N^2M^2: Learning Navigation for Arbitrary Mobile Manipulation Motions in Unseen and Dynamic Environments

Daniel Honerkamp, Tim Welschehold, *Member, IEEE*, and Abhinav Valada, *Member, IEEE*

***Abstract*—Despite its importance in both industrial and service robotics, mobile manipulation remains a significant challenge as it requires seamless integration of end-effector trajectory generation with navigation skills as well as reasoning over long-horizons. Existing methods struggle to control the large configuration space and to navigate dynamic and unknown environments. In the previous work, we proposed to decompose mobile manipulation tasks into a simplified motion generator for the end-effector in task space and a trained reinforcement learning agent for the mobile base to account for the kinematic feasibility of the motion. In this work, we introduce Neural Navigation for Mobile Manipulation (N^2M^2), which extends this decomposition to complex obstacle environments, extends the agent's control to the torso joint and the norm of the end-effector motion velocities, uses a more general reward function and, thereby, enables robots to tackle a much broader range of tasks in real-world settings. The resulting approach can perform unseen, long-horizon tasks in unexplored environments while instantly reacting to dynamic obstacles and environmental changes. At the same time, it provides a simple way to define new mobile manipulation tasks. We demonstrate the capabilities of our proposed approach in extensive simulation and real-world experiments on multiple kinematically diverse mobile manipulators.**

***Index Terms*—Embodied AI, mobile manipulation, reinforcement learning (RL), robot learning.**

I. Introduction

WHILE recent progress in control and perception has propelled the capabilities of robotic platforms to autonomously operate in unknown and unstructured environments [1], [2], [3], [4], this has largely focused on pure navigation tasks [5], [6]. In this work, we focus on autonomous mobile manipulation, which combines the difficulties of navigating unstructured, human-centered environments with the complexity of jointly controlling the base and arm. Mobile Manipulation is commonly reduced to sequential base navigation followed by static arm manipulation at the goal location. This simplification is restrictive as many tasks such as door opening require the joint use of the arm and base and is inefficient as it dismisses simultaneous movement and requires frequent repositioning.

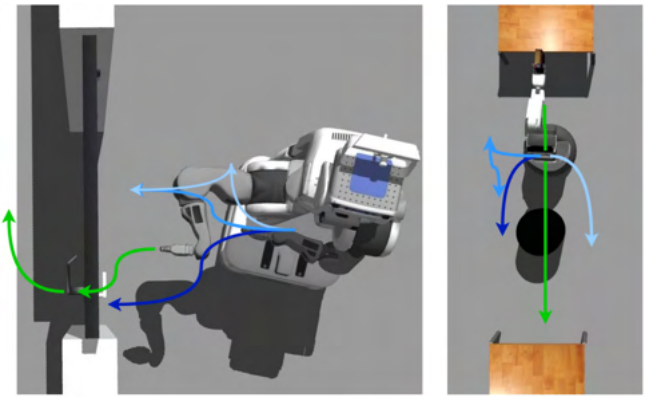

Fig. 1. Mobile manipulation tasks in unstructured environments typically require the simultaneous use of the robotic arm and the mobile base. While it is comparably simple to find EE motions to complete a task (green), defining base motions (blue) that conform to both the robot's and the environment's constraints is highly challenging. We propose an effective approach that learns feasible base motions for arbitrary EE motions. The resulting model is flexible and dynamic and generalizes to unseen motions and tasks.

Mobile manipulation requires a range of capabilities including collision-aware navigation, object interactions, manipulation, exploration of unknown environments, and long-term reasoning. As a result, approaches from a variety of areas such as planning, optimal control, and learning have been proposed. Inverse reachability map (IRM) approaches seek good base positioning for the robot base given the task constraints [7], [8] but often require expert knowledge and can be overly restrictive. Planning approaches [9], [10], [11] come with asymptotic optimality guarantees but scale unfavorably with the size of the configuration space, can have long planning times, and often require frequent replanning in dynamic environments. Model predictive control (MPC) formulations explicitly define and optimize over a range of collision and desirability constraints and recently achieved strong results in mobile manipulation [12], [13]. However, they are computationally expensive, often do not optimize past a limited horizon, and can struggle when objectives oppose each other. Learning-based methods efficiently learn directly from

Manuscript received 15 December 2022; revised 19 April 2023; accepted 31 May 2023. Date of publication 28 June 2023; date of current version 4 October 2023. This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program under Grant 871449-OpenDR and in part by Toyota Motor Europe with an HSR robot for experimental evaluation. This paper was recommended for publication by Associate Editor Y. Bekiroglu and Editor F. Chaumette upon evaluation of the reviewers' comments. *(Corresponding author: Daniel Honerkamp.)*

The authors are with the Department of Computer Science, University of Freiburg, 79110 Freiburg, Germany (e-mail: honerkamp@cs.uni-freiburg.de; twelsche@cs.uni-freiburg.de; valada@cs.uni-freiburg.de).

Code and videos are publicly available at http://mobile-rl.cs.uni-freiburg.de.

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TRO.2023.3284346.

Digital Object Identifier 10.1109/TRO.2023.3284346

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2024, Yokohama, Japan. Cite as T-RO paper.

high-dimensional observations and are well suited to handle unexplored environments [14], [15], [16]. Nevertheless, they either restrict the action space [14], [17] or rely on expert demonstrations [15] to cope with the high-dimensional action space and long-horizon nature of mobile manipulation. Furthermore, the learned behavior is often task-specific, requiring retraining for each novel task.

In this work, we formulate mobile manipulation as a goal conditional reinforcement learning (RL) problem in which the RL agent observes the end-effector (EE) motion and goal and aims to ensure that these motions remain kinematically feasible. Fig. 1 depicts a high-level overview of our approach. We extend the formulation introduced in [18] to unstructured obstacle environments, increase the agent's freedom to control the robots' torso lift joint and the norm of the velocity of the EE motions, generalize the reward function, introduce a regularization to the inverse kinematics (IK) solver to address configuration jumps, and develop a diverse training task. This provides a very simple yet effective way to define new tasks, as the RL agent takes care of all the complexities regarding collision-free navigation and kinematic feasibility. The resulting approach, which we term N^2M^2 (i.e., Neural Navigation for Mobile Manipulation), efficiently learns to solve the navigation for long-horizon mobile manipulation tasks, lasting thousands of steps or up to 5 min in real-time execution, generalizes to unseen tasks and environments in a zero-shot manner, and reacts instantaneously to changes in the environment without any planning times. Finally, the approach is directly applicable to a wide range of kinematics including both holonomic and nonholonomic robotic bases. We show that with appropriate action regularization, our hybrid approach can be trained without a complex simulator and directly transfers to the real world. We demonstrate these capabilities in both extensive simulation and real-world experiments on multiple mobile manipulators, across a wide range of tasks and environments and find that each of the introduced components strongly contributes to the overall large improvements over previous work.

This article makes the following main contributions.

1) We formulate the fulfillment of kinematic feasibility constraints for mobile manipulation tasks in the presence of obstacles as an RL problem.
2) We propose a reactive approach to learn complementary mobile manipulator base motions for arbitrary EE motions on unstructured obstacle maps.
3) We develop a procedurally generated training task and an approach to generate EE motions that can be used across a variety of mobile manipulators with largely varying kinematics and driving models.
4) We demonstrate the capabilities of our approach in extensive simulated and real-world experiments on unseen environments, obstacles, and tasks.
5) We make the code publicly available at http://mobile-rl.cs.uni-freiburg.de.

The rest of this article is organized as follows. Section II discusses existing literature and approaches for mobile manipulation. Section III describes the technical details of our method. Section IV then evaluates its capabilities and compares them to previous approaches. Finally, Section V concludes this article.

II. Related Work

Mobile manipulation tasks require the composition of a vast range of capabilities spanning perception, control, and exploration. In the following, we discuss previous methods from planning, optimal control, and learning to tackle these challenges.

Sequential navigation and manipulation: Due to the difficulties of planning in the conjoint space of the mobile manipulator base and arm, many existing approaches restrict themselves to sequential movements of the base followed by static manipulations with the arm. This decomposition has been popular across approaches based on reachability [8], planning [1], [19], [20], impedance control [21], and RL [14], [22].

Planning: To ensure kinematic feasibility in mobile manipulation tasks, planning-based approaches plan trajectories for the robot in joint space and as such only explore kinematically feasible paths [11], [23]. Sampling-based approaches such as rapidly exploring random trees (RRT) and their variants have been shown to perform well in high-dimensional spaces and come with certain optimality guarantees. However, increasing their configuration spaces and environments can result in long planning times or far from optimal solutions. Operation in unobserved or dynamic environments can trigger costly replanning if the environment changes [24], [25]. While certain constraints such as fixed orientations or task space regions [26] can be incorporated well to plan motions such as opening a door [20], the incorporation of arbitrary (EE) constraints is difficult and often requires expert knowledge and task-specific adaptations. In contrast, our approach can near-instantly react to dynamic changes, offers a natural way to incorporate unexplored environments, and can be directly applied to arbitrary EE motions.

Inverse Reachability maps [7] can be used to seek good positioning for the robot base given the task constraints [8], subsequently, the task is solved stationary. Combinations with planning methods exist [27], but it remains a hard problem to integrate kinematic feasibility constraints in task space mobile motion planning. Welschehold et al. [28] treat the kinematic feasibility of arbitrary gripper trajectories as an obstacle avoidance problem based on a geometric description of the inverse reachability. They analytically modulate the base velocity such that the base stays within feasible regions and orientations relative to the EE.

Optimal Control approaches have demonstrated promising results on complex manipulation tasks that require conjoint movements of the arm and base. MPC-based approaches have demonstrated strong performance on whole-body control tasks such as door opening [13], [29], obstacle avoidance [12], and articulated objects [30]. Constraints are explicitly incorporated into the objective function and optimized over a (usually fixed) rollout horizon. While efficient implementations can incorporate horizons of up to several seconds in near real-time, these approaches still require significant compute. Moreover, they also require simplified collision objects to achieve this and often do not take into account consequences beyond the rollout horizon. In contrast, our approach learns a value function reflective of the

full episode horizon and can perform fast inference with a single forward pass, without reliance on highly optimized implementations. Furthermore, it does not rely on explicit representations of the environment and offers direct extensions to arbitrary input modalities and partially observed environments.

Haviland et al. [31] propose a reactive controller for both holonomic and nonholonomic bases; however, they do not demonstrate any collision avoidance. Redundancy resolution methods specify desirable aspects of different solutions through additional constraints or parameters [32]. In contrast, our approach directly learns to resolve redundancies with regard to long-term optimality, removing the need to specify the explicit desirability of different choices.

Task and motion planning (TAMP) combines low-level motion planning and high-level task reasoning and has resulted in approaches that generalize to many robots and environments [33], [34], [35]. At the same time, it can be computationally demanding, depends on the specification of symbolic actions, and does not generally incorporate uncertainties or partial observability as it requires full information about the 3-D structure of the environment. In contrast, we assume that we can generate the desired EE motions but may act in unexplored or dynamically changing environments.

Obstacle avoidance: Learning-based methods have been shown to be effective for learning obstacle avoidance from high-dimensional sensor inputs such as color images [36], [37], LiDAR [38], [39], depth images [40], or a combination thereof [41]. These approaches have demonstrated the ability to avoid dynamic obstacles such as pedestrians [39], [41], [42]. Sensors such as LiDAR and depth sensors have also shown good transfer from the simulation into the real world [40], [42]. Several approaches have explored the use of 2-D obstacle maps to learn a local planner component to avoid pedestrians based on a 2-D LiDAR [39], to predict a collision map from a 2-D obstacle map based on a laser scan and binary collision events [38] or to directly map from 2-D maps to dense Q-values [43]. LiDAR and depth modalities are commonly used to construct local or global (occupancy) maps. These are then often transformed into egocentric perspectives [39], [43], [44] or into different resolutions [16], [44], [45] before being processed by neural networks. Optimal control approaches have successfully used voxel map representations [46] and signed distance fields [12], [30], [47].

Reinforcement learning: Recently mobile manipulation has also been tackled as an RL task. Relmogen [14] learns subgoals for arm and base but relies on the sequential execution of each and prespecified pushing motions. Kindle et al. [17] learn to directly control both arm and base; however, they restrict the arm movements to a plane. Jauhri et al. [22] learn a base positioning and a discrete decision on whether to use the arm. While these approaches learn effective policies for specific tasks, they cannot easily be applied to novel tasks. In contrast, we address the problem of enabling arbitrary EE trajectories, providing a straightforward technique to define novel tasks with arbitrary motion constraints in task space and the capability for zero-shot generalization to such unseen tasks at test time.

Recently several approaches and benchmarks for complex long-horizon tasks have been proposed. Most similar to our tasks is MoMaRT, which uses imitation learning to control a Fetch robot in simulation and in mostly seen environments [15]. In contrast, we focus on generalization to completely unseen environments as well as the real-world. ManipulaTHOR focus on the higher-level task aspects, using a strongly simplified robot arm and magic grasping action [48]. Rearrangement-style benchmarks define tasks as moving the environment from an initial to a goal state [49]. BEHAVIOR [50] and Habitat-2.0 [51] instantiate a large number of tasks in simulation. These approaches often abstract from the actual kinematics with actions such as "magical" grasping actions that immediately succeed [48], [50], [51] or explicitly provided lower-level motion primitives [50], [52]. In contrast, we assume that the agent is aware of the high-level task goals in the form of access to an EE motion but has to learn to achieve them with the actual robotic kinematics. Second, these approaches achieve limited success even when training separate agents specific to each task. We provide an approach that generalizes to unseen tasks without any retraining or finetuning. A very interesting aspect will be to incorporate and combine our work with the higher-level task focus of these benchmarks.

We formulate our approach as goal conditional RL problem [53], [54], which conditions its policy on a goal state to arrive at this goal. Hierarchical methods [55], [56], [57] abstract complex long-horizon tasks into a high-level policy proposing subgoals and goal-conditional low-level policies. While adapting these methods for mobile manipulation can improve sample efficiency [58], it still has to deal with the complexity and nonstationarity of learning hierarchical policies.

Articulated objects have been a particular focus for mobile manipulation. Mittal et al. [30] estimate the articulation parameters of unseen objects; then, use this to generate keyframes for the EE. Kineverse constructs articulation models for complex kinematics and demonstrates how these can be used to generate motions [59]. These works provide simple ways to generate EE motions for a wide variety of objects, complementing our approach and making it even more broadly applicable.

III. Neural Navigation for Mobile Manipulation

Mobile manipulation in real-world environments is a complex long-horizon task that combines navigation and control in a high-dimensional action space while respecting constraints imposed by the task, the hardware, and the environment. At the core, we decompose the mobile manipulation problem into an EE motion and base velocities learned by an RL agent with the goal to ensure that these concurrent EE and base motions are kinematically feasible. We extend the formulation introduced in [18] to enable the RL agent to simultaneously avoid collisions with the environment and further expand its control to the velocity of the EE motions and the height adjustment of the robot torso. We then generalize the objective and training scheme to learn policies that avoid configuration jumps and generalize to unseen environments while remaining applicable to a diverse set of mobile manipulation platforms. The resulting approach enables it to solve a very broad range of tasks in unstructured real-world environments. An overview of the proposed approach is depicted in Fig. 2.

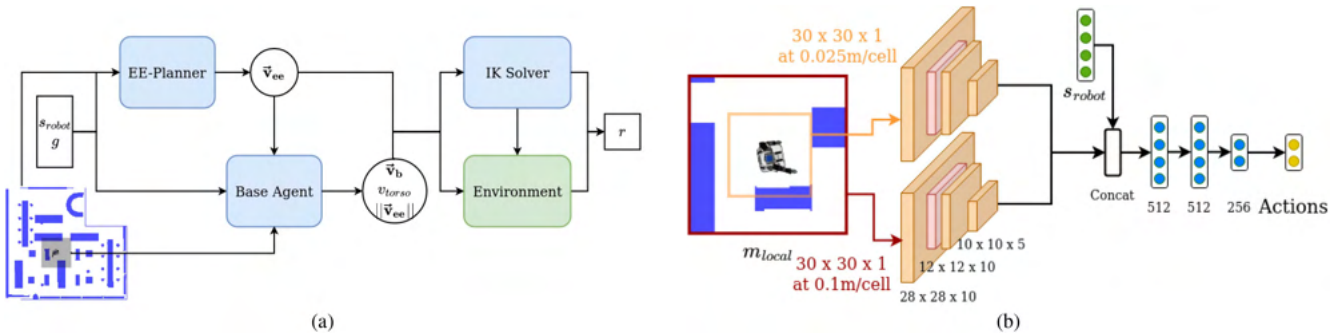

Fig. 2. (a) N^{2}M^2: We decompose mobile manipulation tasks into two components: An EE motion generation and a conditional RL agent that controls the base velocity $\vec{v}_b$, the torso lift joint velocity v_{torso}, and the norm of the velocity of the EE motion, $||\vec{v}_{\text{ee}}||$. The agent receives a local map of the environment (shown in gray) together with the robot state s_{robot}, an EE goal g, and the next EE motions $\vec{v}_{\text{ee}}$. Given the agent's actions, an IK solver then solves for the remaining arm joints and together with the environment returns the reward r. (b) Network architecture of the RL agent, illustrated for the actor. The critic uses the same architecture but additionally conditions on the actions by concatenating them with the robot state and outputs a value estimate instead of a vector of actions. In orange are convolutional layers with kernel size (3, 3) and stride 1. Red are max pool layers with stride 2. All nonfinal layers are followed by a rectified linear unit (ReLU) activation. The networks consist of 935 036 parameters for the actor and 935 283 for each of the critic's two Q-networks.

A. EE Motions

Defining tasks such that they can be achieved by algorithms while remaining generally applicable yet simple to specify for humans is a hard problem. Much of the recent work has focused on addressing this challenge through imitation learning or the definition of task-specific goal states. In the domain of mobile manipulation, outcomes often depend on specific EE operations. As such we define tasks by EE motions. This is, on the one hand, very expressive: A wide variety of tasks can be expressed this way. On the other hand, it provides a simple interface for humans to define and generate novel tasks quickly and unambiguously, as it requires only to reason about motions in simple Cartesian space, instead of a whole-body control problem. We provide a wide variety of example tasks in Section IV. Furthermore, this definition is not overly restrictive in terms of methodology as various types of motion generators for the EE can be employed, such as dynamic-system-based imitation learning, planning-based systems, or even an RL system.

In particular, we only assume access to an arbitrary EE motion generator f_{ee} that, given a current (partial or fully known) map m_{global} of the environment, the current EE pose $\text{ee}_t \in \text{SE}(3)$ and velocities $v_{\text{ee},t} \in \mathbb{R}^6$, and the current EE goal g, outputs next-step velocities $v_{\text{ee},t+1}$ for the EE. This motion generator does not handle the kinematic feasibility of the whole-body motion. Its only constraints are to avoid collisions of the EE with the environment and not to violate fundamental affordances of the robot's hardware, e.g., not creating EE poses outside the reachable space such as poses that are too high to reach regardless of the base positioning. Collision constraints for the robot base as well as kinematic feasibility constraints will be handled by the learned base agent described ahead. We can define this EE motion generation as a function

$$v_{\text{ee},t+1} = f_{\text{ee}}(\text{ee}_t, v_{\text{ee},t}, m_{\text{global}}, g). \tag{1}$$

B. Learning Base Control for Kinematic Feasibility

We formulate mobile manipulation tasks as a goal-conditional RL problem in which the RL agent observes EE motion and goal, and aims to ensure that these motions remain kinematically feasible [18]. The agent controlling the robot's base is operating in a partially observable Markov decision process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, T(s'|s,a), P(o|s), r(s,a))$ where $\mathcal{S}$, $\mathcal{O}$, and $\mathcal{A}$ are the state, observation, and action spaces, T and P describe the transition and observation probabilities, s, s' are the current and next state, respectively, o is the current observation, a is the current action, and r and γ are the reward and discount factor, respectively. The agent's objective is to learn a policy $\pi(a|o)$ that maximizes the expected return $\mathbb{E}_\pi[\sum_{t=1}^{T} \gamma^t r(s_t, a_t)]$. In the goal conditional setting [53], the agent then learns a policy $\pi(a|o,g)$ that maximizes the expected return $r(s,a,g)$ under a goal distribution $g \sim \mathcal{G}$ as $\mathbb{E}_{\pi,\mathcal{G}}[\sum_{t=1}^{T} \gamma^t r(s_t, a_t, g)]$. At each step, an arbitrary EE motion generator produces the next step velocities v_{ee} for the EE. The RL agent receives an observation $o = [v_{\text{ee}}, \text{ee}, \hat{\text{ee}}, g, s_{\text{robot}}, a_{t-1}, m_{\text{local}}]$ consisting of these velocities v_{ee}, the current and resulting desired EE poses ee and $\hat{\text{ee}}$, and an EE goal pose g in the robot's base frame together with the current robot state consisting of the joint angle configurations, the agent's previous actions a_{t-1} and a binary local obstacle map m_{local} centered and oriented around the base and outputs actions $a \sim \pi(a|o,g)$ consisting of base velocities $\mathbf{v_b}$, the torso lift joint velocities v_{torso} and the norm of the EE velocities n_{ee}. In practice, we do not let the agent observe the final EE goal, which can often be far away, but we repeatedly apply the EE motion generator f_{ee} to generate an intermediate goal roughly 1.5 m ahead of the agent.

The objective of the agent is to ensure that these EE motions remain feasible. Given the desired EE motion and the agent's base commands, we use IK to solve for the remaining degrees of freedom in the arm. The IK solver is restricted to solutions that adhere to joint limits and self-collision constraints. This means the agent has to produce base motions such that the EE poses can be achieved while staying within these constraints, failure to do so will result in large deviations from the desired EE poses. To achieve this objective, we extend our previous work [18] in several dimensions. First, we generalize the kinematic feasibility reward as

$$r_{\text{ee}} = -||\hat{\text{ee}}_{t-1} - \text{ee}_t||^2 \cdot d_{\text{ee}}(\hat{\text{ee}}, \text{ee}) \tag{2}$$

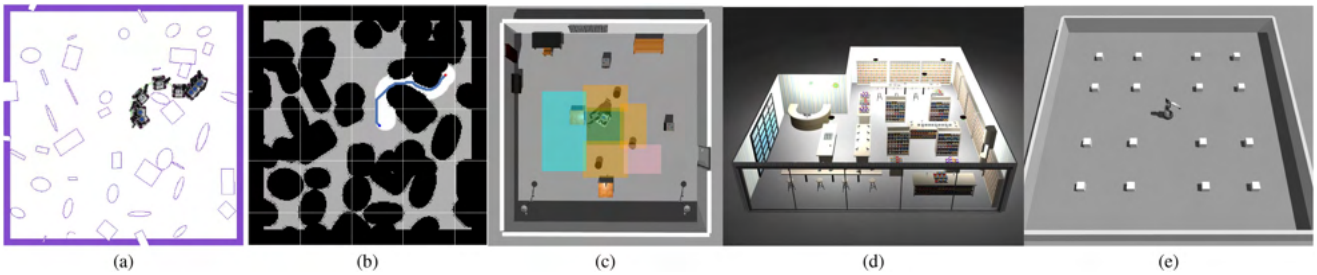

Fig. 3. Depiction of the training and testing environments. (a) Procedurally generated training task. (b) Generated weights and path of the end-effector planner according to (9). The darker the image, the higher the cost. (c) Articulated object environment. The starting area of the robot is shown in green, the spawn area of the obstacles is shown in orange (except for door and spline tasks), and the starting area for the door task is shown in pink. For the p&p task, the robot has to grasp a cereal box on the bottom table and place it onto a table randomly spawned in the cyan area. (d) Bookstore map. (e) Dynamic obstacle task.

where ee and $\hat{\text{ee}}$ are the achieved and desired EE poses with position and orientation components ee_{xyz} and ee_o, d_{rot} calculates a rotational distance of the quaternions $d_{\text{rot}} = 1.0 - \langle \hat{\text{ee}}_o, \text{ee}_o \rangle^2$, and c_{rot} is a constant scaling both components into a similar range. This allows us to explicitly tradeoff precision in the achieved pose with adhering to executable arm motions within the IK solver to address the configuration jumps observed in [18]. We use the flexible BioIK solver and regularize the IK solver with a minimum displacement goal, consisting of the squared distances of each joint, weighted by the reciprocals of the maximum joint velocities [60]. While this restricts the arm joints to stay close to the last time step, for robots with flexible arms, there may still be several possible configurations for a given EE motion. Moreover, the selected arm configuration can be important to fulfill the remaining motions. To increase the RL agent's control over what configuration the arm goes into, we extend its control to the torso lift joint. This joint is often very slow and as a consequence requires a substantial look-ahead to change significantly in height. We extend the agent's action space with a learned torso velocity and use the IK solver only for the remaining arm joints.

An episode terminates early if the gripper deviates too much from the desired pose for more than 20 steps. We optimize this as an infinite horizon task, correcting for the (non-Markovian) early terminations [61] and taking into account that robots are expected to act in the real world in which they will have to continue reaching new goals after fulfilling the current task. To do this, we continue to bootstrap the value functions in the final episode states. Finally, we replace the regularization of the agent's actions' norm with a regularization of the acceleration, incentivizing smooth instead of small actions

$$r_{\text{acc}} = ||a_t - a_{t-1}||^2. \tag{3}$$

To ensure the environment remains Markovian, we extend the observation space with the agent's previous actions. In our experiments, we find this regularization to be essential for the transfer to the real world.

C. Obstacle Avoidance

To perceive the environment, we equip the agent with a local occupancy map of its surroundings. Occupancy maps provide several advantages as follows.

1) They can be constructed from and integrate several sensors, such as LiDAR, RGB-D, and range sensors, ensuring applicability across different robots with different sensors.
2) They offer a geometric abstraction over sensors such as RGB images, which we hypothesize is easier to generalize to unseen objects and environments at test time.
3) They can be constructed in either 2-D or 3-D voxel maps.

In the following, we rely on a 2-D representation, which, as we will show in the experiments, is sufficient for a large range of real-world tasks while keeping computational costs low and during training can be generated without having to rely on a complex simulator.

The agent receives a robot-centered map m_{local} with a side length of 3 m. This map is processed by a map encoder before being concatenated with the robot's proprioceptive state and the goal observation. The map encoder receives two versions of the local map: 1) a coarse one at a resolution of 0.1 m and 2) a fine-grained version with a side length of 0.75 m at a resolution of 0.025 m. While the low resolution enables learning of larger spatial contexts, the fine-grained map reduces the impact of the grid discretization and enables the precise perception of available space in narrow corridors while keeping computational costs minimal. The concatenation of the two maps is passed through three convolutional layers, the first followed by a maxpool, with ReLU activations, and then flattened into a vector. The network is illustrated in Fig. 2(b). We then extend the reward with a collision penalty of

$$r_{\text{coll}} = -10 * \mathbb{1}_{\text{coll}} \tag{4}$$

where $\mathbb{1}_{\text{coll}}$ is a binary collision indicator for the base of the robot. As the RL agent directly learns a mapping from observations to actions, the resulting approach can handle both convex and nonconvex obstacles (see Fig. 3 and video).

D. Controlling Motion Velocities

While we can express discretized motions as a time-stamped sequence of poses in SE(3), many real-world tasks do not depend on a particular execution speed. Furthermore, in cluttered environments, the robot base has to frequently evade obstacles, potentially requiring significantly longer paths than the EE. As a consequence, we propose to also learn to control the norm of the velocity of the EE motions. We extend the agent's action

space by an additional action n_{ee} that controls the norm of the EE velocity after observing the desired next motion by scaling it to

$$v'_{\text{ee}} = n_{\text{ee}} * \frac{v_{\text{ee}}}{||v_{\text{ee}}||}. \tag{5}$$

To incentivize fast motions whenever possible, we add the following reward:

$$r_{\text{vel}} = -(v_{\text{ee, max}} - n_{\text{ee}})^2 \tag{6}$$

where $v_{\text{ee, max}}$ is the maximum velocity the EE is allowed, which we set equal to the maximum base velocity.

To prevent the agent from finding fast movement through difficult poses as a valid strategy to minimize collisions or IK failures, we furthermore transform the penalties for collisions and IK failures from a reward per time step into a reward per distance by multiplying them with the normalization term

$$n_{\text{vel}} = \frac{n_{\text{ee}}}{v_{\text{ee, max}}}. \tag{7}$$

E. Overall Objective

Combining the objectives, the overall reward function is

$$r = n_{\text{vel}} \left(\lambda_{\text{ik}} r_{\text{ik}} + r_{\text{coll}}\right) + \lambda_{\text{vel}} r_{\text{vel}} + \lambda_{\text{acc}} r_{\text{acc}} \tag{8}$$

where λ_{ik}, λ_{vel}, and λ_{acc} are constants to balance the objective (see Table II). We optimize this objective with model-free RL, namely soft-actor critic (SAC), which has been shown to effectively learn robust policies for continuous control tasks [62].

F. Training Environment

To generalize to unseen motions and obstacles, the agent has to be exposed to a wide variety of motions, poses, and obstacles. We introduce a procedurally generated environment that uses elementary shapes to generate obstacles and a simple yet effective obstacle-avoiding EE motion generator. As we will demonstrate in our experiments, the resulting policy generalizes to unseen geometries and motions at test time.

Task: In each episode, we randomly arrange elementary shapes, namely rectangles and ellipses, on an empty map. Each obstacle is placed on a regular grid, offset by a random number drawn from a normal distribution, and placed in a random orientation. The obstacle's width, breadth, and height are drawn from a uniform distribution. An example of such a map is depicted in Fig. 3. Given the obstacle map, we set a goal-reaching task by sampling a random start pose, random initial joint values, and a random goal position within a distance of 0.5 m to 5 m from the start.

While procedurally generated environments have proven very powerful to learn robust policies, they come with their challenges. In particular, we have to ensure that the generated task is solvable. For this, we follow a simple heuristic and reject any goal for which no valid path can be found in a map with an inflation radius of 0.4 m. While more elaborate methods such as curricula [63] and adversarial environment generation [64] have shown promising results, we found it not just simpler but also more effective to directly randomize and maximize the diversity of the environment and goals. During training, we simply integrate the robot kinematics over time and directly use the IK solutions as next joint values for the arm, allowing us to generate data very quickly and without relying on a complex simulator. We will refer to this as an *analytical environment* throughout the remainder of the article. In all other environments, the IK solutions are used as a goal for the low-level arm controllers.

End-effector motions: For training, we propose an instance of a simple obstacle-aware EE motion generator that remains agnostic to the robot. In Section IV, we evaluate how well the behavior learned on these motions generalizes to arbitrary, unseen motions. To ensure that the desired motions are in general feasible, we use two heuristics and develop a refined A*-based planner to implement them as soft constraints, which are as follows.

1) Avoid EE collisions: a minimum distance d_{ee} of the EE from tall obstacles. This can be implemented by inflating the obstacle map, as is common practice in robot navigation. We ignore obstacles that are smaller than the maximum height the EE can reach (minus a small margin) max_z, meaning the EE motion is allowed to pass over obstacles where kinematically possible.
2) Ensure the base can follow the EE motions: We first plan a path from the initial base pose to the EE goal in a map that is inflated by the radius of the robot base. We then define a maximum distance d_{base} that the EE is allowed to deviate from this base path. We set this value to the range of the robot's arm. We implement this by inflating the base path by d_{base} and adding the inverse of this to the weights.

Together, this results in the following weights w of the A*-planner, consisting of the inflated EE path and the inverse of the inflated base path

$$\begin{aligned} w = c * \mathbb{1}\,[&\text{inflate}\,(m_{\text{global},z+}, d_{\text{ee}}) \\ &+ (1 - \text{inflate}\,(m_{\text{basepath}}, d_{\text{base}}))] \end{aligned} \tag{9}$$

where c is a constant, $m_{\text{global},z+}$ is a map with only the obstacles larger than max_z over which the EE cannot pass, and m_{basepath} is a map consisting only of the shortest path from the initial base pose to EE goal obtained by A*. An example of the resulting weights is shown in Fig. 3.

We then generate dense waypoints with A* and interpolate them with a linear dynamic system to produce the actual EE motions. We linearly interpolate the height over the found path from the current height toward $\max(\text{next\_obstacle\_}z + \text{height\_margin}, \text{goal\_}z)$. Rotations are obtained via spherical linear interpolation (slerp) from start to goal orientation. As we demonstrate in the experiments, this results in an effective yet simple method capable of generating motions for complex, real-world floor plans. Note that while more sophisticated methods could generate easier-to-achieve motions in crowded maps, our main interest here is to generate diverse motions for the RL agent. The task is deemed successful if the agent gets within 2.5 c and a rotational distance of 0.05 of the EE goal, has no base collisions or self-collisions, and does not deviate more than 10 c or a rotational distance of 0.05 from the desired motion. We

TABLE I
VELOCITY AND POSE CONSTRAINTS

Parameter	EE-Motion	PR2	TIAGo	HSR
Max. velocity (m/s)	0.2	0.2	0.2	0.2
Max. rotation (rad/s)	0.3	1.0	1.0	1.0
Goal height (m)	-	[0.2, 1.55]	[0.2, 1.5]	[0.2, 1.4]
Restr. height (m)	-	[0.4, 1.0]	[0.4, 1.1]	[0.4, 1.1]

early terminate the episode if the agent violates any of these constraints for more than 20 steps.

IV. EXPERIMENTAL EVALUATIONS

We evaluate our approach across different robotic platforms, different environments, and a wide variety of tasks. In these experiments, we aim to answer the following questions.

1) Do agents trained with our proposed approach generalize to unseen EE motions?
2) Does our approach generalize across robotic platforms with different kinematic abilities?
3) Does the agent learn robust obstacle avoidance that generalizes to a) occupancy maps generated by real-world LiDAR scans and b) unseen obstacles?
4) Does our approach scale to complex, real-world-based obstacle environments?
5) Can the agent react to dynamic obstacles and changing environments?
6) Does our approach transfer to direct execution in the real world?

A. Experimental Robotic Platforms

We train agents for three different robotic platforms differing considerably in their kinematic structure and base motion abilities. The *PR2* robot consists of an omnidirectional base, a height-adjustable torso, and a 7-DOF arm, giving it high mobility and kinematic flexibility. With a base diagonal of 0.91 m, it is comparably large, limiting its maneuverability in narrow spaces. The *Toyota HSR* robot has an omnidirectional base as well, but the arm is limited to 5-DOF including the height-adjustable torso. As a result, it cannot reach all poses in SE3 unless it coordinates movements with its base. The *TIAGo* robot is equipped with a height-adjustable torso similar to the PR2 as well as a flexible 7-DOF arm. However, it uses a differential drive, restricting its mobility compared to the PR2. All three robots are equipped with a base LiDAR sensor, covering 270, 240, and 220 deg on the PR2, HSR, and TIAGo, respectively. The TIAGo additionally has three range sensors pointing backward, each with a field of view of 29 deg and a range of 1 m. In the simulation, we also equip the PR2 and HSR with the same range sensors. The action space for these platforms is continuous, consisting of either one (diff-drive) or two (omni) directional velocities $\mathbf{v}_{\mathbf{b},\{\mathbf{x},\mathbf{y}\}}$, and an angular velocity $\mathbf{v}_{\mathbf{b},\theta}$. This action space is completed with two more actions, one controlling the velocity of the EE motion and the other controlling the torso lift joint. The only exception to this is the HSR for which we do not find a benefit in learning the torso over solving for it with the IK as it only has limited DoF and, therefore, does not learn the torso velocities. Table I lists the constraints we set across the different platforms in the analytical environment.

B. Baselines

We compare our proposed approach against a range of methods from planning, optimal control, and machine learning.

MPC: As the main baseline, we compare against a recent MPC approach [30]. The method uses the SLQ algorithm to minimize the deviation to the desired EE trajectory with additional soft constraints for self-collision, joint velocity, and joint position limits. To ensure a fair comparison, we change all robot collision meshes to simplistic geometries, reduce the number of self-collision constraints to a bare minimum (avoiding EE—torso and EE—head collisions), and set all unused joints as fixed. Collision avoidance is incorporated as a hinge-loss function based on a signed distance field. We provide it with a groundtruth 2-D signed distance field at a resolution of 2.5 c from the obstacle map and model the base collisions as a cylinder centered on the robot base. For the rectangular base of the PR2, we define four more small cylinders at the corners of the base and implement the derivatives with regard to position and orientation. We follow the authors and run the algorithm at a control frequency of 30 Hz with a time horizon of four seconds. The agent has to follow the same EE motions as our approach. As the original code is not publicly available, we reimplement it based on the same optimization library.[1]

E2E: An end-to-end RL agent that does not rely on an IK solver but directly learns to control the whole robot by extending the action space to include the joint velocities for the arm. It receives the same desired EE motions and reward signal but directly acts in the full configuration space of the robot to achieve these motions. In the case of self-collisions, the previous joint values are kept. This is indicated to the agent via a binary variable that is true if there would have been a self-collision.

LKF: The original learning kinematic feasibility approach [18] uses a binary indicator as a reward and does not take into account obstacles. In a range of ablation studies, we extend it until we reach our approach.

RRTConnect: For comparison with classical planning approaches, we evaluate RRTConnect [10]. However, as general motion constraints over time cannot easily be defined in the sampling space, we omit these constraints. The planning algorithm only receives the same start and goal poses as the learning agent and is allowed to freely move to the target pose. As such it has much more freedom to achieve the goal, but, at the same time, is limited in applicability to goal-reaching and pick&place tasks. We use the planner implementations provided by the MoveIt motion planning library [65]. While samplers can, in principle, support arbitrary constraints such as nonholonomic bases, current ROS1 implementations do not support them. As Tiago is not yet ROS2 compatible, we do not evaluate it on this platform.

[1] [Online]. Available: https://github.com/leggedrobotics/ocs2

TABLE II
HYPERPARAMETERS FOR THE DIFFERENT APPROACHES

RL			
tau	1e−3	buffer size	10e5
lr	1e−4	lr end	1e−6
gamma	0.99	steps	10e6
ent coef	learned	train intensity	0.1
batch size	256	hidden layers	[512, 512, 256]
action noise	0.04	frame skip	8 (Tiago)
λ_{ik}	50	c_{rot}	2
λ_{vel}	0.1	λ_{acc}	0.05
MPC			
time horizon	{4 s, 6 s}	frequency	30 Hz
μ collision	{500, 1000, 5000}	δ collision	{0.001, 0.01, 0.1}
μ joint limit	{0.01, 0.1}	δ joint limit	{0.001, 0.01}
min step	{0.01, 0.03}		
RRTConnect			
max waypoint	{0.1, 0.2, 0.4}	range	{default, 1, 10}
plan attempts	{1, 10}	time limit	200 s
Bi2RRT*			
extend step	{0.1, 0.2, 0.4}	time limit	200 s

Notes: For the planner and MPC baselines, we perform a grid search on the obstacle task over these values and then report the results of the best parameters on all tasks.

Bi2RRT:* A bidirectional RRT* algorithm developed for mobile manipulation with the PR2, which achieves efficient sampling by using a two-tree variant of Informed RRT* [11]. As for RRTConnect, we omit the motion constraints. We use the authors' implementation.

C. Evaluation Setup

As the main focus of this work is the base motions, we provide the EE-planner with a groundtruth map of the environment for all tasks except those involving dynamic environments. However, the RL agent is always restricted to the local map (groundtruth in the analytical environment, based on its sensors in all other environments). The planner baselines receive access to the full 3-D groundtruth planning scene. Only for the planner baselines, we additionally simplify all complex collision meshes in the bookstore map with simple geometries to reduce the impact of collision checking on the planning times. We tune the MPC and planner baselines for each robot via grid search on the rnd obstacle task and then evaluate the best set of parameters on all tasks. For our approach, we use a single set of parameters across all robots without any further platform-specific tuning. Hyperparameters for all methods are reported in Table II. The approaches were evaluated on an AMD Ryzen 9 5900X or AMD EPYC 7452 CPU. As the PR2 Gazebo physics no longer matches the real robot,[2] we update it to roughly fit the actual hardware by setting the wheel, and rotation joint efforts to 25 and the wheel controller's proportional gains to 200.

Metrics: A task is deemed successful if the EE reaches the goal without any collisions and never deviates more than 10 cm or a rotational distance d_{rot} of 0.05 from the desired motion. In the analytical environment, we also count configuration jumps as a failure, where we define a jump as exceeding any of the robot's maximum joint velocities. In the analytical environment, we only track base collisions, in the Gazebo simulator, we check all collisions with the robot body. We evaluate each task over 50 episodes. For learning-based methods, we additionally average over five models trained with different random seeds.

[2][Online]. Available: https://github.com/PR2/pr2_controllers/issues/402

D. Generalization to Arbitrary Motions

To evaluate the generalization to arbitrary motions, we generate a range of tasks from different motion systems as follows. *A*-slerp* refers to the A*-based motions used during training, as described in Section III-F.

A-fwd* uses the same EE path but generates different desired EE orientations: Instead of spherical interpolation, it generates motions in which the EE always points in the direction of movement. Only when close to the goal, it uses slerp to achieve the desired final EE orientation. From these motions, we construct the training task, labeled *rnd obstacle*, and a pick&place (*p&p*) task in which the agent has to grasp a cereal box from one table and place it down on another table. An *imitation learning system* learned from a human teacher [66]. The motions are encoded in a dynamic system following a demonstrated hand trajectory to manipulate a certain object. In particular, we use motions to grasp and open a cabinet, grasp and open a drawer and grasp a door handle, push it down, and open the door inward while driving through it. This task is particularly challenging as it requires avoiding both the door frame and the moving door, resulting in a highly constrained and narrow navigation task. With 0.89 m, the door frame is more narrow than the PR2 base-diagonal of 0.91 m. The moving door makes the environment dynamic as it changes the free space. As these motions encode knowledge of how to interact with the objects, we can use them to enable object manipulations that the agent has not seen in training.

Spline interpolation draws five random waypoints in SE(3) with a distance of 1 m to 3 m from the previous and connects them using cubic splines and spherical interpolation. This results in arbitrary motions over a total distance of 5 m to 15 m.

With these motions, we extend the suite of mobile manipulation tasks introduced in [18] while also removing all restrictions on the initial start pose. The task objects are situated in a virtual room and the robot spawns in a random pose and configuration within an area in the center of the room. To evaluate the obstacle avoidance capabilities, we place random obstacles in the path of the agent. We allow the EE motions to directly pass over these obstacles, leading to challenging poses and decisions on how to position the base. The task setup and start area are shown in Fig. 3.

We first evaluate the agents in the analytical environment, abstracting from low-level execution and sensors. The results are shown in Table III. MPC solves a significant amount of episodes across all the robots. In particular, it performs very well on the obstacle-free spline task but it frequently gets stuck around obstacles, only making progress once time progresses and the desired EE pose moves far enough away for the tracking cost term to dominate. This results in a violation of one of the

TABLE III
SUCCESS RATES FOR EXPERIMENTS IN THE ANALYTICAL ENVIRONMENT

		A^*-slerp					Imitation Learning			Spline	Avg
	Agent	rnd obstacle	p&p	bookstore p&p	dynamic obstacle	dynamic p&p	cabinet	drawer	door	spline	
PR2	MPC	54.0	70.0	8.0	64.0	54.0	62.0	76.0	32.0	96.0	57.3
	E2E	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	N²M²	86.0	97.6	70.0	84.4	81.6	97.2	97.6	98.4	80.8	**88.2**
HSR	MPC	64.0	72.0	48.0	60.0	60.0	68.0	68.0	66.0	98.0	67.1
	E2E	25.2	38.8	10.4	17.6	13.2	52.4	14.4	43.2	38.0	28.1
	N²M²	63.2	92.0	68.0	82.0	88.0	93.2	87.2	88.0	84.0	**82.4**
Tiago	MPC	46.0	68.0	0.8	40.0	30.0	62.0	52.0	32.0	86.0	46.3
	E2E	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Ours	61.2	91.6	42.0	54.4	50.4	81.6	89.2	62.4	56.0	**65.4**

Notes: Evaluation on unseen tasks from three different motions systems, an A^*-based system, an imitation system learned from human demonstrations, and spline interpolation of random waypoints. The last column reports the average across all tasks. We evaluate all models on three different robotic platforms, the PR2, HSR, and TIAGo.
Best performing methods are denoted in bold.

constraints: either large deviations from the desired pose, base collisions, or joint limits as it is unable to trade them off optimally in these situations. Due to the difficulties in balancing these soft constraints, in the door opening task, the MPC controller frequently drives over the door frame with the PR2. The E2E model is able to solve a reasonable number of tasks on the HSR. However, as the action space increases with the 7-DoF arms of the PR2 and TIAGo, it is no longer able to succeed in any of the tasks. This matches the fact that recent RL-based mobile manipulation approaches have to restrict the action space to learn. In contrast, our hybrid approach scales effectively with the configuration space of the robot and is able to use the full flexibility of the PR2, achieving the best results on this platform with far above 90% success on many tasks. We achieve similar success rates on the HSR, demonstrating the agent's ability to use the base to complement its limited arm. The most difficult platform is the TIAGo. While our approach does very well on p&p and the imitation learning tasks, performance is significantly lower on the rnd obstacle and spline tasks. We hypothesize that the differential drive induces a much harder exploration problem. In particular, it is much harder to move from one mode of operation to another: e.g., in front of an obstacle, the base has to decide to either turn left or right. Once it has made a decision, it cannot easily change the path without violating the EE motion. As such it becomes much harder to escape local optima in the behavior policy. To better learn long-horizon policies for this robot, we start training with a lower base control frequency of 0.0125 Hz and then linearly increase it to 10 Hz over the course of training. Across all tasks and robots, our approach significantly outperforms the other methods. Also note that, given the procedural generation of the environment, it may not be possible to achieve a perfect score on the rnd obstacle task, as some EE motions may not be possible to fulfill kinematically for some of the robots.

Qualitatively, we find that the agent produces reasonable behavior such as seeking robust poses in the center of its workspace or moving backward until reaching an open space to turn. Examples are shown in Fig. 4 and in the accompanying video. Fig. 5 shows examples of the learned EE velocity norm. We find that the policy learns more complex strategies than simply reducing velocities when close to an obstacle and varies its behavior with the obstacles, the current configuration of the robot, and the desired next EE poses. The bottom subfigure shows the TIAGo agent strongly reduces the velocity as the EE passes over an obstacle while the base has to take a much longer way around the left of the obstacle. Furthermore, the changes in velocity are consistent over time and do not immediately jump from one value to another and the agent learns to use the full range of velocities, not only the extrema of the available range.

E. Ablation: Individual Components

To evaluate the impact of our contributions, we perform extensive comparisons with the original LKF approach. As it cannot avoid obstacles, we train these approaches on a random goal-reaching task in an empty map and compare the ability to follow the motions on a subset of the previous tasks *without obstacles*. Note that results are not directly comparable to those reported in [18], as our task definitions are generally more challenging as they remove restrictions on initial poses and include the torso joint of the PR2. We report both the same success rates as before as well as a success rate that does not penalize configuration jumps as used in [18], which we label *w/jumps*.

The results are shown in Table IV. Row by row we add our contributions. We first switch to the BioIk solver with minimal displacement regularization and the generalized L2 kinematic feasibility reward function. This largely increases the success rate by removing most configuration jumps. For the PR2 and TIAGo, we then additionally learn the torso lift velocities, which have a similarly large impact on the success rate. This is because most jumps occurred in the torso lift joints, which move much slower than the arm joints. Furthermore, for many goals, it is important to plan ahead and to start moving the torso early on in the episode to achieve large height differences. Next, we also learn to control the velocity of the EE motions. This slightly increases success rates, even in the absence of obstacles. We found the impact of this term even more important

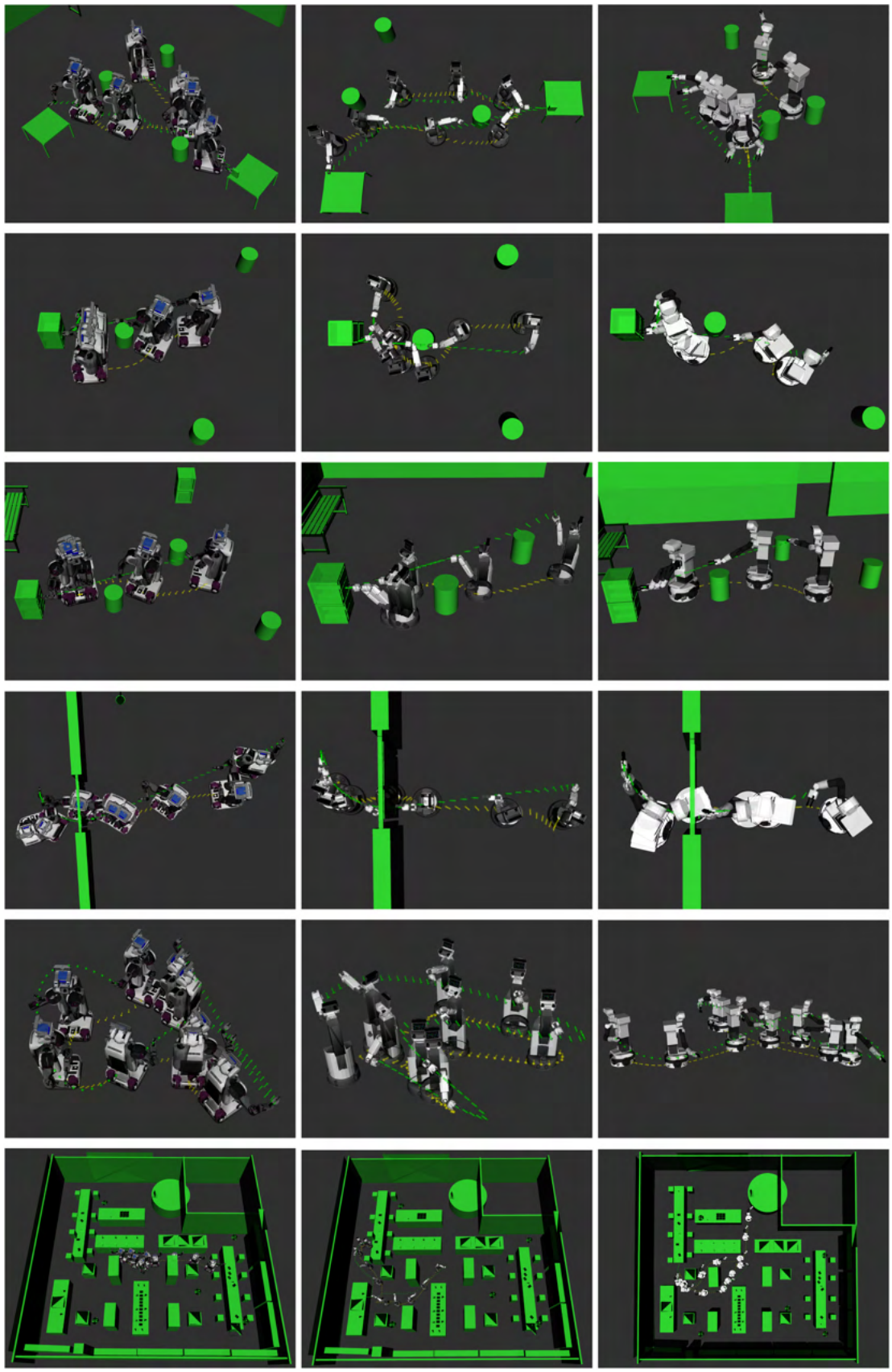

Fig. 4. Example motions produced by N^2M^2 on the different tasks and robots. The base and ee-trajectories are marked in yellow and green, respectively. From left to right: PR2, HSR, and TIAGo. From top to bottom: p&p, cabinet, drawer, door, spline, and bookstore.

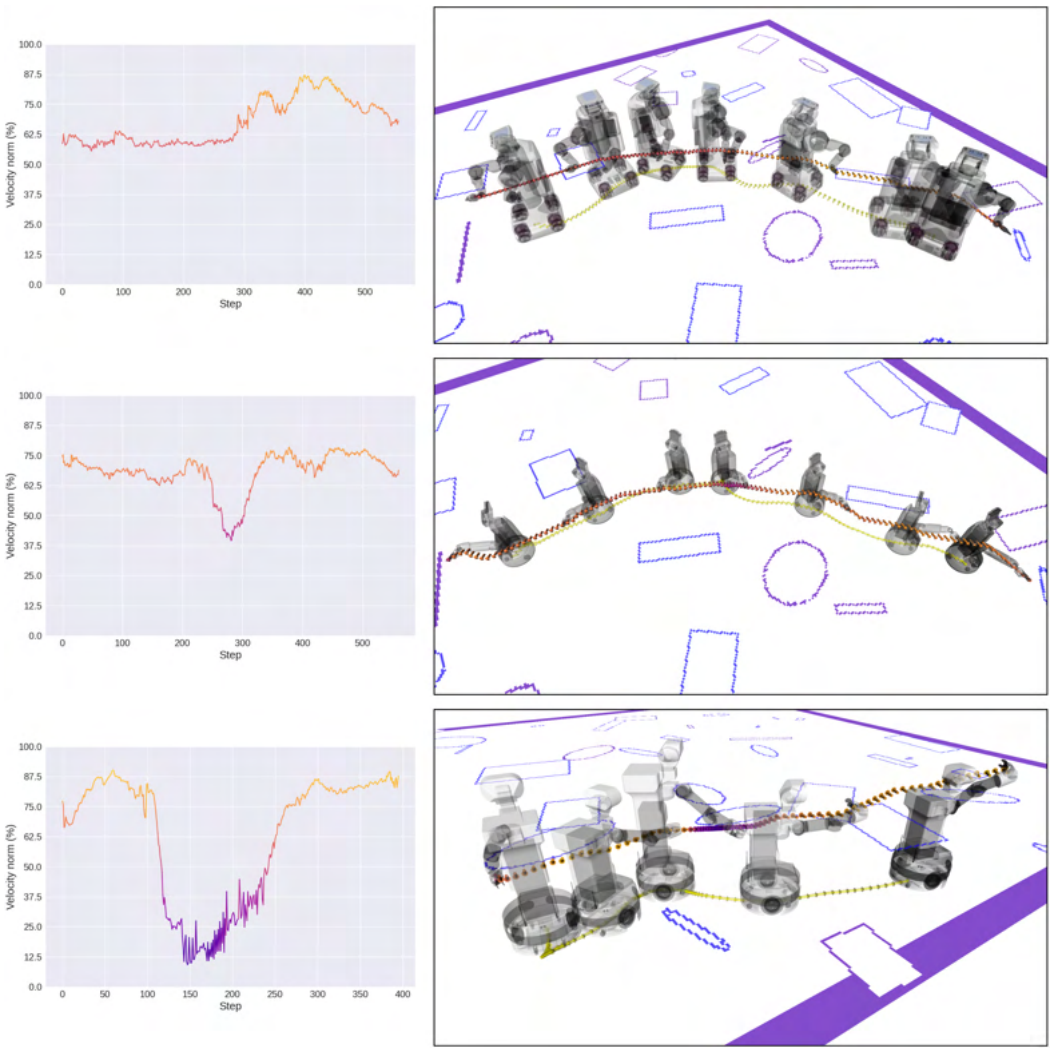

Fig. 5. Analysis of the learned norm of the end-effector velocities in the rnd obstacle task. Left: Velocity norm as a percentage of the max. allowed velocity over environment steps. Right: corresponding trajectory. Marker colors indicate the norm of the end-effector velocity in the same scale as the plots on the left. From top to bottom: PR2, HSR, and TIAGo.

TABLE IV
SUCCESS RATES FOR EXPERIMENTS IN THE ANALYTICAL ENVIRONMENT *WITHOUT* OBSTACLES

	Agent	Linear		A^*-slerp		Imitation				Spline		Avg
		rnd goal		p&p		cabinet		drawer		spline		
		success	w/jumps	success	w/jumps	success	w/jumps	success	w/jumps	success	w/jumps	success
PR2	LKF	0	80	0	46	0	86	0	64	0	42	0.0
	+ bioik, l2, jumps	38	94	8	22	36	82	22	68	0	92	20.8
	+ learn-torso	**92**	92	50	50	72	74	74	74	82	82	74.0
	+ ee-velocity	90	90	72	72	82	82	98	98	66	66	81.6
	+ collision: $\mathbf{N^2M^2}$	88	88	**100**	100	98	98	**100**	100	**72**	72	**91.6**
HSR	LKF	**72**	74	90	92	76	84	80	84	64	80	76.4
	+ bioik, l2, jumps	66	72	94	96	82	92	78	84	82	92	80.4
	+ ee-velocity	**72**	74	90	92	90	92	86	92	80	94	83.6
	+ collision: $\mathbf{N^2M^2}$	**72**	72	**98**	98	**98**	98	**92**	96	**86**	96	**89.2**
Tiago	LKF	14	50	4	76	4	80	0	72	0	4	4.4
	+ bioik, l2, jumps	48	76	42	74	36	92	44	92	20	70	38.0
	+ learn-torso	64	64	**96**	96	80	80	90	90	48	48	75.6
	+ ee-velocity	66	66	86	86	**94**	94	**92**	92	50	50	77.6
	+ collision: $\mathbf{N^2M^2}$	**76**	76	90	90	88	88	86	86	**56**	56	**79.2**

Notes: LKF denotes the previous Learning Kinematic Feasibility approach [18]. Each row adds parts of our contributions to the row above it until we arrive at our proposed model. *w/jumps* denotes success rates ignoring configuration jumps. Evaluated over a single seed.
Best performing methods are denoted in bold.

in the presence of obstacles as the base has to maneuver much more. Finally, we add the occupancy map and train in the new procedurally generated training task. With this, we arrive at our approach, which achieves the best average performance on all robots, highlighting the importance of all components. We hypothesize that this last increase stems from the fact that the new obstacle task largely increases the difficulty and variety of the motions during training. This demonstrates the importance of each individual contribution to arrive at a method that largely increases success rates and robustness over previous work.

F. Scaling to Human-Centered Maps

To further evaluate the generalization to complex objects with unseen geometries and the ability to navigate narrow human-centered environments, we evaluate the agents in a realistic bookstore environment [67] shown in Fig. 3. We define a set of 10 feasible object locations across the whole map and draw random pairs of locations to pick up an object from and place it down. The robot starts in the center of the map, again in a random start configuration. The resulting tasks are very long horizons, taking on average 1300–1500 steps at a control frequency of 10 Hz, and require passing through passages as narrow as 0.85 m. The results are reported in Table III. Our approach is the only method to successfully complete this task on all robots. While success rates are lower than in p&p, both the PR2 and HSR succeed in the majority of episodes. Success rates are somewhat lower on the TIAGo with 42.0%. Difficulties of this task are, on the one hand, the sheer length of each task. On the other hand, the robot has to repeatedly pass through very narrow passages. This is particularly challenging as the agent at the same time has to strictly adhere to the EE motions. As our focus is the learned base behavior, we do not adapt the EE orientations to the environment, even though the pick&place task does not require specific motions while carrying the object. This highlights a large potential to further improve the success rates by learning or adapting the EE motions to the specific task. We leave this for future work.

G. Dynamic Obstacles

To evaluate the reactiveness of the trained agents, we construct two tasks with dynamic obstacles. The obstacles move with a random velocity between 0.1 m and 0.15 m. We replan the EE motion at every time step in MPC-fashion, which easily runs in real time as all the complexities are shifted to the RL agent. This is again a zero-shot transfer: The agent has never seen moving obstacles during training. In the *dynamic obstacles* task, we uniformly spawn 16 dynamic obstacles in an empty room and draw a random goal for the EE, as shown in Fig. 3.

We exclude goals on the very edges of the robot's workspace where its maneuverability is very low, see "Restr. height" in Table I. The *dynamic p&p* task uses the same setup as p&p, but we replace the static obstacles with three dynamic obstacles. The results are included in Table III. We find that our agent is able to very quickly react to these obstacles. We attribute this capability to the fact that the agent has learned a robust policy that prefers to keep some distance to obstacles whenever this is possible. As a result, it tries to do the same whenever a dynamic obstacle approaches. This robustness may be a result of SACs entropy maximization and the action noise that we apply during training. Success rates for the PR2 and HSR are close to those of static obstacle tasks. The TIAGo has more difficulties: The differential drive does not allow it to easily move away from obstacles that move into the side of its base. As it has never seen dynamic obstacles, it has no incentive to anticipate these situations and to proactively place its base accordingly. Additional fine-tuning on dynamic obstacles might be able to induce it with such a sense of foresight. While the MPC approach is able to react to dynamic obstacles, its success rate drops further down to 30%–64% versus the 50.4%–88% of our approach.

H. Comparison to Planners

Planning-based methods are very general and widely used for manipulation tasks. While the specification of arbitrary motion constraints such as defined by the imitation learning motions is difficult as discussed in Section II, these approaches are a powerful baseline for a subset of our tasks: goal-reaching and pick&place in static environments. We compare with two sampling-based planners on the rnd obstacle, pick&place, and bookstore tasks. We remove the EE motion constraints and instead solely specify the task with three goal poses: 1) in front of the object, 2) the grasp pose, and 3) the location to place it down. Note that this is a significantly simpler task as the EE is completely unconstrained in-between the goals. In the second evaluation, we incorporate an additional simple pose constraint to hold the object upright after grasping it that we label as *(upright)*.

Table V compares the success rates, planning time, and the resulting length of the EE trajectory relative to our approach, which still has to fulfill all motion constraints. We find that they perform well in the obstacle and pick&place tasks, even outperforming our approach on the HSR. At the same time, success rates decrease on the PR2. This can be attributed to the larger joint space and the larger base, making the spaces narrower. In contrast, our approach performs better on the PR2 and is able to make use of its flexibility, indicating good scaling with robot complexity. Already adding the simple upright constraint decreases performance on p&p by 6ppt for the PR2. The more restricted HSR completely fails to sample successful paths under this constraint. Moving to the complex bookstore map, performance drops significantly below our approach with success rates of 22% to 42% versus 70%. While planning times are quite low on the other tasks, they now increase to almost 2 min for the PR2. Qualitatively, the planners can produce somewhat unnatural paths, such as waving the whole arm when going from the goal directly in front of the pick object to the pick object. In contrast, our approach seeks out base poses that enable good reactiveness and that are robust to noise in the base movements.

These results highlight two aspects: on the one hand, the difficulty of the bookstore map, and on the other hand, the general applicability of our approach and its beneficial scaling with the size of the configuration space: While having to follow much

TABLE V
EVALUATION OF THE PLANNER BASELINES

	Agent	rnd obstacle			p&p			bookstore p&p		
		success	planning time	rel ee path	success	planning time	rel ee path	success	planning time	rel ee path
PR2	RRTConnect	74.0	3.1 s	1.25	78.0	0.9 s	1.04	22.0	119.6 s	0.97
	RRTConnect (upright)	–	–	–	72.0	3.3 s	0.98	22.0	124.6 s	0.97
	Bi2RRT*	62.5	41.5 s	1.38	84.0	8.6 s	1.17	8.0	115.3 s	1.12
	N²M²	**86.0**	0.1 s	1.00	**97.6**	0.1 s	1.00	**70.0**	0.1 s	1.00
HSR	RRTConnect	**82.0**	0.5 s	0.83	**100.0**	1.0 s	0.99	42.0	12.2 s	0.89
	RRTConnect (upright)	–	–	–	0.0	n.d.	n.d.	0.0	n.d.	n.d.
	N²M²	63.2	0.1 s	1.00	92.0	0.1 s	1.00	**68.0**	0.1 s	1.00

Notes: Motions for the planners are unconstrained while N²M² has to follow the full EE motion. *(upright)* adds a constraint to keep the object upright after picking it up. rel ee path denotes the length of the ee-path relative to N²M². Planning time and rel ee path are only computed over successful episodes. Best performing methods are denoted in bold.

TABLE VI
SUCCESS RATES FOR EXPERIMENTS IN THE GAZEBO ENVIRONMENT

	Agent	A^*-slerp					Imitation Learning			Spline	Avg
		rnd obstacle	p&p	bookstore p&p	dynamic obstacle	dynamic p&p	cabinet	drawer	door	spline	
PR2	MPC	26.0	12.0	0.0	38.0	2.0	0.0	2.0	0.0	18.0	10.9
	E2E	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	N²M²	81.6	87.6	48.8	93.2	74.4	94.0	96.8	60.4	75.6	**79.2**
HSR	MPC	12.0	14.0	0.0	30.0	2.0	0.0	4.0	0.0	42.0	11.6
	E2E	29.6	27.2	19.6	27.2	2.8	39.2	8.4	1.2	7.2	18.0
	N²M²	64.0	92.4	54.4	91.2	78.8	95.6	90.4	85.6	91.6	**82.7**
Tiago	MPC	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
	E2E	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	N²M²	56.0	85.2	54.8	71.2	51.2	78.8	82.4	30.0	56.0	**62.8**

Notes: Evaluation of zero-shot transfer to the Gazebo physics simulator on unseen tasks from three different motions systems, an A^*-based system, an imitation system learned from human demonstrations and spline interpolation of random waypoints. The last column reports the average across all tasks. We evaluate all models on three different robotic platforms, PR2, HSR, and TIAGo. MPC is not evaluated on the TIAGo as it does not possess velocity controllers.
Best performing methods are denoted in bold.

more restrictive motion constraints, it remains competitive with unrestricted state-of-the-art methods and even outperforms them as both robot and map complexity increase. At the same time, it is directly applicable to dynamic scenes, partially observable environments, and arbitrary EE motions as demonstrated in Section IV-D.

I. Executability

To evaluate if the produced motions are readily executable, we transfer the learned policies to the Gazebo physics simulator. The agent's actions are sent to the robot's low-level base and arm controllers at an average frequency of around 30 Hz to 70 Hz. We construct an occupancy map by integrating the LiDAR and range sensor data into a binary costmap. For the dynamic obstacle tasks, the EE motions are generated based on an additional global costmap built from the robot's sensor data. The MPC baseline relies on velocity controllers for the whole robot. We use default velocity controllers of the PR2 and a pseudovelocity controller of the HSR, which underneath utilizes the position controllers. TIAGo does not provide any velocity controllers for the arm, as such we were unable to evaluate the MPC approach on this robot. While our agent can only rely on its sensors, we still provide the MPC approach with the ground-truth signed distance field to reduce complexity. For the dynamic obstacle tasks and the bookstore map, we inflate the local map passed to the RL agent by 3 c to allow the agent to more quickly react to the obstacles. For the narrow door-opening task, we scale the actions of the PR2 by a factor of 0.5.

The results for all tasks are shown in Table VI. In contrast to the analytical environment, the agents have to generalize to unseen physics and low-level controllers, asynchronous execution as well as noisy and partial occupancy maps. Furthermore, any arm collision is now recorded as a failure. Nonetheless, the performance of our approach closely matches the analytical environment with small drops of 9ppt, 0.3ppt, and 2.6ppt in average performance. Looking more closely, the differences stem almost exclusively from the door opening (PR2, TIAGo) and the bookstore (PR2) tasks. In the door opening task, we find that the imprecisions induced by the low-level controllers and asynchronous execution cause the PR2 to slightly touch the door frame in a number of episodes. As this is a very high-precision task for its large base, unseen physics and accelerations can quickly have an impact. For most episodes, success depends on only a few centimeters. Failures on the TIAGo stem largely from collisions between the elbow and the door frame. The robot commonly approaches the door with its arm in one of two configurations. In one of these configurations, the elbow is unable to avoid the door frame, leading to several failures.

As the agent does not take into account arm collisions during training, the learned policy has no means yet to avoid this failure mode. We leave full 3-D collision avoidance to future work. Similar to the door task, the drop in performance in the bookstore map for the PR2 can be attributed to a larger number of collisions in narrow pathways due to the differences in physics and accelerations compared to the training environment as well as a small number of arm collisions with bookshelves. Overall, performance remains high across the large majority of tasks. This is in contrast with the MPC approach, which is significantly impacted by these factors. The approach struggles with unseen physics and imperfect execution of motions, leading to collisions and large deviations from the desired motions on both robots. Once the error terms become too large, the approach is unable to recover good behavior, resulting in large and abrupt constraint violations.

J. Real-World Experiments

We transfer the agents to the real world and evaluate them on the HSR and PR2 robots. For the PR2, we construct an environment consisting of two rooms connected by a door in the hall it is located in. We directly evaluate the HSR in our office building, a common human-centered environment not adapted to the robot's capabilities. Maps of both environments are shown in Fig. 6. Due to their low success rates in the analytical environment and the gazebo environment, both the MPC and E2E baselines pose an increased risk of damaging the robot if executed in the real world. We, therefore, refrained from running them in our real-world experiments.

We mark goal poses for the EE with AR markers and use the robots' head cameras to detect them. If the marker is initially out of sight, the robot receives an initial guess of the target pose. To easily specify goals, we use a prerecorded map and adaptive Monte Carlo localization for localization. We also give the EE planners access to this map, in the form of a static layer in the sensed global costmap. For each episode, we move the robot into a random start pose and draw random initial joint values. We scale the agents' actions by a factor of 0.5 to ensure the safe execution of the motions. For the PR2, we also inflate the local occupancy map by 1 c for the door opening task and 3 c for all the other tasks. Results for all the tasks are reported in Table VII. The experiments are shown in Fig. 6 and in the accompanying video.

Pick&Place: We define positions for several tables, marked orange in Fig. 6, as possible pick-up and place locations, then draw random pairs of these locations for each episode. We then randomly arrange 2–4 obstacles within the map. These obstacles include a large wall segment with a curved star footprint, bins, and a chair for the PR2, and chairs and roll-containers for the HSR. In the second experiment, on top of these static obstacles, we incorporate dynamic obstacles in the form of humans and wheeled objects moved by the authors. For the HSR, we use the A^*-*fwd* planner. We also inflate the local map by 1.5 c and include the map as a static layer in the local map, as the robot does not have any sensors in the back. For the dynamic obstacles, we increase the global map inflation from 0.4 m to 0.5 m for both robots to obtain smoother trajectories for the EE motion generated by the A^*-planner.

TABLE VII
SUCCESS RATES IN THE REAL-WORLD EXPERIMENTS

	Metric	A^*		Imitation Learning			Spline	Total
		p&p static	p&p dynamic	cabinet	drawer	door	spline	
PR2	**Success**	**12**	**12**	**28**	**25**	**23**	**24**	**124**
	Base coll.	2	2	0	0	0	0	4
	Arm coll.	0	0	2	0	3	0	5
	Grasp fail	1	1	0	3	4	0	9
	IK fail	0	0	0	2	0	6	8
	Nr. episodes	15	15	30	30	30	30	150
HSR	**Success**	**11**	**12**	**23**	**24**	**24**	**16**	**110**
	Base coll.	0	2	0	0	1	0	3
	Arm coll.	1	0	0	0	1	0	2
	Grasp fail	0	0	6	3	0	0	9
	IK fail	3	1	1	2	4	4	15
	Safety stop	0	0	0	1	0	0	1
	Nr. episodes	15	15	30	30	30	20	140

Notes: The PR2 is evaluated in an environment consisting of two rooms connected by a door. The HSR is directly evaluated in our offices. Each cell denotes the number of episodes. The last column sums up all tasks.
Best performing methods are denoted in bold.

Both robots achieve high success rates of 76.6%–80% for both the static and dynamic obstacles. They demonstrate well-planned movements around obstacles and behavior such as backing out of confined starting poses if necessary. When confronted with dynamic obstacles, they react quickly and are able to evade obstacles that move directly into their base without breaking the EE motion. Second, the learned behavior is robust to frequently changing EE motions as the A^*-planner adapts to dynamic changes in the shortest path.

Spline: For the spline interpolation, we draw waypoints from within a single room of the environment. Both robots are able to follow these motions with success rates of 80%. The main source of failures is too large deviations from the desired EE motion when having to follow motions at very large heights (PR2) or unusual orientations (HSR). As an additional difficulty, the HSRs gripper frequently moves so low as to be detected as an obstacle by its base LiDAR. However, the learned policy proved robust to this disturbance.

Cabinet and drawer: For the imitation system motions, we construct tasks with the same objects as in the simulation. As the imitation learning system does not incorporate obstacle avoidance into the EE motions, we restrict obstacles to objects with a low height, such that the motions can pass over them and choose start poses in the same room as the target object. We rearrange two obstacles every 3 to 5 episodes, deliberately placing at least one of them such that it constrains the opening motion. The PR2 achieves very high success rates of 93.3% and 83.3% on the cabinet and drawer tasks, respectively. The HSR succeeds in 76.6% and 80% of all the episodes. Its main failure source is the precision of the grasp, grasping slightly in front of the handle, thereby not opening the object. In one episode, the HSR joints hit a safety limit while being in contact with the drawer which led the controllers to be stopped by the software.

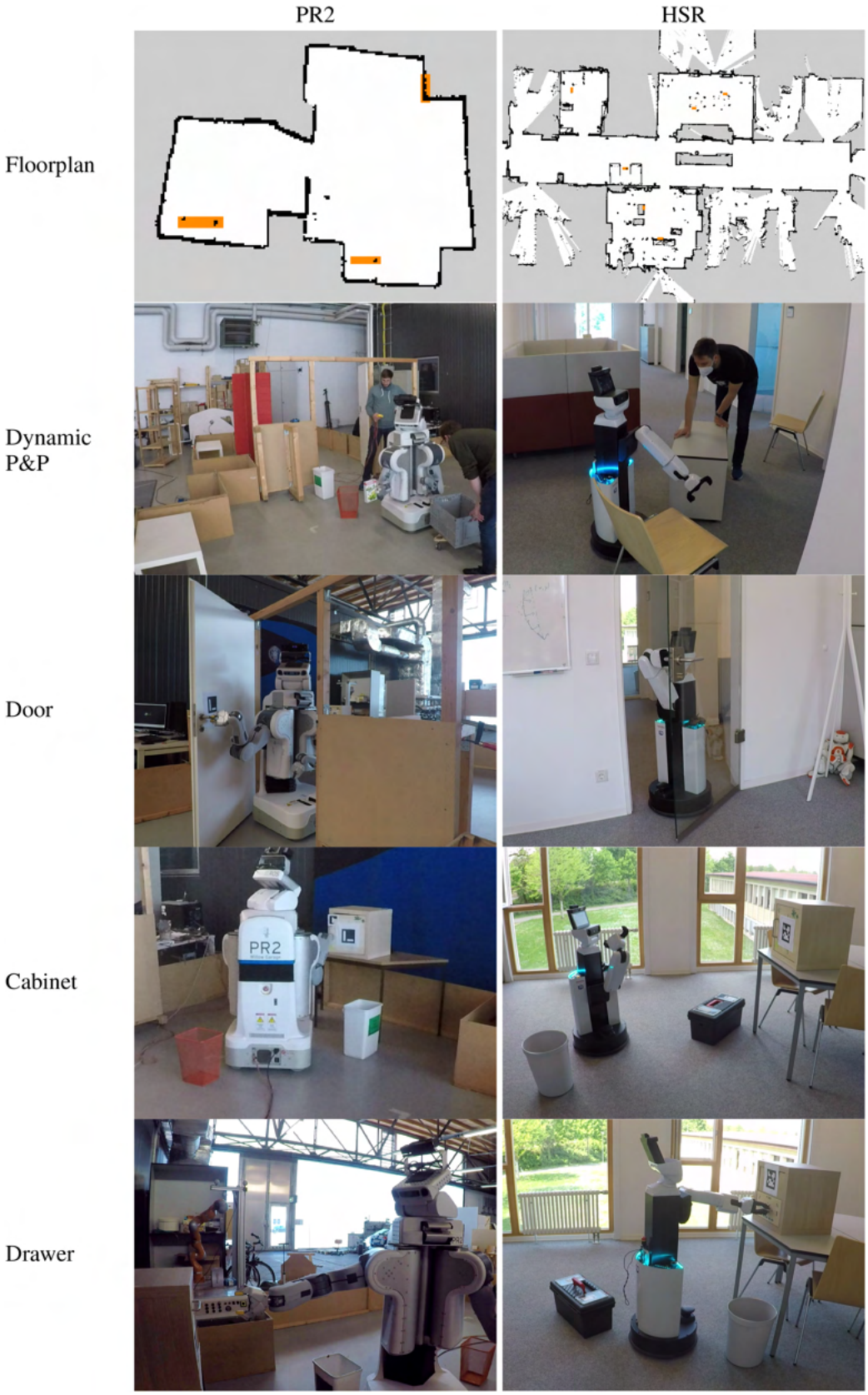

Fig. 6. Real world experiments on the PR2 (left) and HSR (right) robots. From top to bottom: Environment map, pick&place while avoiding static and dynamic obstacles (table locations marked in orange), opening and driving through a door, opening a cabinet, and opening a drawer with static obstacles constraining the base.

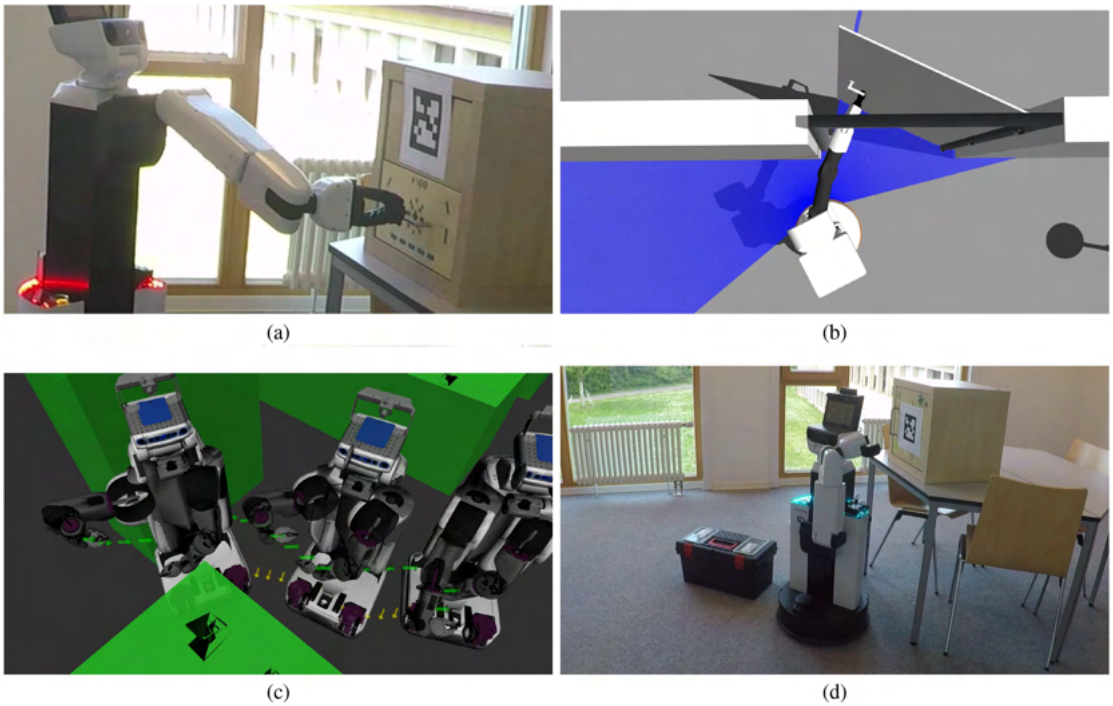

Fig. 7. Examples of failure cases. (a) Independence between base and arm controllers together with noise in localization and actuation's can impact the end-effector precision, which in some cases leads to grasping failures. (b) Collision avoidance focuses on the end-effector and the robot base but does not take into account collisions of the robot arm. Here, the TIAGo's wrist collides with the door frame. (c) Heuristics to generate pick&place motions can produce overly restrictive motions. In the situation shown here, the interpolation results in a desired end-effector pose that is oriented in the opposite of the driving direction (pointing to the right while the path to the goal is further to the left). This leads the base to collide with an obstacle at the left while trying to adhere to this orientation. (d) Agent does not consider head configurations outside of the training distribution. As the HSRs head turns to refocus the cabinet, this leads to a self-collision with the arm.

Door: For the PR2, we use the same door and motions as in the simulation. For the HSR, we use a door within our office, with an even smaller frame width of 0.83 m. As the door's opening radius differs, the learned imitation motions do not directly apply to it. Instead, we sample eight waypoints and interpolate them with the spline interpolator to construct an opening motion for the EE. As its arm is not strong enough to press down the handle, we do not lock the door's spring for the HSR. The PR2 succeeds in 76.6% of all episodes and performs better than in Gazebo. In general, the real hardware seems more reactive than in the simulator. The HSR achieves success rates of 80%. Failure cases for the PR2 include collisions of the elbow with the door itself, slipping off the handle, and not being able to unlock the door lock. The HSR in a few cases maneuvers to the right of the handle, such that it is not able to pass through the door frame without moving the noncontinuous arm roll joints into another configuration (which would require violating the motion constraints). In these situations, it prefers to stay in place until it terminates because the deviations to the desired motion become too large, rather than to produce unsafe behavior such as collisions. The HSR agent proved robust to weird and irregular occupancy maps produced by the glass door.

Across all the tasks, we observe that the learned behaviors directly transfer to the real world with average success rates of 82.6% and 78.6% on the PR2 and HSR, respectively. The agents learned to efficiently avoid obstacles and cause very few base collisions. They rather fail the arm motions than drive into an obstacle. The produced motions are robust to noise such as localization error, which could be significant whenever the number of unmapped obstacles is large. Although the overall motions are smooth, at higher speeds, the arm motions can sometimes become a bit shaky. This has several reasons: the low-level base and arm controllers are independent and as such cannot take into account each other's errors. Second, the hardware introduces several additional noise sources that are not present in the simulation. These include asynchronous control, localization errors, calibration errors, unseen obstacle geometries, and artifacts in the occupancy maps. We found that the acceleration regularization introduced in Section III is essential for smoothness. Finetuning on the real system is a promising avenue to further increase the precision and the feasible velocities on the real systems. The HSRs main failure source is the IK failures. A number of these occurred due to conflicts between the head and arm. Unlike in training in the real world, the head moved to focus the camera on the target object, thereby leading to different self-collision constraints.

K. Limitations

While the approach achieves high success rates across all tasks and environments, we identify a number of failure cases and limitations to address in the future. Table VII provides a quantitative decomposition of the failures in the real world. Fig. 7 shows examples of failure cases.

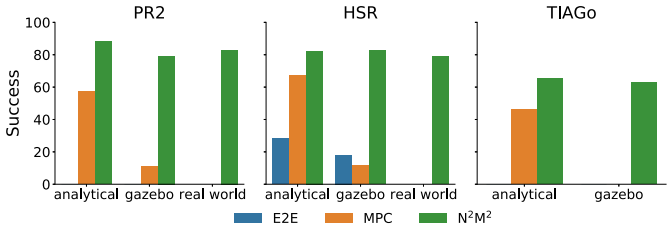

Fig. 8. Average success rates across different robots and environments. Given the low success rates in the analytical environment and the simulated gazebo environment leading to an increased risk of damaging the robot, we did not evaluate the MPC and E2E baselines in the real world.

High-precision navigation: While the approach is able to consistently navigate real-world maps, centimeter-exact navigation such as in the narrow door frame and passages of the bookstore map can still be a challenge. We identify two causes for this: First, as we perform a zero-shot transfer to the real world, the agent is unaware of noise in components such as the low-level controllers or localization. Finetuning in the real world should enable the agent to better take these uncertainties into account. Second, the resolution of the obstacle map is 2.5 cm, which leaves the agent to perceive up to 1.25 cm of free space on each side as occupied. For high-precision tasks, the agent may benefit from further increasing this resolution.

Independent controllers: We rely on out-of-the-box low-level controllers of the different robotic platforms. At the moment neither of these offers whole-body controllers. Instead, we use the independent base and arm controllers. These cannot take into account each other's error terms, sometimes leading to deviations from the desired EE pose or grasp failures, e.g., when the base rotates while the arm is moving [see Fig. 7(a)]. The development of joint low-level controllers is a promising avenue to further stabilize and increase the precision of the EE motions.

Arm collisions: We present an efficient and well-generalizing obstacle avoidance in which the EE motion generator is responsible for collision-free motions for the EE while the RL agent avoids base collisions. However, in certain scenarios, it is still possible for the arm to collide with objects in the environment, as the RL agent does not take into account 3-D obstacle avoidance [see Fig. 7(b)]. We plan to investigate full collision avoidance for the arm using RGB-D data or voxel maps in future work.

Overly restrictive EE motions: Certain tasks such as pick&place do not require the robot to follow exact EE positions or orientations for large parts of the task. The motions we use in this work use effective heuristics and interpolation. While this is simple and efficient for many cases, it can lead to overly difficult EE motions for example in the narrow passages of the bookstore map [see Fig. 7(c)]. In the future, this may be addressed by developing methods that allow the agent to deviate further from the EE motions where unproblematic or by learning EE motions.

Head collisions: In the real world, we control the robots' head cameras to focus on the target objects. This moves the head into unseen configurations that the agent is unaware of. In the case of the HSR, this causes a number of self-collisions with the arm [see Fig. 7(d)]. Conditioning on the full robot state and randomizing the head configurations during training is a promising avenue to alleviate this problem.

L. Generality and Applicability

Solvable tasks: Our method is, in principle, able to provide kinematically feasible navigation for any task for which a motion generator for the EE is available. This applies to a wide variety of mobile manipulation tasks, as we demonstrate a large range of such tasks in our experiments. This modularity enables us to reuse knowledge and generalize to unseen tasks, in contrast to previous learning-based methods that have to retrain from scratch on each new task. However, certain tasks can currently not be solved in this manner, for example, tasks that require specific joint configurations, such as pushing open a door with the elbow joint.

Robotic platforms: The approach is directly applicable to any mobile robotic system that can be decomposed into base and manipulator. In our experiments, we demonstrate this on holonomic and nonholonomic wheeled drives, without any platform-specific hyperparameter tuning. But the same formulation could also be applied to different morphologies such as quadrupeds. The main requirements are that the manipulator is not needed in the navigation motions and the availability of a low-level controller interface to translate velocity or position commands from the agent to the actuators. We provide pretrained agents for the PR2, HSR, and the TIAGo on the project website.[3] We also provide the methods to train agents for other robots, requiring only an implementation of the corresponding environment.

V. Conclusion

We introduced N²M², which extends the formulation of kinematic feasibility for mobile manipulation to complex unstructured environments. We generalized its objective function and extended the agent's control to the velocity of the EE motions and prevent configuration jumps by learning the torso joints and introducing a regularization to the IK. We then introduced a procedurally generated training environment that uses strong randomization and simple elements to produce diverse scenarios. The result is a powerful approach that can successfully act in unseen human-centered real-world environments. In extensive experiments across a variety of robots, physics, and environments, we demonstrated that this approach successfully generalizes in a zero-shot manner to novel tasks, unseen objects and geometries, and dynamic obstacles. By leveraging a hybrid combination of IK and RL, the agent solves tasks with a vast continuous configuration space in which previous state-of-the-art approaches struggle. Our method outperforms these approaches both in the analytical environment and in the transfer to unseen environments on all robotic platforms. Fig. 8 summarizes the success rates across the different robots and environments.

In this work, we have focused on achieving arbitrary motions and used simple robot-agnostic EE motion generators. We purposefully abstracted from optimizing the motions or goals themselves to demonstrate the system's capabilities to achieve challenging motions. In the future, we plan to jointly or iteratively optimize the robot's behaviors and the generated EE motions. A particularly exciting direction is to incorporate this work into hierarchical approaches that learn to produce motions

[3] [Online]. Available: https://mobile-rl.cs.uni-freiburg.de

or subgoals for the EE to achieve high-level goals. Such an approach will benefit from the ability to abstract from complex base behavior to reason in a much simpler space of EE motions. This can be done both within a learning-based paradigm or on the motion planning level of TAMP-based pipelines.

Further work includes the incorporation of partially observable and 3-D obstacle avoidance for the robot arm. The flexibility of the RL approach means that this can be incorporated based on voxel maps or directly learned from camera inputs. The development of joint low-level controllers and finetuning of the learned policies in the real world are promising avenues for further improvements in precision and velocities. Finally, we find that the current RL methods still face difficulties to explore certain high-dimensional continuous action spaces, as exhibited by the TIAGo robot. Methods to alleviate this problem will be important for robotics. A particularly interesting direction for mobile manipulation is the combination of value learning methods with efficient Monte Carlo rollouts, combining the best of MPC and learning-based approaches.

ACKNOWLEDGMENT

The authors would like to thank Adrian Röfer and Niklas Wetzel for their feedback on the initial draft of the article.

REFERENCES

[1] K. Blomqvist et al., "Go fetch: Mobile manipulation in unstructured environments," 2020, *arXiv:2004.00899*.

[2] J. V. Hurtado, L. Londoño, and A. Valada, "From learning to relearning: A framework for diminishing bias in social robot navigation," *Front. Robot. AI*, vol. 8, 2021, Art. no. 650325.

[3] K. Sirohi, R. Mohan, D. Büscher, W. Burgard, and A. Valada, "EfficientLPS: Efficient LiDAR panoptic segmentation," *IEEE Trans. Robot.*, vol. 38, no. 3, pp. 1894–1914, Jun. 2022.

[4] F. R. Valverde, J. V. Hurtado, and A. Valada, "There is more than meets the eye: Self-supervised multi-object detection and tracking with sound by distilling multimodal knowledge," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 11612–11621.

[5] J. Duan, S. Yu, H. L. Tan, H. Zhu, and C. Tan, "A survey of embodied AI: From simulators to research tasks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 2, pp. 230–244, Apr. 2022.

[6] A. Younes, D. Honerkamp, T. Welschehold, and A. Valada, "Catch me if you hear me: Audio-visual navigation in complex unmapped environments with moving sounds," *IEEE Robot. Automat. Lett.*, vol. 8, no. 2, pp. 928–935, Feb. 2023.

[7] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *Proc. Int. Conf. Robot. Automat.*, 2013, pp. 1970–1975.

[8] F. Paus, P. Kaiser, N. Vahrenkamp, and T. Asfour, "A combined approach for robot placement and coverage path planning for mobile manipulation," in *Proc. Int. Conf. Intell. Robots Syst.*, 2017, pp. 6285–6292.

[9] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011.

[10] J. Kuffner and S. LaValle, "RRT-Connect: An efficient approach to single-query path planning," in *Proc. Int. Conf. Robot. Automat.*, 2000, vol. 2, pp. 995–1001.

[11] F. Burget, M. Bennewitz, and W. Burgard, "BI2RRT*: An efficient sampling-based path planning framework for task-constrained mobile manipulation," in *Proc. Int. Conf. Intell. Robots Syst.*, 2016, pp. 3714–3721.

[12] J. Pankert and M. Hutter, "Perceptive model predictive control for continuous mobile manipulation," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 6177–6184, Oct. 2020.

[13] M. V. Minniti, R. Grandia, K. Fäh, F. Farshidian, and M. Hutter, "Model predictive robot-environment interaction control for mobile manipulation tasks," in *Proc. Int. Conf. Robot. Automat.*, 2021, pp. 1651–1657.

[14] F. Xia, C. Li, R. Martín-Martín, O. Litany, A. Toshev, and S. Savarese, "ReLMoGen: Leveraging motion generation in reinforcement learning for mobile manipulation," in *Proc. Int. Conf. Robot. Automat.*, 2021, pp. 4583–4590.

[15] J. Wong et al., "Error-aware imitation learning from teleoperation data for mobile manipulation," in *Proc. Conf. Robot Learn.*, 2022, vol. 164, pp. 1367–1378.

[16] F. Schmalstieg, D. Honerkamp, T. Welschehold, and A. Valada, "Learning long-horizon robot exploration strategies for multi-object search in continuous action spaces," in *Proc. Int. Symp. Robot. Res.*, 2022, pp. 52–66.

[17] J. Kindle, F. Furrer, T. Novkovic, J. J. Chung, R. Siegwart, and J. Nieto, "Whole-body control of a mobile manipulator using end-to-end reinforcement learning," 2020, *arXiv:2003.02637*.

[18] D. Honerkamp, T. Welschehold, and A. Valada, "Learning kinematic feasibility for mobile manipulation through deep reinforcement learning," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 6289–6296, Oct. 2021.

[19] R. Diankov, S. S. Srinivasa, D. Ferguson, and J. Kuffner, "Manipulation planning with caging grasps," in *Proc. IEEE Int. Conf. Humanoid Robots*, 2008, pp. 285–292.

[20] M. Arduengo, C. Torras, and L. Sentis, "Robust and adaptive door operation with a mobile robot," *Intell. Serv. Robot.*, vol. 14, no. 3, pp. 409–425, 2021.

[21] J. Liu et al., "Garbage collection and sorting with a mobile manipulator using deep learning and whole-body control," in *Proc. Int. Conf. Humanoid Robots*, 2021, pp. 408–414.

[22] S. Jauhri, J. Peters, and G. Chalvatzaki, "Robot learning of mobile manipulation with reachability behavior priors," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 8399–8406, 2022.

[23] F. Burget, A. Hornung, and M. Bennewitz, "Whole-body motion planning for manipulation of articulated objects," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1656–1662.

[24] O. Arslan and P. Tsiotras, "Use of relaxation methods in sampling-based algorithms for optimal motion planning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 2421–2428.

[25] M. Otte and E. Frazzoli, "RRTx: Real-time motion planning/replanning for environments with unpredictable obstacles," in *Algorithmic Foundations of Robotics XI*. Berlin, Germany: Springer, 2015, pp. 461–478.

[26] D. Berenson, S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *Int. J. Robot. Res.*, vol. 30, no. 12, pp. 1435–1460, 2011.

[27] D. Leidner, A. Dietrich, F. Schmidt, C. Borst, and A. Albu-Schäffer, "Object-centered hybrid reasoning for whole-body mobile manipulation," in *Proc. Int. Conf. Robot. Automat.*, 2014, pp. 1828–1835.

[28] T. Welschehold, C. Dornhege, F. Paus, T. Asfour, and W. Burgard, "Coupling mobile base and end-effector motion in task space," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–9.

[29] J. Sleiman, F. Farshidian, M. V. Minniti, and M. Hutter, "A unified MPC framework for whole-body dynamic locomotion and manipulation," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4688–4695, Jul. 2021.

[30] M. Mittal et al., "Articulated object interaction in unknown scenes with whole-body mobile manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022.

[31] J. Haviland, N. Sunderhauf, and P. Corke, "A holistic approach to reactive mobile manipulation," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 3122–3129, Apr. 2022.

[32] R. Ancona, "Redundancy modelling and resolution for robotic mobile manipulators: A general approach," *Adv. Robot.*, vol. 31, no. 13, pp. 706–715, 2017.

[33] C. R. Garrett et al., "Integrated task and motion planning," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 4, pp. 265–293, 2021.

[34] M. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum, "Differentiable physics and stable modes for tool-use and manipulation planning," in *Proc. Robot.: Sci. Syst.*, 2018.

[35] C. R. Garrett, T. Lozano-Perez, and L. P. Kaelbling, "FFRob: Leveraging symbolic planning for efficient task and motion planning," *Int. J. Robot. Res.*, vol. 37, no. 1, pp. 104–136, 2018.

[36] S. Bansal, V. Tolani, S. Gupta, J. Malik, and C. Tomlin, "Combining optimal control and learning for visual navigation in novel environments," in *Proc. Conf. Robot Learn.*, 2020, vol. 100, pp. 420–429.

[37] T. El-Gaaly, C. Tomaszewski, A. Valada, P. Velagapudi, B. Kannan, and P. Scerri, "Visual obstacle avoidance for autonomous watercraft using smartphones," in *Proc. Auton. Robots Multirobot Syst. Workshop*, 2013.

[38] M. Kollmitz, D. Büscher, and W. Burgard, "Predicting obstacle footprints from 2D occupancy maps by learning from physical interactions," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 10256–10262.
[39] R. Guldenring, M. Görner, N. Hendrich, N. J. Jacobsen, and J. Zhang, "Learning local planners for human-aware navigation in indoor environments," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2020, pp. 6053–6060.
[40] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter, "Learning a state representation and navigation in cluttered and dynamic environments," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5081–5088, Jul. 2021.
[41] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2018, pp. 3052–3059.
[42] U. Patel, N. Kumar, A. J. Sathyamoorthy, and D. Manocha, "DWA-RL: Dynamically feasible deep reinforcement learning policy for robot navigation in dense mobile crowds," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 6057–6063.
[43] J. Wu et al., "Spatial action maps for mobile manipulation," in *Proc. Robot.: Sci. Syst.*, 2020.
[44] T. Chen, S. Gupta, and A. Gupta, "Learning exploration policies for navigation," in *Proc. Int. Conf. Learn. Representations*, 2019.
[45] Y. Goel et al., "Predicting dense and context-aware cost maps for semantic robot navigation," in *Proc. IROS PNARUDE (Percep. Navigation Auton. Robot. Unstructured Dyn. Environments) Workshop*, 2022.
[46] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean signed distance fields for on-board MAV planning," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2017, pp. 1366–1373.
[47] L. Han, F. Gao, B. Zhou, and S. Shen, "FIESTA: Fast incremental Euclidean distance fields for online motion planning of aerial robots," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2019, pp. 4423–4430.
[48] K. Ehsani et al., "ManipulaTHOR: A framework for visual object manipulation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 4497–4506.
[49] D. Batra et al., "Rearrangement: A challenge for embodied AI," 2020, *arXiv:2011.01975*.
[50] S. Srivastava et al., "BEHAVIOR: Benchmark for everyday household activities in virtual, interactive, and ecological environments," in *Proc. Conf. Robot Learn.*, 2022, pp. 477–490.
[51] A. Szot et al., "Habitat 2.0: Training home assistants to rearrange their habitat," *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 251–266, 2021.
[52] C. Gan et al., "The ThreeDWorld transport challenge: A visually guided task-and-motion planning benchmark for physically realistic embodied AI," in *Proc. Int. Conf. Robot. Automat.*, Philadelphia, PA, USA, 2022, pp. 8847–8854, doi: 10.1109/ICRA46639.2022.9812329.
[53] L. P. Kaelbling, "Learning to achieve goals," in *Proc. 13th Int. Joint Conf. Artif. Intell.*, 1993, pp. 1094–1098.
[54] T. Schaul, D. Horgan, K. Gregor, and D. Silver, "Universal value function approximators," in *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 37, 2015, pp. 1312–1320.
[55] R. S. Sutton, D. Precup, and S. Singh, "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning," *Artif. Intell.*, vol. 112, no. 1/2, pp. 181–211, 1999.
[56] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *Proc. Nat. Conf. Artif. Intell.*, 2017.
[57] L. P. Kaelbling, "Hierarchical learning in stochastic domains: Preliminary results," in *Proc. Int. Conf. Mach. Learn.*, 1993, vol. 951, pp. 167–173.
[58] C. Li, F. Xia, R. Martin, and S. Savarese, "HRL4IN: Hierarchical reinforcement learning for interactive navigation with mobile manipulators," in *Proc. Conf. Robot Learn.*, 2019, pp. 603–616.
[59] A. Röfer, G. Bartels, W. Burgard, A. Valada, and M. Beetz, "Kineverse: A symbolic articulation model framework for model-agnostic mobile manipulation," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 3372–3379, Apr. 2022.
[60] P. Ruppel, N. Hendrich, S. Starke, and J. Zhang, "Cost functions to specify full-body motion and multi-goal manipulation tasks," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 3152–3159.
[61] F. Pardo, A. Tavakoli, V. Levdik, and P. Kormushev, "Time limits in reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, vol. 80, pp. 4045–4054.
[62] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, vol. 80, pp. 1861–1870.
[63] P. Klink, C. D'Eramo, J. R. Peters, and J. Pajarinen, "Self-paced deep reinforcement learning," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 9216–9227, 2020.
[64] M. Dennis et al., "Emergent complexity and zero-shot transfer via unsupervised environment design," in *Proc. Conf. Neural Inf. Process. Syst.*, 2020, pp. 13049–13061.
[65] D. Coleman, I. Sucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: A MoveIt! case study," *J. Softw. Eng. Robot.*, vol. 5, no. 1, pp. 3–16, 2014.
[66] T. Welschehold, C. Dornhege, and W. Burgard, "Learning mobile manipulation actions from human demonstrations," in *Proc. Int. Conf. Intell. Robots Syst.*, 2017, pp. 3196–3201.
[67] AWS RoboMaker, "AWS-RoboMaker-bookstore-world," 2019. [Online]. Available: https://github.com/aws-robotics/aws-robomaker-bookstore-world

Daniel Honerkamp received the M.S. degree in computational statistics and machine learning from University College London, London, U.K., in 2018. He is currently working toward the Ph.D. degree in computer science with Robot Learning Lab headed by Abhinav Valada, Freiburg im Breisgau, Germany.

His research interests include reinforcement learning and autonomous decision making for mobile manipulation and embodied agents.

Tim Welschehold (Member, IEEE) received the Diploma degree in physics and the Ph.D. degree in computer science from the University of Freiburg, Freiburg, Germany, in 2013 and 2020, respectively.

He is currently a Postdoctoral Researcher with Autonomous Intelligent Systems group headed by Wolfram Burgard. He is a member of the BrainLinks-BrainTools Center. His research revolves around reinforcement learning, imitation learning, and representation learning in robotics with a focus on mobile manipulation and dynamical systems.

Abhinav Valada (Member, IEEE) received the M.S. degree in robotics from Carnegie Mellon University, Pittsburgh, PA, USA, in 2013, and the Ph.D. degree in computer science from the University of Freiburg, Freiburg, Germany, in 2019.

He is currently an Assistant Professor and the Director of Robot Learning Lab, University of Freiburg. He is a member of the Department of Computer Science, a Principal Investigator with BrainLinks-BrainTools Center, and a core faculty in the European Laboratory for Learning and Intelligent Systems (ELLIS) unit, Freiburg. His research lies at the intersection of robotics, machine learning, and computer vision with a focus on tackling fundamental robot perception, state estimation, and control problems using learning approaches in order to enable robots to reliably operate in complex and diverse domains.