Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Model Predictive Interaction Control for Industrial Robots

Tobias Gold, Andreas Völz, Knut Graichen

Chair of Automatic Control,
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
(tobias.gold@fau.de, andreas.voelz@fau.de, knut.graichen@fau.de)

Abstract: This paper discusses the use of model predictive control (MPC) for industrial robot applications with physical robot-environmental interaction. A model predictive interaction control (MPIC) scheme is introduced that deals both with the prediction of the robot motion and the forces between robot and environment. With regard to the robot motion, either the rigid body dynamics, a simplified model, or a cascaded control structure can be employed. The external forces or torques are treated as additional state variables whose dynamics are based on the elastic behavior of the contact surface. Since the force prediction depends on the knowledge of the environmental stiffness, a method for online estimation is discussed. The approach allows to realize different tasks as motion control, compliance control, direct force control as well as hybrid force/motion control by adjusting the weighting factors in the cost function. The implementation is based on the nonlinear MPC software GRAMPC and the library PINOCCHIO for computation of rigid body dynamics. Besides comparing the different robot dynamics models, the approach is demonstrated for a hand-guiding and a table wiping task.

Keywords: Nonlinear model predictive control, physical robot-environment interaction, force control, motion control, hybrid force/motion control, compliance control, stiffness estimation

1. INTRODUCTION

Model predictive control (MPC) is widely used in the robotics community. This is, on the one hand, due to its ability to handle linear or nonlinear multiple-input multiple-output systems with state and input constraints. On the other hand, the formulation of the control task as optimization problem offers a high degree of flexibility. The drawback is the high computational demand for the online solution of the optimization problem. Some examples for the application of MPC in robotics are the stabilization of the trunk of a humanoid robot in Koenemann et al. (2015) or the imitation of the human gait in Schultz and Mombaur (2010). Especially the consideration of contact restrictions in MPC can lead to advantages over alternative methods as shown in Mordatch et al. (2012).

In the case of industrial robots, the use of classical control methods prevails. However, the system is per se constrained by mechanical limits, such as maximum joint positions and velocities. Furthermore, there are often task-specific limitations e.g. workspace bounds or the maximum payload at the tip of robot. In motion planning, these constraints as well as the rigid body dynamics are already taken into account. For example, Verschueren et al. (2016) show the time-optimal motion planning for a six degree of freedom robot under consideration of task-specific boundaries. An application of MPC for the time-optimal transition between operating points and their stabilization is discussed in Zhao et al. (2004), at least for simple robot examples. Lee et al. (2005), by contrast, present an approach using recursive algorithms for the dynamic calculation of arbitrarily complex robot structures for optimization.

For tasks that require physical interaction between the robot and the environment, e.g. grinding or welding, the application of model predictive control is more difficult, since the robot has to predict the resulting forces and torques. In Wahrburg and Listmann (2016) a model predictive admittance controller is proposed that constrains the occurring force magnitudes and allows to handle environments with different stiffness. The work of Matschke et al. (2017) combines predictive path-following with force control, whereby a linear spring model is used to predict the forces. Similarly, in Kazim et al. (2018) predictive path-following is combined with admittance control by including the admittance dynamics in the optimal control problem. However, due to the high effort for the optimization, the application of such approaches is often restricted to systems with few degrees of freedom.

This paper introduces a novel formulation for model predictive interaction control (MPIC) that includes the dynamics of the robot and an additional model for the prediction of external forces and torques. Hereby, a spring model that describes the elastic behavior of the interaction object is used to derive the force dynamics. Similar as in Jung et al. (2001), the required object stiffness may be estimated from position and force measurements. Different control tasks such as motion control, compliance control, force control or hybrid force/motion control can be realized with the same MPIC approach by appropriately designing the cost function. To tackle the computational demand for typical industrial robots, the implementation is based on

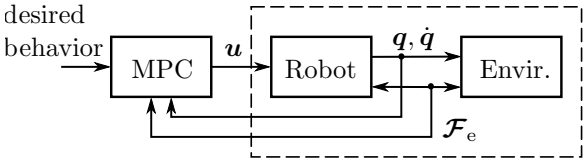

Fig. 1. MPC of robots in interaction with an environment.

state-of-the-art software packages. The optimization problem is solved using the nonlinear model predictive control software GRAMPC (Englert et al., 2019) that implements an augmented Lagrangian algorithm with an inner gradient method. The computation of the rigid body dynamics as well as their analytic derivatives is performed using the toolbox PINOCCHIO (Carpentier et al., 2019) which applies the computationally efficient articulated body algorithm and the recursive Newton-Euler algorithm. The proposed MPIC approach is demonstrated for a hand-guiding and a table wiping task. Furthermore, the real-time feasibility is investigated using different models for the robot dynamics.

2. INTERACTION MODELING

This section discusses the model of the interaction between a robot and an environment. The considered system is composed of the robot system and the environmental system as pictured in Figure 1. The robot subsystem describes the motion of the robot, whereas the environmental subsystem describes the behavior of the forces.

2.1 Modeling of soft contact interaction

Many materials provide a elastic behavior which is described by the stress and strain under an acting force. In the further course, the focus is solely orientated on the linear-elastic material behavior, which is described by Hooks' law

$$\sigma = E\epsilon \tag{1}$$

with the internal stress $\sigma = \frac{F_\mathrm{ext}}{A}\,[\mathrm{N/mm^2}]$, the proportional factor $E\,[\mathrm{N/mm^2}]$ and the strain $\epsilon = \frac{\Delta l}{l_0}$ under the assumption of a homogeneous and isotropic material. The law may be reformulated into the force-deformation relation

$$F_\mathrm{ext} = \frac{EA}{l_0}\Delta l = K_\mathrm{e}\Delta l\,, \tag{2}$$

with the spring stiffness K_e and a deformation Δl. The relationship shows that the linear-elastic material behavior is described by a spring equation with the stiffness K_e according to the material properties.

The single degree of freedom force model will be now extended to the general six degree of freedom form for the robotic interaction case. For reasons of clarity, further interactions are considered between the robot tip and an environmental object. Nevertheless, the methodology also includes other forces and moments at the robot structure. The wrench

$$\boldsymbol{\mathcal{F}}_\mathrm{ext} = [\boldsymbol{F}_\mathrm{ext}^\mathrm{T}, \boldsymbol{m}_\mathrm{ext}^\mathrm{T}]^\mathrm{T} \tag{3}$$

is the 6×1 vector of the contact forces $\boldsymbol{F}_\mathrm{ext}$ and moments $\boldsymbol{m}_\mathrm{ext}$ acting between the robot end-effector and the environment. The pose vector

$$\boldsymbol{p}(\boldsymbol{q}) = [\boldsymbol{t}(\boldsymbol{q})^\mathrm{T}, \boldsymbol{\phi}(\boldsymbol{q})^\mathrm{T}]^\mathrm{T} = \boldsymbol{T}_e^b(\boldsymbol{q}) \tag{4}$$

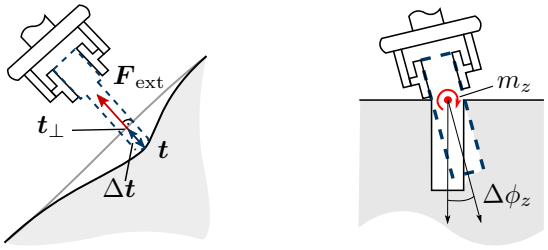

Fig. 2. Schematic model of a compliant environment with translational and rotational deformation.

composes the end-effector position $\boldsymbol{t}$ and the orientation, denoted by the Euler angles $\boldsymbol{\phi}$, which is calculated by the coordinate transformation of the joint position to the Cartesian world frame. The Cartesian motion of the pose

$$\dot{\boldsymbol{p}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = [\boldsymbol{v}(\boldsymbol{q}, \dot{\boldsymbol{q}})^\mathrm{T}, \boldsymbol{\omega}(\boldsymbol{q}, \dot{\boldsymbol{q}})^\mathrm{T}]^\mathrm{T} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}\,, \tag{5}$$

described by the translational and the angular velocities of the end-effector $\boldsymbol{v}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and $\boldsymbol{\omega}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, is determined by the product of the Jacobian $\boldsymbol{J}(\boldsymbol{q})$ and the joint velocity $\dot{\boldsymbol{q}}$.

The spring model (2) can be used in the operational space by calculating the distance between surface pose and end-effector pose as pictured in Figure 2. The following topic will be first introduced in the translational case and be extended later to the rotational case. The model (2) is used to calculate the force

$$\boldsymbol{F}_\mathrm{ext} = \boldsymbol{K}_\mathrm{trans}(\boldsymbol{t} - \boldsymbol{t}_\perp) = \boldsymbol{K}_\mathrm{trans}\Delta\boldsymbol{t} \tag{6}$$

from the penetration of the end-effector into an elastic environment $\Delta\boldsymbol{t}$ with the positive semi-definite translational environmental stiffness matrix $\boldsymbol{K}_\mathrm{trans}$. The reference position $\boldsymbol{t}_\perp$ of the undeformed object is formed at the origin of the normal of the undeformed object surface $\boldsymbol{N} \in \mathbb{R}^3$ through the end-effector position $\boldsymbol{t}$.

The aim of the approach is to handle the force as a state variable in the MPIC. To this end, the dynamic behavior of the force has to be considered. Under the assumption of an end-effector motion in normal direction, the reference position will be constant, i.e. $\boldsymbol{t}_\perp = const.$ Thus, the time derivative of the force is determined according to

$$\dot{\boldsymbol{F}}_\mathrm{ext} = \boldsymbol{K}_{trans}\dot{\boldsymbol{t}} = \boldsymbol{K}_{trans}\boldsymbol{v}\,. \tag{7}$$

This means that the change of the force is proportional to the Cartesian end-effector velocities.

For the rotational case, a torque along a single axis $i \in (x, y, z)$ is considered. The torsional torque

$$m_{\mathrm{ext},i} = K_{\mathrm{tors},i}\Delta\phi_i \tag{8}$$

is calculated due to rotational deformation $\Delta\phi_i$ and a torsional stiffness $K_{\mathrm{tors},i}$ of the environment along a single axis. The extension to the rate of change of the torques

$$\dot{m}_{\mathrm{ext},i} = K_{\mathrm{tors},i}\,\omega_i \tag{9}$$

is similar to the procedure in the translational case using the angular velocity ω_i. The vector $\boldsymbol{m}_\mathrm{ext} = [m_{\mathrm{ext},x}, m_{\mathrm{ext},y}, m_{\mathrm{ext},z}]^\mathrm{T}$ consists of the torques along the three Cartesian axes. Hence, the full wrench dynamics $\dot{\boldsymbol{\mathcal{F}}}_\mathrm{ext}$ is build by

$$\dot{\boldsymbol{\mathcal{F}}}_\mathrm{ext} = \begin{bmatrix}\dot{\boldsymbol{F}}_\mathrm{ext}\\ \dot{\boldsymbol{m}}_\mathrm{ext}\end{bmatrix} = \begin{bmatrix}\boldsymbol{K}_\mathrm{trans}\boldsymbol{v}\\ \boldsymbol{K}_\mathrm{rot}\,\boldsymbol{\omega}\end{bmatrix} = \boldsymbol{K}_e\dot{\boldsymbol{p}}\,. \tag{10}$$

The stiffness of the compliant object and the environmental surface play an important role for determining the

spring model. The next section will discuss the online estimation of the stiffness during an interaction with an object.

2.2 Stiffness estimation

The accuracy of the force prediction highly depends on the knowledge of the environmental stiffness, which is often not a-priori available. To this end, an approach for stiffness estimation is presented, where the actual interaction with its resulting force and deformation is evaluated in the manner of Jung et al. (2001).

The resistance against elastic deformation of an environment due to an applied force is described by the stiffness. The value can therefore be calculated in the scalar case by the relation of the force and the resulting deformation according to

$$K = \frac{F_{\text{ext}}}{t - t_{\perp}} = \frac{F_{\text{ext}}}{\Delta t}\,, \tag{11}$$

see e.g. (Spong et al., 2005). Principally, the actual stiffness is determined with every new force and position measurement. However, especially the force measurement is noisy, which requires a filtering of the estimation.

To this end, a Kalman filter is used to filter the noisy stiffness calculation of (11) with a constant autonomous parameter model. The initialization of the stiffness estimation K^0 may be done with an approximated stiffness values of the expected interactions. The standard time-discrete Kalman filter algorithm is updated in every sample step if a force respectively torque threshold $F_{\text{ext},i} > F_{\min}$ or $m_{\text{ext},i} > m_{\min}, \forall i \in [x, y, z]$ is exceeded. The initial value of $\boldsymbol{K}_e$ should be chosen to a high value to overestimate the force in the initial phase. Thus, the robot will stop faster in case of a contact.

Besides the force measurement, the penetration of the end-effector in the environmental object is a necessary value for stiffness estimation. To this end, the reference position $\boldsymbol{t}_{\perp}$ of the undeformed environment is the base for calculating the value. One way for finding this pose is to supervise a threshold for detecting the first contact. If the penetration motion is done solely in direction of the acting force, the reference pose is constant over the task. However, if a relative motion with respect to the acting force complementary directions exist, the reference position is changing and has to be estimated.

The surface of an environmental object can be modeled for example as a plane E in the Cartesian space. To this end, the reference position $\boldsymbol{t}_{\perp}$ defines the support vector of the plane and the force vector at the point of contact detection defines the corresponding normal vector $\boldsymbol{N}$. With both information, the plane of the environmental surface at the contact point is defined.

In the next step, the distance of the end-effector position to the plane is determined by the approach of the dropped perpendicular foot. Thereto, a help line $\boldsymbol{h} = \boldsymbol{t} + c\boldsymbol{N}$ standing vertically on the plane E is built up. Afterwards, the intersection point of E and $\boldsymbol{h}$ has to be found and the distance of this point to the end effector position $\boldsymbol{t}$ determined. This distance is the penetration depth $\Delta\boldsymbol{t}$ of the end-effector in the environment.

3. MODEL PREDICTIVE INTERACTION CONTROL

The model predictive interaction control scheme deals with the control of a coupled robot-environment system during interaction tasks. As pictured in Figure 1, the coupling between the robot subsystem and the passive environmental subsystem is established by the force acting on the robot structure. This section discusses the model predictive control scheme for interaction tasks. In the first step, the basic functionality of MPC is described with a generic formulation of the optimization problem. This will be concretized step by step for example by the system dynamics, composed of the robot and the interaction dynamics. Further components of the optimization problem such as constraints and the choice of the cost function will be dealt with in the further course. Finally, the numerical solution of the optimization problem and the robot dynamics calculation is discussed.

3.1 Basic structure

Model predictive control is based on the iterative (suboptimal) solution of a dynamic optimization problem

$$\min_{\boldsymbol{u}} \quad J(\boldsymbol{u}; \boldsymbol{x}_k) = V(\boldsymbol{x}(T)) + \int_0^T l(\boldsymbol{x}(\tau), \boldsymbol{u}(\tau))\,\mathrm{d}\tau \tag{12a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(\tau) = \boldsymbol{f}(\boldsymbol{x}(\tau), \boldsymbol{u}(\tau))\,, \qquad \boldsymbol{x}(0) = \boldsymbol{x}_k \tag{12b}$$

$$\boldsymbol{x}(\tau) \in \mathcal{X}\,,\ \boldsymbol{u}(\tau) \in \mathcal{U} \tag{12c}$$

over a moving horizon $\tau \in [0, T]$ with length T. For this purpose, a control trajectory $\boldsymbol{u}$ is calculated which minimizes the cost function (12a). The minimization is constrained by the system dynamics (12b) and the admissible set (12c) of the states $\boldsymbol{x}$ and control variables $\boldsymbol{u}$. The optimization in the MPC sense initializes the states with the actual measured or estimated quantities $\boldsymbol{x}_k$. The initial solution of the control variable is taken from the optimization of the last time step. The first part of the optimal solution is executed on the controlled system up to the next time step. In the following, the individual parts of the problem (12) are introduced in general and specified in the Section 4 according to three applications.

3.2 Robot dynamics

The dynamic model

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}_{\text{J}} + \boldsymbol{\tau}_{\text{ext}}, \tag{13}$$

of a robot arm with n joints describes the relationship between the resulting motion due to the applied generalized forces. For this purpose, the generalized coordinates $\boldsymbol{q} \in \mathbb{R}^n$ and $\dot{\boldsymbol{q}} \in \mathbb{R}^n$, the states of the system, describe the joint position and the velocity of the robot. The joint torques $\boldsymbol{\tau}_{\text{J}} \in \mathbb{R}^n$ are the input of the system and the external torques $\boldsymbol{\tau}_{\text{ext}} \in \mathbb{R}^n$ will be in the comment matter of view a disturbance. This definition is revisited later in the paper. The external torques are the result of an external force on the robot structure. In the case of an external force at the end-effector, the external torques $\boldsymbol{\tau}_{\text{ext}} = \boldsymbol{J}(\boldsymbol{q})^{\text{T}}\boldsymbol{\mathcal{F}}_{\text{ext}}$ are obtained from the wrench $\boldsymbol{\mathcal{F}}_{\text{ext}}$ using the transposed Jacobian.

3.3 Interaction Dynamics

In this section, the previous view of the external force $\boldsymbol{\mathcal{F}}_{\text{ext}}$ as a disturbance is reinterpreted. In the case of an

interaction task, the external force can be regarded as a state, since it is used to describe the behavior of the overall system. On the one hand, the interaction is considered solely via the torques in the joint space. In Section 2.1, the torsional spring model was introduced to calculate the torques around a axis. According to (9), the dynamic model of the external torques

$$\dot{\boldsymbol{\tau}}_{\text{ext}} = \boldsymbol{K}_{\text{J}}\dot{\boldsymbol{q}}, \ , \qquad \forall \boldsymbol{K}_{\text{J}} \geq \boldsymbol{0} \tag{14}$$

describes the behavior of the interaction in joint space. The virtual joint stiffness $\boldsymbol{K}_{\text{J}} = \text{diag}(k_{\text{J},1}, \ldots, k_{\text{J},n}) \in \mathbb{R}^{n\times n}$ is a degree of freedom of this formulation and has to be chosen to a desired joint stiffness. However, the joint space formulation leads to the fact that the information about Cartesian forces and moments are not taken into account and instead each individual joint is considered decoupled.

The virtual joint stiffness is considered as constant so far. This assumption holds as long as the robot is in interaction. In free motion, the virtual stiffness has to be zero because a free motion does not cause any external torques. To this end, a selection of the virtual stiffness

$$k_{\text{J},i}(t) \begin{cases} = 0, \text{ if } \quad \tau_{\text{ext},i}(t) = 0, \\ > 0, \text{ else}, \end{cases} \tag{15}$$

with respect to the actual external torques is required. This leads to a problem without torque prediction in (12b).

The definition of the interaction in Cartesian space results in the modeling of the force dynamics

$$\dot{\boldsymbol{\mathcal{F}}}_{\text{ext}} = \boldsymbol{K}_{\text{e}}\boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}\,. \tag{16}$$

according to (10). The matrix $\boldsymbol{K}_{\text{e}} = \text{diag}(\boldsymbol{K}_{\text{trans}}, \boldsymbol{K}_{\text{tors}}) \in \mathbb{R}^{6\times 6}$ describes the stiffness of the interaction object. Basically, the model shows how the forces increase or decrease in the individual Cartesian directions during a particular motion $\dot{\boldsymbol{p}} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$. Similar to the joint space formulation, the virtual Cartesian stiffness $\boldsymbol{K}_{\text{e}}$ is either constant for a pure interaction control or time variant with a selection in the manner of (15) according to

$$K_{\text{e},i}(t) \begin{cases} = 0, \text{ if } F_{\text{ext},i}(t) = 0 \text{ or } m_{\text{ext},i}(t) = 0, \\ > 0, \text{ else}\,. \end{cases} \tag{17}$$

Another possibility is to estimate the environmental stiffness and adapt them online. This approach is introduced in the Section 2.2. The stiffness will be updated in every sampling step, but will be constant over the prediction horizon.

3.4 Constraints

Restrictions play an important role in robotic applications, since for example certain maximum values of quantities are set in norms. On the one hand, state variables of the robot system as well as the control variables are limited by upper and lower bounds. These robot boundaries are formulated as box constraints according to

$$\boldsymbol{q} \in [\boldsymbol{q}^{-}, \boldsymbol{q}^{+}]\,, \tag{18a}$$

$$\dot{\boldsymbol{q}} \in [\dot{\boldsymbol{q}}^{-}, \dot{\boldsymbol{q}}^{+}]\,, \tag{18b}$$

$$\boldsymbol{\tau}_{\text{J}} \in [\boldsymbol{\tau}_{\text{J}}^{-}, \boldsymbol{\tau}_{\text{J}}^{+}]\,. \tag{18c}$$

In addition, certain restrictions have to be observed depending on the specific task. An example for this is the limitation of the Cartesian working space or the limitation of a maximum Cartesian speed. These constraints are summarized to

$$\boldsymbol{p}(\boldsymbol{q}) \in [\boldsymbol{p}^{-}, \boldsymbol{p}^{+}]\,, \tag{19a}$$

$$\dot{\boldsymbol{p}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in [\dot{\boldsymbol{p}}^{-}, \dot{\boldsymbol{p}}^{+}]\,. \tag{19b}$$

Note that the constraints (19) are now nonlinear with respect to the states due to the kinematic transformation.

3.5 Cost function design

The cost function of the optimization problem (12a) depends on the actual control task. By penalizing the deviation from a desired behavior, the optimization problem tries to minimize these deviations. A common choice is to penalize the weighted norm of a quantity $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_{\text{des}}$ according to

$$\|\tilde{\boldsymbol{x}}\|_{\boldsymbol{A}}^2 = \frac{1}{2}\tilde{\boldsymbol{x}}^{\text{T}}\boldsymbol{A}\tilde{\boldsymbol{x}} \tag{20}$$

with the positive (semi)-definite weighting matrix $\boldsymbol{A}$. The cost function now consists of the respective penalization terms

$$\begin{aligned} l(\boldsymbol{x}, \boldsymbol{u}) = l_{\text{pos}}(\boldsymbol{q}) + l_{\text{vel}}(\dot{\boldsymbol{q}}) + l_{\text{torque}}(\boldsymbol{\tau}_{\text{ext}}) \\ + l_{\text{force}}(\boldsymbol{\mathcal{F}}_{\text{ext}}) + l_{\text{ctr}}(\boldsymbol{u})\,. \end{aligned} \tag{21}$$

Note that the choice of the individual penalty terms will depend on the respective application and can therefore be set also to zero. Some example applications are presented in Section 4.

3.6 Numerical solution

The optimization problem (12) is solved with the GRAMPC toolbox (Englert et al., 2019). The toolbox is based on an efficient implementation to enable embedded real-time optimization problems with constraints using an augmented Lagrangian formulation. The idea of the method is to consider the dual system using the augmented Lagrangian function instead of the original problem (12). In order to solve the problem, the augmented Lagrangian function is sequentially maximized with respect to the Lagrangian multipliers and minimized using a projected gradient method with respect to the controls.

An important part of the optimization is the numerical integration of the system dynamics (12b). The problem of solving the resulting motion with respect to the applied generalized forces is solved by the forward dynamics

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}(\boldsymbol{q})^{-1}\left(\boldsymbol{\tau}_{\text{J}} + \boldsymbol{\tau}_{\text{ext}} - \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q})\right), \tag{22}$$

where the joint accelerations are determined with respect to the joint torque input $\boldsymbol{\tau}_{\text{J}}$ and the external torque $\boldsymbol{\tau}_{\text{ext}}$. Due to the sparsity of $\boldsymbol{M}(\boldsymbol{q})$ and $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ as well as the inversion of the position dependent joint space inertia matrix $\boldsymbol{M}(\boldsymbol{q})$, the closed form solution of the forward dynamic of a six degree of freedom (DOF) robot is computationally inefficient. Instead, this problem is solved by computationally efficient recursive forward dynamic algorithm

$$\ddot{\boldsymbol{q}} = \text{FD}(\mathit{model}, \boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau}_{\text{J}}, \boldsymbol{\mathcal{F}}_{\text{e}}) \tag{23}$$

based on the Newton-Euler equations. A suitable algorithm for this purpose is the computational efficient Articulated Body Algorithm (ABA) (Featherstone, 2008). The algorithm carries out a three-pass recursion. In the first forward pass, the kinematic quantities are determined for

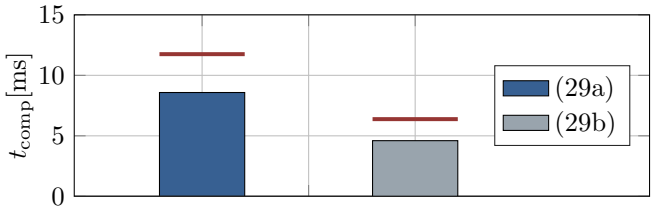

Fig. 3. Computation times for the MPC motion control with different robot dynamic formulations (29).

the next recursions. The next backward pass calculates the spatial forces acting on the N bodies. The last forward recursion determines the spatial body accelerations, to get the joint accelerations. The main advantage is that the algorithm does not depend on the joint space inertia matrix $\boldsymbol{M}(\boldsymbol{q})$. Instead, the constant (3×3) body inertia matrices of the individual links will be used within the algorithm. This will lead to an algorithmic complexity of $\mathcal{O}(N)$ which is linear in the number of robot joints.

As discussed above, the optimization is based on a gradient method which evaluates the first order optimality conditions. The Jacobian of the system dynamic, the equality and inequality conditions as well as the cost function with respect to the states and controls are required for this purpose. One possibility for computing the partial derivatives of the forward dynamics is via finite differences or automatic differentiation as used, for example, in Giftthaler et al. (2017). The analytical calculation of derivatives of the rigid body algorithms for inverse and forward dynamics were introduced by Carpentier and Mansard (2018). It is used that the partial derivatives of the calculated quantities in the recursion steps can be calculated directly from the partial derivatives of the kinematic quantities. To this end, the approach relies on the partial derivatives of the individual calculations within the recursions in the forward and backward passes. Thus, the partial derivatives of the dynamics can also be calculated using a recursive algorithm, which is built up in forwards and backwards passes over the kinematic tree. An implementation of the analytical calculation of the derivatives is given by the C++ framework Pinocchio (Carpentier et al., 2019).

4. APPLICATIONS

This section discusses the use of the MPIC for three applications, a pure motion control, a hand-guiding of the robot and a hybrid force motion scenario. In the first scenario the algorithmic effort of an MPC using the full rigid body dynamic (22) shall be discussed. Furthermore, the use of the torque dynamic (14) as well as the force dynamic (16) is dealt with by one application each.

The applications were implemented at a light weight robot arm with 6 joints. A six degree of freedom (DOF) force/torque sensor was mounted on the tip. However, the approach is not limited to measured force/torques. Estimated quantities can also be used, e.g. following the approach of Gold et al. (2019). In all cases, the control was running with a sampling time of 0.01 s.

4.1 Motion control

The first application deals with the computational effort of the MPC for different levels of detail of the robot dynamics. For the description of the robot system, the state space

$$\boldsymbol{x}_{\text{mc}} = \begin{bmatrix} \boldsymbol{q} \\ \dot{\boldsymbol{q}} \end{bmatrix} \tag{24}$$

is spanned by the joint position and velocity and the input of the rigid body system will be the transmitted joint torque $\boldsymbol{\tau}_{\text{J}}$. Since in many systems the joint torque cannot be specified directly and instead the motor torque $\boldsymbol{u}_{\text{mc}} = \boldsymbol{\tau}_{\text{m}}$ serves as the input, the friction in the motor and transmission must also be taken into account. This leads to a joint torque approximation according to

$$\boldsymbol{\tau}_{\text{J}} \approx \boldsymbol{\tau}_{\text{m}} - \boldsymbol{\tau}_{\text{f}}(\dot{\boldsymbol{q}}) \,, \tag{25}$$

whereby the friction is dominated by Coulomb and viscous friction. The Coulomb friction is thereby approximated using a tanh function. This allows to define the dynamic model of motion control as

$$\dot{\boldsymbol{x}}_{\text{mc}} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} \,. \tag{26}$$

The forward dynamics $\ddot{\boldsymbol{q}} = FD(\cdot)$ are calculated according to Section 3.6 using the toolbox Pinocchio.

Due to the mechanical structure of the robot, the admissible position space of each joint is constrained. Also for mechanical reasons, the maximum velocity of the joint is limited. These restrictions must be taken into account in the optimization problem using the box constraint (18). In addition, the absolute maximum motor torque is also bounded, which is considered by the input constraint

$$\boldsymbol{\tau}_{\text{m}} \in [\boldsymbol{\tau}_{\text{m}}^{-}, \boldsymbol{\tau}_{\text{m}}^{+}] \,. \tag{27}$$

The cost function for joint space motion control

$$l_{\text{mc}}(\boldsymbol{x}, \boldsymbol{u}) = \|\tilde{\boldsymbol{q}}\|^2_{\boldsymbol{Q}_q} + \|\tilde{\dot{\boldsymbol{q}}}\|^2_{\boldsymbol{Q}_{\dot{q}}} + \|\tilde{\boldsymbol{u}}_{\text{mc}}\|^2_{\boldsymbol{R}} \tag{28}$$

contains parts for the states and the control. With a penalization of a position or a velocity deviation $\tilde{\boldsymbol{q}} = \boldsymbol{q} - \boldsymbol{q}_{\text{des}}$ / $\tilde{\dot{\boldsymbol{q}}} = \dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_{\text{des}}$, the optimization problem aims to achieve a desired position respective velocity as accurately as possible. The penalization of the control variable influences the aggressiveness of the control. The parameters used at the motion control application are summarized in Table 1.

Table 1. Optimization parameters for the motion control application.

prediction horizon	T	0.5 s
sampling points	N_{hor}	50
max. gradient iterations	i_{grad}	5
max. multiplier iterations	i_{mult}	1
position weights	$\boldsymbol{Q}_q$	$\text{diag}([0, \ldots, 0])$
velocity weights	$\boldsymbol{Q}_{\dot{q}}$	$\text{diag}([10^5, \ldots, 10^5])$
motor torque weights	$\boldsymbol{R}$	$\text{diag}([10^{-2}, \ldots, 10^{-2}])$

Two levels of detail are considered for the robot dynamics

$$\ddot{\boldsymbol{q}}_{\text{full}} = \boldsymbol{M}(\boldsymbol{q})^{-1} \left(\boldsymbol{\tau}_{\text{J}} - \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) \right) \,, \tag{29a}$$

$$\ddot{\boldsymbol{q}}_{\text{simpl}} = \boldsymbol{M}_{\text{J}}^{-1}(\boldsymbol{\tau}_{\text{J}} - \boldsymbol{g}(\boldsymbol{q})) \,, \tag{29b}$$

for the further computation time comparison in Figure 3. To this end, a scenario with a four times operating point

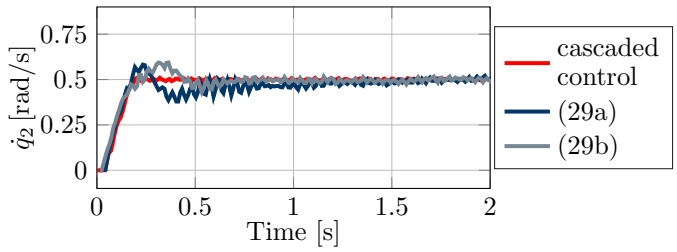

Fig. 4. Velocity step responses at joint 2 of the cascaded independent joint control and the MPC with robot model (29a) and (29b).

change $\dot{\boldsymbol{q}}_{\text{des}}$ over 20 s without an interaction $\boldsymbol{\mathcal{F}}_{\text{e}} = \boldsymbol{0}$ was executed at the robot. The controllers were computed on a Ubuntu 18.04 OS with Intel(R) Core(TM) i5-8250U CPU. First, the full rigid body dynamics (29a) with its implementation by the ABA is used. The average computing time is 8.1 ms which is within the sampling time of 10 ms. However, it can be seen that a maximum computing time of 11.7 ms was required, so that the calculation was no longer within the required sampling time. The second dynamics formulation (29b) achieves a significant gain in computing time. The reason for this is that instead of the configuration-dependent mass matrix $\boldsymbol{M}(\boldsymbol{q})$ of the rigid body, only the constant joint inertia $\boldsymbol{M}_{\text{J}}$ were used. The gravitational vector was calculated using the recursive Newton-Euler algorithm (RNEA). The calculation time is reduced to almost half of 4.6 s.

The computation time comparison shows that a model-predictive control is possible under consideration of the RBD. However, the maximum computing time of (29a) shows that if the maximum available computing time is violated, relapse strategies must prevail. For example, the (sub)optimal solution from the last calculation step may be used for sporadic violations. Instead of the first control variable, the next one is given to the system.

The step response of the velocity of both dynamic expansion stages are compared to a cascaded independent joint controller. The outer velocity control loop is sampled with 10kHz. The step responses can be seen in Figure 4. The rise time of the step responses is for all three curves in the range of 140 ms. However, both trajectories of the MPC show an overshoot which has stabilized after further 0.5 s. Furthermore, a high oscillation is recognizable in all three signals. Ripples that originate from the Harmonic Drive transmission are the reason for this. The better quality of the independent joint control trajectory is mainly attributable to the significantly higher sampling rate. In addition, it is visible that the quality of the step response for (29a) is worse than for (29b). This is due to the fact that the optimization of (29a) with configuration-dependent mass matrix $\boldsymbol{M}(\boldsymbol{q})$ converges more slowly than for the constant inertia matrix $\boldsymbol{M}_{\text{J}}$ in (29b). Because of the relatively large joint inertia, the mass matrix is quasi linearized, whereby the rigid body inertia has only a small influence.

It is obvious that the additional computing effort is only worthwhile if the modeled effects are also excited in operation. Especially the inertia effects of the rigid body occur mainly with a high dynamic impact. This means that the

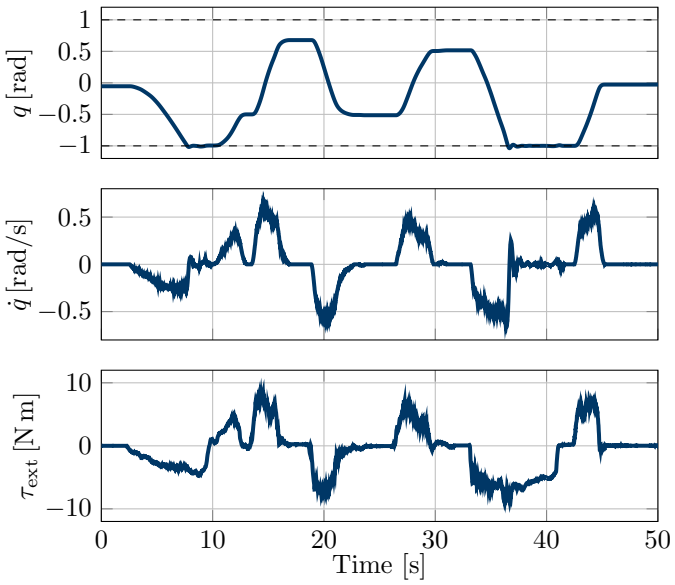

Fig. 5. Plots of the trajectories q_1, $\dot{q}_1$ and τ_{ext1} of joint 1 in a hand-guiding scenario.

consideration of the complete model is only advantageous for highly dynamic applications. In the following, procedures for the physical robot environmental interaction are presented. These methods are mainly operated in a slow dynamic range, so that the independent joint control is a good choice.

4.2 Hand-guiding

The application of hand-guiding is an important skill for modern light weight robots. Teaching of the robots is more and more done by guiding the robot to the desired pose. To this end, the robot has to detect an external force and to react to the force with an evasive motion. The control should be able to minimize the external torques. This leads to the extension of the state vector

$$\boldsymbol{x}_{\text{hg}} = \begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{\tau}_{\text{ext}} \end{bmatrix}, \quad \boldsymbol{u}_{\text{hg}} = \dot{\boldsymbol{q}}\,. \tag{30}$$

by the external moments. The hand-guiding is operated in a slow dynamic range with an underlying velocity control as described above, whereby the robot is modeled by an integrator with the velocity as input. Referred to (30), the dynamic model

$$\dot{\boldsymbol{x}}_{\text{hg}} = \begin{bmatrix} \boldsymbol{u}_{\text{hg}} \\ \boldsymbol{K}_{\text{J}}\boldsymbol{u}_{\text{hg}} \end{bmatrix} \tag{31}$$

consists of the robot and torque dynamics. The quantities of joint position and velocity are also limited by the system in the hand-guiding case according to (18a) and (18b). These limits are considered in the optimization problem using box constraints for (12c).

The control task can be understood as a minimization of the unmodeled external torques. Modeled torques are for example torques resulting from a desired payload at the end-effector. This leads to the following cost function design

$$l_{\text{hg}}(\boldsymbol{x}, \boldsymbol{u}) = \|\tilde{\boldsymbol{\tau}}_{\text{ext}}\|^2_{\boldsymbol{Q}_{\tau_{\text{ext}}}} + \|\boldsymbol{u}_{\text{hg}}\|^2_{\boldsymbol{R}}\,, \tag{32}$$

where the desired external torques are calculated according to

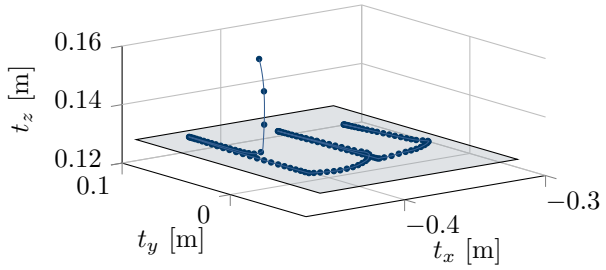

Fig. 6. Path plot of the end-effector during the board cleaning application.

$$\boldsymbol{\tau}_{\text{ext des}} = \boldsymbol{J}(\boldsymbol{q})^{\mathrm{T}} \boldsymbol{F}_{\text{payload}} \tag{33}$$

with the modeled payload forces. In Table 2, the optimization parameters are given for the hand-guiding case.

Figure 5 shows the trajectories of the joint position, velocity and the external moment for joint 1 in a hand-guiding scenario. The proportional relationship (14) between the external torque and the resulting velocity is clearly visible. The position constraint is satisfied as shown in Figure 5. The comparison of velocity and external torque shows the influence of the constraint on the velocity response. In the area of an active bound, e.g. at approx. 7 s or 36 s, the external torques do not result in a velocity, which is visible by the divergent qualitative trajectories in these areas.

Table 2. Optimization parameters for the hand-guiding application.

prediction horizon	T	0.5 s
sampling points	N_{hor}	50
max. gradient iterations	i_{grad}	2
max. multiplier iterations	i_{mult}	2
external torque weights	$\boldsymbol{Q}_\tau$	$\text{diag}([100, 100, 100, 10, 1, 1])$
motor torque weights	$\boldsymbol{R}$	$\text{diag}([10^0, \ldots, 10^0])$

4.3 Hybrid motion/force control

The aim of a hybrid motion/force control is the use of different control strategies for complementary directions. On the one hand, desired Cartesian directions are motion controlled for example with position or velocity control. The complementary directions are force controlled.

This task leads to a state space which contains the robot states as well as the Cartesian force/moment states for describing the interaction. To this end, the following state space

$$\boldsymbol{x}_{\text{hmf}} = \begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{\mathcal{F}}_{\text{ext}} \end{bmatrix}, \quad \boldsymbol{u}_{\text{hmf}} = \dot{\boldsymbol{q}} \tag{34}$$

is defined. The joint velocity is still the control variable for this task. Since the interaction takes place in Cartesian space, the interaction is also modeled in the same space

$$\dot{\boldsymbol{x}}_{\text{hmf}} = \begin{bmatrix} \boldsymbol{u}_{\text{hmf}} \\ \boldsymbol{K}_{\text{e}} \boldsymbol{J}(\boldsymbol{q}) \boldsymbol{u}_{\text{hmf}} \end{bmatrix}. \tag{35}$$

As with the previous two tasks, the robot quantities are limited as well. This means that (18a) and (18b) hold also for this task. The task specific constraints (19) can be defined for the hybrid motion/force control too, but are not considered in this case.

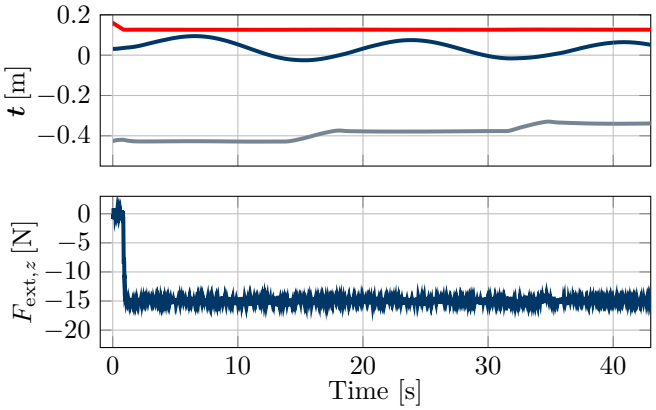

Fig. 7. Trajectories of the Cartesian end-effector positions (t_x, t_y and t_z) and force in z direction during the board cleaning application.

In the cost function the actual task is defined, namely the Cartesian position control or the force control of respective directions. This can be achieved by choosing complementary entries of the weighting matrices $\boldsymbol{Q}_p$ and $\boldsymbol{Q}_{\mathcal{F}}$ in the cost function

$$l_{\text{hmf}}(\boldsymbol{x}, \boldsymbol{u}) = \|\tilde{\boldsymbol{p}}\|^2_{\boldsymbol{Q}_p} + \|\tilde{\boldsymbol{\mathcal{F}}}_{\text{ext}}\|^2_{\boldsymbol{Q}_{\mathcal{F}}} + \|\boldsymbol{u}_{\text{hmf}}\|^2_{\boldsymbol{R}}. \tag{36}$$

The motion/force control is used for a board cleaning application. To this end, a sponge is mounted on the end-effector. The aim is now to apply a force of 15N on the sponge to guarantee sufficient contact with the surface. The z direction is force controlled for this purpose and the wiping movement is carried out with a position controlled direction. In our case the y direction. A compliance control is applied to the last direction, the x direction, to change the linear wiping movement path by a physical human input. The direction is therefore force controlled with a desired force of zero and a constant virtual stiffness value $K_{\text{e},x}$ which is a parametric degree of freedom. The parameters for optimization are given in Table 3.

Table 3. Optimization parameters for the hybrid motion/force control application.

prediction horizon	T	0.5 s
sampling points	N_{hor}	50
max. gradient iterations	i_{grad}	3
max. multiplier iterations	i_{mult}	2
position weights	$\boldsymbol{Q}_p$	$\text{diag}([0, 10^3, 0, 10^3, 10^3, 10^3])$
interaction force weights	$\boldsymbol{Q}_{\mathcal{F}}$	$\text{diag}([0.01, 0, 0.03, 0, 0, 0])$
motor torque weights	$\boldsymbol{R}$	$\text{diag}([10, \ldots, 10])$

In Figure 6 the path of the end-effector is pictured in 3D space. The robot starts with a free motion in z direction at the point $(x, y, z) = (-0.43, 0.03, 0.16)$ with a velocity of 40 mm/s. The corresponding trajectories of the position and the z direction force are shown in Figure 7. The free motion is visible by the zero force at the beginning. The contact occurs after approx. 0.8 s with a rapid increase of the interaction force. The desired target value is reached after another approx. 0.3 s and is kept during the entire interaction. The specified wiping movement in the y direction can be seen in the corresponding Cartesian position. The compliance control in the x direction is visible at 15 s and 33 s. In this case, a force was applied in the x direction,

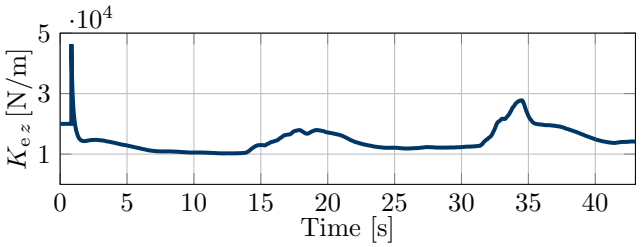

Fig. 8. Online estimation of the interaction stiffness.

which results in a corresponding motion in this direction. Figure 6 shows this motion by shifting the wiping motion in the x direction.

In order to predict the interaction force according to (16), the environmental stiffness was estimated in the z direction by the online method described in Section 2.2. The trajectory of the estimation may be seen in Figure 8, where the stiffness estimation was initialized with a start estimate of 20000 N/m. It is advisable to initialize the stiffness too high, which leads to a more sensitive behavior at the beginning of the contact. This will cause the robot to slow down faster. The trajectory in Figure 8 shows a peak in the initial phase of the contact. In the transient case of the interaction, inertia and dissipative effects occur in addition to the modeled spring forces. This leads to an increased force compared to the spring model, which overestimates the stiffness in this phase. However, this does not represent a disadvantage for practical applications, since the initial overestimation leads to a faster deceleration of the motion. As a result, negative force peaks at the beginning of the contact do not occur or only occur to a reduced extent. Subsequently, the estimate is leveled out at around 10000 N m. The human input for shifting the wiping movement in x direction at 15 and 33 s leads to interference forces in the z direction. These can also be seen in the stiffness estimation due to the temporary overestimation.

5. CONCLUSION

Model predictive control (MPC) is a powerful tool for the control of bounded nonlinear systems. Industrial robot applications are an important field that falls into this system class. The paper shows, that the usage of MPC provides some improvements especially for robot-environment interaction. To this end, a generic framework for interaction control tasks was presented. The model predictive interaction control (MPIC) scheme is for example applicable for classical motion control, compliance control, direct force control as well as hybrid forms. It has been shown that the application of MPIC for force control on materials with elastic behavior shows good control performance. The paper also points out, that a MPC motion control using the full rigid body dynamics is possible in case of a 6 DOF robot but shows less advantages in the normal dynamic range.

The next steps will be the application of model predictive control for highly dynamic tasks. In addition, the methodology of MPIC will be extended to the case of interaction with a rigid environment. A further area of research is the environmental modeling from past contact cases.

REFERENCES

Carpentier, J. and Mansard, N. (2018). Analytical Derivatives of Rigid Body Dynamics Algorithms. In *Robotics: Science and Systems*.

Carpentier, J., Saurel, G., Buondonno, G., et al. (2019). The pinocchio C++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In *Proc. of SII*, 614–619.

Englert, T., Völz, A., Mesmer, F., Rhein, S., and Graichen, K. (2019). A software framework for embedded nonlinear model predictive control using a gradient-based augmented Lagrangian approach (GRAMPC). *Optimization and Engineering*, 20(3), 769–809.

Featherstone, R. (2008). *Rigid Body Dynamics Algorithms*. Springer, New York.

Giftthaler, M., Neunert, M., Stäuble, M., Frigerio, M., Semini, C., and Buchli, J. (2017). Automatic differentiation of rigid body dynamics for optimal control and estimation. *Advanced Robotics*, 31, 1225–1237.

Gold, T., Völz, A., and Graichen, K. (2019). External torque estimation for an industrial robot arm using joint torsion and motor current measurements. In *Proc. of Joint Conference on MECHATRONICS and NOLCOS*, 879–884.

Jung, S., Hsia, T.C., and Bonitz, R.G. (2001). Force tracking impedance control for robot manipulators with an unknown environment: Theory, simulation, and experiment. *International Journal of Robotics Research*, 20(9), 765–774.

Kazim, K.J., Bethge, J., Matschke, J., and Findeisen, R. (2018). Combined predictive path following and admittance control. In *Proc. of ACC*, 3153–3158.

Koenemann, J., Del Prete, A., Tassa, Y., et al. (2015). Whole-body model-predictive control applied to the HRP-2 humanoid. In *Proc. of IROS*, 3346–3351.

Lee, S.H., Kim, J., Park, F.C., Kim, M., and Bobrow, J.E. (2005). Newton-type algorithms for dynamics-based robot movement optimization. *Transactions on Robotics*, 21(4), 657–667.

Matschke, J., Bethge, J., Zometa, P., , and Findeisen, R. (2017). Force feedback and pathfollowing using predictive control: Concept and application to a lightweight robot. In *Proc. of IFAC World Congress*, 10243–10248.

Mordatch, I., Todorov, E., and Popović, Z. (2012). Discovery of complex behaviors through contact-invariant optimization. *Transactions on Graphics*, 31(4), 43:1–43:8.

Schultz, G. and Mombaur, K. (2010). Modeling and optimal control of human-like running. *Transactions on Mechatronics*, 15, 783–792.

Spong, M., Hutchinson, S., and Vidyasagar, M. (2005). *Robot Modeling and Control*. Wiley, New York.

Verschueren, R., van Duijkeren, N., Swevers, J., and Diehl, M. (2016). Time-optimal motion planning for n-DOF robot manipulators using a path-parametric system reformulation. In *Proc. of ACC*, 2092–2097.

Wahrburg, A. and Listmann, K. (2016). MPC-based admittance control for robotic manipulators. In *Proc. of CDC*, 7548–7554.

Zhao, J., Diel, M., and Longman, R. (2004). Nonlinear model predictive control of robots using real-time optimization. In *Proc. of AIAA/AAS*, 1–18.