# Leveraging 3D Reconstruction for Mechanical Search on Cluttered Shelves

**Anonymous Author(s)**
Affiliation
Address
email

**Abstract:** Finding and grasping a target object on a cluttered shelf is a challenging manipulation skill. When the target is occluded by other unknown objects and initially not visible, the robot needs to reposition the obstructing objects by pushing or pick-and-place actions. Additionally, a narrow shelf space filled with multiple objects leads to a very small collision-free space, limiting the manipulator's workspace. Existing works for the most part focus on developing specialized tools with suction grippers to overcome these challenges. In this paper, we use a standard two-finger gripper and 6-DoF grasping for pick-and-place, and develop a mechanical search algorithm based on 3D reconstruction of the scene. Specifically, we define two functions, an existence function and a graspability function, and formulate a model-based optimal control problem. To estimate the two functions and construct approximate dynamics models for them, we exploit a pre-trained pushing dynamics model, depth rendering, and collision detection, which are all based on the 3D reconstruction results. Our methods not only show comparable performance to when objects are assumed to be known in simulation but also, through real-world experiments, show the real-world applicability and robustness of our method.

**Keywords:** Mechanical search, Object rearrangement, Prehensile and Non-prehensile manipulation

## 1 Introduction

Finding and grasping the desired target object on a cluttered shelf – where the target is occluded by other unknown objects and initially not visible to a vision sensor – is a challenging manipulation skill. The robot needs to reposition obstructing objects to identify the target's location and then grasp it while avoiding collisions with shelves and other objects. This problem poses a significant challenge due to the shelf's geometric characteristics, which allow visual observation only from the front direction and restrict the workspace of the manipulator.

Previous research on finding and grasping has primarily focused on (i) less challenging environments such as a flat table or bin [1, 2, 3, 4, 5, 6, 7, 8] or (ii) a more complex shelf environment but with restrictive assumptions that all object shapes and locations are known (e.g., assuming the shelf's inside can be seen through its transparent top) [9, 10, 11, 12, 13, 14, 15, 16]. Recent works have considered more practical shelf environments where non-target objects are unknown – which are referred to as the *mechanical search* –, but they rely on tailored long stick-shaped tools to manipulate objects in a narrow shelf space [17, 18, 19]. The task of finding and grasping the target object in the challenging cluttered shelf environment using a standard robot gripper, without specialized tools, remains an unresolved problem.

In this paper, to the best of our knowledge, we are the first to propose a practical method for finding and grasping the target on the cluttered shelf filled with unknown objects, by using the standard

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

robot gripper (see Figure 1). As in other studies, we employ two types of actions: pushing and pick-and-place actions. However, our methods differ from previous studies [18, 19] where specialized tools with suction grippers are developed. Instead, we use the standard two-finger gripper and apply 6-DoF grasping for picking, which introduces novel challenges in our research.

In particular, when implementing pick-and-place actions, the most challenging part is the ignorance of the environment given only a partial visual observation. Because the shelf space is not only very narrow but also filled with multiple unknown objects, identifying the collision-free space and generating collision-free robot grasping paths – which include a path to approach an object and a path out from the shelf while holding the object – is very challenging. In this regard, the noticeable difference between our method compared to the existing methods is that we exploit a pre-trained 3D recognition model and use the 3D reconstruction results for the mechanical search.

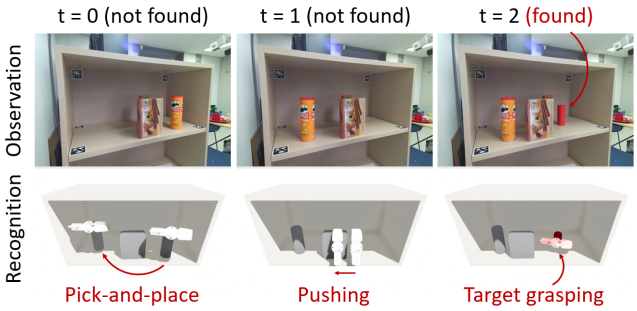


Figure 1: A 3D recognition-based mechanical search and grasping of the target object (red cylinder).

In this paper, we propose a general framework for mechanical search and grasping and provide practical algorithms where we leverage a 3D object recognition model. Specifically, we introduce two indicator functions, denoting the target object's candidate pose by $x \in \mathrm{SE}(3)$, (i) an existence function $f(x)$ that indicates if the target can exist at $x$ and (ii) a graspability function $g(x)$ that indicates if the target at $x$ is graspable. Then our framework is formulated to rearrange the surrounding objects until we have only one existable and graspable candidate pose $x$, i.e., $f(x) = 1$ and $g(x) = 1$ at only one $x \in \mathrm{SE}(3)$. We approach this problem as the model-based optimal control, assuming the dynamics models for the two functions $f_{t+1} = \mathcal{F}(f_t, a_t), g_{t+1} = \mathcal{G}(g_t, a_t)$ – where $a_t$ represents the robot action – are given, for a suitably-designed objective function.

To construct approximate dynamics models $\mathcal{F}, \mathcal{G}$ and estimate the functions $f_t, g_t$ from a partial visual observation $o_t$, we use the recently-introduced 3D object recognition model based on the *superquadric* primitives [20]. By placing the estimated superquadrics on the simulation such as the PyBullet and using the pre-trained pushing dynamics model [21], we obtain approximate dynamics models $\hat{\mathcal{F}}, \hat{\mathcal{G}}$ and function estimates for $f_t, g_t$. To avoid the accumulation of estimation errors when solving the optimal control, we use the model predictive control with a short time horizon, solving iterative optimizations.

Experiments on both simulation and real-world environments confirm the effectiveness of our 3D reconstruction-based find and grasp methods. One reasonable concern is that the success rate of our method will greatly depend on the recognition results. Through extensive experiments in simulation, we verify that our methods show similar performance compared to the case where all the information about the objects is assumed to be known. Additionally, we verify the real-world applicability and robustness of our method despite the presence of noise in visual information in real-world environments.

## 2 A General Framework for Mechanical Search and Grasping

In this section, we propose a general framework for mechanical search and provide two specific algorithms: (i) search-and-grasp and (ii) search-for-grasp, using the model-based optimal control formulation. The search-and-grasp method executes actions first to find the target and then executes additional actions to rearrange the objects to make the target graspable, while the search-for-grasp method integrates the search and grasp processes into a cohesive framework so that the target object's graspability is taken into account in the search phase.

Our focus lies specifically on scenarios where the target object is known – where its information is given as the color and geometry – and is fully occluded by other unknown objects. We assume that the robot can interact with objects either by pushing or pick-and-place actions to rearrange the objects. When the robot performs the pick-and-place action, it is only allowed to place an object in another empty space on the shelf (i.e., object stacking is not allowed). Also, when the robot rearranges the surrounding non-target objects, the target object is assumed to remain stationary. The goal of the method is, by rearranging the surrounding objects, to make the target object graspable not only visible.

We denote the target object's 6-dof pose by $x \in \mathrm{SE}(3)$. Instead of dealing with the continuous pose space $\mathrm{SE}(3)$, to simplify numerical computations, we restrict our attention to a finite subset $\mathcal{X} \subset \mathrm{SE}(3)$. The two core components in our framework are (i) an existence function $f : \mathcal{X} \to \{0, 1\}$ and (ii) a graspability function $g : \mathcal{X} \to \{0, 1\}$. The existence function $f(x)$ indicates whether the target object can be present at the pose $x$ or not. For example, given an observation shown in Figure 2 (*Upper*), consider two candidate target poses $x \in \mathrm{SE}(3)$ (i.e., the red cylinders in Figure 2 *Lower*). For $f(x)$ to be 1, the rendered image must match the observation (there are other conditions as well, details are in the next section), otherwise, $f(x) = 0$. In practice, considering the discretization resolution of $\mathcal{X}$, $f(x)$ is considered to be 1 if the target object pose can exist near $x$. Naturally, it captures the uncertainty of the target object's pose because $\sum_{x \in \mathcal{X}} f(x)$ represents the number of possible object poses; the greater $\sum_{x \in \mathcal{X}} f(x)$, the more uncertain. We assume $\mathcal{X}$ is sufficiently densely discretized so that $\sum_{x \in \mathcal{X}} f(x)$ is lower bounded by 1 (i.e., the target object must exist in at least one $x \in \mathcal{X}$). The graspability function $g(x)$ indicates whether the target object at the pose $x$ is graspable or not.

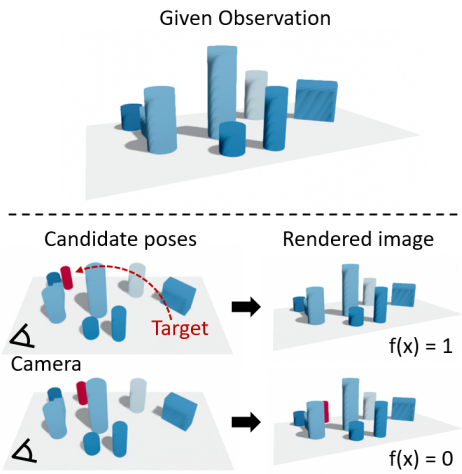


Figure 2: Illustration on the existence function $f(x)$. *Upper*: Observation. *Lower*: Candidate poses and hypothetical rendering results.

Denote the existence and graspability functions at (discrete) time-step $t$ by $f_t(x)$ and $g_t(x)$ and a pushing or pick-and-place action by $a_t$, we assume that the dynamics models for $f_t, g_t$ are given and denoted by $\mathcal{F}$ and $\mathcal{G}$ such that $f_{t+1}(x) := \mathcal{F}(f_t, a_t)(x)$ and $g_{t+1}(x) := \mathcal{G}(g_t, a_t)(x)$. Additionally, we assume that the uncertainty of the target object pose is not increasing in $t$, i.e., $\sum_{x \in \mathcal{X}} f_t(x) \geq \sum_{x \in \mathcal{X}} \mathcal{F}(f_t, a_t)(x)$, regardless of $a_t$. This non-increasing uncertainty assumption comes from the prior assumption that the target object does not move when an action is applied because it is impossible for an object suddenly can exist where it could not originally exist.

Given initial $f_0, g_0$ and dynamics models $\mathcal{F}$ and $\mathcal{G}$, we formulate a **search-for-grasp method** as a model-based optimal control. Specifically, the goal is to find a sequence of actions $a_0, \ldots, a_T$ for a fixed time $T$, so that $f_T(x^*) = 1$ for only one $x^* \in \mathcal{X}$ – if there are more than one, then we cannot specify at which $x$ the target object exists – and that is graspable, i.e., $g_T(x^*) = 1$. For this purpose, we formulate the following model-based optimal control problem:

$$\min_{a_1, \ldots, a_T} \sum_{x \in \mathcal{X}} f_T(x) + \alpha f_T(x)(1 - g_T(x)), \tag{1}$$

such that $f_{t+1}(x) = \mathcal{F}(f_t, a_t)(x)$ and $g_{t+1}(x) = \mathcal{G}(g_t, a_t)(x)$ and $\alpha$ is a hyperparameter. Minimizing the first term in (1) makes $f_t$ have only one $x^*$ such that $f_T(x^*) = 1$ – since the existence function is lower bounded by 1 –, and minimizing the second term enforces $g_T(x^*) = 1$. Without the second term, even though the target object becomes visible, there is no guarantee that it will be graspable.

In practice, we prefer to perform a minimum number of actions needed for the task, rather than for a pre-determined fixed length $T$. We introduce a discount factor $\gamma \in (0, 1]$ and propose a modified version of the optimal control problem:

$$\min_{a_1,\ldots,a_T} \sum_{t=0}^{T-1} \gamma^t (\mathcal{J}_{t+1} - \mathcal{J}_t) = \gamma^T \mathcal{J}_T + \frac{1-\gamma}{\gamma} \sum_{t=1}^{T} \gamma^t \mathcal{J}_t - \mathcal{J}_0, \tag{2}$$

where $\mathcal{J}_t := \sum_{x \in \mathcal{X}} f_t(x) + f_t(x)(1 - g_t(x))$. As $\gamma \to 1$, the new objective function converges to the original one in (1) since $\mathcal{J}_0$ is constant. For $\gamma < 1$, $\mathcal{J}_t$ for $t < T$ are now taken into account, and thus minimizing (2) leads to accomplishing the task quickly. A $\gamma$ that is too small can make the problem difficult, so an appropriate level of $\gamma$ needs to be found.

Additionally, we propose a computationally lighter version, a **search-and-grasp method**. In the cost term $\mathcal{J}_t$ in (2), the computation cost of $g_t(x)$ is very high since it requires the planning of multiple collision-free paths (as we explain in the next section in more details). And $g_t(x)$ must be evaluated for all $x$ such that $f_t(x) = 1$, thus as $\sum_x f_t(x)$ is bigger, the computation cost increases. In the search-and-grasp method, we first find the target by using the first term only $\mathcal{J}_t = \sum_x f_t(x)$. Then, after the target is found at $x^*$, we use the second term $\mathcal{J}_t = 1 - g_t(x^*)$ to rearrange the objects to make the target graspable. Since the target pose is already specified at the search phase, in the second stage, we need to compute the graspability function $g_t$ only at one $x^*$, which significantly reduces the total computation time.

# 3 3D Object Recognition-based Mechanical Search

In this section, as one practical way to implement our framework, we propose an object recognition-based approach. We assume that a scene contains multiple unknown objects and a known target object, and an RGB-D camera is fixed in a specific position and can only capture one side of the scene (e.g., a shelf containing objects from the front view). From the partial view information of the scene, our strategy is to first recognize the shapes and poses of the objects and then to use the recognition results to estimate the existence and graspability functions $f, g$ and the dynamics models $\mathcal{F}$ and $\mathcal{G}$.

## 3.1 Object Recognition via Superquadrics

In this paper, we employ the shape class called the superquadrics, a family of geometric shapes which represent diverse shapes ranging from boxes, cylinders, and ellipsoids to other symmetric complex shapes. The implicit function of a superquadric surface at the canonical pose (i.e., $T = I_{4\times4}$ is the identity matrix) has the following form:

$$\left( \left| \frac{x}{a_1} \right|^{\frac{2}{e_2}} + \left| \frac{y}{a_2} \right|^{\frac{2}{e_2}} \right)^{\frac{e_2}{e_1}} + \left| \frac{z}{a_3} \right|^{\frac{2}{e_1}} - 1 = 0, \tag{3}$$

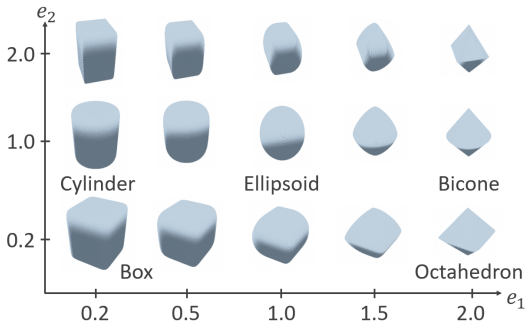

Figure 3: Examples of superquadrics.

where $q = (a_1, a_2, a_3, e_1, e_2) \in \mathbb{R}^5$ is the shape parameter; $a_1, a_2, a_3$ controls the sizes and $e_1, e_2$ controls the geometric shapes (some examples are shown in Figure 3). Given a pose of the object $T \in \mathrm{SE}(3)$, the implicit surface equation is transformed accordingly. Recently, given only a partial observation of an object, a model that predicts the 3d full shape of the object by using the superquadric representation has been proposed [20]; we use this model for 3D object recognition. We include details about the model architectures and the training details in Appendix C.

### 3.2 Existence and Graspability Function Estimates

Given a partial visual observation $o$, let a set of estimated superquadric shape parameters and poses be denoted by $s = \{(q^i, T^i)\}$. Further, we assume that a classification result on whether the target object is visible or not in $o$ is given and denoted by $c \in \{0, 1\}$ – if visible (i.e., $c = 1$), we assume that the target object's pose can be estimated accurately. Using these estimates $s, c$, we construct the estimates for the existence and graspability functions denoted by $\hat{f}(x; s, c)$ and $\hat{g}(x; s)$.

The key idea is to locate the recognized superquadric objects in the simulation, as well as the target object at a hypothetical pose $x \in \mathcal{X}$. Then we perform depth rendering, collision detection, and grasp planning to construct $\hat{f}(x; s, c)$ and $\hat{g}(x; s)$. Specifically, if $c = 1$, the existence function $\hat{f}$, since we know at which $x^*$ the target object exist, is 1 only at $x^*$, i.e., $\hat{f}(x^*; s, c = 1) = 1$ and 0 at other $x \in \mathcal{X}$. If $c = 0$, $\hat{f}(x; s, c = 0)$ is defined to be 1 if (i) depth rendering results with and without the target object at $x$ are identical and (ii) there is no collision between the recognized objects, the environment, and the target object. Otherwise $\hat{f}(x; s) = 0$. The graspability function $\hat{g}(x; s)$ is defined to be 1 if we can find a collision-free grasping trajectory of the robot gripper, where all possible collisions between the robot arm, gripper, environment, and multiple objects should be taken into account.

### 3.3 Approximate Dynamics Models $\hat{\mathcal{F}}$ and $\hat{\mathcal{G}}$

In this section, we construct the approximate dynamics models for $f_t$ and $g_t$ by using (i) the dynamics of $s_t$ denoted by $s_{t+1} = \mathcal{S}(s_t, a_t)$ – where we use the PyBullet simulation for the pick-and-place action and pre-trained SE(2)-equivariant pushing dynamics model for pushing action [21] – and (ii) the function estimates $\hat{f}, \hat{g}$ in Section 3.2. Since $\hat{f}$ takes the classification label $c$ as an input, we need a dynamics model for $c_t$, but it is hardly possible to know if the target will be visible or not in the future given an action $a_t$. We take a conservative strategy and assume that the visibility of the target object does not change at $t + 1$, i.e., $c_{t+1} = c_t$. Then, for $f_t$ and $g_t$ at time $t$, the approximate dynamics models are defined as follows:

$$\mathcal{F}(f_t, a_t)(x) := f_t(x)\hat{f}(x; s_{t+1}, c_{t+1}); \qquad \mathcal{G}(g_t, a_t)(x) := \hat{g}(x; s_{t+1}), \tag{4}$$

where $s_{t+1} = \mathcal{S}(s_t, a_t)$ and $c_{t+1} = c_t$. The existence function $\mathcal{F}(f_t, a_t)(x)$ is 1 if both $f_t(x)$ and $\hat{f}(x; s_{t+1}, c_{t+1})$ are 1, which guarantees the non-increasing uncertainty condition for $\mathcal{F}$, i.e., $\sum_{x \in \mathcal{X}} f_t(x) \geq \sum_{x \in \mathcal{X}} \mathcal{F}(f_t, a_t)(x)$.

### 3.4 Sampling-based Model Predictive Control

Given an initial observation $o_0$ and function estimates $f_0(x) = \hat{f}(x; s_0, c_0), g_0(x) = \hat{g}(x; s_0)$, by using the approximate dynamics models $\hat{\mathcal{F}}, \hat{\mathcal{G}}$, we can solve the optimal control problem in (2). However, since both the function estimates and dynamics models have estimation or numerical errors, optimizing $a_1, \ldots, a_T$ at once is susceptible to the accumulated errors. Instead, we perform a model predictive control where we iterate the following procedure for $T$ times: for $t = 0, \ldots, T-1$ (i) update initial estimates of $f_t, g_t$ from a new observation $o_t$, (ii) solve (2) with a short time horizon $M < T$, and (iii) take only the first action $a_t$.

**Implementation Details:** To solve the iterative optimization in MPC (where we set $T = 10$), we use $M = 3$ and take a sampling-based approach. The actions space $\mathcal{A}$, whose elements are either pushing or pick-and-place actions, is defined based on the object recognition results and as a discrete set that can be efficiently searched through sampling (details are included in Appendix B). We stop the MPC iteration if the target is visible and graspable, so the total number of actions can be less than $T$. The objective function is slightly modified so that, in $\mathcal{J}_t = \sum_{x \in \mathcal{X}} f_t(x) + f_t(x)(1 - g_t(x))$, the graspability function $g_t(x)$ – which originally could take 0 or 1 – now can take $1, 0, -1, -2, -3, \ldots$ where $g(x) = -n$ indicates that generated grasping trajectories for the target at $x$ collides with at least $n - 1$ objects. This modified objective function provides more dense signals, making the optimization problem easier to solve, while not changing the optimal solution.

# 4 Experiments

In this section, we conduct a comparative analysis of the two proposed methods, the search-and-grasp and search-for-grasp methods, and verify that both methods show robust performance to find and grasp the occluded target object in both simulation and real-world environments.

**Environment.** The simulation and real-world experiments share the same robot, the same camera, and the same shelf environment; the simulation experiments are conducted by the Pybullet simulator. We use the 7-dof Franka Emika Panda robot with a parallel-jaw gripper and an Azure Kinect DK camera sensor looking into the shelf at a fixed location (See Figure 4). The raw input visual observation from the camera is an RGB-D, which is then converted to point cloud data. In the simulation environment, we assume that the ground-truth segmentation masks of objects are available; for real-world experiments, we train a depth image-based segmentation network. We use cylinder-shaped and cube-shaped objects of various sizes which are visualized in Figure 4. All objects are assumed to stand upright on the shelf. Throughout the experiments, we restrict our attention to a cylindrical target object. Considering the rotational symmetry of the target object and the assumption that the target object stands upright, the target pose space $\mathcal{X}$ can be reduced to $\mathbb{R}^2$.

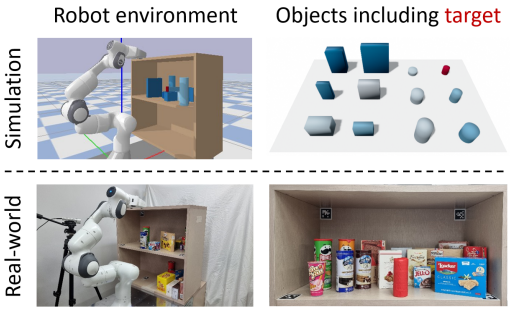


Figure 4: The left column shows the simulation and real environments, and in the right column, objects used in each environment are visualized. In particular, the target object is marked in red in the simulation; the red-taped can is the target object in the real experiment.

**Action space.** The action space $\mathcal{A}$ is composed of either pushing or pick-and-place actions. A pushing action is defined by $(T_{\text{push}}, d) \in \text{SE}(3) \times \mathbb{R}^3$, where $T_{\text{push}} \in \text{SE}(3)$ is an initial gripper pose and $d \in \mathbb{R}^3$ is a displacement of the gripper with a fixed orientation. A pick-and-place action is defined by $(T_{\text{grasp}}, T_{\text{place}}) \in \text{SE}(3) \times \text{SE}(3)$, which are gripper poses when it grasps and places the object, respectively. Specifically, the gripper first approaches $T_{\text{grasp}}$ straightforwardly in the direction the gripper faces the front, grasps the object and retrieves back along the same route of approaching. A similar process is repeated in the placing stage with $T_{\text{place}}$. If the gripper collides with other objects, it is judged to be invalid and is not sampled. For details on the sampling method from the action space, refer to Appendix C.1.

**Evaluation metrics.** We report the find- and grasp-success rates separately. If a sequence of our actions makes the target visible within 10-time steps, it is considered as a find-success. If it makes the target not only visible but also graspable and we can find a collision-free path for taking out the object within 10-time steps, it is considered a grasp success. Additionally, we measure the average number of actions required to find or grasp the target.

## 4.1 Simulation Experiments Results

In this section, we empirically verify that the proposed methods successfully find the target object and grasp it in the simulation experiments. The Figure 5 shows an example of how our recognition-based search-for-grasp method acts on the simulation experiment.

In the simulation, we first randomly place both the surrounding objects and the target object in the main simulation experiment. Specifically, We first select some random objects from a predefined object set and then spawn them on the random positions and orientations sampled from the uniform distribution on the shelf bottom and z-axis rotation matrix with uniform rotation angle distribution, respectively. And then we sample a random position of a target from the uniform distribution on the region where $f(x) = 1$.

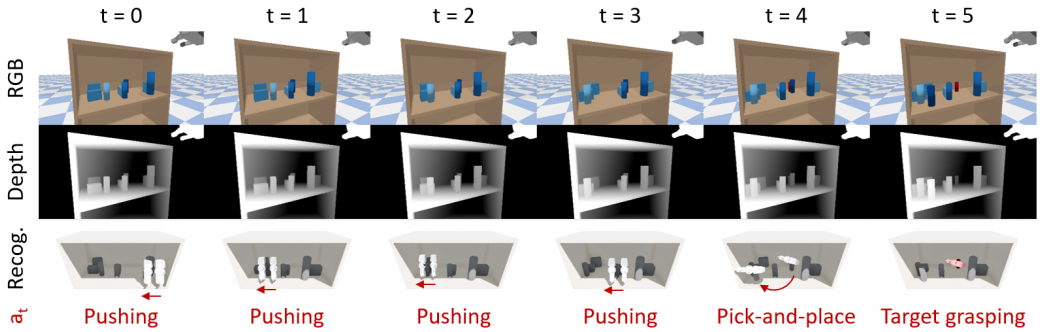


Figure 5: An example trajectory of simulation manipulation. Each column shows the camera input and action selection at each time step. In the simulation, surrounding objects are blue and the target object is red.

| | | The number of objects | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 2 | | 4 | | 6 | | 8 | |
| **METHOD** | | Find | Grasp | Find | Grasp | Find | Grasp | Find | Grasp |
| O-Search-and-Grasp | Succ. | 0.98 | 0.96 | 1.0 | 0.88 | 1.0 | 0.84 | 0.98 | 0.66 |
| | Steps | 1.163 | 1.132 | 1.32 | 2.136 | 1.86 | 3.286 | 1.694 | 3.485 |
| O-Search-for-Grasp | Succ. | 1.0 | 0.98 | 1.0 | 0.82 | 1.0 | 0.8 | 1.0 | 0.66 |
| | Steps | 1.24 | 1.408 | 1.36 | 1.854 | 1.66 | 2.5 | 1.74 | 3.212 |
| R-Search-and-Grasp | Succ. | 1.0 | 0.96 | 0.96 | 0.84 | 0.98 | 0.66 | 0.98 | 0.56 |
| | Steps | 1.46 | 1.551 | 1.562 | 2.065 | 1.653 | 3.3 | 2.102 | 3.73 |
| R-Search-for-Grasp | Succ. | 1.0 | 0.98 | 1.0 | 0.88 | 1.0 | 0.72 | 0.98 | 0.6 |
| | Steps | 1.34 | 1.531 | 1.74 | 2.543 | 1.8 | 2.4 | 1.653 | 3.846 |

Table 1: Simulation manipulation results

Note that some initial placement of the objects may be such that the target object is not graspable in the given action number limit, so the possible success rate for a given dataset may be lower than 1. To provide the upper limit of grasping success rate, we also implemented *oracle* methods, where all information about surrounding non-target objects is accessible. In the oracle experiments, the recognition module is not used and the ground-truth poses and shapes of the surrounding objects are directly used for MPC policy.

We experiment with each method 50 times for 2, 4, 6, and 8 surrounding objects, respectively. All experiments using the same number of objects share starting scenarios. The success rate of finding and grasping the target and the mean number of actions for each task are shown in the table 1. We put the letter 'O' in front of the method name in case of experiments using the oracle, and 'R' in case of using the recognition module.

The discussions that can be drawn from the result table are summarized as follows. First, the performance difference between Oracle and recognition is not significant, which means that the recognition-based method shows competitive performance. Second, throughout the experiments for recognition-based methods, the search-for-grasp method shows a better-grasping success rate than the search-and-grasp method. This seems to be because graspability is considered in the search process, and the result meets our expectations. Finally, the search-for-grasp method did not ensure a reduction in the average number of action steps for grasping. This phenomenon may have occurred due to investing numerous actions in attempting to transform failed scenarios in the search-and-grasp method into successful ones.

### 4.2 Real-world Experiments Results

In this section, we adopt the R-Search-for-Grasp method and conduct the task of finding and grasping the desired target object in real-world environment. Figure 6 shows a real-world manipulation

result. In this example, our approach can find an appropriate series of actions that successfully find and grasp the target object (red cylinder). The target object is occluded by the two objects, and the target object is found through three pushing actions. Since the target object found is not graspable at $t = 3$, an additional pushing action is applied and then the target object can be grasped.

**Failure cases.** Table 2 shows the manipulation success rates in real-world experiments. We design 5 test scenarios for each number of objects, and used object configurations for each experiment are in Appendix C. A few failure failure cases occur, especially when the number of the surrounding objects increases. Most of the failure reasons we observe are (i) that there's no solution of rearranging the objects in our designed action space and (ii) failures due to incorrect 3D recognition (e.g., erroneous updates of existence and graspability maps).

| Num. | Find | Grasp |
|---|---|---|
| 3 | 5/5 | 5/5 |
| 4 | 5/5 | 4/5 |
| 5 | 4/5 | 2/5 |
| 6 | 3/5 | 2/5 |

Table 2: Search-for-grasp real-world manipulation results.

### 4.3 Limitations and Future Directions

First, in the actual implementation of the graspability function, we only considered the trajectory where the gripper holds the object in a straight line. This assumption can potentially hinder the algorithm's performance since it may incorrectly determine that an object cannot be grasped, even when a feasible grasp is possible without colliding with another object. Introducing methods like Rapidly-exploring Random Trees (RRT) to the graspability indicator can help address this limitation and enable successful grasping in such situations. Secondly, if the recognition module recognizes an object as inaccurately large, the pose map may be underestimated, which may cause problems in object search. To solve this problem, the uncertainty of the recognition module prediction can be measured and reflected when updating the pose map. Lastly, since we use a superquadric representation, our algorithm is not able to use for objects that cannot be represented by a single superquadric. It can be solved by applying several existing studies that express one object as a set of several superquadrics.

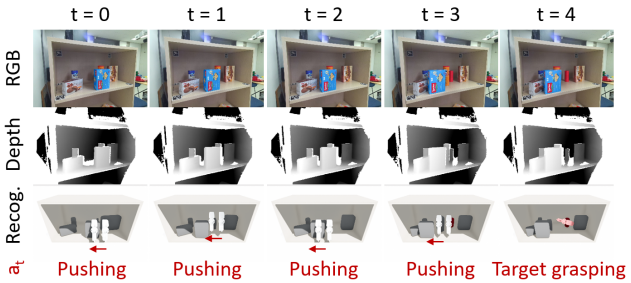


Figure 6: Search-for-grasp real-world manipulation results

## 5 Conclusion

We have proposed a general mechanical search and grasping framework and implemented practical algorithms, the search-and-grasp and search-for-grasp, by using the pre-trained 3D object recognition model. Our method does not rely on specialized tools and are not restricted to the suction gripper as the previous studies, but rather uses the standard two-finger gripper and 6-DoF grasping for pick-and-place actions. We have confirmed that our proposed method works well in both simulation and real-world shelf environments.

## References


[1] L. L. Wong, L. P. Kaelbling, and T. Lozano-Pérez. Manipulation-based active search for occluded objects. In *2013 IEEE International Conference on Robotics and Automation*, pages 2814–2819. IEEE, 2013.

[2] M. Gupta, T. Rühr, M. Beetz, and G. S. Sukhatme. Interactive environment exploration in clutter. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5265–5272. IEEE, 2013.

[3] M. R. Dogar, M. C. Koval, A. Tallavajhula, and S. S. Srinivasa. Object search by manipulation. *Autonomous Robots*, 36:153–167, 2014.

[4] J. K. Li, D. Hsu, and W. S. Lee. Act to see and see to act: Pomdp planning for objects search in clutter. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5701–5707. IEEE, 2016.

[5] Y. Xiao, S. Katt, A. ten Pas, S. Chen, and C. Amato. Online planning for target object search in clutter under partial observability. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8241–8247. IEEE, 2019.

[6] A. Price, L. Jin, and D. Berenson. Inferring occluded geometry improves performance when retrieving an object from dense clutter. In *Robotics Research: The 19th International Symposium ISRR*, pages 376–392. Springer, 2022.

[7] M. Danielczuk, A. Kurenkov, A. Balakrishna, M. Matl, D. Wang, R. Martín-Martín, A. Garg, S. Savarese, and K. Goldberg. Mechanical search: Multi-step retrieval of a target object occluded by clutter. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1614–1621. IEEE, 2019.

[8] M. Danielczuk, A. Angelova, V. Vanhoucke, and K. Goldberg. X-ray: Mechanical search for an occluded object by minimizing support of learned occupancy distributions. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9577–9584. IEEE, 2020.

[9] J. Lee, Y. Cho, C. Nam, J. Park, and C. Kim. Efficient obstacle rearrangement for object manipulation tasks in cluttered environments. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 183–189. IEEE, 2019.

[10] J. Lee, C. Nam, J. Park, and C. Kim. Tree search-based task and motion planning with prehensile and non-prehensile manipulation for obstacle rearrangement in clutter. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8516–8522. IEEE, 2021.

[11] E. R. Vieira, D. Nakhimovich, K. Gao, R. Wang, J. Yu, and K. E. Bekris. Persistent homology for effective non-prehensile manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 1918–1924. IEEE, 2022.

[12] S. Park, Y. Chai, S. Park, J. Park, K. Lee, and S. Choi. Semi-autonomous teleoperation via learning non-prehensile manipulation skills. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 9295–9301. IEEE, 2022.

[13] C. Nam, J. Lee, Y. Cho, J. Lee, D. H. Kim, and C. Kim. Planning for target retrieval using a robotic manipulator in cluttered and occluded environments. *arXiv preprint arXiv:1907.03956*, 2019.

[14] C. Nam, J. Lee, S. H. Cheong, B. Y. Cho, and C. Kim. Fast and resilient manipulation planning for target retrieval in clutter. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3777–3783. IEEE, 2020.

[15] C. Nam, S. H. Cheong, J. Lee, D. H. Kim, and C. Kim. Fast and resilient manipulation planning for object retrieval in cluttered and confined environments. *IEEE Transactions on Robotics*, 37 (5):1539–1552, 2021.

[16] Y. Yang, H. Liang, and C. Choi. A deep learning approach to grasping the invisible. *IEEE Robotics and Automation Letters*, 5(2):2232–2239, 2020.

[17] H. Huang, M. Dominguez-Kuhne, V. Satish, M. Danielczuk, K. Sanders, J. Ichnowski, A. Lee, A. Angelova, V. Vanhoucke, and K. Goldberg. Mechanical search on shelves using lateral access x-ray. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2045–2052. IEEE, 2021.

[18] H. Huang, M. Danielczuk, C. M. Kim, L. Fu, Z. Tam, J. Ichnowski, A. Angelova, B. Ichter, and K. Goldberg. Mechanical search on shelves using a novel "bluction" tool. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6158–6164. IEEE, 2022.

[19] H. Huang, L. Fu, M. Danielczuk, C. M. Kim, Z. Tam, J. Ichnowski, A. Angelova, B. Ichter, and K. Goldberg. Mechanical search on shelves with efficient stacking and destacking of objects. In *Robotics Research*, pages 205–221. Springer, 2023.

[20] S. Kim, T. Ahn, Y. Lee, J. Kim, M. Y. Wang, and F. C. Park. Dsqnet: A deformable model-based supervised learning algorithm for grasping unknown occluded objects. *IEEE Transactions on Automation Science and Engineering*, 2022.

[21] S. Kim, B. Lim, Y. Lee, and F. C. Park. Se (2)-equivariant pushing dynamics models for tabletop object manipulations. In *Conference on Robot Learning*, pages 427–436. PMLR, 2023.