Effective and Robust Non-prehensile Manipulation via Persistent Homology Guided Monte-Carlo Tree Search

Ewerton R. Vieira[1]([✉]), Kai Gao[2], Daniel Nakhimovich[2], Kostas E. Bekris[2],
and Jingjin Yu[2]

[1] Department of Mathematics, Rutgers, NJ, USA
`er691@rutgers.edu`
[2] Department of Computer Science, Rutgers, NJ, USA

Abstract. Performing object retrieval in real-world workspaces must tackle challenges including *uncertainty* and *clutter*. One option is to apply prehensile operations, which can be time consuming in highly-cluttered scenarios. On the other hand, non-prehensile actions, such as pushing simultaneously multiple objects, can help to quickly clear a cluttered workspace and retrieve a target object. Such actions, however, can also lead to increased uncertainty as it is difficult to estimate the outcome of pushing operations. The proposed framework in this work integrates topological tools and Monte-Carlo Tree Search (MCTS) to achieve effective and robust pushing for object retrieval. It employs persistent homology to automatically identify manageable clusters of blocking objects without the need for manually adjusting hyper-parameters. Then, MCTS uses this information to explore feasible actions to push groups of objects, aiming to minimize the number of operations needed to clear the path to the target. Real-world experiments using a Baxter robot, which involves some noise in actuation, show that the proposed framework achieves a higher success rate in solving retrieval tasks in dense clutter than alternatives. Moreover, it produces solutions with few pushing actions improving the overall execution time. More critically, it is robust enough that it allows one to plan the sequence of actions offline and then execute them reliably on a Baxter robot.

Introduction video: youtu.be/00kEztqytRU.
Code: github.com/DanManN/planning_baxter/.

Keywords: Manipulation · Topology · Monte-Carlo Tree Search

1 Introduction

Retrieving a target object from a messy and constrained space, such as taking out a bottle of water from a fridge, requires a robotic arm to relocate other

objects blocking access. Humans routinely perform such tasks with a high degree of success, and the manipulation primitives used to execute such tasks are not limited by pick-and-place-based rearrangements. Instead, they often involve non-prehensile manipulation, such as pushing and pulling actions. Endowing robots with such skills is highly desirable, especially if they are tasked to carry out ordinary human tasks. Humans execute such manipulation operations, including grouping objects before performing a push, in a naturally robust manner, while keeping the number of actions low.

This work proposes strategies for solving object retrieval problems focusing on *efficiency and robustness.* An example run is shown in Fig. 2, in which the robot must retrieve the target object located at the lower portion of the workspace (a Pringles can of a specific flavor). Our study tackles the challenge by exploring two related sub-strategies: (i) clustering objects to inform the choice of pushing actions, and (ii) searching through and evaluating feasible pushing operations before executing them in the real world. The first is motivated by human behaviors of implicitly grouping objects before pushing them simultaneously. Topological tools, such as persistent homology, have been successfully employed to comprehensively identify manageable object clusters for selecting the proper pushing actions [19]. The second direction leverages

Fig. 1. Top left: the experimental setup. Top right: Scene S2 is one of the initial configurations for the experiments. The robot must move cluttered objects blocking access to a target object $\mathcal{T}$ placed at the bottom close to the wall. (2) to (5) demonstrate the pushing actions executed after a given plan by the proposed method. A simple grasping plan is performed from (6) to (7).

Monte-Carlo Tree Search (MCTS) to explore the feasible identified pushing actions with a high reward level in terms of the number of obstacles removed from the path to the target and how dispersed the clusters are after the performed actions. The reward metric is intuitive since a higher number of obstacles removed results in a higher chance to solve the task faster, and dispersing the clusters leads to easier pushing actions for consecutive steps.

The proposed framework integrates MCTS and the informed actions/rewards provided by Persistent Homology. It is referred to here as **PHIM** (**P**ersistent **H**omology **I**nformed actions and rewards for **M**onte-Carlo tree search). For real-world experiments, the pipeline obtains the initial configuration from perception and then planning via PHIM to produce a high-level sequence of operations that result in successful execution reliably. The solution is executed on the Baxter robot. This can be done either by performing the plan open-loop or optionally re-planning after each pushing action execution. Given the experimental evaluation,

PHIM achieves a significantly higher success rate in solving cluttered problems in constrained, shelf-like workspaces than alternatives, due to its long-horizon planning capability and solution robustness. Furthermore, plans produced by PHIM demonstrate are visually more natural and human-like.

In summary, this work contributes: **1) PHIM** (**P**ersistent **H**omology **I**nformed actions and rewards for **M**onte-Carlo tree search) is a principled framework that integrates topological tools and Monte Carlo methods to achieve long-horizon planning for object retrieval via non-prehensile actions.
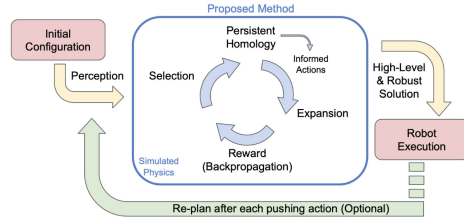

Fig. 2. Proposed framework integrating Persistent Homology for informed actions/rewards and Monte-Carlo tree search.

2) In comparison to existing methods, including methods that only employ topological tools but do not perform planning over multiple horizons, PHIM delivers significantly improved solution optimality in terms of the success rate and the number of actions taken. At the same time, PHIM incurs little extra computations cost.
3) PHIM demonstrated superior robustness to uncertainty (e.g., in comparison to [19], which lacks real robot experiments) as its plans can be directly executed on a real Baxtor robot with inherently motor uncertainty plus additional perception uncertainty.

2   Related Work

Working in a constrained and cluttered environment requires manipulation tasks, such as pick-n-place and object rearrangement. Prehensile manipulation generally assumes accurate pose estimation of objects, which is challenging for cluttered setups with action uncertainty. In contrast, non-prehensile actions can be used in a cluttered and constrained workspace with uncertainty to rearrange multiple objects using a single operation, thus allowing large-scale object manipulation [8]. Sometimes they are preferred over picking [5,12,13,17,18] since pushing can be executed with simpler and smaller end-effectors for constrained workspaces. Non-prehensile actions, however, are less predictable and increase uncertainty regarding object placement; this motivates solutions for estimating the outcome of pushing actions [7,23,24].

Previous works in reaching through clutter focused on recognizing objects to be relocated to allow a collision-free region to reach a target [10,11]. Uncertainty arising from occlusion or sensor noise complicates this problem, and efforts aim to minimize their effects by inferring object shape [16,21], reason about pose uncertainty [20], or using probabilistic filtering [15]. To achieve higher success rates, human interaction has been proposed to guide a high-level plan, which proposes an ordered sequence of objects to be pushed to approximate intermediate goal positions [14]. For faster kinodynamic planning, non-prehensile actions are allowed to have quasi-static interactions [4].

The proposed method takes advantage of persistent homology to inform the selection of efficient and robust pushing actions, which outperform, in terms of planning time and the number of actions, prior efforts in the same domain [4,14,19]. Moreover, it provides high-level solutions that are robust under uncertainty (pose, arm movements, non-prehensile manipulation of objects). Related problems, where the target has to be pushed to a desired goal, have been explored [1]. Some approaches learn an optimal policy from visual input [22] or aim to compute effective manipulation state/action sequences [3].

3   Problem Setup and Notation

Let $\mathcal{W} \subset \mathbb{R}^3$ be a bounded region with a cross-section given by a polygon and $\partial \mathcal{W}$ is its boundary, where *movable obstacles* $\mathcal{O} = \{o_1, \ldots, o_n\}$ and a target object $\mathcal{T}$ are approximated by cylinders residing inside $\mathcal{W}$. Assume that an arm with a gripper g can reach a target object at any
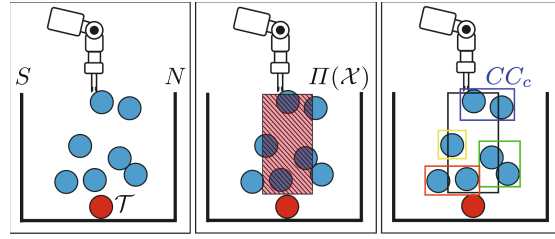

Fig. 3. (left) The workspace $\mathcal{W}$ and the robot arm for the configuration of Fig. 1, where $\mathcal{T}$ is the target. (middle) The path region $\Pi(\mathcal{X})$. (right) Connected components in $\Pi(\mathcal{X})$ using $r = 0.116$, where $CC_c = CC_c(\mathcal{X}, r)$ is the closest component to the gripper.

position in $\mathcal{W}$. Furthermore, the arm can access the interior of $\mathcal{W}$ from only one face of $\partial \mathcal{W}$, where due to limited accessibility, it cannot perform overhand grasps or lift the objects. Collisions between $\partial \mathcal{W}$ and the arm are not permitted. In other words, the robot respects the workspace constraints and may move along a 2D plane below the center of mass of the objects to avoid toppling them. The grasping of the target $\mathcal{T}$ requires the arm's last link to be collision-free with all obstacles, including the boundary $\partial \mathcal{W}$. The robot is a Baxter with a 7-DoF articulated robotic arm with a gripper attached at its end. The task is to reach and grasp the target object $\mathcal{T}$ with a minimal set of actions used to move obstacles blocking access to $\mathcal{T}$.

The configuration $\mathcal{X}$ is defined as $\mathcal{X} = \{\mathcal{P}, p_\mathcal{T}, s\} = \{(p_1, \ldots, p_n), p_\mathcal{T}, s\} \in \mathcal{W}^{n+1} \times SE(2)$, where $p_i \in \mathcal{W}$ defines the object o_i's coordinates and $p_\mathcal{T}$ defines the $\mathcal{T}$'s coordinate, respectively, while s is the position and orientation of the robot's gripper g. To simplify the notation, $\mathcal{X}[o_i] = p_i$ and $\mathcal{X}[\mathcal{T}] = p_\mathcal{T}$ represent that o_i and $\mathcal{T}$ are at position p_i and $p_\mathcal{T}$, respectively. $\mathcal{X}[g] = s$ represents that the gripper g of the robot is at pose $s \in SE(2)$. Let $V(p)$ be the subset of $\mathcal{W}$ occupied by an object at position p. A configuration $\mathcal{X} \in \mathcal{W}^{n+1} \times SE(2)$ is *feasible* if: no objects are overlapping; objects are not dropped, toppled, or deformed. The arm is allowed to push multiple objects simultaneously, and the resulting action must lead to a feasible configuration.

4   Method

Persistent homology is a data analysis tool that provides topological features of a space at different spatial resolutions. It is often applied for analysis of point-cloud data (e.g., see [2], and [9]). Briefly, it considers expanding balls of radius r centered at each point of a collection of points. As the balls grow, track the unions of all these balls as they overlap each other, given a 1-parameter family of spaces. For each radius r, one can build a Vietoris-Rips complex (abstract simplicial complex) using the information given by the intersection of the r-balls (for more details, see [9]). In this setting, persistent homology is the homology of the Vietoris-Rips complex as a function of r. Intuitively, persistent homology counts the number of connected components (object clusters) and holes of various dimensions and keeps track of how they change with parameters.
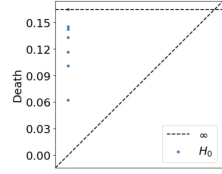

Fig. 4. The persistence diagram for the positions of the obstacles of Fig. 1.

To find clustering that persists, it is sufficient to focus on the zeroth-homology (zero-dim. homology) where the coefficient $\mathbb{Z}_2$ is ideal for computing connected components (clusters). While the radius r increases, the zero-dimensional persistent homology stores when the ball in one connected component first intersects a ball of a different connected component, merging both into one, see Fig. 5.

Non-prehensile Manipulation Approach. Given a target $\mathcal{T} \in \mathcal{O}$, denote $\Pi(\mathcal{X}) \subset \mathcal{W}$ as the *path region*, where the robot arm will push the obstacles to clear the path to the target. The idea is to use persistent homology to find clusters (connected components) inside $\Pi(\mathcal{X})$ and use this information to push objects. Since $\Pi(\mathcal{X})$ is smaller than the workspace, the arm has to fit inside $\Pi(\mathcal{X})$ to perform the action. Thus the shape and size of $\Pi(\mathcal{X})$ depends on $\mathcal{W}$ and the width w of the arm. For simplicity, $\mathcal{W}$ is considered to be a rectangular shelf described by the parallel walls of the shelf N and S as in Fig. 3.

To describe $\Pi(\mathcal{X})$, first select axis x to be associated with the depth of the shelf and y to correspond to the width of the shelf with $y = 0$ and $y = w$ for points at the south wall S and north wall N, respectively, where w is denoted by the width of the shelf. Then, $\Pi(\mathcal{X})$ is defined to be the rectangle given by $(\mathcal{X}[g]_x, \mathcal{X}[\mathcal{T}]_y - w)$ and $(\mathcal{X}[\mathcal{T}]_x, \mathcal{X}[\mathcal{T}]_y + w)$, where $\mathcal{X}[\mathcal{T}] = (\mathcal{X}[\mathcal{T}]_x, \mathcal{X}[\mathcal{T}]_y)$ and
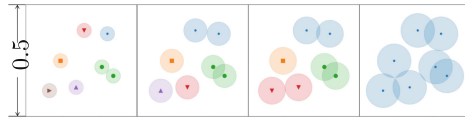

Fig. 5. Examples of clusters (connected components) that persist obtained for four different radii r in the persistence diagram in Fig. 4. (left) For $r = 0.062$, there are 6 different clusters shown by different markers. (left-middle) For $r = 0.1$, there are 4 clusters. (right-middle) For $r = 0.116$, two clusters. (right) Only one cluster for $r \geq 0.144$.

$\mathcal{X}[g] = (\mathcal{X}[g]_x, \mathcal{X}[g]_y)$, see Fig. 3. When the target is too close to the walls S and N, it is possible to apply a planar rotation matrix $R_{-\phi}$ to obtain $\Pi(\mathcal{X})$ with incidence angle ϕ as in [19].

Having $\Pi(\mathcal{X})$, obstacles can be clustered by applying persistent homology inside $\Pi(\mathcal{X})$. Define $\mathcal{CC}(\mathcal{X}, r)$ to be the collection of connected components (clusters) for the radius $r > 0$. It is enough to use the zero-dimensional persistent homology to obtain the connected components and their generators. The goal is for the arm to try to push all obstacles in the closest connected component, denoted by $\mathcal{CC}(\mathcal{X}, r)_c$, to the griper $\mathcal{X}[g]$. The performed action a is successful when all obstacles in $\mathcal{CC}(\mathcal{X}, r)_c$ are removed from $\Pi(\mathcal{X})$.

Due to the unpredictability of a non-prehensile action a, it is advantageous to bestow a reward on each action performed based on the number of obstacles removed from the path region and the number of connected components created by the pushing action a. The reward function rwd in MCTS for a given action a is defined by $\mathrm{rwd}(a) = b(a) + t(a) \cdot m(a)$, where $b(a) = \#\Pi(ch(\mathcal{X}, a)) - \#\Pi(\mathcal{X})$ is the number of obstacles removed from the path region; $m(a) = \max(\#\mathcal{CC}(ch(\mathcal{X}, a), r) - \#\mathcal{CC}(\mathcal{X}, r), 0)$ is the number of connected components created by the action a; and $t(a) = 1$ if $b(a) > 0$ or 0 elsewhere. Note that the term $m(a)$ adds a positive reward for the configuration that increases the number of clusters (connected components), resulting in a configuration of sparse obstacles that leads to subsequent easier pushing actions.

The pushing actions are based on the circumscribed rectangle that contains $\mathcal{CC}(\mathcal{X}, r)_c$, denoted by $\mathcal{R}(\mathcal{X}, r)$. The actions are either by a sweeping movement from the bottom of $\mathcal{R}(\mathcal{X}, r)$ to the top of $\mathcal{R}(\mathcal{X}, r)$ or from top to bottom of $\mathcal{R}(\mathcal{X}, r)$, denote the action by up and down, respectively.

Observe that for a given configuration $\mathcal{X}$, each radius $r > 0$ may define a different $\mathcal{R}(\mathcal{X}, r)$, and without a proper discretization of the radii, the tree will grow exponentially. The persistence diagram provides an informed manner to select radii that lead to robust actions and bounds the exponential growth. The radii that present such properties are the collection of radii that persist under a selected hyper-parameter ν, more specifically, a *persistent radius* for a given parameter $\nu > 0$ is a radius $r > 0$ such that the number of connected components in the persistence diagram between r and $r + \nu$ remains the same. Denote R_ν by the set of all persistent radii for a given value $\nu > 0$. In the persistence diagram in Fig. 4, $R_{0.015} = \{0.062, 0.1, 0.116, 0.144\}$. Note that R_ν always contains the radius where the last connected component dies, thus $R_\nu \neq \emptyset$.

Another important parameter is the gripper width h, where for radii less than h the gripper cannot position between two clusters, thus R_ν can be further decreased by removing all radii less than h from it, denote this new collection of radii by $R(\mathcal{X}) = R_{\nu,h}(\mathcal{X})$. For example, in the persistence diagram of Fig. 4, $R_{\nu,h}(\mathcal{X}) = \{0.062, 0.1, 0.116, 0.144\}$ for $\nu = 0.015$ and $h = 0.05$. In [19], there are two proposed algorithms: PHIA takes the minimal radius of $R_{\nu,h}$ and performs the pushing actions, and PHIS builds a tree with nodes being the configurations and propagates by performing the pushing actions for each radius in $R_{\nu,h}$ then it selects the path that leads to success and minimizes the number of actions.

Given a configuration $\mathcal{X}$, the set of all possible actions is represented by $R(\mathcal{X}) \times \{\mathsf{up}, \mathsf{down}\}$, hence the expansion stage of MCTS for $\mathcal{X}$ produces children $ch(\mathcal{X}, a)$ where $a = (r, d) \in R(\mathcal{X}) \times \{\mathsf{up}, \mathsf{down}\}$, r a radius and d a direction.

Actions a may result in failure either by dropping/toppling obstacles or pressing obstacles against the wall. In all cases, a child node obtained by unfeasible action is marked as a failure, and no reward is given. The framework that integrates the informed actions and rewards by Persistent Homology and MTCS is named in short by **PHIM** (**P**ersistent **H**omology **I**nformed actions and rewards to **M**onte-Carlo tree search). For sample pushing actions planned by PHIM, see Fig. 1.

5   Experiments

To evaluate the proposed method, 25 real-world and 50 simulated experiments were performed and compared against baselines as described below. The real-world scenarios are described by S1, S2, S3, S4, and S5 in Figs. 6 and 1. The simulated experiments are labeled M1 through M10 as depicted in Fig. 7.

Experimental Setup: The simulated experiments were run on a Ubuntu workstation with *Intel Core i5-8259U 3.8* Ghz *16* GB *RAM*. Gazebo was used for simulating the actions, and MoveIt was used for forming motion planning queries to the Bi-EST planner [6] in OMPL. For real-world experiments, the Baxter robot was used to perform the pushing actions and grasp the target. The choice of Baxter was ideal for the real experiment since it performs the moving actions with a small degree of imprecision, which helps to show the robustness of the proposed approach. An overview of the workspace is provided by an Intel Realsense D435 RGB-D camera. For each object, its position is detected with a 2D fiducial marker at the top using Chilitags.

Fig. 6. (top row) the randomized baselines fail to find a plan; (bottom left - S4) GRTC-H fails to find a plan, and OOA fails to execute the plan; (bottom right - S5) GRTC-H, OOA, and PHIA perform the most actions. Scene S2 is given in Fig. 1.

Choice of Parameter: 1 target and 7 obstacles with enough height to have the center of mass above the contact points performed by the pushing actions of Baxter's arm. Bricks are used to create artificial walls to simulate a shelf workspace with a desirable size, and it allows recording and running perception over the table. The gripper can lift the target a bit to perform the retrieve action. However, it is not allowed to retrieve while passing over the obstacles. The objects have enough weight to provide friction and simulate the real grasping.

Fig. 7. Randomized baselines M2 and M5 in simulation with varying shape obstacles.

The hyper-parameter ν for the persistent homology algorithms is selected to be 0.015 m, the margin of error for the arm's pushing actions, which is empirically chosen to be the difference between the desirable position minus the actual position after performing the actions.

Another useful hyper-parameter to consider is the width of the gripper h. The selection of h dictates if the gripper can be successfully fitted between two clusters (connected components). During the experiments, h is considered the real width plus an extra margin of error. Thus, h is selected to be greater than $0.03 + 0.015$ m ($h = 0.05$ m). Furthermore, for the exploration parameter in the Monte-Carlo tree search, the conventional choice of $\sqrt{2}$ is chosen. The overall planning time threshold for all methods is 500 s. Above this threshold, failure is reported and real-world experiment is not performed.

Collisions: Soft collisions between obstacles and walls are allowed. Nevertheless, the simulated actions prevent hard collisions and the arm pressing the obstacles against the wall. In fact, actions that resulted in the arm pressing the obstacles were the only issue presented when the simulated experiments were performed in real-world scenarios.

It was surprising that the challenge of avoiding dropping and toppling was easily solved. The proposed solution to fix pressing objects against the wall is to consider the walls in a simulated environment as movable objects with five times the weight of one obstacle. Any action resulting in moving the wall is

Fig. 8. Real-world actions (top) and planning time (bottom) performed by each method for scenes S1 to S5 of Figs. 1 and 6, for GRTC-H, OOA, PHIA, PHIS, and PHIM (ours), where applicable. All PHI* methods do reasonably well; PHIM always provides the best solution quality.

considered unfeasible. Incorporating this extra condition increased the chances of failing during the planning phase. However, experiments showed that the PHIM approach could find high-level solutions with minimal pushing actions without significantly increasing planning time.
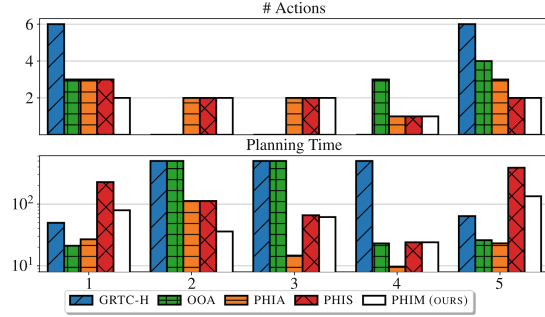
Comparisons: The proposed PHIM is compared against PHIA, PHIS, OOA [19] and the modified GRTC-Heuristic [14]. All algorithms were adapted to prevent the arm from pressing the objects against the wall. Without this extra condition, PHIA, OOA, and GRTC-Heuristic fail in all real-world experiments. This additional check condition does not require changing the algorithms. The algorithm OOA is a simple approach to push each obstacle one by one in the path region $\Pi(\mathcal{X}_0)$ of the initial configuration $\mathcal{X}_0$. And modified GRTC-Heuristic (GRTC-H) is based on the algorithm used in [14] that pushes obstacles in a straight line to their goal region outside of the path region $\Pi(\mathcal{X}_0)$, where the goal is randomly selected to be outside of $\Pi(\mathcal{X}_0)$ without overlapping any other obstacles. Both algorithms are randomized approaches to execute pushing actions without using clustering information from the persistent homology.

Robustness: To evaluate the robustness of the proposed method, the real-world experiments had the position of each object slightly different from the positions given to the planner to solve, adding a noise of 3cm to the tag-based position. Moreover, the planner simulated the whole collection of pushing actions to solve the retrieval problem without re-planning after every step. In other words, the planner solved the problem in a simulated world in an offline manner. This choice of offline planning is motivated by the robustness of the algorithms based on persistent homology.

Discussion: The success rate for GRTC-H is comparatively low, mainly due to taking a long time planning feasible actions (see Fig. 8). The planning time for OOA is the second fastest as it does not perform a search and only finds the closest obstacle to the gripper. However, it has a high chance of failing during execution as it is not robust. When it pushes an obstacle and if another one is near, it can push them together, thereby changing the configuration for the next action. Thus, it results in a lower success rate.
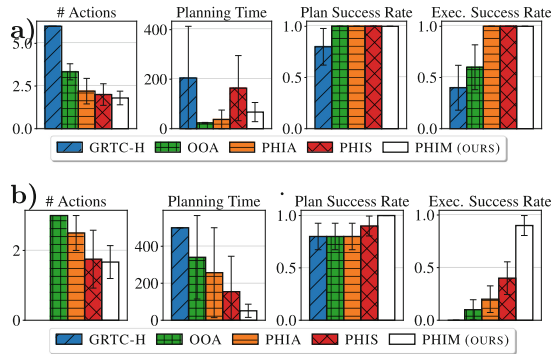

PHIM plans fewer actions than the other methods and has a 100% success rate either in simulation or in real-world execution per Fig. 9. It achieves such performance without substantially increasing the planning time, showing that the rewards based on information collected by persistent homology are quite useful. In summary, when comparing all methods, PHIM accomplishes the most robust actions as well as

Fig. 9. Left to right: average number of actions, planning time, planning success rate, and execution success rate for **a)** real-world and **b)** simulated experiments. PHIM provides the best solution quality and much faster than PHIS.

reduces the number of actions needs to solve the problem without significantly increasing the planning time.

6   Conclusion

This paper introduces a robust and efficient method for planning non-prehensile manipulation actions in clutter. It uses topological tools and MCTS to obtain high-quality solutions for object retrieval problems. The solutions are robust under perturbations and gracefully handle the uncertainty associated with objects' poses and arm actions. Furthermore, it allows to successfully plan offline actions using physics simulations that are successfully executed online. Real-world experiments show that PHIM has a higher success rate and is faster in finding robust solutions than alternatives. It reduces the number of pushing actions required and decreases planning time by avoiding the consideration of actions that result in time-expensive simulated tasks.

The proposed framework can be directly applied to highly dense clusters of objects. However, such a scenario is expected to perform similarly to the other methods since highly dense setups have a trivial topological shape to explore (fewer ways to cluster). A possible direction to be explored is to select obstacles to be removed by pick-n-place tasks, resulting in easier follow-up pushing actions to clean the path to the target.

References

1. Cosgun, A., Hermans, T., Emeli, V., Stilman, M.: Push planning for object placement on cluttered table surfaces. In: IROS 2011, pp. 4627–4632. IEEE (2011)
2. Edelsbrunner, H.: A Short Course in Computational Geometry and Topology. SAST, Springer, Cham (2014). https://doi.org/10.1007/978-3-319-05957-0
3. Haustein, J.A., Arnekvist, I., Stork, J., Hang, K., Kragic, D.: Learning manipulation states and actions for efficient non-prehensile rearrangement planning. arXiv preprint arXiv:1901.03557 (2019)
4. Haustein, J.A., King, J., Srinivasa, S.S., Asfour, T.: Kinodynamic randomized rearrangement planning via dynamic transitions between statically stable states. In: ICRA 2015, pp. 3075–3082. IEEE (2015)
5. Havur, G., Ozbilgin, G., Erdem, E., Patoglu, V.: Geometric rearrangement of multiple movable objects on cluttered surfaces: a hybrid reasoning approach. In: ICRA 2014, pp. 445–452. IEEE (2014)
6. Hsu, D., Latombe, J.C., Motwani, R.: Path planning in expansive configuration spaces. In: ICRA 1997, vol. 3, pp. 2719–2726. IEEE (1997)
7. Huang, B., Han, S.D., Boularias, A., Yu, J.: DIPN: deep interaction prediction network with application to clutter removal. In: ICRA 2021 (2021)
8. Huang, E., Jia, Z., Mason, M.T.: Large-scale multi-object rearrangement. In: ICRA 2019, pp. 211–218. IEEE (2019)
9. Kaczynski, T., Mischaikow, K., Mrozek, M.: Computational Homology. AMS, vol. 157. Springer, New York (2004). https://doi.org/10.1007/b97315

10. Lee, J., Cho, Y., Nam, C., Park, J., Kim, C.: Efficient obstacle rearrangement for object manipulation tasks in cluttered environments. In: ICRA 2019, pp. 183–189. IEEE (2019)
11. Nam, C., Lee, J., Cho, Y., Lee, J., Kim, D.H., Kim, C.: Planning for target retrieval using a robotic manipulator in cluttered and occluded environments. arXiv preprint arXiv:1907.03956 (2019)
12. Nieuwenhuisen, M., et al.: Mobile bin picking with an anthropomorphic service robot. In: ICRA 2013, pp. 2327–2334. IEEE (2013)
13. Pan, Z., Hauser, K.: Decision making in joint push-grasp action space for large-scale object sorting. arXiv preprint arXiv:2010.10064 (2020)
14. Papallas, R., Dogar, M.R.: Non-prehensile manipulation in clutter with human-in-the-loop. In: ICRA 2020, pp. 6723–6729. IEEE (2020)
15. Poon, J., Cui, Y., Ooga, J., Ogawa, A., Matsubara, T.: Probabilistic active filtering for object search in clutter. In: ICRA 2019, pp. 7256–7261. IEEE (2019)
16. Price, A., Jin, L., Berenson, D.: Inferring occluded geometry improves performance when retrieving an object from dense clutter. arXiv preprint arXiv:1907.08770 (2019)
17. Shome, R., et al.: Towards robust product packing with a minimalistic end-effector. In: ICRA 2019, pp. 9007–9013. IEEE (2019)
18. Song, C., Boularias, A.: Object rearrangement with nested nonprehensile manipulation actions. In: IROS 2019, pp. 6578–6585. IEEE (2019)
19. Vieira, E.R., Nakhimovich, D., Gao, K., Wang, R., Yu, J., Bekris, K.E.: Persistent homology for effective non-prehensile manipulation. In: ICRA 2022, pp. 1918–1924 (2022)
20. Wang, R., Mitash, C., Lu, S., Boehm, D., Bekris, K.E.: Safe and effective picking paths in clutter given discrete distributions of object poses. In: IROS 2020, pp. 5715–5721. IEEE (2020)
21. Wong, L.L., Kaelbling, L.P., Lozano-Pérez, T.: Manipulation-based active search for occluded objects. In: ICRA 2013, pp. 2814–2819. IEEE (2013)
22. Yuan, W., Stork, J.A., Kragic, D., Wang, M.Y., Hang, K.: Rearrangement with nonprehensile manipulation using deep reinforcement learning. In: ICRA 2018, pp. 270–277. IEEE (2018)
23. Zhou, J., Hou, Y., Mason, M.T.: Pushing revisited: differential flatness, trajectory planning, and stabilization. Int. J. Robot. Res. **38**(12–13), 1477–1489 (2019)
24. Zhou, J., Mason, M.T., Paolini, R., Bagnell, D.: A convex polynomial model for planar sliding mechanics: theory, application, and experimental validation. Int. J. Robot. Res. **37**(2–3), 249–265 (2018)