IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Perceptive Model Predictive Control for Continuous Mobile Manipulation

Johannes Pankert*, Marco Hutter*

***Abstract*— A mobile robot needs to be aware of its environment to interact with it safely. We propose a receding horizon control scheme for mobile manipulators that tracks task space reference trajectories. It uses visual information to avoid obstacles and haptic sensing to control interaction forces. Additional constraints for mechanical stability and joint limits are met. The proposed method is faster than state of the art sampling based planners, available as opensource and can be implemented on a broad class of robots. We validate the method both in simulation and through extensive hardware experiments with a multitude of mobile manipulation platforms. The resulting software package is released with this paper.**

I. Introduction

Due to the aging workforce and more stringent safety regulations, there is an increasing need for automation in fields like building construction and industrial inspection [1], [2]. While companies are starting to deploy robots for monitoring and measuring [3], there are few reported cases where commercial mobile robots are used to perform manipulation tasks. Such robots have to be mobile since the workspaces of fixed based manipulators are too restrictive. There are solutions for discrete tasks such as drilling or welding, where it is sufficient for a robot to move to a target location and then perform the manipulation task [4], [5]. In this work, we want to enable robots to perform continuous processes such as cleaning, spray-painting or grinding that exceed the workspace boundaries of a stationary robotic arm. Construction sites and industrial plants are challenging environments since they are unstructured with both static and dynamic obstacles. Avoiding obstacles is an essential requirement for the deployment of a robot in those environments. A popular solution to motion planning with obstacle avoidance is using sampling-based planners. They can provide globally optimal solutions [6] but have high computational costs. Our method only generates locally optimal motion plans, but it is fast and reactive to changing or unknown environments.
The targeted applications can be grouped into two classes: For tasks like spray-painting or visual inspection, the robot does not need to touch the environment. In contrast, cleaning, grinding, or troweling plaster requires the robot to be in contact with its environment and control the interaction force. Since the tools are heavy and the required interaction forces can be high, the modern lightweight robots with torque-controllable joints that are popular in the research community are not strong enough to perform the tasks. Instead, we propose an admittance control approach that can be used on any robotic manipulator, including industrial-sized arms that can support the required payloads.

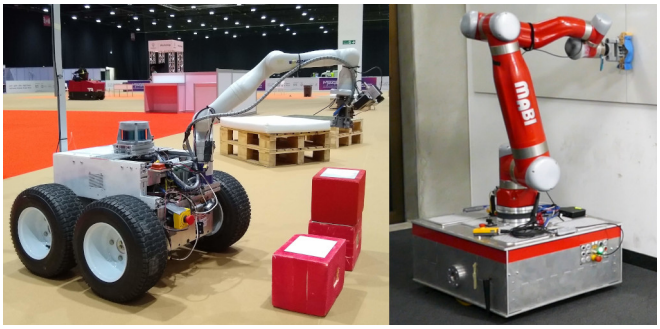


Fig. 1: Two mobile robots performing manipulation tasks.
Left: *Swaco* picking up a brick for the *MBZIRC2020* robotics competition.
Right: *MabiMobile* cleaning a whiteboard by controlling the interaction force.

We present a model predictive control strategy (MPC) that has an awareness of its environment and can plan whole-body motion for a mobile manipulator while avoiding collisions. Through task-space admittance control, it can track desired interaction forces and torques.

A. Related Work

1) Sequential Linear Quadratic Model Predictive Control: In this work, we use Sequential Linear Quadratic Model Predictive Control (SLQ-MPC), a flavor of non-linear MPC. The algorithm has initially been proposed in a discrete-time formulation [7]. Later, a continuous-time formulation was proposed along with the open-source implementation *OCS2* that we use in this work [8]. The first work on task space control of a mobile manipulator with SLQ used equality constraints to enforce end-effector tracking [9]. Modeling the tracking task as a quadratic cost instead of a constraint greatly improves the robustness against disturbances [5]. In this work, we take inspiration from *Grandia et al.* and introduce soft inequality constraints with Relaxed Barrier Functions (RBF) to the MPC cost [10].

2) Signed Distance Fields for Collision Avoidance: Signed distance fields are a popular choice in representing a robot's environment. They have successfully been employed in trajectory optimization for collision avoidance [11], [12]. In those works, signed distance fields were created offline from existing mesh models. With *Voxblox*, a framework was presented that can generate Euclidian Signed Distance Fields (ESDF) incrementally from depth sensor data [13].

*Robotic Systems Lab, ETH Zuerich {pankertj, hutter}@ethz.ch

This research was supported by the Swiss Federal Railways (SBB) and the Swiss National Science Foundation through the National Centre of Competence in Digital Fabrication (NCCR dfab).

The new ESDF-based mapping framework *FIESTA* has been presented recently and the authors report faster online map generation compared to *Voxblox*[14]. In this work, we build on *Voxblox* and show how we can shorten distance and gradient query times by caching gradients for each voxel. The same technique can also be used with *FIESTA* or other ESDF mapping implementations.

3) Model Predictive Admittance Control: Controlling the interaction wrench of contact by displacement of the robot's end-effector, admittance control, has been studied for many years [15]. In recent years, different groups developed model predictive admittance control strategies [16], [17]. Those works showed promising results both in simulation and in hardware experiments for fixed based robots. For mobile robots, interaction force control results have been presented using various techniques such as task-space impedance control [18] and dynamic model predictive control [19]. Those approaches require torque-controllable actuators that are not available in most high-payload robots.

B. Contributions

We propose a fast motion planning framework for mobile manipulators that can run as a model predictive controller. The framework has been tested extensively on different mobile robots for a large variety of applications and has been published as open-source software[1].

To our best knowledge, this is the first demonstration of collision avoidance in SLQ. We compare our method to state of the art sampling based planners *RRTX*[20] and *BiFMT*[21] and show that it converges faster and produces shorter whole-body paths. We show that by caching gradients of an ESDF, queries to the map are fast enough to use them in an MPC. The caching is lightweight and runs along with the online mapper. We furthermore show that by using an admittance scheme, we can control interaction-forces of a mobile robot without torque-controllable joints while planning future motion and respecting joint and stability constraints.

II. Model Predictive Control

A model predictive control (MPC) module generates control inputs for the robot to follow an end-effector trajectory while respecting several constraints. In the following sections, we describe the SLQ algorithm, the system model used, the cost-function, and our soft constraints mechanism.

A. Sequential Linear Quadratic Model Predictive Control

The SLQ algorithm performs forward roll-outs of the current control policy over a prediction horizon with the full non-linear system dynamics. The system dynamics are then linearized around the obtained state and input trajectories. Quadratic approximations of the cost functions are computed. With those approximations, we solve the algebraic or differential Riccati equations to obtain the next affine control policy. The affine policies consist of time-varying state feedback laws and input feed-forward terms.

[1] https://github.com/leggedrobotics/perceptive_mpc

B. System Model

The type of robot we are targeting with our approach consists of a mobile base with a manipulator with n degrees of freedom (DOF). The base can turn in place but has a non-holonomic constraint and cannot drive sideways. These properties are typical for differential drive, skid steer, or tracked robots. A kinematic model is used in the MPC. Joint positions describe the state of the arm $\boldsymbol{x_{arm}} \in \Re^n$. In contrast to prior work [9], [5], the full base pose $\boldsymbol{x}_{base} \in SE(3)$ is used in the system model. A quaternion $\boldsymbol{q}_{base}$ encodes the base orientation and a $\Re^3$ vector $\boldsymbol{r}_{base}$ its position in a static world frame.

Using the full base pose instead of a reduced state $[x, y, \varphi_{yaw}]$ allows the robot to negotiate uneven terrain and generalizes our work for any type of mobile base. The arm is controlled with joint velocity references $\boldsymbol{u}_{arm} = [\dot{\varphi}_1, \ldots, \dot{\varphi}_n]$. The base's wheel speeds are calculated from the desired base twist, the forward velocity, and the turning rate $\boldsymbol{u}_{base} = [v, \dot{\varphi}_{base}]$.

These considerations lead to the following system model:

$$\boldsymbol{x} = [\boldsymbol{x_{base}}, \boldsymbol{x_{arm}}]^T \tag{1}$$

$$\boldsymbol{u} = [v, \dot{\varphi}_{base}, \dot{\varphi}_1, \ldots, \dot{\varphi}_n]^T \tag{2}$$

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{q_{base}} \boxdot 0.5k\dot{\varphi}_{base} \\ \boldsymbol{q_{base}} \cdot [v, 0, 0]^T \\ \boldsymbol{u_{arm}} \end{bmatrix} \tag{3}$$

"$\boxdot$" denotes the Hamilton quaternion product, "i, j, k" the quaternion imaginary units and "$\cdot$" a vector rotation by a quaternion.

C. Cost Function

The cost function integrates the end-effector tracking error $C_{ee\_tracking}$, the constraint barrier functions B_i and the weighted control effort $\boldsymbol{u}^T \boldsymbol{R} \boldsymbol{u}$ over the time horizon T:

$$J(\boldsymbol{x}, \hat{\boldsymbol{x}}, \boldsymbol{u}) = \int_{\tau=t_0}^{t_0+T} L(\boldsymbol{x}(\tau), \hat{\boldsymbol{x}}(\tau), \boldsymbol{u}(\tau)) d\tau \tag{4}$$

$$L(\boldsymbol{x}, \hat{\boldsymbol{x}}, \boldsymbol{u}) = C_{ee\_tracking} + \sum B_i + \boldsymbol{u}^T \boldsymbol{R} \boldsymbol{u} \tag{5}$$

In the following sections we will describe how we formulate cost terms for end-effector tracking, collision avoidance and compliance to joint and task-space constraints.

1) Task space tracking: Deviations of the robot's end-effector pose $\boldsymbol{\xi_{ee}}$ to a desired pose $\hat{\boldsymbol{\xi}}_{\boldsymbol{ee}}$ are penalized. The desired pose may be time-dependent and changes over the controller's time horizon. For the translational error $\boldsymbol{e_{pos}}$, we use the difference between the two position vectors. To compute the rotational error, we use an orientation error formulation on the orientation quaternions of the current and the desired pose [22]:

$$\begin{aligned} \boldsymbol{e_O}(\boldsymbol{q}, \hat{\boldsymbol{q}}) = & q_w \cdot [\hat{q}_x \ \hat{q}_y \ \hat{q}_z]^T - \hat{q}_w \cdot [q_x \ q_y \ q_z]^T \\ & + [\hat{q}_x \ \hat{q}_y \ \hat{q}_z]^T \times [q_x \ q_y \ q_z]^T \end{aligned} \tag{6}$$

The robot's end-effector pose is a function of the current state. We use Robcogen [23] to compute the forward kinematics from the robot's base to the end-effector and multiply

with the world to base transform from $\boldsymbol{x_{base}}$.
The sum of squares of these error functions form the end-effector tracking term of our cost function:

$$C_{ee\_tracking}(t, \boldsymbol{x}) = \|\boldsymbol{e_{pos}}(t, \boldsymbol{x})\|_2^2 + \|\boldsymbol{e_O}(t, \boldsymbol{x})\|_2^2 \quad (7)$$

For the SLQ algorithm to converge to a stable control law, the second derivative of the cost function with respect to the state must be positive semi-definite. Since the robot's forward kinematics are generally not positive semi-definite, we approximate the second derivatives of the cost term in a Gauss-Newton fashion with $\ddot{C}_{ee\_tracking} \approx 2\dot{\boldsymbol{e}}_{\boldsymbol{pos}}\dot{\boldsymbol{e}}_{\boldsymbol{pos}}^T + 2\dot{\boldsymbol{e}}_{\boldsymbol{o}}\dot{\boldsymbol{e}}_{\boldsymbol{o}}^T$

D. Soft Constraints

While tracking the end-effector target, the generated motion plans must additionally respect constraints. The SLQ algorithm has no notion of hard inequality constraints, but we can create cost-functions that penalize violations of constraints $z_i = g_i(\boldsymbol{x}, \boldsymbol{u}, t) \geq 0$ with Relaxed Barrier Functions (RBF) [24]:

$$B_i(z_i) = \begin{cases} -\mu ln(z_i) & z_i > \delta \\ \mu\beta(z_i; \delta) & z_i <= \delta \end{cases} \quad (8)$$

β is the quadratic function for which $B \in \mathcal{C}^2$.The parameters μ and δ can be used as tuning parameters for each individual constraint but we achieved good results with $\mu = 5 \cdot 10^{-3}$ and $\delta = 10^{-4}$ for all constraints in this work.
To ensure positive semi-definiteness of the overall cost-function we approximate the second derivatives of the cost terms with a first order approximation of g:

$$\ddot{(B_i(g))} \approx \dot{g}\ddot{B}_i(g)\dot{g}^T \quad (9)$$

A set of linear constraints restricts the allowed joint positions as well as the velocity commands for the robot's base and the arm. Additionally, we use the constraint mechanism for collison avoidance and to ensure meachanical stability as described in the following sections.

E. Collision Avoidance

We use the RBF soft constraint mechanism to avoid collisions of the robot with obstacles. We define a set of collision spheres to approximate the robot. At the center of those spheres, we query the distance to the closest obstacle from a Euclidean Signed Distance Field (ESDF). We generate the ESDF online with the mapping framework *Voxblox* [13]. In the standard implementation, *Voxblox* interpolates distances of neighboring 8 voxel centers at query time. Jacobians are then computed with the finite differences of the interpolated values.
We modified the framework to cache gradients along with the signed distances in each voxel. We can then evaluate the distances at each point assuming a local linear model:

$$d(\boldsymbol{r}) = d_k + \dot{d}_k(\boldsymbol{r} - \boldsymbol{r}_k) \quad (10)$$

where k denotes the index of the voxel containing the query point and $\boldsymbol{r}$ the cartesian coordinate of the query point.
The distance function is piece-wise differentiable. For continuous gradients, we also investigated caching of second-order derivatives and used a quadratic approximation to compute $d(\mathrm{x})$. The collision constraint functions are:

$$g_i(\boldsymbol{x}) = d(FK_i^{world}(\boldsymbol{x})) - r_i \quad (11)$$

with the forward kinematics FK_i^{world} to the center of the collision sphere i and r_i the radius of the sphere.
We compute the forward kinematics for the entire robot along the kinematic chain and reuse intermediate results between different collision checkpoints. Adding more collision checkpoints does therefore not introduce a significant overhead.

F. Task Space Admittance Control

Tracking a desired end-effector pose while maintaining a desired contact wrench is not possible in general since both objectives conflict. To harmonize both tasks, we propose a task space admittance control scheme that enables the user to balance the two objectives.
Let $\hat{\Xi} = [\hat{\xi}^0, \ldots, \hat{\xi}^m]$ with $\hat{\xi}^i = [\hat{\boldsymbol{r}}^i, \hat{\boldsymbol{q}}^i]$ be a desired end-effector path in task space. We displace the path by an admittance term $\delta\xi = [\delta\boldsymbol{r}, \delta\boldsymbol{q}]$ $with$ $\delta\boldsymbol{r}, \delta\boldsymbol{q} \in \Re^3$ that depends on the current and past wrench tracking errors $\delta\boldsymbol{w} = \hat{\boldsymbol{w}} - \boldsymbol{w}$:

$$[\delta\boldsymbol{r}_i, \delta\boldsymbol{q}_i]^T = K_p\delta\boldsymbol{w} + K_i \int \delta\boldsymbol{w}dt \quad (12)$$
$$\text{foreach } \hat{\xi}^i \text{ in } \hat{\Xi} : \tilde{\xi}^i = \hat{\xi}^i \boxplus \delta\xi$$

$\boldsymbol{w} = [\boldsymbol{f}_{ee}, \boldsymbol{\tau}_{ee}]$ is the measured end-effector wrench, $\hat{\boldsymbol{w}}$ the desired wrench. The gain matrices K_p and K_i define the properties of our admittance controller. A non-zero entry in K_p causes a spring-like behavior in the respective spatial direction. The integrator gains in K_i allow for stationary force tracking. We introduce anti-windup limits to the integrator to restrict the maximum position displacement in the presence of non-zero integrator gains if the desired force cannot be achieved (i.e., if there is no contact).
The augmented path $\tilde{\Xi} = [\tilde{\xi}^0, \ldots, \tilde{\xi}^n]$ is sent to the whole-body MPC module.

G. Mechanical Stability

Large external wrenches applied at the end-effector to the environment can threaten the mechanical stability of the robot. We add a stability constraint to the MPC problem to prevent the robot from falling. If the Zero Moment Point (ZMP) remains in the support polygon of the robot, it is guranteed to be mechanically stable [25]. We compute the location of the ZMP $\boldsymbol{r}_{ZMP}$ considering the torque induced by gravity f_g at the center of mass (COM) $\boldsymbol{r}_{COM}$ and the planned interaction wrench at the end-effector $(\boldsymbol{f}_{EE}, \boldsymbol{\tau}_{EE})$. Dynamic effects are neglected in this work since the planned motion is relatively slow. By definition, the x and y component of the sum of moments is 0 at the ZMP which gives us

following identity:

$$\mathbf{0} \overset{!}{=} \boldsymbol{n} \times (\boldsymbol{r}_{COG} - \boldsymbol{r}_{ZMP}) \times \boldsymbol{f}_g - (\boldsymbol{r}_{EE} - \boldsymbol{r}_{ZMP}) \times \boldsymbol{f}_{EE} - \boldsymbol{\tau}_{EE}) \tag{13}$$

$$\boldsymbol{r}_{ZMP} = \frac{\boldsymbol{n} \times (\boldsymbol{r}_{COG} \times \boldsymbol{f}_g - \boldsymbol{r}_{EE} \times \boldsymbol{f}_{EE} - \boldsymbol{\tau}_{EE})}{\boldsymbol{n} \cdot (\boldsymbol{f}_g - \boldsymbol{f}_{EE})} \tag{14}$$

$\boldsymbol{n}$ is the surface normal vector, all quantities are noted in base frame, originated at the center of the support polygon. We inscribe a circle of radius r_{sc} into the support polygon to enforce that the ZMP stays inside with an inequality constraint:

$$g_{ZMP}(\boldsymbol{x}) = r_{sc}^2 - \|\boldsymbol{r}_{ZMP}\|_2^2 \geq 0 \tag{15}$$

Note that $\boldsymbol{r}_{COG}$ and $\boldsymbol{r}_{EE}$ depend on the current state $\boldsymbol{x}$. In case, high interaction wrenches are commanded, the constraint lets the robot adapt its configuration to operate safely.

III. Experiments

We evaluated the performance of our proposed method in many experiments, both in simulation and on different robotic platforms. The accompanying video[2] shows footage of those tests. In the following section, we first introduce our evaluation robots. We then describe the experiments to evaluate task-space trajectory tracking, collision avoidance, and interaction force control.

A. Robotic Platforms

The two robots used for this work are shown in Figure 1. Both platforms use the same control strategy. While we performed the lab experiments with *MabiMobile*, we took *Swaco* to the *MBZIRC 2020* robotics competition.

1) MabiMobile: MabiMobile is a mobile robot with a differential drive base with supporting castor wheels and a 6-DOF manipulator (*Mabi Speedy 12* by Mabi Robotics). The robot uses an Intel Realsense T265 for localization. A six-axis F/T sensor (*SenseOne* by BOTA Systems) is mounted between the end-effector tool and the arm. Joint velocities are sent to the manipulator with a rate of $250\,\mathrm{Hz}$. The current joint positions as well as the measured wrench from the sensor are transmitted to the onboard computer at the same rate via an Ethercat bus system. All hardware experiments were conducted on a single Intel NUC with a dual core i7-5557U processor.
Various tools can be attached to the end-effector for different applications. We use a spray gun to demonstrate end-effector tracking, a Robotiq 2F gripper to open a industrial cabinet and a springloaded sponge tool to wipe a whiteboard.
Along with the tool, there is a Intel Realsense D435i depth camera attached to the end-effector for online mapping.

[2] https://youtu.be/cTXytsWyFxE

TABLE I: Linear and rotational task-space tracking errors while following 6 reference paths of $1.5\,\mathrm{m}$ length in a hardware experiment.

$\mu_{pos}[\mathrm{m}]$	$\sigma_{pos}[\mathrm{m}]$	$\mu_{rot}[^\circ]$	$\sigma_{rot}[^\circ]$
0.0156	0.0035	2.3981	0.5039
0.0184	0.0111	2.3668	0.5848
0.0150	0.0036	2.4146	0.5038
0.0155	0.0031	2.4924	0.5349
0.0208	0.0042	3.1252	0.7177
0.0170	0.0039	2.7039	0.5628

2) Swaco: Swaco consists of a skid-steer base and a 7-DOF Kinova *Gen-3* manipulator. The robot is equipped with a LIDAR, two depth cameras, an IMU, and an RTK GPS receiver. During the *MBZIRC 2020* competition, we used this robot with the presented control toolbox to build a brick wall and extinguish fires.

B. MPC Implementation

Unless stated otherwise, the experiments were conducted with $\boldsymbol{R} = \mathbf{I}_8$. We implemented the MPC control module with the OCS2 toolbox[3] and used its SLQ solver [8] to compute optimal solutions with a rate of $20\,\mathrm{Hz}$ over a time horizon of $T = 2\,\mathrm{s}$. The necessary derivatives and Hessians were computed with CppAD Code Gen[4].
In contrast to our previous work, we do not follow the optimal whole-body trajectories with a separate tracking controller. Instead, we evaluate the affine policies with the latest state estimate at a rate of $250\,\mathrm{Hz}$ and send the computed control inputs to the motor controllers.
The whole-body MPC tracks task space trajectories. If the target application does not require a specific timing, we augment the reference paths with timestamps such that a maximum linear and angular velocity is not exceeded.

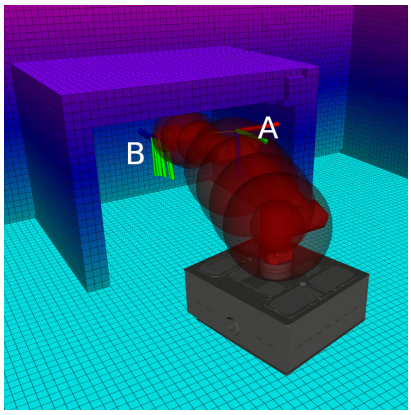
C. Trajectory Tracking

We mounted a paintbrush to the end-effector and sprayed a pattern onto a whiteboard to showcase a continuous manipulation task. To solely evaluate the controller performance and disregard the problem of global localization, we command the spraying path relative to the initial end-effector pose.
We measured the accuracy of the position and orientation tracking on 6 reference trajectories of $1.5\,\mathrm{m}$ length. Table I shows the mean positional and rotational deviations and their standard deviations of 200 recorded end-effector poses from the reference trajectory.

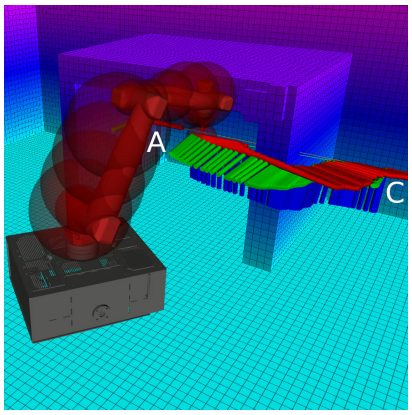
D. Collision Avoidance

We first evaluate the performance of the caching extension to *Voxblox* and compare them to the default implementation as a baseline. Afterwards, we demonstrate in a gazebo simulation experiment that given the same task, different behavior emerges when planning with an without collision constraints. Finally, we show in a hardware experiment, that

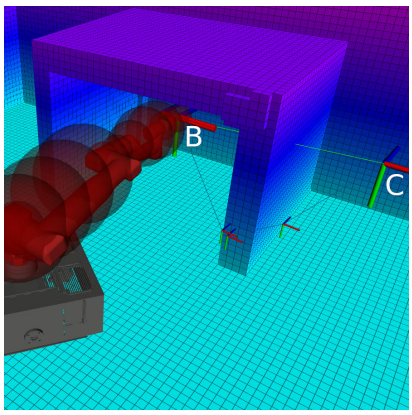
[3] https://bitbucket.org/leggedrobotics/ocs2
[4] https://github.com/joaoleal/CppADCodeGen



(a) Robot reaching underneith the table in experiment A to B with the SLQ method.



(b) Experiment A to C: The visualized path shows the planned end-effector poses of the whole-body motion plan.



(c) Experiment B to C: The light blue line shows the *RRTX* end-effector path that the *SLQ-MPC* tracks.

Fig. 2: Collision avoidance in ESDF maps: The robot tracks a desired end-effector path between a start and a goal poses (A,B,C). Collision checks are performed at 12 points on the manipulator, depicted with red spheres.

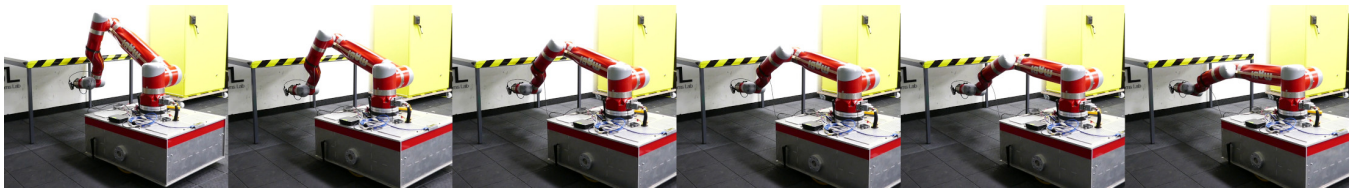


Fig. 3: Collision avoidance hardware experiment: The robot reaches underneath the table. The base first moves back such that the arm does not collide with the top of the table.

the whole pipeline runs in real-time and how collisions with an obstacle are avoided.

1) Voxblox Caching Benchmark: For the following experiments, we generated a ESDF from a gazebo world. The ESDF contains about 10^6 voxels with a side length of $5\,\mathrm{cm}$. Computing the gradients for each voxel took $575\,\mathrm{ms}$ in total. Caching the hessians as well took $2887\,\mathrm{ms}$. Note that in an online scenario, we would typically not recompute all gradients and hessians when the ESDF changes but only update the affected region.

We measured querying times of distances and gradients with the *Voxblox* standard interpolation methods and our caching methods. Table II shows the results for 1,000,000 queries each. Querying distances and gradients is significantly faster using the local linear or quadratic models based on the cached values rather than employing the interpolation scheme of the default *Voxblox* implementation. Even though the query time of the linear and the quadratic model are very similar, we opted for the linear models to get distance and gradients for collision avoidance. There was no visible improvement in the smoothness of the generated MPC trajectories when using the cached hessians as well.

2) Simulation Experiment: We compare the proposed method to two standard approaches in path planning. Our simulation environment contains a table and we define three end-effector poses A, B, and C. The benchmark task is to plan a collision free whole-body trajectory between those poses. Collision checks are performed on 12 points of the arm. We interpolate linearly between the start and goal pose for an end-effector reference path as the input to the SLQ whole-body planner. Alternatively, we plan a collision free end-effector path with a RRTX planner in $\Re^3$ and use this path as an input to our method. We refer to this method as *SLQ_RRTX_EE*. The end-effector orientation remains to be interpolated between start and goal pose. A planning horizon of $1\,\mathrm{s}$ is sufficient to find a near optimal reference path for our test scenarios. The SLQ planning horizon is set to $8\,\mathrm{s}$ which is sufficient to plan the entire trajectory for each benchmark task.

The two methods we compare our work to are the sampling based planners *RRTX*[20] and *BiFMT* [21]. For both planners, we define a $SE(2) \times \mathbb{R}^6$ state space and employ

TABLE II: Comparison of the query times between the default *Voxblox* implementation and our method with linear and quadratic models based on cached gradients and hessians. We report durations for 10^6 queries each. For all three query types, our methos is significantly faster than the default implementation.

Method	T[ms]
Distance Interpolation (default)	491
Distance, Linear Model (our method)	**58**
Distance Interpolation, Gradient at voxel center (default)	1047
Distance and Gradient, Linear Model (our method)	**71**
Distance and Gradient Interpolation (default)	3559
Distance and Gradient, Quadratic Model	**68**

uniformly distributed sampling within the robots joint limits and bounded base workspace $[-2.5\,\mathrm{m}, 2.5\,\mathrm{m}]^2$. The sampling time for *RRTX* is set to $200\,\mathrm{s}$. We count a plan as successful if the final pose does not deviate from the goal pose by $0.5\,\mathrm{m}$ and $0.5\,\mathrm{rad}$. The *BiFMT* requires a valid goal configuration to perform the bidirectional search. We provide collision free start and goal configurations for each goal pose A, B, and C. The number of samples to be drawn is fixed and set to 10000. All other parameters are set to default as specified in the *OMPL*[26] version 1.4.2. The optimization criterion for both planners is a minimal path length in the configuration space. We use the same metric to compare the results of our planner with the baseline.

For each combination for planner and task, 10 experiments were conducted. Table III reports the averaged results. The SLQ method is successful in all trials for 2 tasks. It outperforms the baseline solutions both by yielding in shorter paths and with a faster computation time. The *per-iteration* time is short enough to run the planner as a model predictive controller. The task B to C can not be solved with the *SLQ* method, since the obstacle avoidance would require a large deviation from the end-effector reference path. The *SLQ_RRTX_EE* method can solve all posed tasks and yields slightly shorter paths and significantly faster convergence compared to the plain *SLQ* method. *RRTX* does not perform well on the tasks because randomly sampling valid states in the goal region is difficult in the high dimensional configuration space. *BiFMT* works better since the goal state is already fixed and it merely needs to find valid intermediate states.

3) Robot experiment: We repeated the A to B experiment on hardware. The robot receives a command to reach underneath a table and has to adapt its posture to not collide with the table. Figure 3 shows the motion sequence during the experiment.

Voxblox processes incoming point-clouds from the depth camera at a rate of $1\,\mathrm{Hz}$ updates the ESDF. We cache gradients at the same rate. Since the map is smaller than in simulation, caching times are neglectable. In contrast to the simulation experiment, we commanded a 6-Dof end-effector pose trajectory. The emerging behavior is similar to the one observed in simulation. Once the obstacles are close enough, such that a collision would occur within the planning horizon, the robot moves back with the mobile base and stretches our the base to avoid a collision of its forearm with the edge of the table.

E. Task-Space Admittance Control

The admittance control scheme requires the controller to quickly react to changes of the measured interaction force. The update rate of the MPC is therefore increased to $100\,\mathrm{Hz}$.

1) Unlocking a door: For an integration test, we commanded the robot to open the door of an industrial cabinet. It first reaches for the handle by tracking a task space trajectory and then turns the door handle with the admittance control strategy. The experiment is featured in the video. Admittance gains of $K_p = 10^{-3}\mathbf{I}_3$, $K_i = 10^{-2}\mathbf{I}_3$ and anti-windup limits of $3\,\mathrm{N}$ were chosen to control forces in all directions. For the torques, we chose $K_p = 10^{-2}\mathbf{I}_3$, $K_i = 10^{-1}\mathbf{I}_3$ and anti-windup limits of $1\,\mathrm{N\,m}$. The strategy for turning the handle was to command an end-effector path that roughly describes the desired motion. Due to uncertainties in the handle location, the grasping pose and the robot's state estimate, the path could not be executed by just using position control. The desired wrench was set to 0. The P-gains would cause a steady-state error in wrench tracking and render a spring-like behavior between the handle and the current pose reference. The I-gains damp the system. Low integrator limits prevent a stationary exact wrench tracking which would harm the pose tracking objective.

The admittance controller and some foam we attached to the handle provided the necessary compliance to turn the handle. Without the foam, we observed that the controller quickly went unstable. Because the combined system of robot and door is still very stiff, we had to command slow reference motion and set the high control input weights $\boldsymbol{R} = 10^3\mathbf{I}$.

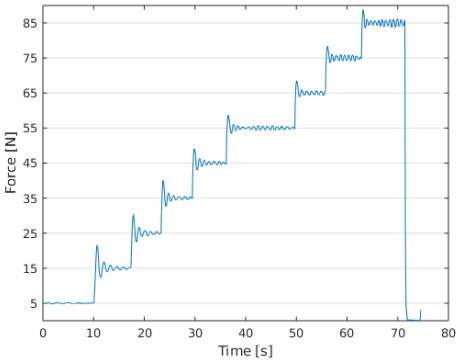


Fig. 4: The robot exerting forces with its end-effector . The desired interaction force increases in $10\,\mathrm{N\,m}$ increments. A wide range of forces can be tracked with high accuracy.

2) Continuous Interaction Force Control: For the following experiments, we use a compliant end-effector tool. It allows for lower input weights of $\boldsymbol{R} = 10^1\mathbf{I}$ which makes the controller more reactive.

The robot exerts force onto a white-board. While not changing the end-effector position reference, we increased the desired force in $10\,\mathrm{N}$ steps starting with $5\,\mathrm{N}$. Figure 4 shows the measured interaction force perpendicular to the whiteboard over time. After overshooting on changes in the force reference, the former is being tracked with high accuracy. In the time interval $[39\,\mathrm{s}, 49\,\mathrm{s}]$ the mean measured force was $54.995\,\mathrm{N}$ with a standard deviation of $0.369\,\mathrm{N}$. Similar accuracies can be measured for the other force references.

In another series of experiments, we command an end-effector pose reference to move $1.5\,\mathrm{m}$ along the whiteboard, move $20\,\mathrm{cm}$ up and move $1.5\,\mathrm{m}$ while exerting a constant interaction-force of $30\,\mathrm{N}$. Figure 5 shows the recorded forces over time, along with the end-effector position in world-frame. After an initial peak in force when getting in contact

TABLE III: Comparison of our method (SLQ and SLQ with an RRTX end effector plan) against the baseline solutions RRTX and BiFMT. We report average results from 10 experiments each. For SLQ and SLQ_RRTX_EE, we report the computation time till convergence and additionally the time per iteration that is relevant for running the planner in a receding horizon fashion.

Experiment	**Algorithm**	**Success Rate**	**Path Length**$[AU]$	**Clearance** [m]	**Computation Time:** [s]	**Iterations**
AB	RRTX	0.1	8.273	0.169	200	-
AB	BiFMT	1	6.240	0.142	19.403	-
AB	SLQ	1	3.561	0.056	1.376 (0.074)	18.7
AB	SLQ_RRTX_EE	1	3.326	0.069	0.396 (0.035) + 1.0	11
AC	RRTX	0.5	10.394	0.158	200	-
AC	BiFMT	1	6.016	0.209	16.925	-
AC	SLQ	1	5.383	0.058	1.018 (0.056)	18.1
AC	SLQ_RRTX_EE	1	4.819	0.0524	0.4009 (0.027) + 1.0	14.7
BC	RRTX	0.6	10.138	0.16	200	-
BC	BiFMT	1	7.390	0.187	23.106	-
BC	SLQ	0	*1.478*	*-0.001*	*0.838 (0.064)*	*13*
BC	SLQ_RRTX_EE	1	4.0134	0.098	0.100 (0.014) + 1.0	7

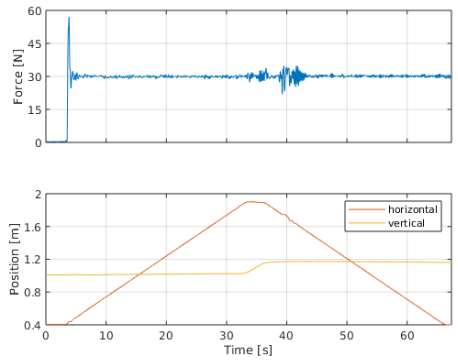


Fig. 5: Continuous end-effector motion along a wall, while maintaining a constant interaction force. The upper plot shows the interaction force in z direction over time. The lower plot shows the tool's vertical and horizontal position on the wall. After overshooting during contact establishment, the desired force is tracked reliably. At $t = 40\,\mathrm{s}$, there is a small perturbation due to play in the end-effector tool from which the controller quickly recovers.

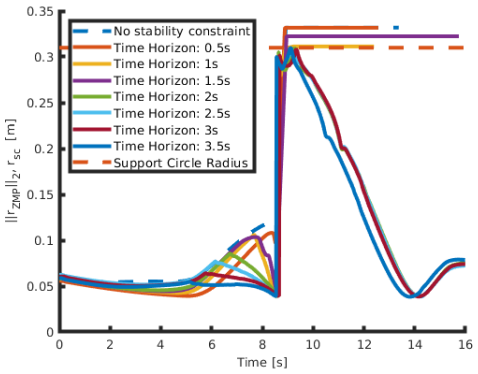


Fig. 6: Distance of the ZMP from the center of the support polygon. At $t = 8.5\,\mathrm{s}$, the robot grasps a heavy object. Without the stabiliy constraint, the ZMP moves outside the support polygon and the robot falls. The larger the prediction horizon of the MPC, the better it can prepare for the grasp by shifting its COM. For time horizons larger than $1.5\,\mathrm{s}$, the constraint is satified and the robot does not fall.

with the wall, the robot tracks the reference force reliably. The mean force measured from $t = 5\,\mathrm{s}$ on is $29.91\,\mathrm{N}$ and the standard deviation $0.42\,\mathrm{N}$. A small disturbance is caused by an undesired flipping of the sponge around $t = 40\,\mathrm{s}$. Recordings of this experiment are included in the video.

F. Mechanical Stability

We assess the effectiveness of the ZMP constraint in a simulated grasping experiment. After an initial rotation of the end-effector by $90\,\mathrm{deg}$, the robot has to reach $1\,\mathrm{m}$ to the front, grasp an object of $100\,\mathrm{kg}$ and retreat $1\,\mathrm{m}$. The MPC is informed about the additional mass by increasing the reference force $\boldsymbol{f}_{EE}$ to $1000\,\mathrm{N}$ in the direction of gravity at the grasping time.

In a base-line experiment without the ZMP constraint, we observe that the robot stretches out the arm when approaching the target and then falls after grasping the object. With the constraint active, we conducted a series of experiments in which we subsequently increase the planning horizon from $0.5\,\mathrm{s}$ in $0.5\,\mathrm{s}$ increments to $3.5\,\mathrm{s}$. The support circle radius is set to $30\,\mathrm{cm}$ for the soft constraint. We count an experiment as successful, if the ZMP does not leave a circle of radius $31\,\mathrm{cm}$. Figure 6 shows the distance of the ZMP to the center of the support polygon over time for the different experiments. The stability constraint causes the robot to move to a configuration where the COM balances out the large external force at the end-effector. A time horizon of $2\,\mathrm{s}$ or more gives adequate time to prepare for the expected external force and the robot remains in balance. For shorter time horizons, velocity limits prevent the robot from shifting into a safe configuration prior to the grasping moment and the robot falls.

IV. Conclusions

We presented a perceptive model predictive control strategy for mobile robots. End-effector task-space trajectories can be tracked while respecting joint and stability constraints. In multiple hardware experiments, we showed how this

Fig. 7: First results in robotic plastering. The robot applies plaster on the wall with a spraying nozzle.

enables mobile robots to execute tasks that exceed the workspace boundaries of fixed based systems.

Querying cached distances and gradients from a euclidean signed distance field proved to be an efficient method of introducing awareness of the robot's surrounding to a receding horizon controller. We achieve an order of magnitude speed-up at query time compared to the state of the art solution. This allows us to plan around obstacles as shown in the hardware experiment, while preserving the advantages of a reactive controller.

The admittance control module enables precise interaction force control with a position-controlled manipulator on a mobile base. To our best knowledge, this is the first demonstration of receding horizon admittance control on a mobile robot. We want to use those capabilities for construction robotics tasks such as plastering. Figure 7 shows the first successful results for spraying plaster. A mobile robot will trowel the plaster with the presented method. Future work could model the contact of the compliant end-effector tool and add a spring compression state to the model predictive controller.

The open-source toolbox allows for fast integration of the presented control strategies into other robotic platforms. This simplifies the deployment of mobile manipulators in different application domains.

REFERENCES

[1] "Imagining construction's digital future | McKinsey," https://www.mckinsey.com/industries/capital-projects-and-infrastructure/our-insights/imagining-constructions-digital-future.

[2] J. Santagate and K. Prouty, "Five Key Reasons to Consider Robotics for Industrial Inspections," Sep. 2018.

[3] "Trimble, Hilti Partner on Site Robot Project," https://www.constructionequipment.com/ TrimbleHiltiPartneron-SiteRobotProject.

[4] N. Hack, T. Wangler, J. Mata-Falcón, K. Dörfler, N. Kumar, A. N. Walzer, K. Graser, L. Reiter, H. Richner, J. Buchli *et al.*, "Mesh mould: An on site, robotically fabricated, functional formwork," in *Second Concrete Innovation Conference (2nd CIC), Paper*, no. 19, 2017.

[5] A. R. Gawel, H. Blum, J. Pankert, K. Kraemer, L. Bartolomei, S. Ercan, F. Farshidian, M. Chli, F. Gramazio, R. Siegwart, M. Hutter, and T. Sandy, "A Fully-Integrated Sensing and Control System for High-Accuracy Mobile Robotic Building Construction," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov. 2019, p. 8.

[6] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning," *arXiv:1105.1186 [cs]*, May 2011.

[7] M. Neunert, C. de Crousaz, F. Furrer, M. Kamel, F. Farshidian, R. Siegwart, and J. Buchli, "Fast nonlinear Model Predictive Control for unified trajectory optimization and tracking," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 1398–1404.

[8] F. Farshidian, M. Neunert, A. W. Winkler, G. Rey, and J. Buchli, "An Efficient Optimal Planning and Control Framework For Quadrupedal Locomotion," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 93–100, May 2017.

[9] M. Giftthaler, F. Farshidian, T. Sandy, L. Stadelmann, and J. Buchli, "Efficient kinematic planning for mobile manipulators with non-holonomic constraints using optimal control," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. Singapore, Singapore: IEEE, May 2017, pp. 3411–3417.

[10] R. Grandia, F. Farshidian, R. Ranftl, and M. Hutter, "Feedback MPC for Torque-Controlled Legged Robots," *arXiv:1905.06144 [cs]*, Aug. 2019.

[11] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *2009 IEEE International Conference on Robotics and Automation*. Kobe: IEEE, May 2009, pp. 489–494.

[12] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 4569–4574.

[13] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 1366–1373.

[14] L. Han, F. Gao, B. Zhou, and S. Shen, "FIESTA: Fast Incremental Euclidean Distance Fields for Online Motion Planning of Aerial Robots," *arXiv:1903.02144 [cs]*, Jul. 2019.

[15] D. Lawrence, "Impedance control stability properties in common implementations," in *1988 IEEE International Conference on Robotics and Automation Proceedings*, Apr. 1988, pp. 1185–1190 vol.2.

[16] J. Matschek, J. Bethge, P. Zometa, and R. Findeisen, "Force Feedback and Path Following using Predictive Control: Concept and Application to a Lightweight Robot," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 9827–9832, Jul. 2017.

[17] A. Wahrburg and K. Listmann, "MPC-based admittance control for robotic manipulators," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, Dec. 2016, pp. 7548–7554.

[18] D. Leidner, A. Dietrich, F. Schmidt, C. Borst, and A. Albu-Schäffer, "Object-centered hybrid reasoning for whole-body mobile manipulation," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 1828–1835.

[19] M. V. Minniti, F. Farshidian, R. Grandia, and M. Hutter, "Whole-Body MPC for a Dynamically Stable Mobile Manipulator," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3687–3694, Oct. 2019.

[20] M. Otte and E. Frazzoli, "RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, Jun. 2016.

[21] J. A. Starek, J. V. Gomez, E. Schmerling, L. Janson, L. Moreno, and M. Pavone, "An asymptotically-optimal sampling-based algorithm for Bi-directional motion planning," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 2072–2078.

[22] B. Siciliano, Ed., *Robotics: Modelling, Planning and Control*, ser. Advanced Textbooks in Control and Signal Processing. London: Springer, 2009.

[23] M. Frigerio, J. Buchli, D. G. Caldwell, and C. Semini, "RobCoGen: A code generator for efficient kinematics and dynamics of articulated robots, based on Domain Specific Languages," *Journal of Software Engineering for Robotics (JOSER)*, vol. 7, no. 1, pp. 36–54, Jul. 2016.

[24] C. Feller and C. Ebenbauer, "Relaxed Logarithmic Barrier Function Based Model Predictive Control of Linear Systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 3, pp. 1223–1238, Mar. 2017.

[25] M. Vukobratović and B. Borovac, "Zero-moment point — thirty five years of its life," *International Journal of Humanoid Robotics*, vol. 01, no. 01, pp. 157–173, Mar. 2004.

[26] I. A. Sucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics Automation Magazine*, vol. 19, no. 4, pp. 72–82, Dec. 2012.