

# Planning Base Poses and Object Grasp Choices for Table-Clearing Tasks Using Dynamic Programming

Sune Lundø Sørensen<sup>1</sup> <sup>a</sup>, Lakshadeep Naik<sup>2</sup> <sup>b</sup>, Peter Khiem Duc Tinh Nguyen<sup>2</sup>,  
Aljaz Kramberger<sup>2</sup> <sup>c</sup>, Leon Bodenhausen<sup>2</sup> <sup>d</sup>, Mikkel Baun Kjærgaard<sup>1</sup> <sup>e</sup> and Norbert Krüger<sup>2,3</sup> <sup>f</sup>

<sup>1</sup>*SDU Software Engineering, The Mærsk McKinney Møller Institute, University of Southern Denmark, Campusvej 55, Odense, Denmark*

<sup>2</sup>*SDU Robotics, The Mærsk McKinney Møller Institute, University of Southern Denmark, Campusvej 55, Odense, Denmark*

<sup>3</sup>*Danish Institute for Advanced Studies (DIAS), Odense M, Denmark*

**Keywords:** Base Pose Planning, Mobile Manipulation, Dynamic Programming, World Modeling, Perceptual Anchoring.

**Abstract:** Given a setup with external cameras and a mobile manipulator with an eye-in-hand camera, we address the problem of computing a sequence of base poses and grasp choices that allows for clearing objects from a table while minimizing the overall execution time. The first step in our approach is to construct a world model, which is generated by an anchoring process, using information from the external cameras. Next, we developed a planning module which – based on the contents of the world model - is able to create a plausible plan for reaching base positions and suitable grasp choices keeping execution time minimal. Comparing our approach to two baseline methods shows that the average execution cost of plans computed by our approach is 40% lower than the naive baseline and 33% lower than the heuristic-based baseline. Furthermore, we integrate our approach in a demonstrator, undertaking the full complexity of the problem.


## 1 INTRODUCTION


Grasping arbitrary objects from tables (see Figure 1) in unconstrained environments is a challenging task, which can take place in the e.g. the hospitality industry where tables need to be cleared in a restaurant or a university canteen (Khatib, 1999). Solving the task requires to address a number of difficult sub-problems. (1) Planning an optimal path for the robot platform in terms of performance and execution time, (2) Visual pose estimation based on multiple cameras potentially combined with visual exploration, (3) Pre-grasp and Grasp planning and (4) informed decisions on grasp alternatives and grasp execution despite the expected uncertainties in pose estimation. On top of these challenges, it is required to integrate the sub-components into a stable system. Because of these complexities, today such systems are not in use in real


world unconstrained environments.


In many situations external cameras installed in the rooms where the robot is operating can be exploited (see Figure 1(a)), since these are nowadays available in many buildings. Given such a setup with external cameras and a camera on the robot arm, we address first problem (1) of planning the optimal series of actions for a mobile manipulator to clear a table, minimizing navigation and manipulation time. The actions consist of navigating to a base pose and grasping the object and storing it on the robot itself.


Our approach consists of three steps. First, a symbolic world model of the scene is constructed, using the information from the external cameras. The world model is built and maintained by an anchoring process which associates the information from the external cameras. The world model also contains the pre-defined grasp poses and the inverse reachability map, which are needed to plan the base poses and grasp choices. Next, we compute the grasping and navigation costs. Finally, the contents of the world model and the costs are used to solve the planning problem using dynamic programming with memoization. Compared to previous works we do not assume the order in which the objects should be grasped to


<sup>a</sup>  <https://orcid.org/0000-0003-2874-0660>

<sup>b</sup>  <https://orcid.org/0000-0002-2614-8594>

<sup>c</sup>  <https://orcid.org/0000-0002-4830-4885>

<sup>d</sup>  <https://orcid.org/0000-0002-8083-0770>

<sup>e</sup>  <https://orcid.org/0000-0001-5124-744X>

<sup>f</sup>  <https://orcid.org/0000-0002-3931-116X>

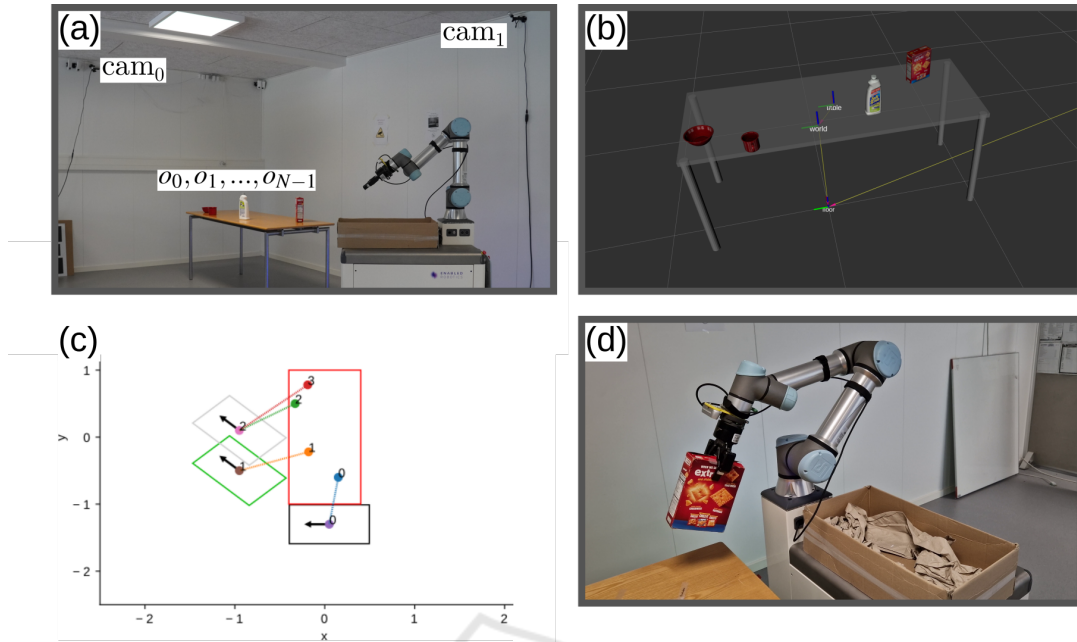


Figure 1: (a): The robot is tasked with clearing the objects  $o_0, o_1, \dots, o_{N-1}$  from the table. The external cameras,  $cam_0$  and  $cam_1$  in the upper left and right corners provide an overview of the scene. (b): A world model is built using the information from the external cameras. (c): Using the world model's contents we propose a solution for computing a sequence of actions for clearing the objects, minimizing execution time. In this case, the plan contains three base poses (smaller rectangles with arrows) for picking the four objects (numbered dots inside the bigger rectangle, which is the table). (d): The robot executes the plan and uses a eye-in-hand camera to accurately pose estimate the objects before grasping.

be known beforehand. To show the efficiency of our method we also implement two baseline methods and compare the execution costs.

The remainder of this paper is structured as follows: Section 2 discuss related works, Section 3 presents formally the problem we address, Section 4 presents our approach, and Section 5 presents the results of evaluating the approach and comparing it against two baseline methods. In Section 6 we demonstrate our approach in an table-clearing application addressing problems (1)-(4). Lastly, Section 7 discusses the results and concludes.

## 2 STATE OF THE ART

This work involves a few different disciplines. In this section we describe the related works, starting with world modeling (Section 2.1), explicit base pose planning (Section 2.2), and finally sequence planning (Section 2.3).

### 2.1 World Modeling

World modeling - and in particular perceptual anchoring - using information from external, static and

robot cameras have been addressed in several contexts. Daoutis et al. (Daoutis et al., 2012) describe a frame-work for cooperative perceptual anchoring, in a setup of stationary agents (e.g., external cameras) and a mobile agent (e.g., a mobile manipulator). (Wong et al., 2015) use clustering to associate data from multiple viewpoints. They use several variations of a Dirichlet process mixture-model and incorporate information about the different views. In this work we use the Jonker-Volgenant algorithm (Crouse, 2016) and a similarity measure based on Euclidean distance and class.

### 2.2 Explicit Base Pose Planning

The selection of a base pose for grasping an object depends on the existence of a valid Inverse Kinematics Solution to attain the desired grasp pose. Typically, this is accomplished through the utilization of Inverse Reachability Maps (IRM) (Makhal and Goins, 2018; Vahrenkamp et al., 2013). However, employing IRM for planning an appropriate base pose is computationally demanding, primarily due to the inherent complexity of the 6D search space. Hence, recent works (Jauhri et al., 2022) have introduced learning-based approaches to predict base poses for grasping.

Some alternative approaches involve planning manipulator configurations in conjunction with the base pose (Reister et al., 2022; Vafadar et al., 2018). Additionally, certain methodologies also consider uncertainty in both the robot’s localization and the object pose during base pose planning (Meng et al., 2021; Stulp et al., 2012).

In this work, we plan the optimal base pose for grasping an object, while simultaneously considering the cost associated with navigating to the selected base pose from the robot’s prior base pose and also the future base poses that must be visited to successfully complete the task.

### 2.3 Sequence Planning

The problem of planning base poses for a mobile manipulator and selecting the optimal execution order of tasks (in this work, grasping) is a highly complex one. The main reason is that there is often a large number of possibilities to consider. Previous works have tried to address this problem in a limited setup. For example in (Reister et al., 2022) the grasping sequence is assumed to be known a priori and only local optimization is performed i.e. minimize the time to navigate to the next object and grasp it. Xu et al. (Xu et al., 2020) address the problem in the context of picking parts from trays for assembling. They select base poses from the intersections of base poses from where the objects can be grasped to minimize the movement of the mobile base, but do not consider manipulation time. Further, to compensate for localization uncertainty they ignore intersections smaller than a certain threshold. The final plan is then found by computing the shortest path between all of the base poses. (Harada et al., 2015) have also tried to minimize the number of base poses without considering the combined time cost of navigation and manipulation.

In this work, we present a method that provides both the base poses and the optimal sequence in which these poses should be visited. Notably, our approach does not make assumptions inherent in prior works, such as the sequence in which objects should be grasped is known or the assumption that minimal base poses inevitably lead to an optimal solution.

## 3 PROBLEM DEFINITION

We use in this work three types of poses: *Objects pose* is the 6D pose of e.g., an object. The *grasp pose* of an object is the pose the gripper should be in to grasp the object. Finally, a *base pose* refers to the SE2 pose

of the mobile base of the robot. We assume that we have up to  $M$  external stationary cameras as shown in Figure 1(a). Further, we assume that there are  $N$  instances of different objects with poses  $o_1, o_2, \dots, o_N$  on the table that the robot needs to grasp and that each object has a grasp pose  $g_1, g_2, \dots, g_N$ . It is assumed that all external and robot cameras are RGB-D cameras with RGB and depth information, and that there are pipelines in place to detect and pose estimate objects from the RGB and depth information.

The problem can then be formulated as: Given  $N$  objects with poses  $o_0, o_1, \dots, o_{N-1}$  on the table, find first for each object the set  $\mathcal{A}_n$  of possible actions, where each action contains an object grasp pose  $g_n$  and a base pose  $b_k$ ,  $\{b_k, g_n\}$ , as shown in Equation (1).

$$\begin{aligned} \mathcal{A}_1 &= \{\{b_k, g_1\}\}_{k=1}^{K_1} \\ \mathcal{A}_2 &= \{\{b_k, g_2\}\}_{k=1}^{K_2} \\ &\vdots \\ \mathcal{A}_N &= \{\{b_k, g_N\}\}_{k=1}^{K_N} \end{aligned} \quad (1)$$

Then, taking the union of all  $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_N$  yields  $\mathcal{A}$ , as shown in Equation (2), which contains all possible actions.

$$\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \dots \cup \mathcal{A}_N \quad (2)$$

Then, from the power set  $\mathcal{P}(\mathcal{A})$ , which gives all subsets of  $\mathcal{A}$  including  $\emptyset$  and  $\mathcal{A}$  itself, we can construct a set  $\mathcal{D}$ , as shown in Equation (3) by keeping only valid sets of  $\mathcal{P}(\mathcal{A})$ . A valid set  $a$  of actions includes grasping all objects ( $|a \cap \mathcal{A}| = N$ ) and each object only once ( $|a \cap \mathcal{A}_n| = 1 \forall n$ ).  $a$  may have the same base pose across multiple actions.

$$\mathcal{D} = \{s \in \mathcal{P}(\mathcal{A}) \mid |s \cap \mathcal{A}| = N \text{ and } |s \cap \mathcal{A}_n| = 1 \forall n\} \quad (3)$$

$\mathcal{D}$  contains all valid sets of actions which satisfies the aforementioned criteria, which means that they can be executed in one sequence. Each set of actions in  $\mathcal{D}$  can be executed in  $N!$  different sequences. Therefore, we take all possible sequences of each element of  $\mathcal{D}$ , which yields the set  $\mathcal{S}$  of all valid sequences in which actions can be executed.

Equation (4) shows an example of a sequence  $s$  where four objects with grasp poses  $g_1, g_2, g_3, g_4$  are grasped from three base poses  $b_5, b_{17}, b_{42}$ .

$$s = (\{b_{17}, g_2\}, \{b_{17}, g_4\}, \{b_5, g_3\}, \{b_{42}, g_1\}) \quad (4)$$

For a sequence  $s \in \mathcal{S}$  we define  $s(i)$  to be the  $i$ th action in the sequence,  $g(s(i))$  to be the grasp pose of the object to be grasped for that action and  $b(s(i))$

to be the base pose from where the object should be grasped.

Finally, the problem is to find the sequence  $s^*$  which reduces the overall task execution time. Equation (5) formally states the problem, where  $C_{\text{seq}}(s)$  is the total execution cost of a sequence  $s$  which is defined in Section 4.3.1.

$$s^* = \underset{s \in \mathcal{S}}{\text{argmin}} C_{\text{seq}}(s) \quad (5)$$

## 4 PROPOSED APPROACH

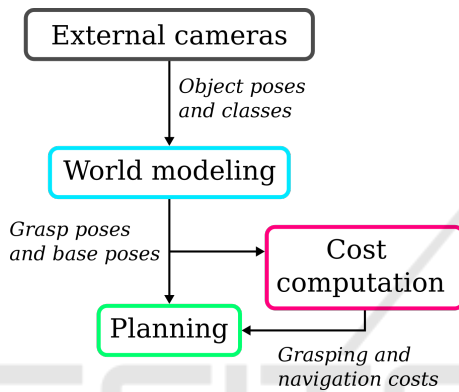


Figure 2: The components of our approach.

Our approach has three steps as shown in Figure 2. First, we construct a *world model* from the data from the external cameras, to have a stable representation of the objects in the scene, and augment it with predefined grasp poses and inverse reachability maps. Next, we compute the *grasping and navigation costs* for use in the *planning algorithm*. Finally, the planning algorithm uses the information from the world model and the costs to determine the time-optimal action sequence of base poses and grasp choices. We assume that the following information is obtained beforehand:

- The dimensions of the table and its pose
- The 6D transformations between the cameras

In the following sections, we will describe the steps of the approach in more detail, starting with the world model.

### 4.1 The World Model

The world model (WM) has two important functions: (1) It provides a stable symbolic representation of the objects and the table, and (2) it contains predefined grasp poses  $g_1, g_2, \dots, g_S$ , one for each object class

$c_1, c_2, \dots, c_S$  and an inverse reachability map. This enables it to provide all the necessary information for solving the planning problem. Each object is represented with an *anchor*  $\alpha_n = \{o_n, c_s\}$ , where  $o_n$  is the estimated 6D pose in the world frame and  $c_s$  is the object's class. The world model receives object detections and positions estimates from the object detector for each of the different camera observations. These are then processed by a data association (DA) process to determine which measurements originate from the same object. The DA process uses the Jonker-Volgenant algorithm (Crouse, 2016) as described in (Sørensen and Kjærgaard, 2023).

The associations are then fed to the multi-view 6D object pose estimation process that fuses the poses estimate of the associated objects to obtain a full 6D pose  $o_n$  for each object. Finally, each anchor candidate will either initiate the creation of a new anchor or be associated to an existing depending on the state of the world model, using an anchoring process. The anchoring process starts by comparing each candidate to each anchor in the WM using the same similarity measure as the DA process (Sørensen and Kjærgaard, 2023). Next, it also uses the Jonker-Volgenant algorithm (Crouse, 2016) to compute the assignments of new observations to anchors. New observations which are not associated by the algorithm will instantiate new anchors.

#### 4.1.1 Predefined Grasps and Inverse Reachability Maps

As mentioned in Section 4.1 the WM contains a predefined a grasp pose  $g_s$  for each object class  $c_s$ . The grasp pose is defined in the object frame.

In addition to the predefined grasps, the WM also contains an Inverse Reachability Map (IRM). Each robot manipulator has a unique IRM (Makhal and Goins, 2018; Vahrenkamp et al., 2013) that contains suitable robot base poses for reaching a grasp pose  $g_s$ . The map is continuous but in order to make the planning problem feasible, we discretize it in translation using step-size  $\lambda$  and orientation using step-size  $\phi$ . Thus, for a grasp pose  $g_s$  for  $o_n$  in the robot base frame, it contains a set of 6D base poses (Makhal and Goins, 2018) such that from each base pose there exists an inverse kinematic solution to the grasp pose. Figure 3 shows a subset of base poses for the UR5e robot in its Inverse Reachability Map. The red cube in the middle of the figure represents the object  $o_n$  with grasp pose  $g_s$ .  $b_0, b_1, \dots, b_J$  are the base poses from where the UR5e can grasp the object.

Using the grasp pose  $g_s$  and pose  $o_n$  of each object, we can compute an IRM for each object. Each of these are then filtered, yielding an IRM for each ob-

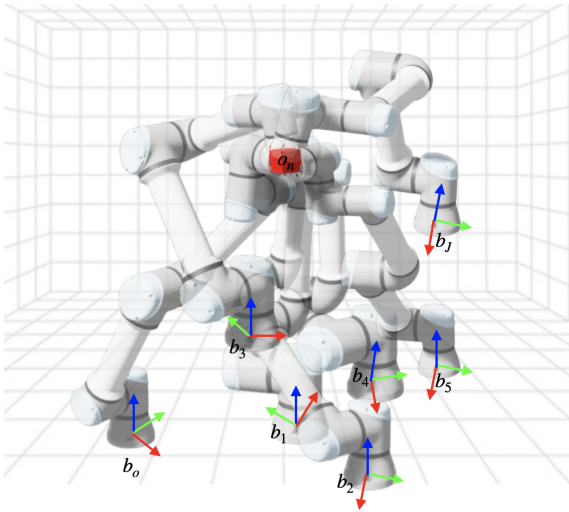


Figure 3: Inverse Reachability Map for UR5e robot as it is stored in the world model. Note that only subset of base poses have been shown here.

ject, which contains only the robot base poses where the mobile base is on the ground and not in collision with the table. From these, the WM can now provide the sets  $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_N = \mathcal{A}$  of possible actions for each of the  $N$  objects.

## 4.2 Cost Computation

The next step in our approach is to compute the navigation and grasping costs. It involves approximating the cost  $C_{\text{grasp}}(g(s(i)))$  of grasping object  $g(s(i))$  from base pose  $b(s(i))$  using grasp pose  $g_s$  and the cost  $C_{\text{nav}}(b(s(i-1)), b(s(i)))$  of navigating from  $b(s(i-1))$  to  $b(s(i))$ . To begin with, the time cost  $C_{\text{grasp}}(g(s(i)))$  of grasping the object from each valid base pose in the  $\mathcal{A}$  is computed. Figure 4 illustrates the possible base poses for grasping the objects at different locations on the table.

Once the grasping costs are computed for all the objects in the scene, we compute the navigation cost  $C_{\text{nav}}(b(s(i-1)), b(s(i)))$ . The navigation cost estimates the time required for navigating between the two base poses. Therefore, for all the  $K$  unique base poses across  $\mathcal{A}$ , the navigation costs need to be computed for the  $K \times K$  different combinations.

Thus, if the robot is at base pose  $b(s(i-1))$  the total cost of performing the next action  $s(i)$  is given by:

$$C_{\text{action}}(s(i-1), s(i)) = C_{\text{nav}}(b(s(i-1)), g(s(i))) + C_{\text{grasp}}(g(s(i))) \quad (6)$$

## 4.3 Computing the Sequence

The core challenge now is to compute the base poses and object grasping sequence. We formulate this as an optimization problem, where the objective is to minimize navigation and manipulation time and solve it using dynamic programming with memoization. The algorithm has three inputs described previously in Sections 4.1 and 4.2:

- A set  $S$  of all possible valid action sequences.
- Estimated action costs  $C_{\text{action}}(s(i-1), s(i))$

Finally, the base poses and navigation costs are used in a dynamic programming with memoization algorithm (DP) (Held and M. Karp, 1962). Figure 1(c) illustrates the solution to the scene shown in Figure 1(a). The algorithm is described next.

### 4.3.1 The Optimization Algorithm

As stated in the problem definition in Section 3, the algorithm has to compute an ordered sequence  $s^*$  of  $N$  object actions. The sequence should minimize the total cost  $C_{\text{seq}}(s^*)$  of all actions in  $s^*$ :

$$C_{\text{seq}}(s) = \sum_{i=1}^{|s|} C_{\text{action}}(s(i-1), s(i)) \quad (7)$$

where the  $C_{\text{action}}(s(i-1), s(i))$  consists of manipulation and navigation costs as described in Section 4.2. For  $i = 1$  we define

$$C_{\text{action}}(s(0), s(1)) = C_{\text{nav}}(b_0, b(s(1))) + C_{\text{grasp}}(g(s(1))) \quad (8)$$

where  $b_0$  is the robot's initial base pose. The grasping cost  $C_{\text{grasp}}(s(i))$  is set to a high cost if  $g(s(i))$  not reachable from  $b(s(i))$ . To solve the minimization problem, we define a function  $C_{DP}(s(i))$  representing the cost of executing all actions in a sequence  $s$  up to and including  $s(i)$  as:

$$C_{DP}(s(i)) = \begin{cases} C_{\text{action}}(s(i-1), s(i)) & \text{if } i = 1 \\ \text{mincost}(s(i)) + C_{\text{grasp}}(s(i)) & \text{if } i > 1 \end{cases} \quad (9)$$

which depends on a function  $\text{mincost}(s(i))$ , see Equation (10), that finds the previous action  $s(i-1)$  which has the minimum cost, i.e. the minimum time to grasp the previous object from the previous base pose and to navigate from there to the current base pose.

$$\text{mincost}(s(i)) = \arg \min_{s(i-1)} C_{\text{nav}}(s(i-1), s(i)) + C_{DP}(s(i-1)) \quad (10)$$

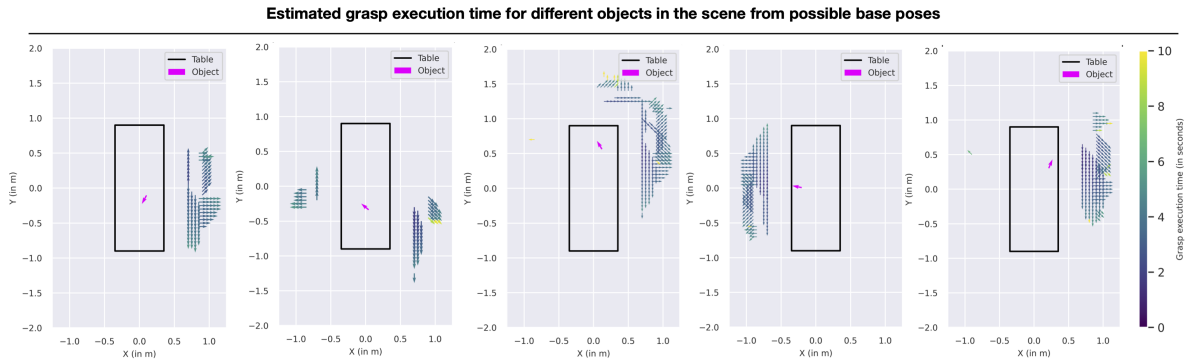


Figure 4: Manipulation costs computed for different objects in the scene.

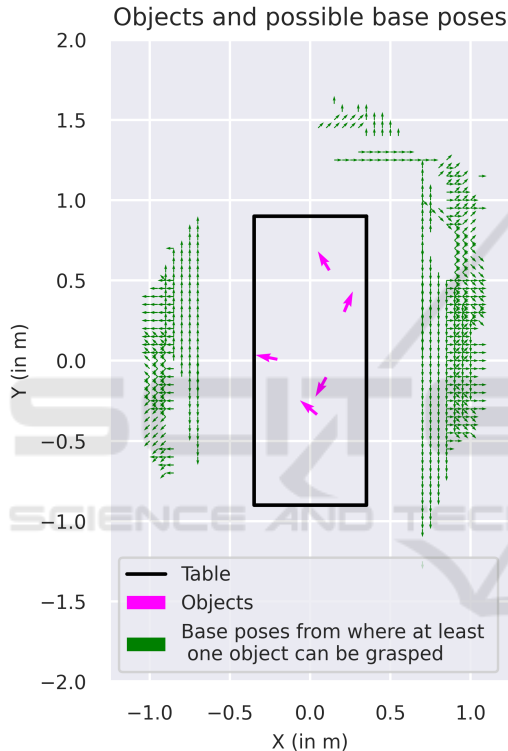


Figure 5: Base poses (green arrows) from where at least one object (pink arrows) could be grasped. Navigation cost is computed between these base poses.

Note that  $\text{mincost}(s(i))$  depends on the cost of the previous actions  $C_{DP}(s(i-1))$  and thus the problem is solved recursively in a top-down fashion. The solution is calculated in polynomial time using dynamic programming with memoization. There are two phases. First, the cost  $C_{\text{seq}}(s)$  is computed recursively using  $C_{DP}(s(i))$  for the all unique actions (valid combinations of a grasp and base pose) at the end of a sequence in  $\mathcal{D}$  (which contains all valid sequences, see Equation (3) in Section 3). Thus, for all sequences ending with a particular action, we now have the one with the minimum cost. The computational and mem-

ory requirements are significantly reduced by using memoization, i.e., storing and reusing the cost of partial sequences that appear in multiple  $s \in \mathcal{D}$ . This is possible because the nature of the problem allows it; For a given action,  $s(i)$ , the cost of all possible actions depends on the previous actions taken (i.e., which objects have already been grasped and from which base pose(s)). Finally, the optimal sequence  $s^*$  is simply the one with the lowest cost  $C_{\text{seq}}(s)$ .

## 5 RESULTS

In this section we evaluate the execution cost of the proposed approach and compare it against two baseline methods; One heuristic-based and one more naive method. We generate 25 scenes with five randomly placed objects and a random starting pose for the robot and apply the three methods. The next sections describe the details of the evaluation.

**Prerequisites:** Experiments were conducted on five different objects from the YCB benchmark for manipulation (Calli et al., 2015). Top grasp poses were selected for grasping all the objects. Inverse Reachability Maps were computed with a resolution of  $\lambda$  cm translation and  $\phi$  degrees rotation. Object pose estimates were determined using two RealSense D455<sup>1</sup> RGB-D cameras using (Naik et al., 2022). The costs were computed using the NVIDIA Isaac simulator<sup>2</sup>. For the manipulation cost, grasp execution time was computed using the Lula Trajectory generator. The navigation cost between the two base poses was computed using a heuristic based on the distance between the two bases while considering the geometry of the robot and the scene.

**Baselines:** We compared our method against two

<sup>1</sup><https://www.intelrealsense.com/depth-camera-d455>

<sup>2</sup><https://developer.nvidia.com/isaac-sim>

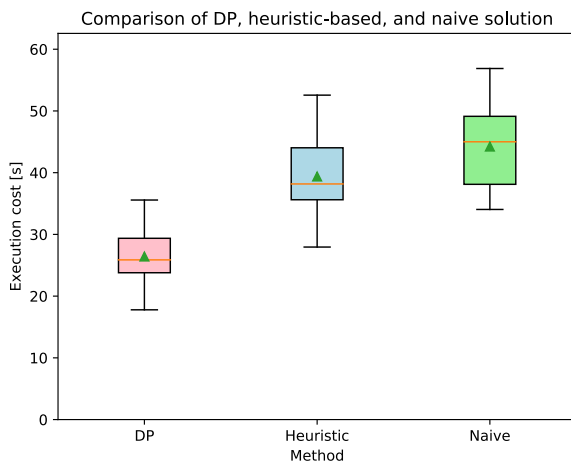


Figure 6: Box plots of the execution cost of the dynamic programming and the heuristic-based solutions, using  $\lambda = 0.25$  m discretization in translation and  $\phi = 45^\circ$  in orientation. The median is shown with a horizontal line in the boxes and the means with green triangles.

baselines. The first baseline is a heuristic-based method. This method selects the base with the lowest navigation cost from its current pose and grasps all possible objects from there. It then continues to select the next base pose from there where new objects can be grasped. The other baseline is simpler. It first selects the nearest base pose along the table's edge and then in turn selects base poses clockwise around the table, checking every  $q$  cm if any objects can be grasped.

**Experimental Setup:** The scenes used for evaluation are generated by sampling a random pose for five different objects (a mug, bleach bottle, mustard bottle, bowl and a cracker box) on the table's surface. A random collision-free starting base pose for the robot is also sampled for each scene.

## 5.1 Quantitative Results

Figure 6 shows box plots of the execution cost for the proposed method and the two baselines. The colored boxes extend from the 1st to the 3rd quartile with the line showing the median and the green triangles showing the mean execution cost. It can be seen that the execution cost of our approach is 40% lower than the naive baseline and 33% lower than the heuristic-based baseline.

### Ablation Study: Cost Computation Time

The most time-consuming part of our approach is the cost computation. Figure 7 shows box plots for the time used for computing costs, the execution cost and the percentage of grasped objects, for a scene with

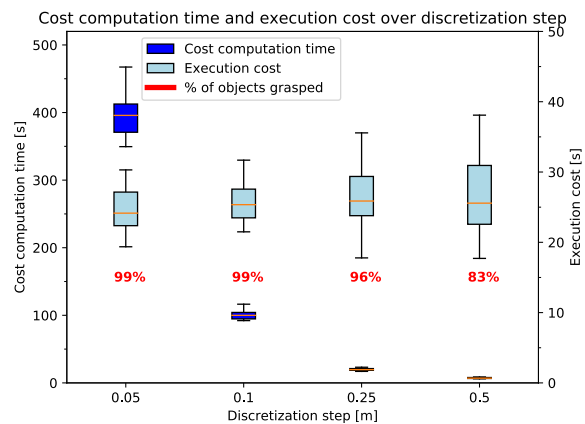


Figure 7: Computation time for costs with different levels of discretization of translation. The discretization of the orientation is kept constant at  $45^\circ$ . The percentages in red tell how many of the 125 objects across the 25 scenes were reachable.

different translation discretization steps,  $\lambda$ . It shows that even with a rather low resolution of  $\lambda = 0.25$  m the cost computation time is only around 18 seconds and still the execution time and percentage of objects grasped is comparable to smaller values of  $\lambda$ .

## 6 APPLICATION

In addition to the evaluation in Section 5, we have also demonstrated our approach in a lab environment covering the full complexity of the problem at hand. The setup consists of two RGB-D cameras mounted on the walls and a mobile manipulator with a gripper and an RGB-D camera, as shown in Figure 1(a). First, for each camera, the objects on the table are detected using Detectron2 (Wu et al., 2019) and their position estimated. This information is sent to the world model, which performs the data association. The pose estimation pipeline then uses the association to perform a 6D pose estimate of the objects, which is then returned to the world model, see Figure 1(b). We then run the planning algorithm as described in Section 4 to obtain the table-clearing plan, see Figure 1(c). The robot then executes the plan by first driving to the first base pose in the plan and starting a visual exploration to achieve a good view of the object to grasp and a pose estimate with low uncertainty. When the uncertainty is below a pre-defined threshold, the robot then attempts to grasp the object, see Figure 1(d). This is repeated for all base poses and objects in the plan. A video of the demonstration can be viewed on YouTube<sup>3</sup>.

<sup>3</sup><https://youtu.be/NcxZ-c6hcvQ>

## 7 CONCLUSION

In this paper, we have presented a novel approach to planning a table-clearing task for a mobile manipulator, in a setup with external cameras and a robot camera. We first model the scene in a world model using the information from the external cameras. Based on the world model, we use dynamic programming to plan an sequence of bases poses and grasp choices which minimize the overall execution time. Evaluating the approach on 25 different scenes and comparing it to two baseline methods shows that our approach computes plans with a 33% lower execution cost than the heuristic-based baseline and 40% lower than the naive approach. Limitations of our approach, which should be addressed in future work, include the cost computation time and including objects' geometry in the planning, to avoid collisions between objects when removing them from the table.

## ACKNOWLEDGMENTS

This work is funded by the Innovation Fund Denmark through the FacilityCobot project. The authors would also like to thank the I4.0 lab of The University of Southern Denmark for lending us the robot used in this work.

## REFERENCES

- Calli, B., Walsman, A., Singh, A., Srinivasa, S., Abbeel, P., and Dollar, A. M. (2015). Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols. *arXiv preprint arXiv:1502.03143*.
- Crouse, D. F. (2016). On implementing 2D rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696.
- Daoutis, M., Coradeschi, S., and Loutfi, A. (2012). Cooperative Knowledge Based Perceptual Anchoring. *International Journal on Artificial Intelligence Tools*, 21(03):1250012.
- Harada, K., Tsuji, T., Kikuchi, K., Nagata, K., Onda, H., and Kawai, Y. (2015). Base position planning for dual-arm mobile manipulators performing a sequence of pick-and-place tasks. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 194–201. IEEE.
- Held, M. and M. Karp, R. (1962). A Dynamic Programming Approach to Sequencing Problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210.
- Jauhri, S., Peters, J., and Chalvatzaki, G. (2022). Robot Learning of Mobile Manipulation With Reachability BehaviorPriors. *IEEE RA-L*, 7(3):8399–8406.
- Khatib, O. (1999). Mobile manipulation: The robotic assistant. *Robotics and Autonomous Systems*, 26(2-3):175–183.
- Makhal, A. and Goins, A. K. (2018). Reuleaux: Robot base placement by reachability analysis. In *2018 Second IEEE International Conference on Robotic Computing (IRC)*, pages 137–142. IEEE.
- Meng, Y., Chen, Y., and Lou, Y. (2021). Uncertainty aware mobile manipulator platform pose planning based on capability map. In *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 123–128. IEEE.
- Naik, L., Iversen, T. M., Kramberger, A., Wilm, J., and Krueger, N. (2022). Multi-view object pose distribution tracking for pre-grasp planning on mobile robots. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 1554–1561.
- Reister, F., Grotz, M., and Asfour, T. (2022). Combining navigation and manipulation costs for time-efficient robot placement in mobile manipulation tasks. *IEEE RA-L*, 7(4):9913–9920.
- Stulp, F., Fedrizzi, A., Mösenlechner, L., and Beetz, M. (2012). Learning and reasoning with action-related places for robust mobile manipulation. *Journal of Artificial Intelligence Research*, 43:1–42.
- Sørensen, S. L. and Kjærgaard, M. B. (2023). Quantifying the Accuracy of Collaborative IoT and Robot Sensing in Indoor Settings of Rigid Objects. *Under review*, page 8.
- Vafadar, S., Olabi, A., and Panahi, M. S. (2018). Optimal motion planning of mobile manipulators with minimum number of platform movements. In *IEEE Int. Conf. on Industrial Technology*.
- Vahrenkamp, N., Asfour, T., and Dillmann, R. (2013). Robot placement based on reachability inversion. In *IEEE ICRA*.
- Wong, L. L., Kaelbling, L. P., and Lozano-Pérez, T. (2015). Data association for semantic world modeling from partial views. *The International Journal of Robotics Research*, 34(7):1064–1082.
- Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., and Girshick, R. (2019). Detectron2. <https://github.com/facebookresearch/detectron2>.
- Xu, J., Harada, K., Wan, W., Ueshiba, T., and Domae, Y. (2020). Planning an efficient and robust base sequence for a mobile manipulator performing multiple pick-and-place tasks. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11018–11024.