Planning Robot Placement for Object Grasping*

Manish Saini[1], Melvin Paul Jacob[1], Minh Nguyen[2], and Nico Hochgeschwender[2]

[1] Hochschule Bonn-Rhein-Sieg, Germany
{manish.saini,melvin.paul}@smail.inf.h-brs.de
[2] University of Bremen, Germany
{minh.nguyen,nico.hochgeschwender}@uni-bremen.de

Abstract. When performing manipulation-based activities such as picking objects, a mobile robot needs to position its base at a location that supports successful execution. To address this problem, prominent approaches typically rely on grasp poses being provided by a planner for a target object, which are then analysed to identify the best robot placements for achieving each grasp pose. In this paper, we propose instead to first find robot placements that would not result in collision with the environment and from where picking up the object is feasible, then evaluate them to find the best placement candidate. Our approach takes into account the robot's reachability, as well as RGB-D images and occupancy grid maps of the environment for identifying suitable robot poses. The proposed algorithm is embedded in a service robot application, in which a person points to select the target object for grasping. We evaluate our approach with a series of grasping experiments, against an existing baseline implementation that commands the robot to a fixed navigation goal. The experimental results demonstrate the validity of our approach, which can serve as a baseline for developing and evaluating more advanced techniques in the future.

Keywords: Robot Placement, Motion Planning

1 Introduction

A common task for mobile robots in domestic and household scenarios is object retrieval, e.g. fetching a cup from a dining table or kitchen counter. Solutions for this task typically involve the robot navigating to a predetermined location and then initiating the picking-up behaviour. Such approaches often do not take into account factors of the environment or physical limitations of the robot, which can result in the robot trying to grasp the object from unfavourable positions, for instance, a far table edge rather than one that is close to the object. Such sub-optimal placements can increase the difficulty of the manipulation task, increasing the risk of failure or even making the task infeasible, e.g. when the

* This work was supported by the Bonn-Aachen International Center for Information Technology (b-it).

target object cannot be reached without the robot colliding with the table or with other objects.

To overcome these challenges, we propose in this paper a robot placement planning approach that identifies locations from where a mobile robot can grasp an object more easily. Our approach takes into account occupancy grid maps and RGB-D data to check whether the placement candidates can be navigated to or may be at risk of collision during the manipulation motion. The candidates are then evaluated using a criterion that characterizes the difficulty of the overall navigation and manipulation motion, and the best candidate is selected for carrying out the pickup behaviour. The planning algorithm is embedded in a service robot application, in which a human operator chooses the target object using hand gestures. We evaluate our approach with a series of grasping experiments using the Toyota Human Support Robot (HSR) and compare it against a baseline implementation which grasps from a predefined navigation goal. In summary, our contributions in this paper are two-fold:

- We developed a base placement planning algorithm distinct from prominent approaches addressing the same problem. Our evaluation on a real robot validates the effectiveness of our approach, which can serve as a baseline for developing and evaluating more advanced methods in the future.
- We integrated the proposed planning approach in a Human-Robot Interaction (HRI) scenario for addressing RoboCup tasks which require the robot to retrieve an object from an open environment [5].

The remainder of this article is organized as follows. Related studies will be discussed in the next section. Sec. 3 describes in detail our robot placement planning algorithm and how we carry out the pickup behaviour using the generated placement candidates. Sec. 4 describes our experimental setup and evaluates the results of our grasping experiments. Finally, Sec. 5 includes concluding remarks and discusses possible extensions to our approach in the future.

2 Related Work

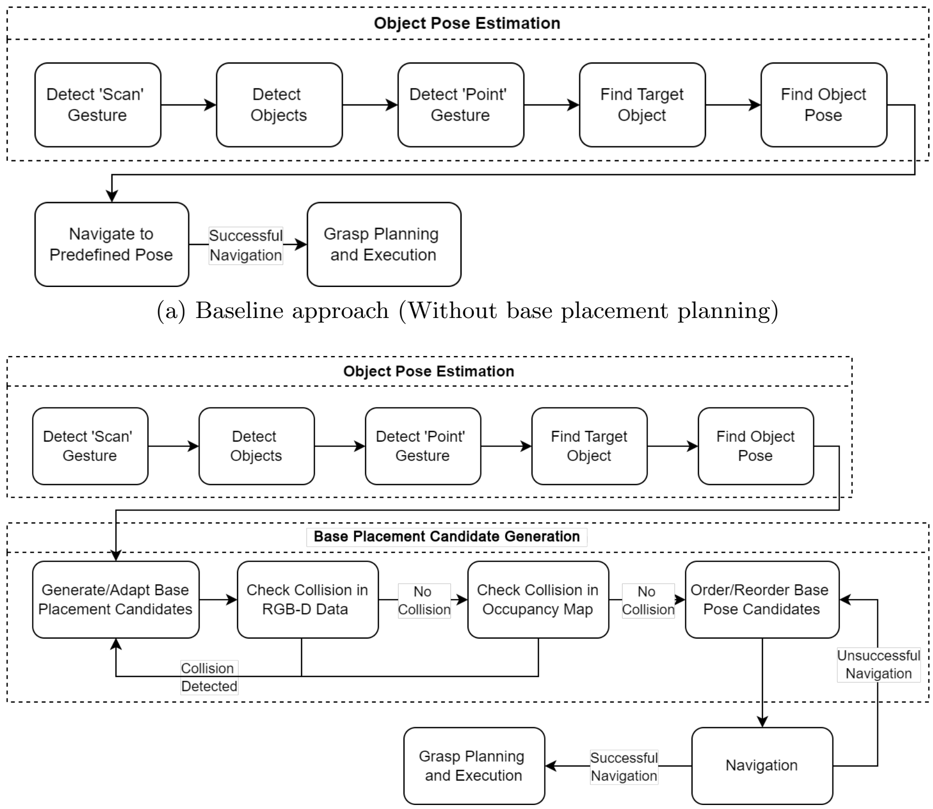
The traditional approach to improve object grasping performance is grasp planning, which aims to find placement and configuration of end effectors that can best satisfy some criteria relevant to the grasping task [1,10,14]. Grasp planning can be broadly categorized into analytical [14] and data-driven [1,10] methods, with the former typically analysing the target object's shape and/or hand-object contact to evaluate different properties of grasp candidates, such as stability or task compatibility. Empirical or data-driven methods [1,10], on another hand, sample and evaluate grasp candidates based on some existing grasp experience, either characterized by a heuristic or generated in simulation or on a real robot.

Another approach for improving the performance of the robot's grasping behaviour is to consider environmental features during motion planning or control of the robot arm [3]. This can be done in combination with Dynamic Movement Primitives (DMP), in which complex trajectories, e.g. motion of a robot manipulator, are modelled using a system of second-order linear Ordinary Differential Equation (ODE). A learnable forcing term is then introduced to adapt these trajectories to some goal-oriented behaviours using well-defined attractor dynamics [7]. Data from proximity sensors can then be incorporated into the motion plan, e.g. for obstacle avoidance [3], by introducing a coupling term to the DMP formulation. This coupling term can either be derived analytically via computation of the potential field [4, 6, 16] or learned from sensory data [15].

To address the problem of finding an optimal base placement for grasping an object, several approaches utilize inverse reachability maps (IRM) introduced in [17], which invert a representation of the robot's workspace using its kinematics for a given grasp candidate. The resulting robot placements are then evaluated using some quality metric, typically to minimize the chance of self-collision or singularity. Related research has improved upon various aspects of IRM, including quality metrics for possible robot placements [21] and representations of the robot workspace [9, 21]. Burget and Bennewitz [2] extend the approach with constraints specific to humanoid robots to find optimal foot placements for a grasp candidate. Reister et al. [13] apply IRM to the problem of finding time-efficient base placements for picking and placing several objects consecutively, where the placements are evaluated by a combination of manipulation and navigation costs, and reachability inversion is used for the manipulation portion of the cost function. Paus et al. [12] combine IRM with coverage path planning to minimize the number of base repositioning needed to achieve optimal coverage with the manipulator. IRM is also used as priors in machine learning approaches, e.g. for learning approximations of the robot's inverse reachability [18] or training motion models [8]. Aside from IRM-based approaches, Yamazaki et al. [20] evaluate different "motions" for a target end-effector pose by analysing the error distributions associated with carrying out the motion, which consists of moving the base along a trajectory and the arm to a fixed joint configuration. These approaches all rely on the candidate grasp pose being available, e.g. from a grasp planner, which differs from our approach of finding base placements before planning the grasp motion.

3 Approach

Our approach to finding base placements for picking up objects is embedded in a service robot application, illustrated in Fig. 1b). Here, a human operator chooses the target object using a pointing gesture, then our planning algorithm finds a base placement based on the estimated pose of the object, and finally, the according navigation and manipulation motions are planned and executed using existing solutions available on the robot.



(a) Baseline approach (Without base placement planning)

(b) Proposed approach (With base placement planning)

Fig. 1. Execution Flow: The figures describe the flow of the algorithm in the baseline and proposed approaches. The baseline approach lacks base placement planning and instead always goes to a predefined pose defined before initiating pickup. While the proposed approach generates base placement candidates (BPCs) based on the RGB-D data and occupancy grid map before initiating pickup.

3.1 Object Pose Estimation

The object of interest is specified to the robot using gestures. For gesture recognition, we adapt and train an existing MLP architecture[3] to classify pixel values of hand landmarks detected using the MediaPipe library[4] as either *Scan*, *Point*, or *Stop*. Initially, the component waits for a *Scan* gesture from the operator, after which it detects objects in the scene. Next, when the *Point* gesture is recognized, a line is fitted to the detected landmarks of the pointing finger and extends from the fingertip until intersecting either the bounding box of a detected object or the image boundary. The object whose bounding box is intersected is then the target object, and its position in the map frame is estimated from the corresponding depth data in the RGB-D image using the Open3D library [22].

3.2 Base Placement Candidate (BPC) Generation

Algorithm 1: Robot placement planning algorithm.

Param: Radius of robot footprint r_r, robot height h_r
Param: Minimum and maximum reach of robot d_{min}, d_{max}
Param: Angle increment for generating radial vectors θ
Input: Object position in map frame ${}^m\mathbf{p}_o$
Input: Detected object width and height w_o, h_o
Input: RGB-D image of the scene $\mathcal{I}$
Input: Occupancy grid map $\mathcal{G}$
Output: Set of base placement candidates $\mathcal{S}$

Set of robot placement candidates: $\mathcal{S} \leftarrow \emptyset$
Generate set of radial unit vectors: ${}^m\mathbf{p}_o, \theta \mapsto \mathcal{V}$
foreach $\hat{\mathbf{v}}_i \in \mathcal{V}$ **do**
    Initial robot position candidate w.r.t. object: ${}^o\mathbf{p}_{ri} \leftarrow d_{min}\hat{\mathbf{v}}_i$
    while $\|{}^o\mathbf{p}_{ri}\| < d_{max}$ **do**
        if $\neg$ `HasCollisionRisk`(${}^o\mathbf{p}_{ri}$, r_r, h_r, w_o, h_o, $\mathcal{I}$, $\mathcal{G}$) **then**
            Pose candidate: ${}^o\mathbf{q}_{ri} \leftarrow \{{}^o\mathbf{p}_{ri}, {}^o\mathbf{o}_r\}$, s.t. robot points in $-\hat{\mathbf{v}}_i$
            `/* Transform to map frame and add to set of candidates */`
            $\mathcal{S} \leftarrow \mathcal{S} + \{{}^m\mathbf{q}_{ri}\}$
            `break()`
        end
        `/* Risk of collision detected, try position further away */`
        ${}^o\mathbf{p}_{ri} \leftarrow (\|{}^o\mathbf{p}_{ri}\| + r_r)\hat{\mathbf{v}}_\mathbf{i}$
    end
end
return $\mathcal{S}$

[3] `https://github.com/Kazuhito00/hand-gesture-recognition-using-mediapipe`
[4] `https://developers.google.com/mediapipe/solutions/vision/hand_landmarker`

Having obtained the object position in the map frame ${}^{m}\mathbf{p}_o$, we generate a set of potential base placements from where the pickup behaviour can be executed and rank them using a metric characterizing the difficulty of carrying out the corresponding navigation and manipulation motions. Algorithm 1 shows the pseudocode for our base placement planning component. For the remainder of this paper, notations of geometric relations will follow the ${}^{originframe}relation_{targetframe}$ convention, where poses are denoted with $\mathbf{q}$, positions $\mathbf{p}$, and orientations $\mathbf{o}$.

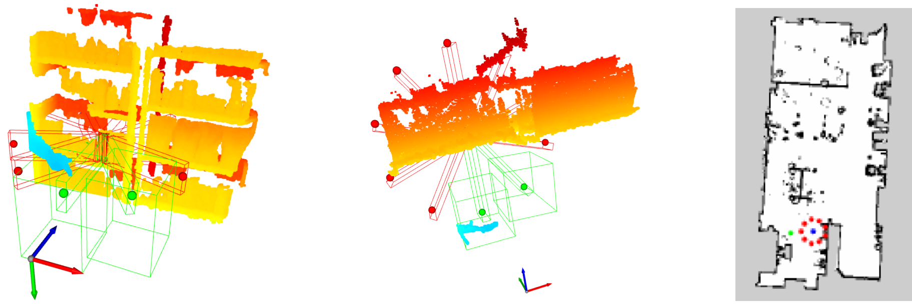
Generating robot placement candidates We first generate BPCs, one along each of radial unit vectors $\mathcal{V}$ spaced at equal angular intervals θ[5], originating from the projected object's position and pointing outward. Along each $\hat{\mathbf{v}}_i \in \mathcal{V}$, a BPC ${}^{o}\mathbf{q}_{ri}$ (w.r.t. the object) is generated within a predefined distance range $d_{min} \leq \|{}^{o}\mathbf{p}_{ri}\| \leq d_{max}$ and points towards the object, where the distance range corresponds to the robot's reachable workspace. This is done by initializing the position candidate to be at distance d_{min} (minimum safe distance for grasping) from the object and increasing the distance by r_r until d_{max} (max reach of the robot) if the risk of collision during pickup behaviour is detected. Repeating this process for each of the radial vectors, we acquire a set of BPCs $\mathcal{S}$.

Checking for risk of collision For each BPC ${}^{o}\mathbf{p}_{ri}$, we process the RGB-D image used for object detection to check if there is a risk of collision when the robot reaches the object (`HasCollisionRisk` in Algorithm 1). This is done by filtering the point cloud to select only points in the space that the robot may occupy during the pickup behaviour, and eliminate the candidate if the number of points found is higher than a fixed threshold k_{obs}. To this end, we perform the check twice, once for the robot body at the base placement pose and once for the manipulator during the reaching motion. The checked regions are visualized as cuboids in Fig. 2, where valid base placement candidates are coloured green, and pruned ones red. The larger cuboids are of size $2r_r \times 2r_r \times h_r$, corresponding to the robot's dimensions. Each smaller cuboid extends from the object position to the position candidate at the object's height, and has dimensions corresponding to the object's detected width and height w_o, h_o.

In addition to RGB-D data, we also process the occupancy grid map provided by the robot to check if placing the robot at ${}^{o}\mathbf{p}_{ri}$ may result in a collision. This is done by transforming a circle of radius r_r centred at ${}^{o}\mathbf{p}_{ri}$ to the grid coordinates of the occupancy map and check if the corresponding cells are occupied. In our experiments, we used only the static occupancy grid map to check for collision, but this step can also be performed on dynamic occupancy maps, if available, to check for dynamic obstacles.

If either of the above checks detects a risk of collision, that BPC is removed from consideration. Otherwise, the corresponding pose candidate at position ${}^{o}\mathbf{p}_{ri}$ and pointing towards the object, ${}^{o}\mathbf{q}_{ri}$, is added to the set of candidates $\mathcal{S}$.

[5] This angle can be calculated based on the robot's reach and footprint radius. For example, the angle to achieve a chord length of $2r_r$ on a circle of radius d_{min} is $\theta = 2\arcsin\frac{r_r}{2d_{min}}$



(a) 3D Front View (b) 3D Top View (c) Occupancy Grid Map View

Fig. 2. Checking robot placement candidates for risk of collision during the pickup behaviour using RGB-D data and occupancy grid map. `RGB-D data:` The regions checked for obstacles are visualized by cuboids. Candidates and corresponding cuboids that may result in the collision are coloured red. The larger cuboids correspond to the robot's dimensions. Each smaller cuboid extends from the object position to the position candidate at the object's height, and has dimensions corresponding to the object's detected width and height w_o, h_o. `Occupancy Grid Map:` Red dots represent the candidates checked for collision, while the green dot represents the robot's current position.

3.3 Navigation and Object Grasping

Algorithm 2 outlines how we carry out the object pickup using the robot placement candidates provided by Algorithm 1. The candidates are evaluated based on the combined distance from the robot's current position to the pose candidate and the distance from the pose candidate to the object, with the assumption that a longer distance would result in a more challenging grasp and closure distance than a limit would result in a collision. The `MotionCost` function in Algorithm 2 performs this evaluation to determine the best candidate. The resulting best candidate will be removed from $\mathcal{S}$ and used as the navigation goal for the robot to carry out the corresponding motions to pick up the object. If navigation fails, the remaining candidates will be reevaluated based on the current robot pose, and a new best candidate will be chosen for executing the pickup behaviour. This process is repeated until either the robot navigates successfully and begins the grasping motion, or if no candidate is left in $\mathcal{S}$. Planning and execution of the navigation motion is done using a modified version of the `move_base` ROS package[6] provided by Toyota. The subsequent manipulation motion is planned and executed using our existing solution previously described in [11].

4 Experimental results and evaluation

To evaluate our approach, we carry out a set of grasping experiments using our robot placement planning algorithm and compare the results to a baseline grasp

[6] `https://wiki.ros.org/move_base`

Algorithm 2: Object pickup based on planned robot placements.

Input: Object position in map frame ${}^{m}\mathbf{p}_o$
Input: Set of base placement candidates $\mathcal{S}$
while $\mathcal{S} \neq \emptyset$ **do**
  Read robot current pose ${}^{m}\mathbf{q}_r^c$
  Best candidate: ${}^{m}\mathbf{q}_r^* \leftarrow \arg\min_{{}^{m}\mathbf{q}_r}$ `MotionCost(`${}^{m}\mathbf{q}_r$, ${}^{m}\mathbf{p}_o$, ${}^{m}\mathbf{q}_r^c$`)` $\forall\, {}^{m}\mathbf{q}_r \in \mathcal{S}$
  $\mathcal{S} \leftarrow \mathcal{S} - \{{}^{m}\mathbf{q}_r^*\}$
  Navigation result: $r_{nav} \leftarrow$ `Navigate(`${}^{m}\mathbf{q}_r^*$`)`
  if $r_{nav} = false$ **then** `continue()`;
  Pickup result: $r_{pickup} \leftarrow$ `Pickup(`${}^{m}\mathbf{p}_o$`)`
  return r_{pickup}
end
1. how much distance from robot to propose pose 2. distance between proposed location and object

planning approach, in which the robot navigates to a predefined location and initiates the same object pickup behaviour. Our experiments are carried out using a Human Support Robot (HSR) [19] from Toyota, which can be seen in Fig. 3 along with our experimental setup. The remainder of this section will describe this setup and discuss the experimental results in detail.

4.1 Experimental setup and procedure

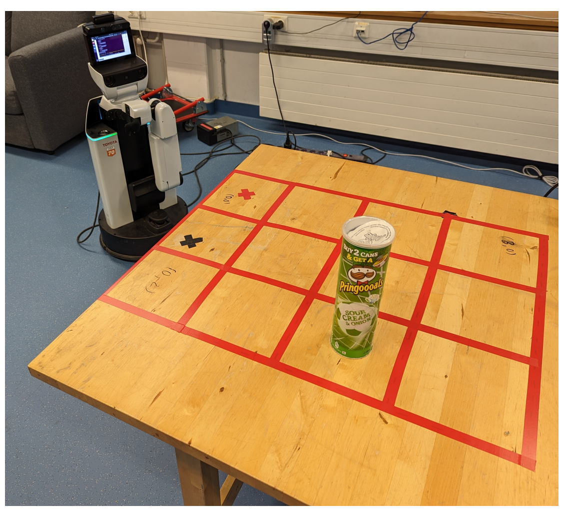


Fig. 3. Experimental Setup

In our experiments, the objects are placed on a standard dining table of dimensions (H $\times$ W $\times$ L) 74 $\times$ 80 $\times$ 180 cm. To control for different object placements on the table surface, we tape a 3-by-4 grid on the table, whose cells are squares of dimensions 20 $\times$ 20 cm, as shown in Fig. 3. Here, the long sides of the grid are 20 cm away from the long edges of the table, and one short side of the grid aligns with one short edge of the table. We define a 2D discrete coordinate system for referring to specific grid cells, in which a cell can be denoted (x, y) s.t. $x \in [0, 2]$ and $y \in [0, 3]$ are integers. In Fig. 3, cell $(0, 0)$ is marked with a red "x", and the x-axis of the coordinate system aligns with the table's short edge.

For each approach, we perform five grasps of the same object for each grid cell, i.e. 120 grasp attempts in total. A grasp is considered successful if the object remains in the gripper after the pickup behaviour, i.e. after the manipulator returns to its recovery configuration. If the robot is unable to hold on to the object, or if collision with the table occurs, the attempt is considered a failure. For the proposed robot placement planning approach, a human operator presents a "Scan" gesture to the robot to initiate object detection, then points to select the target object. The robot then plans the robot placement poses and carry out the corresponding motions as described in Sec. 3. For the baseline approach, the robot detects and estimates the object's pose, navigates to a predefined location close to the short edge of the table, and initiates the pickup behaviour.

4.2 Results and Discussion

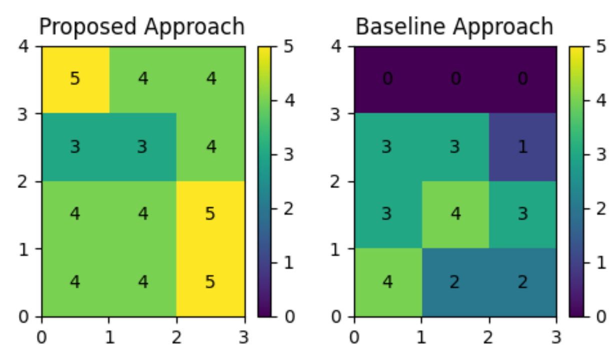


Fig. 4. Heatmap of approach's success out of 5 trials in each grid, corresponding to the experimental setup described in Sec. 4.1. With the baseline approach, the robot can not grasp from the table's longer edges and is unable to grasp when an object is placed far away. The proposed algorithm plans base placements according to the object's pose estimation, resulting in consistent grasp performance for all grid cells.

The results of our grasping experiments are shown in Table 1 and the heatmap of same is given in Fig.4. Overall, the proposed approach had a success rate of

Table 1. Number of successful and failed attempts during the grasping experiments.

Cell	**Proposed Approach**		**Baseline Approach**	
	Success	Failure	Success	Failure
(0, 0)	4	1	4	1
(1, 0)	4	1	2	3
(2, 0)	5	0	2	3
(0, 1)	4	1	2	3
(1, 1)	4	1	4	1
(2, 1)	5	0	3	2
(0, 2)	3	2	3	2
(1, 2)	3	2	3	2
(2, 2)	4	1	1	4
(0, 3)	5	0	0	5
(1, 3)	4	1	0	5
(2, 3)	4	1	0	5

81.7% compared to 40% of the baseline approach. Attempts using the baseline approach are more likely to fail when the object is placed in the grids far from the short edge of the table, i.e. far from the predefined navigation goal. When placed on the last 2 rows (cells $(x, 2)$ and $(x, 3)$), the object is near the limit of the robot's reach, resulting in the robot frequently colliding with the table. In contrast, the proposed approach allows the robot to navigate to other edges of the table before grasping, which makes picking up the object in the last row possible and even at a high success rate.

The experiments described thus far validate the soundness of the proposed robot placement planning algorithm and show how it enables a more flexible means to approach picking up objects, compared to the existing baseline approach. However, they have not clearly shown the benefit of utilizing both RGB-D data and occupancy grid maps in the planning process. This motivates additional experiments, e.g. grasping from a cluster of objects or with obstacles, to show whether the proposed approach can plan robot placements that reduce the risk of collision.

5 Conclusion and Future Work

In this paper, we present an approach to finding a robot placement from where it can pick up an object. Approaches addressing the same problem typically assume an end-effector pose is provided by some grasp planner, then find and evaluate robot placements based on this grasp pose. Instead, we propose to directly find valid robot placement candidates using the estimated object position, RGB-D data and occupancy grid maps, and evaluate them using a simple metric. We evaluate our approach with a series of grasp experiments on the Toyota HSR, in which we compare the approach to an existing baseline implementation that sends the robot to a fixed location before initiating the pickup behaviour.

The results validate the effectiveness of our approach in addressing the base placement problem for mobile manipulation and can serve as a baseline for developing and evaluating more advanced methods in the future. Furthermore, the algorithm is embedded in a Human-Robot Interaction (HRI) pipeline, in which a human operator chooses the target object using hand gestures. The pipeline allows an alternative HRI method, complementing standard speech recognition, for Robocup tasks that involve object retrieval in open environments.

As future work, several aspects of the proposed planning algorithm can be extended. First, we used a simple distance range in our approach to represent the robot's reachability, but more elegant methods, such as reachability maps, can be employed that consider the robot's full kinematics. Furthermore, while the cuboid between the object and a pose candidate serves as a simple approximation of the space occupied by the robot's arm while reaching for the object, it does not reflect the actual trajectory that the arm will follow in practice. A more advanced approach may check for obstacles along the planned end-effector trajectory, e.g. to avoid collision in case the trajectory would go under the table surface. Finally, instead of just combining the distance from the pose candidate to the object and the robot's current pose, better metrics can be employed to evaluate the candidates that take into account the potential navigation and manipulation motions that the robot needs to execute, such as one proposed by Yamazaki et al. [20].

References

1. Bohg, J., Morales, A., Asfour, T., Kragic, D.: Data-Driven Grasp Synthesis—A Survey. IEEE Trans. Robot. **30**(2), 289–309 (2014)
2. Burget, F., Bennewitz, M.: Stance selection for humanoid grasping tasks by inverse reachability maps. In: 2015 IEEE Int. Conf. on Robotics and Automation (ICRA). pp. 5669–5674 (2015)
3. Frank, F., Paraschos, A., Van Der Smagt, P., Cseke, B.: Constrained Probabilistic Movement Primitives for Robot Trajectory Adaptation. IEEE Trans. Robot. **38**(4), 2276–2294 (2022)
4. Ginesi, M., Meli, D., Calanca, A., Dall'Alba, D., Sansonetto, N., Fiorini, P.: Dynamic Movement Primitives: Volumetric Obstacle Avoidance. In: 2019 19th Int. Conf. Adv. Robot. ICAR. pp. 234–239 (2019)
5. Hart, J., Moriarty, A., Pasternak, K., Kummert, J., Hawkin, A., Hassouna, V., Pena Narvaez, J.D., Ruegemer, L., von Seelstrang, L., Van Dooren, P., Garcia, J.J., Mitzutani, A., Jiang, Y., Matsushima, T., Polvara, R.: Robocup@home 2024: Rules and regulations (2024)
6. Hoffmann, H., Pastor, P., Park, D.H., Schaal, S.: Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance. In: 2009 IEEE International Conference on Robotics and Automation. pp. 2587–2592 (2009)
7. Ijspeert, A.J., Nakanishi, J., Hoffmann, H., Pastor, P., Schaal, S.: Dynamical Movement Primitives: Learning Attractor Models for Motor Behaviors. Neural Computation **25**(2), 328–373 (2013)
8. Jauhri, S., Peters, J., Chalvatzaki, G.: Robot Learning of Mobile Manipulation With Reachability Behavior Priors **7**(3), 8399–8406

9. Jun Dong, Trinkle, J.C.: Orientation-based reachability map for robot base placement. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1488–1493 (2015)
10. Kleeberger, K., Bormann, R., Kraus, W., Huber, M.F.: A Survey on Learning-Based Robotic Grasping. Curr Robot Rep **1**(4), 239–249 (2020)
11. Mitrevski, A., Padalkar, A., Nguyen, M., Plöger, P.G.: "lucy, take the noodle box!": Domestic object manipulation using movement primitives and whole body motion. In: RoboCup 2019: Robot World Cup XXIII. pp. 189–200 (2019)
12. Paus, F., Kaiser, P., Vahrenkamp, N., Asfour, T.: A combined approach for robot placement and coverage path planning for mobile manipulation. In: 2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS). pp. 6285–6292
13. Reister, F., Grotz, M., Asfour, T.: Combining navigation and manipulation costs for time-efficient robot placement in mobile manipulation tasks. IEEE Robotics and Automation Letters **7**(4), 9913–9920 (2022)
14. Sahbani, A., El-Khoury, S., Bidaud, P.: An overview of 3D object grasp synthesis algorithms. Robotics and Autonomous Systems **60**(3), 326–336 (2012)
15. Sutanto, G., Su, Z., Schaal, S., Meier, F.: Learning Sensor Feedback Models from Demonstrations via Phase-Modulated Neural Networks. In: 2018 IEEE Int. Conf. Robot. Autom. ICRA. pp. 1142–1149 (2018)
16. Tan, H., Erdemir, E., Kawamura, K., Du, Q.: A potential field method-based extension of the dynamic movement primitive algorithm for imitation learning with obstacle avoidance. In: 2011 IEEE International Conference on Mechatronics and Automation. pp. 525–530 (2011)
17. Vahrenkamp, N., Asfour, T., Dillmann, R.: Robot placement based on reachability inversion. In: 2013 IEEE International Conference on Robotics and Automation. pp. 1970–1975 (2013)
18. Welschehold, T., Dornhege, C., Paus, F., Asfour, T., Burgard, W.: Coupling mobile base and end-effector motion in task space. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1–9 (2018)
19. Yamamoto, T., Terada, K., Ochiai, A., Saito, F., Asahara, Y., Murase, K.: Development of Human Support Robot as the research platform of a domestic mobile manipulator. Robomech J **6**(1), 4 (2019)
20. Yamazaki, K., Suzuki, S., Kuribayashi, Y.: Approaching motion planning for mobile manipulators considering the uncertainty of self-positioning and object's pose estimation. Robotics and Autonomous Systems **158**, 104232 (2022)
21. Yao, H., Laha, R., Figueredo, L.F.C., Haddadin, S.: Enhanced Dexterity Maps (EDM): A New Map for Manipulator Capability Analysis **9**(2), 1628–1635
22. Zhou, Q.Y., Park, J., Koltun, V.: Open3d: A modern library for 3d data processing (2018)