Safe and Effective Teleoperation for High Degree-of-Freedom Robots using MPC

Federico Pizarro Bejarano[1], Lukas Brunke[1,2], and Angela P. Schoellig[1,2]

***Abstract*—Teleoperation is becoming increasingly important not only for robotic deployment, but also for collecting demonstration data for imitation learning and foundation-model-style robot training. However, for high degree-of-freedom robots such as mobile manipulators, whole-body teleoperation remains cumbersome: commanding every degree of freedom directly is slow and cognitively demanding, while many existing interfaces rely on specialized hardware, motion tracking, or robot-specific setups. We propose a partial-intent predictive teleoperation framework in which the user specifies only partial intent, such as end-effector motion, base motion, or a subset of joint velocities, and the remaining degrees of freedom are completed autonomously. Our teleoperation framework combines an intent-tracking objective with autonomous completion costs, enabling a single optimization-based approach to support different teleoperation modes without redesign. We focus on mobile manipulation as the primary application domain and use it to illustrate how our framework supports safe and flexible teleoperation for high degree-of-freedom robots.**

I. Introduction

Teleoperation remains an essential tool for deploying robots in unstructured environments, especially when human judgment is needed for task execution, supervision, or recovery. It is also becoming increasingly important as a mechanism for collecting demonstration data for imitation learning and large-scale robot learning pipelines. For static manipulators, effective teleoperation interfaces are now widely available, but mobile manipulators remain substantially harder to operate because the user must coordinate a large number of degrees of freedom across the base, arm, and perception stack [1].

Whole-body teleoperation of high-degree-of-freedom (DoF) robots is often slow and cognitively demanding, and many existing solutions reduce that burden only by introducing additional hardware, motion-tracking setups, or robot-specific interfaces, which makes data collection expensive and harder to scale [1]. Honerkamp et al. [1] similarly argue that low-cost, low-friction teleoperation is crucial for both practical use and high-quality demonstration collection.

This motivates a broader view of teleoperation: the user specifies only the part of the behavior that expresses intent, while the robot completes the remaining motion autonomously. For example, if a user commands the base of a mobile manipulator, the arm may need to stabilize a carried object [2] or maintain camera viewpoint; if the user commands end-effector motion, the base may need to reposition to preserve reachability, manipulability, or collision avoidance. In these scenarios, the non-teleoperated degrees of freedom have their own objectives and constraints.

Our teleoperation framework is inspired by model predictive safety filters, which minimally modify commanded inputs to enforce constraints [3], [4]. However, classical safety filters assume the policy commands the full actuation space. Here, the operator commands only a subset of the available inputs, and the optimization must jointly decide how to interpret, track, and complete those commands over a horizon while satisfying objectives and constraints on the non-controlled DoFs. This changes safe teleoperation from pure command correction to intent-aware motion completion.

The main contributions of this paper are threefold. First, we present a general partial-intent predictive teleoperation framework based on masked tracking of user-selected input dimensions. Second, we extend the multi-step tracking perspective of model predictive safety filters [4] to partial-input teleoperation, including decaying horizon weights that prioritize near-term intent and reduce chattering. Third, we show how the same framework can support multiple teleoperation modes for high-DoF robots, particularly mobile manipulators, without retraining or redesign.

II. Related Work

A. Safety Filters and Constraint Enforcement

Safety filters provide a principled way to enforce constraints around an arbitrary controller by modifying commanded inputs only when necessary [5], [6]. Control barrier functions (CBFs) offer forward-invariance guarantees for safe sets [7], but can become sensitive near active constraints and may lead to nonsmooth interventions or chattering [8]. Model predictive safety filters (MPSFs) instead solve a finite-horizon optimization problem to certify that a safe backup plan exists [3], and recent multi-step variants improve smoothness by optimizing intent tracking over several future steps [4].

Our work borrows this multi-step tracking viewpoint; however, in classical safety filters, the external policy proposes the full actuation vector and the filter modifies that command to preserve safety. In our setting, the operator specifies only part of the actuation space, and the remaining inputs are autonomous decision variables by design. The controller therefore does not merely filter a complete action; it completes a partial command into a feasible full-system motion.

[1]The authors are with the Learning Systems and Robotics Lab (www.learnsyslab.org), University of Toronto, Canada, and affiliated with the University of Toronto Robotics Institute and the Vector Institute for Artificial Intelligence in Toronto. {federico.pizarrobejarano, lukas.brunke, angela.schoellig}@robotics.utias.utoronto.ca

[2]Lukas Brunke and Angela P. Schoellig are also with the Technical University of Munich and the Munich Institute for Robotics and Machine Intelligence (MIRMI).

B. Shared Control and Teleoperation

Shared-control methods combine human input with autonomous decision making, typically to improve safety, usability, or task success. One common approach is *blended control*, in which human and autonomous inputs are fused, for example through weighted averaging or an MPC objective that arbitrates between a human and a separate autonomous actor [9]. Another family of methods allocates authority between the human and robot through variable-autonomy mechanisms such as EMICS and HierEMICS [10], [11].

More closely related to our setting are recent teleoperation methods for mobile manipulation that reduce operator burden by assigning only task-relevant dimensions to the human. Honerkamp et al. [1] propose MoMa-Teleop, in which the user controls end-effector motion while a pretrained reinforcement learning policy controls the robot base to realize whole-body behavior with standard low-cost interfaces. This is the closest conceptual predecessor to our work: it validates partial-intent teleoperation as a useful paradigm for high-DoF robots. However, the autonomy module is learned, tied to a specific decomposition, and does not expose objectives and constraints explicitly or allow for re-tuning parameters without retraining.

These approaches provide useful design patterns, but they usually emphasize either arbitration between agents or a specific learned decomposition. In contrast, our teleoperation framework treats the teleoperation interface as a selection of controlled dimensions: the user commands a subset of the robot's DoFs, while the remaining DoFs are controlled autonomously according to constraints and secondary objectives. This perspective is especially natural for high-DoF systems such as mobile manipulators, where different interfaces may command the base, the end effector, or both.

III. Problem Setup

We consider a discrete-time dynamical system

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \tag{1}$$

where $\mathbf{x}_k \in \mathbb{X} \subset \mathbb{R}^n$ is the state and $\mathbf{u}_k \in \mathbb{U} \subset \mathbb{R}^m$ is the full control input. The robot is subject to state and input constraints

$$\mathbf{x}_k \in \mathbb{X}_c, \qquad \mathbf{u}_k \in \mathbb{U}_c. \tag{2}$$

At time step k, the teleoperator does not command the full input. Instead, the interface provides a partial command $\mathbf{u}_k^{\text{tele}} \in \mathbb{R}^m$ together with a binary mask $\mathbf{M}_k \in \{0,1\}^{m\times m}$ that selects the input dimensions actively controlled by the user. The matrix $\mathbf{M}_k$ is diagonal, with ones on teleoperated dimensions and zeros elsewhere. This representation allows the teleoperation mode to change online, for example when switching from base to end-effector teleoperation.

The commanded dimensions encode user intent and should be tracked as closely as possible. The remaining dimensions must be chosen autonomously to satisfy additional objectives such as maintaining grasp stability, avoiding collisions, preserving manipulability, or improving viewpoint. We therefore separate the control objective into three parts:

1) tracking user intent on the masked input dimensions,
2) maintaining feasibility and safety through state and input constraints, and
3) optimizing autonomous objectives on the non-teleoperated degrees of freedom.

Our teleoperation framework deliberately avoids fixing a specific robot decomposition. The commanded variables can correspond to base velocities, end-effector twists, individual joints, or any other selected control channels. As a result, the same optimization structure applies across multiple teleoperation interfaces and robot embodiments.

IV. MPC for Partial-Intent Teleoperation

Given the current state $\mathbf{x}_k$ and the current user command $\mathbf{u}_k^{\text{tele}}$, we first construct a hypothetical future teleoperation trajectory over the MPC horizon. Following [4], we track not only the current input but also an inferred sequence of future inputs, typically weighted more strongly at near-term steps:

$$\hat{\mathbf{u}}_{0|k}^{\text{tele}} = \mathbf{u}_k^{\text{tele}}, \tag{3}$$

$$\hat{\mathbf{u}}_{i|k}^{\text{tele}} = \Psi(\mathcal{I}_k, i), \quad i = 1, \dots, N-1, \tag{4}$$

$$Q_i \succeq 0, \quad i = 0, \dots, N-1, \tag{5}$$

where $\mathcal{I}_k$ denotes the information available for intent inference, such as the recent input history, the active teleoperation mode, task context, or a simple extrapolation of the current command, and Ψ denotes the mechanism used to predict the user's future inputs. The matrices Q_i weight the relative importance of tracking the inferred intent at different steps in the horizon. In practice, a common choice is to set $Q_i = \gamma^i Q$ for $i = 0, \dots, N-1$ with $\gamma \in (0, 1]$. This prioritizes near-term agreement with the user more strongly than long-horizon agreement, which can reduce chattering and improve responsiveness while still allowing the controller to plan ahead for feasibility and smoothness.

Using this predicted trajectory, we solve the following MPC optimization. The first optimized input $\mathbf{u}_{0|k}^\star$ is applied in receding-horizon fashion.

$$\min_{\mathbf{x}_{i|k}, \mathbf{u}_{i|k}} \sum_{i=0}^{N-1} \Big(\|\mathbf{M}_k(\mathbf{u}_{i|k} - \hat{\mathbf{u}}_{i|k}^{\text{tele}})\|_{Q_i}^2 \tag{6a}$$

$$+ \ell_{\text{auto}}(\mathbf{x}_{i|k}, \mathbf{u}_{i|k}, \mathbf{M}_k) \Big) + V_f(\mathbf{x}_{N|k}, \mathbf{M}_k) \tag{6b}$$

$$\text{s.t. } \mathbf{x}_{0|k} = \mathbf{x}_k, \tag{6c}$$

$$\mathbf{x}_{i+1|k} = \mathbf{f}(\mathbf{x}_{i|k}, \mathbf{u}_{i|k}), \quad i = 0, \dots, N-1, \tag{6d}$$

$$\mathbf{x}_{i|k} \in \mathbb{X}_c, \quad \mathbf{u}_{i|k} \in \mathbb{U}_c, \quad i = 0, \dots, N-1, \tag{6e}$$

$$\mathbf{x}_{N|k} \in \mathbb{X}_f. \tag{6f}$$

Here, $\mathbf{x}_{i|k}$ and $\mathbf{u}_{i|k}$ denote the predicted state and input at step i of the horizon, initialized from the current state $\mathbf{x}_k$, N is the horizon length, $\mathbf{M}_k$ is the diagonal mask selecting the teleoperated input dimensions, $\ell_{\text{auto}}(\dots)$ is the autonomous stage cost on the non-teleoperated behavior, $V_f(\dots)$ is the terminal cost, and $\mathbb{X}_f$ denotes the terminal set.

The masked tracking term is the central modeling choice. Because $\mathbf{M}_k$ zeros out non-teleoperated dimensions, the optimizer is penalized only for deviating from the inferred user-intent trajectory on the selected channels. The unmasked

dimensions are then determined by the autonomous objective and the constraints. This preserves the semantics of the teleoperation interface: the controller respects the dimensions actually commanded by the user while still planning proactively through the inferred future intent trajectory.

The autonomous objective ℓ_{auto} is task dependent. For a mobile manipulator, it may include terms that stabilize an object, maintain camera viewpoint, avoid singular configurations, regularize base motion, or keep the end effector close to a nominal posture. Importantly, these objectives act only through the dimensions that are not currently teleoperated or through coupled system dynamics; they do not overwrite the commanded intent channels.

The formulation also naturally supports mode switching. When the teleoperation interface changes, only the mask $\mathbf{M}_k$ needs to be updated. The underlying optimizer, constraints, and autonomous costs remain unchanged. This modularity is one of the practical advantages of phrasing teleoperation as masked MPC rather than as a separate controller for each control mode.

V. Evaluation Setup

We evaluate our proposed framework and the RL baseline in a controlled mobile-manipulation goal-reaching benchmark. The evaluation is designed to test whether our framework preserves commanded end-effector intent while completing the remaining whole-body motion under the same task conditions used by the RL baseline.

A. Proposed Controller Setup

Our approach is structured as a whole-body nonlinear MPC built with CasADi and `acados` [12]. The experiments are conducted on an omnidirectional Clearpath Ridgeback base with a 6-DoF UR10 arm. We model the robot with $n_q = 9$ generalized coordinates, comprising planar base pose $(x_{\text{base}}, y_{\text{base}}, \theta_{\text{base}})$ and six arm joint positions $\theta_{1:6}$, and $n_v = 9$ generalized velocities, comprising base planar velocities $(\dot{x}_{\text{base}}, \dot{y}_{\text{base}}, \dot{\theta}_{\text{base}})$ and six arm joint velocities $\dot{\theta}_{1:6}$. The controller state is $\mathbf{x} = [\mathbf{q}; \mathbf{v}] \in \mathbb{R}^{18}$ and the internal MPC input is an acceleration command $\mathbf{u} \in \mathbb{R}^9$. In this mobile-manipulation instantiation, the teleoperation command is a generalized velocity command, denoted by $\mathbf{v}_k^{\text{tele}} \in \mathbb{R}^9$. Although the MPC optimizes generalized accelerations internally, the executable command sent to the robot is obtained by interpolating the predicted velocity trajectory $\mathbf{v}^\star_{0:N-1|k}$; the acceleration input is only an internal optimization variable. At each MPC update, the optimizer predicts full state and input trajectories over a horizon of $N = 10$ steps with $dt = 0.1$ s.

The teleoperation command can specify any subset of the generalized velocity variables. Various methods can be used to predict the teleoperation command over the horizon, denoted by $\hat{\mathbf{v}}^{\text{tele}}_{i|k}$ for $i = 0, \dots, N-1$ [4]. In practice, we found that holding the teleoperation command constant across the prediction horizon (i.e., $\hat{\mathbf{v}}^{\text{tele}}_{i|k} = \mathbf{v}^{\text{tele}}_k$) worked well.

The stage cost ℓ_{auto} regularizes generalized velocities and accelerations for non-teleoperated joints. The teleoperation-tracking objective is weighted significantly higher so that following the commanded velocity remains the primary objective. Additional costs can be added, such as pose tracking for non-teleoperated joints. Safety is enforced through state and input box constraints together with signed-distance collision constraints using a safety margin of 0.10 m for both self-collision and static obstacles. State constraints and collision constraints are softened with shared linear and quadratic slack penalties for feasibility, whereas input constraints remain hard.

B. RL Baseline Setup

We use a goal-conditioned RL baseline inspired by N^2M^2/MoMa-Teleop [1], [13]. The task is goal reaching in free space, without obstacle-avoidance to isolate completion behavior: at the start of each episode, a 6D goal pose is sampled, and a deterministic end-effector planner generates a desired twist by linearly interpolating position and using spherical linear interpolation for orientation. The learned policy then outputs 3D normalized base control $\mathbf{a}_t = [a_x, a_y, a_\psi] \in [-1, 1]^3$, mapped to physical base velocities with limits $v_x, v_y \in [-0.4, 0.4]$ m/s and $\omega_z \in [-0.75, 0.75]$ rad/s.

The policy is trained with Soft Actor-Critic (SAC) [14] in a custom Gymnasium/PyBullet environment. The observation follows the same semantic blocks as [13] and includes the commanded twist, current/desired/goal end-effector poses, robot generalized positions, and previous action.

The reward emphasizes kinematic feasibility and smoothness through an IK tracking term together with action-magnitude and action-change regularization. We measure position error for the end-effector as $d_p = \|\mathbf{p}_{\text{curr}} - \mathbf{p}_{\text{goal}}\|_2$ and orientation error as $d_r = 1 - |\langle q_{\text{curr}}, q_{\text{goal}} \rangle|^2$. Episodes terminate successfully when the end-effector reaches the goal pose within fixed position and orientation tolerances, or unsuccessfully at a maximum horizon of $T_{\max} = 1000$ steps; early failure is triggered by cumulative violations of the desired planner pose. Training uses MLP actor/critic networks, replay-based off-policy learning, automatic entropy tuning, and 5×10^5 environment steps.

C. Benchmark and Metrics

For head-to-head comparison, the RL baseline and the proposed framework are evaluated on the same simulator, robot model, deterministic end-effector command generator, goal sampler, command limits, success thresholds, and episode horizon. Each episode starts from the robot home configuration and samples a goal pose from a fixed distribution. Goal position is sampled uniformly in $x, y \in [-1.0, 1.0]$ and $z \in [0.7, 1.0]$ while goal orientation is sampled by drawing a random axis and a rotation angle in $[0, \pi]$. For each comparison, both methods are evaluated on identical sampled goals, yielding paired rollouts under the same conditions.

An episode is successful if both

$$d_p = \|\mathbf{p}_{ee} - \mathbf{p}_g\|_2 \leq 0.1 \text{ m}, \tag{7}$$

$$d_r = 1 - |\langle q_{ee}, q_g \rangle|^2 \leq 0.05, \tag{8}$$

and otherwise terminates at $T_{\max} = 1000$ simulation steps. We evaluate each controller over 100 episodes using the same goal set and shared success criteria.

TABLE I: Head-to-head goal-reaching evaluation over 100 episodes with identical sampled goals for RL and the proposed framework. Values are mean $\pm$ standard deviation, with min/max shown for the final error metrics.

Metric	RL	Proposed
Success rate (%)	97.0	97.0
Final position error (m)	0.129 ± 0.341	0.098 ± 0.047
	$(0.021,\ 3.503)$	$(0.032,\ 0.416)$
Final orientation error	0.037 ± 0.089	0.028 ± 0.016
	$(0.000,\ 0.903)$	$(0.000,\ 0.050)$
Episode length (steps)	340.5 ± 134.6	340.6 ± 136.8
Arm control effort	62.6 ± 82.4	16.7 ± 14.2
Base path length (m)	1.35 ± 0.84	1.64 ± 0.72
Full-command jerkiness	0.681 ± 0.277	0.074 ± 0.069
Wall time (s)	1.82 ± 0.66	3.40 ± 1.34

From Table I, we observe that both methods achieved the same success rate (97%); however, our framework produced tighter final-error distributions, lower arm effort, and markedly lower full-command jerkiness than the RL baseline. The RL baseline also exhibited much larger worst-case position and orientation errors, whereas our framework remained more consistent across episodes. Although the RL baseline is faster in wall-clock runtime, our framework provides better robustness and control quality at similar task success.

Beyond these quantitative metrics, our framework offers practical advantages for teleoperation. Safety constraints, collision margins, and task trade-offs are specified explicitly in the optimization problem rather than being absorbed into a learned policy or indirectly encoded through reward shaping, making our framework more interpretable, easier to tune, and easier to debug. It also does not require retraining when robot limits, costs, teleoperation modes, or command profiles change; adapting the behavior amounts to modifying masks, constraints, or objective weights. In our additional experiments, policies trained with one simulated teleoperator degraded substantially when the command profile became faster or jerkier, whereas our framework remained stable.

D. Behavioral Flexibility Showcase

In addition to the quantitative goal-reaching benchmark, we include three qualitative scenarios that illustrate the flexibility of our framework. Across all three scenarios, the robot model, solver stack, and safety-constrained whole-body optimization remain unchanged; only the task-level configuration, including masks, references, and scene settings, is modified.

1) Case A - End-effector teleoperation with autonomous base collision avoidance: In this scenario, the simulated teleoperator commands the end-effector forward, while low-height static obstacles are placed in the way of the robot. Our framework prioritizes end-effector tracking and leaves base motion to autonomous completion under constraints. As a result, the base deviates from a naive straight-line route, avoiding collisions while preserving the commanded end-effector behavior as closely as feasible.

2) Case B - Base teleoperation with autonomous arm collision avoidance: In the second scenario, teleoperation is used to control the base, while the arm is adjusted autonomously. The base moves forward, forcing the arm to move to avoid colliding with obstacles. This demonstrates the complementary allocation of autonomy relative to Case A: the same framework now treats base motion as the externally prioritized teleoperation and autonomously moves the arm to satisfy constraints.

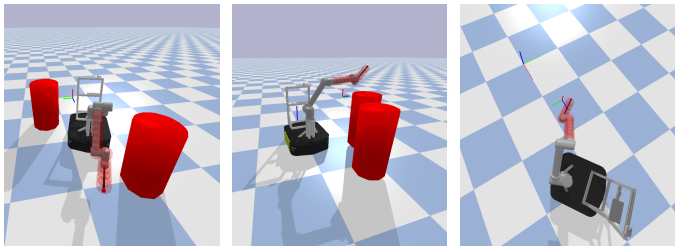

Fig. 1: Qualitative flexibility showcase of the proposed framework under three tasks. **Left:** end-effector teleoperation with autonomous base collision avoidance. **Middle:** base teleoperation with autonomous arm collision avoidance. **Right:** circular base motion with end-effector stabilization at a fixed target. The same framework is used in all cases; only the task-level masks, references, and scene settings are changed.

3) Case C - Circular base motion with end-effector stabilization: In the third scenario, the base is teleoperated along a circular path, while the end effector has an additional objective to remain near a fixed position. This demonstrates the simultaneous execution of a base teleoperation tracking with a secondary end-effector stabilization goal, without changing the controller architecture.

VI. Conclusion

We presented a partial-intent predictive teleoperation framework for high-DoF robots in which the user commands only a subset of the available control dimensions and the remaining motion is completed autonomously through a safety-constrained predictive optimization. This formulation unifies intent tracking, autonomous completion, and constraint satisfaction in a single framework, and supports different teleoperation modes without retraining or redesign. In mobile-manipulation experiments, our framework matched the success rate of a learned baseline while producing smoother and more consistent whole-body behavior.

VII. Future Work

An important next step is to complete the experimental validation on the physical robot. This includes transferring the experiments already presented in simulation to hardware, as well as evaluating in settings that more directly reflect the motivating applications for shared autonomy and data collection. First, we plan to deploy our framework on tray-carrying mobile manipulation tasks [2], where autonomous completion can be used to stabilize carried objects while preserving intuitive user control of the base or end effector. Second, we plan to conduct a user study on the physical platform to measure workload, ease of use, task completion, and safety relative to more direct or learned teleoperation alternatives. Third, we plan to test whether demonstrations collected with our framework on the real robot yield higher-quality data for imitation learning by comparing downstream policy performance, consistency, and robustness against data collected through more conventional teleoperation pipelines.

REFERENCES

[1] D. Honerkamp, H. Mahesheka, J. O. von Hartz, T. Welschehold, and A. Valada, "Whole-body teleoperation for mobile manipulation at zero added cost," *IEEE Robotics and Automation Letters*, 2025.

[2] A. Heins and A. P. Schoellig, "Keep it upright: Model predictive control for nonprehensile object transportation with obstacle avoidance on a mobile manipulator," *IEEE Robotics and Automation Letters*, 2023.

[3] K. P. Wabersich and M. N. Zeilinger, "Linear model predictive safety certification for learning-based control," in *IEEE Conference on Decision and Control*, 2018.

[4] F. Pizarro Bejarano, L. Brunke, and A. P. Schoellig, "Multi-step model predictive safety filters: Reducing chattering by increasing the prediction horizon," in *IEEE Conference on Decision and Control*, 2023.

[5] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Transactions on Automatic Control*, 2019.

[6] K. P. Wabersich, A. J. Taylor, J. J. Choi, K. Sreenath, C. J. Tomlin, A. D. Ames, and M. N. Zeilinger, "Data-driven safety filters: Hamilton-jacobi reachability, control barrier functions, and predictive methods for uncertain systems," *IEEE Control Systems Magazine*, 2023.

[7] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *European Control Conference*, 2019.

[8] L. Brunke, S. Zhou, and A. P. Schoellig, "Preventing inactive cbf safety filters caused by invalid relative degree assumptions," *IEEE Transactions on Automatic Control*, 2025.

[9] E. Jabbour, M. Vulliez, C. Preault, and V. Padois, "A model predictive control approach to blending in shared control," *IEEE International Conference on Robotics and Automation*, 2024.

[10] M. Chiou, N. Hawes, and R. Stolkin, "Mixed-initiative variable autonomy for remotely operated mobile robots," *ACM Transactions on Human-Robot Interaction*, 2021.

[11] D. Panagopoulos, G. Petousakis, A. Ramesh, T. Ruan, G. Nikolaou, R. Stolkin, and M. Chiou, "A hierarchical variable autonomy mixed-initiative framework for human-robot teaming in mobile robotics," in *IEEE International Conference on Human-Machine Systems (ICHMS)*, 2022.

[12] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. van Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, and M. Diehl, "acados: a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, 2021.

[13] D. Honerkamp, T. Welschehold, and A. Valada, "N^2m^2: Learning navigation for arbitrary mobile manipulation motions in unseen and dynamic environments," *IEEE Transactions on Robotics*, 2023.

[14] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv:1801.01290*, 2018.