A. ten Pas, K. Saenko, and R. Platt, "High precision grasp pose detection in dense clutter," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 598–605.

[17] T. Hermans, J. M. Rehg, and A. Bobick, "Guided pushing for object singulation," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2012, pp. 4783–4790.

[18] K.-S. Hwang, J. Ling, and W.-H. Wang, "Adaptive reinforcement learning in box-pushing robots," in *Proc. IEEE Conf. on Automation Science and Engineering (CASE)*. IEEE, 2014, pp. 1182–1187.

[19] E. Johns, S. Leutenegger, and A. J. Davison, "Deep learning a grasp function for grasping under gripper pose uncertainty," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4461–4468.

[20] K. N. Kaipa, A. S. Kankanhalli-Nagendra, N. B. Kumbla, S. Shriyam, S. S. Thevendria-Karthic, J. A. Marvel, and S. K. Gupta, "Enhancing robotic unstructured bin-picking performance by enabling remote human interventions in challenging perception scenarios," in *Proc. IEEE Conf. on Automation Science and Engineering (CASE)*. IEEE, 2016, pp. 639–645.

[21] D. Katz, M. Kazemi, J. A. Bagnell, and A. Stentz, "Clearing a pile of unknown objects using interactive perception," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2013, pp. 154–161.

[22] B. Kehoe, D. Berenson, and K. Goldberg, "Toward cloud-based grasping with uncertainty in shape: Estimating lower bounds on achieving force closure with zero-slip push grasps," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2012, pp. 576–583.

[23] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, "A survey of research on cloud robotics and automation," *IEEE Transactions on automation science and engineering*, vol. 12, no. 2, pp. 398–409, 2015.

[24] J. Kenney, T. Buckley, and O. Brock, "Interactive segmentation for manipulation in unstructured environments," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2009, pp. 1377–1382.

[25] S. Krivic, E. Ugur, and J. Piater, "A robust pushing skill for object delivery between obstacles," in *Proc. IEEE Conf. on Automation Science and Engineering (CASE)*. IEEE, 2016, pp. 1184–1189.

[26] M. Laskey, C. Chuck, J. Lee, J. Mahler, S. Krishnan, K. Jamieson, A. Dragan, and K. Goldberg, "Comparing human-centric and robot-centric sampling for robot deep learning from demonstrations," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2017, pp. 358–365.

[27] M. Laskey, J. Lee, C. Chuck, D. Gealy, W. Hsieh, F. T. Pokorny, A. D. Dragan, and K. Goldberg, "Robot grasping in clutter: Using a hierarchy of supervisors for learning from demonstrations," in *Proc. IEEE Conf. on Automation Science and Engineering (CASE)*. IEEE, 2016, pp. 827–834.

[28] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *Int. Journal of Robotics Research (IJRR)*, vol. 34, no. 4-5, pp. 705–724, 2015.

[29] M. Levi, *The mathematical mechanic: using physical reasoning to solve problems*. Princeton University Press, 2009.

[30] W. H. Li and L. Kleeman, "Autonomous segmentation of near-symmetric objects through vision and robotic nudging," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2008, pp. 3604–3609.

[31] K. M. Lynch, "The mechanics of fine manipulation by pushing," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 1992, pp. 2269–2276.

[32] K. M. Lynch and M. T. Mason, "Controllability of pushing," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, vol. 1. IEEE, 1995, pp. 112–119.

[33] ——, "Stable pushing: Mechanics, controllability, and planning," *Int. Journal of Robotics Research (IJRR)*, vol. 15, no. 6, pp. 533–556, 1996.

[34] J. Mahler and K. Goldberg, "Learning deep policies for robot bin picking by simulating robust grasping sequences," in *Conference on Robot Learning*, 2017, pp. 515–524.

[35] 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 *Proc. Robotics: Science and Systems (RSS)*, 2017.

[36] J. Mahler, M. Matl, X. Liu, A. Li, D. Gealy, and K. Goldberg, "Dex-net 3.0: Computing robust robot suction grasp targets in point clouds using a new analytic model and deep learning," *arXiv preprint arXiv:1709.06670*, 2017.

[37] J. A. Marvel, K. Saidi, R. Eastman, T. Hong, G. Cheok, and E. Messina, "Technology readiness levels for randomized bin picking," in *Proceedings of the Workshop on Performance Metrics for Intelligent Systems*. ACM, 2012, pp. 109–113.

[38] M. T. Mason, *Mechanics of robotic manipulation*. MIT press, 2001.

[39] D. Omrčen, C. Böge, T. Asfour, A. Ude, and R. Dillmann, "Autonomous acquisition of pushing actions to support object grasping with a humanoid robot," in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*. IEEE, 2009, pp. 277–283.

[40] D. Prattichizzo and J. C. Trinkle, "Grasping," in *Springer handbook of robotics*. Springer, 2008, pp. 671–700.

[41] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2011, pp. 1–4.

[42] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *Int. Journal of Robotics Research (IJRR)*, p. 0278364917735594, 2017.

[43] Z. Teng and J. Xiao, "Surface-based detection and 6-dof pose estimation of 3-d objects in cluttered scenes," *IEEE Trans. Robotics*, vol. 32, no. 6, pp. 1347–1361, 2016.

[44] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez, "More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 30–37.

[45] K.-T. Yu, N. Fazeli, N. Chavan-Dafle, O. Taylor, E. Donlon, G. D. Lankenau, and A. Rodriguez, "A summary of team mit's approach to the amazon picking challenge 2015," *arXiv preprint arXiv:1604.03639*, 2016.