Institute of Parallel and Distributed Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Master Thesis

Approximate constrained model predictive control of compliant motion involving contact

Patric Skalecki

Course of Study: Technische Kybernetik

Examiner: Prof. Dr. Marc Toussaint

Supervisor: Prof. Dr. Marc Toussaint

Commenced: November 2, 2016

Completed: May 2, 2017

CR-Classification: I.2.8, G.1.6

Abstract

The goal of this work is to investigate an approach to model predictive control in constrained settings, where the motion involves constraints and compliant motion is desired. Specifically, the fixed Lagrangian should be evaluated on a simulated two link serial robot model performing a motion that involves sliding.

For this task, it is needed to setup a simulation environment for the modeled system, to implement the control strategy and to analyze this control strategy with respect to the stiffness along the planned trajectory. Further on, there will be a theoretical investigation of the closed-loop system regarding stability. As an advanced test scenario a position error in the sliding surface is introduced. Hence, only position control will fail, since either the system could be damaged due to a *high-gain feedback* behavior or the contact and thus the sliding motion will not occur. Therefore, it is investigated whether it is possible to extend the control structure in order to ensure that the sliding motion takes place as desired.

Zusammenfassung

Das Ziel dieser Arbeit ist es, einen Ansatz zur modellprädiktiven Regelung in beschränkten Szenarien, bei welchen die Bewegung beschränkt ist und weiche Bewegungen gewünscht sind, zu untersuchen. Im Speziellen soll die feste Lagrange-Funktion an einem zweisegmentigem seriellem Roboter, welcher eine gleitende Bewegung durchführt, untersucht werden.

Für diese Aufgabe ist es notwendig eine Simulationsumgebung für das modellierte System aufzusetzen, die Regelstrategie zu implementieren und das Analysieren dieser Regelstrategie bezüglich der Steifheit entlang der geplanten Trajektorie. Weitergehend wird es eine theoretische Untersuchung des geschlossenen Kreises hinsichtlich der Stabilität geben. Als weiterführendes Testszenario wird ein Positionsfehler in der Oberfläche, welche zum Gleiten genutzt werden soll, eingeführt. Daher wird eine reine Positionsregelung fehlschlagen, weil entweder das System durch ein *high-gain feedback* Verhalten beschädigt werden könnte oder der Kontakt und somit die gleitende Bewegung nicht eintreten wird. Daher wird untersucht, ob es möglich ist die Regelstruktur so zu erweitern, dass die gleitende Bewegung wie gewünscht stattfindet.

Contents

List of Figures

1 Introduction

Control in robotics for trajectory tracking is well known nowadays. In industry, there exist very precise robots for fabrication tasks and several other purposes. But when the planned trajectory has contact involved, the control strategies used in industry are really stiff. This means, if the contact point is not precisely known then these robots would either not go in contact because the planned trajectory stays above or would force the end-effector into the object. Such a behavior is often undesired, e.g. for tasks like object interaction with soft/ fragile objects. Especially, this becomes more and more important for tasks, where robots and humans interact. There it is really important, that the robot can be operated safely without the *fear* that it tries to interpenetrate the human, even if the human is not where the robot expects him to be. Another very challenging task is legged motion, where it is essential to hit the contact point in a manner that the robot does not fall or fails to complete the planned task. There it might be impossible to generate a precise preplaned trajectory if the terrain is changing or can't be observed precisely due to perception issues like the lack of high precision sensors or malfunction. Thus, compliant motion control is a challenging task in modern robotics and is still an active research field.

One simple and often used approach to this problem is to heuristically design controllers based on motion primitives. This can be achieved by trying to reproduce natural behavior motions. A simple example scenario is a end-effector of a robotic arm which should slide on a table. There a simple motion primitive is sense if the contact takes place at the desired time and if not move the end-effector down until the contact takes place. Then execute the rest of the task. But this scheme requires often expert knowledge and does not generalize to different tasks once a controller is designed for a specific task.

As a more generic approach the problem can also be viewed from the trajectory optimization side. Here it is well known how to design objectives for planning a trajectory around obstacles for various tasks. But it is clear that a position controller alone on the trajectory is not able to execute the desired task, if compliant motion is required. For instance a PID position controller stabilizing the error between robot position and reference trajectory would result in the behavior mentioned above, thus it would not be possible to perform a compliant execution of the motion. Due to the close relation between trajectory optimization and optimal control, it is easy to transfer the objective into an optimal control problem and hence design a control trajectory. But this is still an open-loop controller not using the information for any sensors or state estimates. The *naive* idea of a complete replanning of the whole task trajectory is too costly and thus infeasible to execute in the real time loop. The central question is how to use the new information and execute the task in an approximate optimal way.

In the control theory community there exists a control scheme called *model predictive control*, short MPC. This scheme can handle nonlinear system dynamics and constraints explicitly. Here a short horizon optimal control problem is solved in every time step and then only the first time step is applied to the real system. Then this procedure repeats iteratively. Through the insights on trajectory optimization it is easier to define meaningful objectives. Usually the MPC scheme is an approximation or overestimation of the underlying optimal control problem with infinite horizon. Therefore my supervisor proposed in [Tou16] an approach to MPC based on the Bellman's principle and quadratic approximation of the fixed Lagrangian for a underlying fixed horizon optimal control problem. This scheme will be derived in more detail later.

In this work I will derive an algorithm to apply the *approximate constraint model predictive control* scheme and evaluate this method on two numerical examples, a simple trajectory optimization between two points and a two link serial robot. The purpose of the first example is getting insights into the control scheme and the latter is to evaluate the physical behavior in the contact phase. The special interest of the evaluation is to judge how compliant the resulting behavior is. Especially if some constraints are uncertain or even unknown.

In the next chapter I will evaluate some of the related work, which motivated us to look deeper into the topic. There, I want to point out different approaches how controllers are designed to handle contact situations for communities with different backgrounds. In chapter 3 the basics on efficient trajectory optimization using the k-order constrained motion optimization framework and on standard model predictive control will be derived. Chapter 4 presents a derivation of approximate constraint model predictive control. Here the cost-to-go approximation is explained in detail and a design scheme for the application of the control strategy is presented. The evaluation of the approximate constraint MPC controller on two simulation examples with special focus on the compliance in contact situation will be in chapter 5. Chapter 6 discusses the results of the evaluation, alternative control structures and the gained insights through the work. In the end a conclusion will be given.

2 Related Work

Here some of the work motivated us to look deeper into the topic is presented. Also I want to give a brief overview of the different approaches how people with different backgrounds tackle the control problem involving contact. In Figure 2.1 the different types of approaches to this task are shown. These domains are not directly separated and often share common ideas which is indicated by the intersection. In the following, some elected approaches are reviewed and their main targets and drawbacks are pointed out.

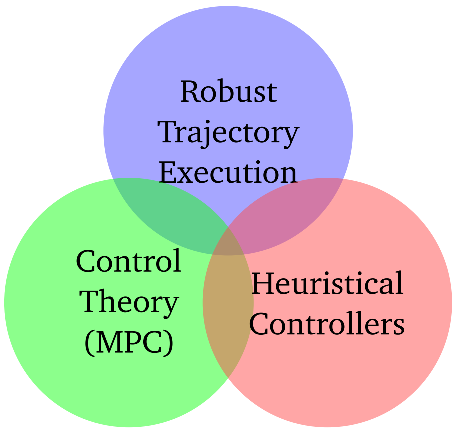


Figure 2.1: Pictorial representation of the different approaches for designing a controller.

First, consider an approach for robust trajectory execution. One example for this is the DIRCON-Algorithm proposed by [PKT16]. This Algorithm is evaluated on the ATLAS robot in a walking and climbing task. The authors consider a constrained Lagrangian system $\dot{x}(t) = f(x(t), u(t))$ with the constraint $\Phi(x(t)) = 0$, where $x(t)$ describes the position and velocity of the robot in state space. The design phase for the overall task follows in three steps:

1. *Constrained Direct Collocation*
   A standard optimal control problem is solved via direct collocation for generating the optimal state-control trajectories.

2. *Design an equality-constrained LQR*
   A time-varying linear system constrained in the manifold $F(t)x(t) = 0$ based on the linearization around the trajectory form step 1. is projected via a transformation matrix $P(t)$ into an unconstrained time-varying linear system. Then a LQR is designed for this unconstrained system.

3. *QP Feedback Controller*
   Based on the solution of the time-varying LQR and the Hamilton-Jacobi Bellman equation the cost for the QP is derived.

Step 1. and 2. can be computed offline and only step 3. needs to be executed in real time. The QP Feedback Controller is designed to explicitly handle input constraints and compensate friction. Also minor variations in the planned contact time can be handled. It also incorporates a no-slip condition. Thus the main target of the authors is to design a generic design scheme for executing a preplanned trajectory in situations where the contact point is known precisely. Hence the main disadvantage is that the environment needs to be known precisely when planning the trajectory in order to have the precise knowledge when and where the contact takes place and to be able to execute the trajectory.

In the second domain of approaches, the Heuristical Controllers, there are controllers which are designed to also execute a motion but not necessary stiff around the trajectory. This means, that the trajectory is an *optimal* plan for an uncertain environment and the target is to reproduce the contact profile to achieve the task. Usually, here are controllers which try to mimic a specific information seeking behavior. An interesting approach can be found in [TRB+14]. There, the authors design a switching controller between a position and force controller depending whether the desired contact takes place or not. If mismatches between knowledge of the environment and the real one occur, they mimic a *natural* behavior. Specifically this means, that if no contact takes place (e.g. no contact force is measured) at a time where the planned trajectory commands to be in contact, then a PD behavior is designed, which pulls the end-effector in the constraint along the constraint gradient direction. Also the task space trajectory is used as a feedforward control, which creates only movement orthogonal to the constraint gradient. In the case where contact is sensed and no contact is desired, a PD behavior, similar to the previous case, is designed pushing the end-effector away from the constraint. The two remaining cases are the application of the force or position controller if contact or not is desired and takes place, respectively. This switching behavior of the dual-execution controller is illustrated in Figure 2.2. The authors implemented this controller on a 7-DOF Kuka arm with a 4-DOF Barrett hand. They were able to move a finger to a point on the table. They achieved a good result for different table heights, which are unknown to the controller. But such controllers use an optimal plan for the task and do not necessary result in an optimal behavior (closed-loop trajectory) of the robot. This is due to the

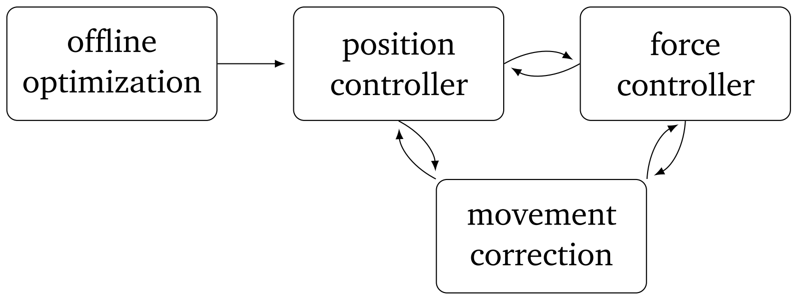


Figure 2.2: Control architecture for the dual execution controller.

movement correction part, which is allowed to change the execution of the optimal plan. Also it is not general how to design the movement correction, since just pulling into the constraint may violate other objectives or constraints. One example could be the pick up of an obstacle. If the obstacle is too far away the robot could fall if only the arm is stretched forward. Then, more complex movement correction strategies are required. Thus expert knowledge is needed to design these movement corrections.

For the control theory domain the focus is on model predictive control, since this scheme is generic and directly handles nonlinearities. This is possible because it is based on a repetitive solution of a small horizon optimal control problem. Thus, the resulting closed-loop trajectories will be near to optimal or at least better than simply designing motion patterns by hand. Rather than explaining this control scheme I refer to section 3.2 or to [MA17], where the general idea of model predictive control is explained in detail. Based on the knowledge from [Mül15], [MA17] and the *approximate constraint model predictive control* scheme proposed by [Tou16] I will discuss the different approaches to MPC. In the MPC community the MPC term refers to the stabilization of a set-point. Here typically the involving cost function needs to be positive definite with respect to this set-point. Then stability follows from Lyapunov when the cost function is used as Lyapunov function and attractivity from Barbalat's lemma. As a relaxation of the standard MPC scheme the positive definiteness property in the cost is dropped. The resulting MPC scheme is known as economic MPC in the control theory literature. In this scheme the closed-loop behavior might not necessary be convergent to a set-point, e.g. if a periodic movement has a lower cost than a stationary set-point. Applications of MPC schemes in robotics are also known. One example is [ELT+13] and [TET12], where the authors designed a MPC controller for a humanoid robot in simulation for different tasks. They use as optimization framework their iterative LQG scheme [TL05] and explain in detail how they create the objectives as a weighted sum of scalar objectives for different tasks. With this MPC scheme they were able to achieve different motions like getting up of the ground or walking as concatenation of smaller movements. For walking this means a repetition of four states right step, right stance, left step and left stance.

For the above stated general MPC schemes, the drawback is that fixed time horizon tasks are not obvious to encode. This is even harder if an economic cost is used and the overall target is not visible in the optimization horizon. Thus the target must be encoded in the terminal cost/ cost-to-go term of the MPC optimization. In the tutorial paper [Tou16] an approximate constrained MPC scheme is proposed which designs a quadratic cost-to-go term based on the Lagrangian of the finite time optimal control problem. This method seems promising and is a different approach to MPC compared to the standard methods for designing a MPC controller. Here a finite time optimal control problem is solved offline and afterwards a controller tailored to execute this task is generically designed. Further informations on how this is achieved are given in chapter 4.

3 Background

This chapter presents the background used in this work. Here a framework for efficient path optimization is explained. Furthermore a relation to optimal control is shown. As second topic the standard model predictive control approach is presented. The extensions to the economic MPC scheme are also shown. This scheme is similar to the method presented in chapter 4.

3.1 k-order Constrained Path Optimization

For model predictive control it is necessary to deal with fast optimization algorithms. Therefore it is important to create structured optimization problems. For this purpose consider the k-order constrained path optimization framework.

A path/ trajectory x consists of T time steps for n dimensional position x_t in joint space $X \subseteq \mathbb{R}^n$. Hence the path x is a $n \times T$ dimensional optimization variable over the domain $\mathbb{R}$. This leads to a general nonlinear program

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & g(x) \le 0 \\ & h(x) = 0, \end{aligned}$$

where $f : \mathbb{R}^{n\times T} \mapsto \mathbb{R}$ is the objective and $g : \mathbb{R}^{n\times T} \mapsto \mathbb{R}^{d_g}$ and $h : \mathbb{R}^{n\times T} \mapsto \mathbb{R}^{d_h}$ are the inequality and equality constraints, respectively. Further, let all functions be sufficiently smooth, in order to use a Newton method for optimization.

To introduce structure in the nonlinear program we assume the k-order Markov assumption

$$\begin{aligned} f(x) &= \sum_{t=0}^{T-1} f_t(x_{t-k:t}) \\ g(x) &= \bigotimes_{t=0}^{T-1} g_t(x_{t-k:t}) \\ h(x) &= \bigotimes_{t=0}^{T-1} h_t(x_{t-k:t}), \end{aligned}$$

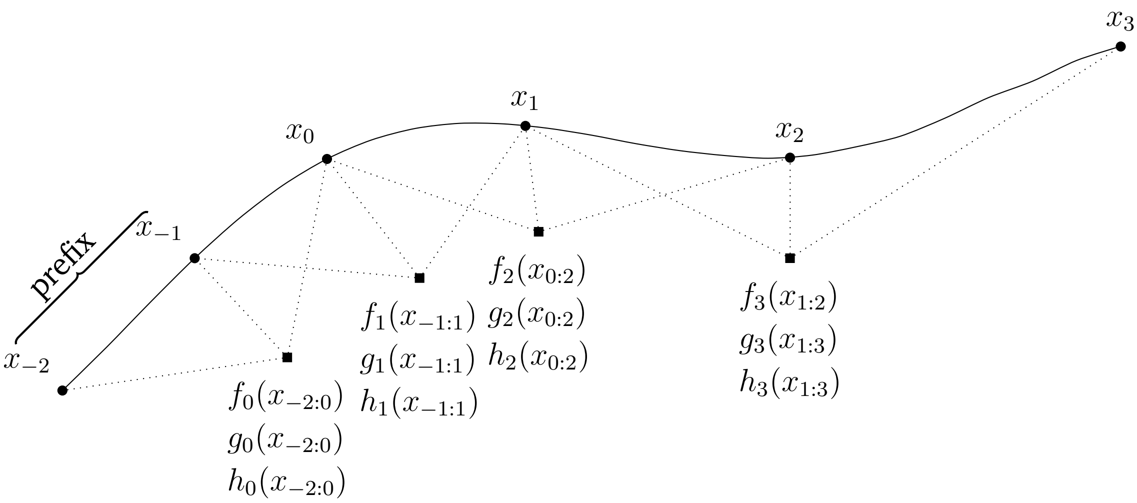


Figure 3.1: Visualization of the structure for a 2-order constrained path optimization problem.

Adapted form: [Tou16] Figure 1

where $x_{t-k:t}$ is the tuple notation for $\{x_{t-k}, \ldots, x_t\}$ and $\otimes$ denotes that $g(x)$, $h(x)$ consists of stacked constraints $g_t(x_{t-k:t})$, $h_t(x_{t-k:t})$ for every time step t. This also implies that the overall objective $f(x)$ is a sum of objectives for every time step and all functions at a time step t only depend on the current and the last k time steps. In the first k time steps the dependency for f_t leaves the range of the optimization variable. Hence define the prefix $x_{-k:-1}$ and encode there e.g. the initial state for the considered system. The k-order structure is depicted in Figure 3.1 for a 2-order Markov assumption. The dotted line indicates the local dependency of the involving functions at a time step. This leads to the k-order Motion Optimization (KOMO)

$$\begin{aligned} \min_x \quad & \sum_{t=0}^{T-1} f_t(x_{t-k:t}) \\ \text{s. t.} \quad & g_t(x_{t-k:t}) \leq 0 \quad \forall_{t=0}^{T-1} \\ & h_t(x_{t-k:t}) = 0 \quad \forall_{t=0}^{T-1}. \end{aligned}$$

For further details on the KOMO framework consider [Tou14b] or [Tou16].

For such optimization problems there exist efficient solvers, which can handle even larger dimensions (dim$(x) > 50$). These solvers exploit the banded Lagrangian being introduced by the KOMO framework. One such solver is the any time augmented Lagrangian, which uses a row-shifted storage scheme for the gradients and hessians and approximate the hessian of $f(x)$ using the Gauss-Newton approximation for a squared f_t as cost. More information on this solver can be found in [Tou14b] or [Tou14a].

3.1.1 Relation to Optimal Control

According to the lectures [Ebe14] and [Arn16] optimal control problems are usually stated in continuous time, state-space ($q = \begin{pmatrix} x & \dot{x} \end{pmatrix}^\top$ for second order systems like mechanical systems) and use a control input u. An optimal control problem in Lagrange Form is given by

$$\begin{aligned}
\min_{u(t)} \quad & \int_{t_0}^{t_f} f_0(q(t), u(t), t) dt \\
\text{s. t.} \quad & \dot{q}(t) = f(q(t), u(t), t) \quad \forall t \in [t_0, t_f] \\
& g(q(t), u(t), t) \leq 0 \qquad \forall t \in [t_0, t_f] \\
& \Psi(q(t_0), q(t_f)) = 0,
\end{aligned}$$

where $f_0 : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R} \mapsto \mathbb{R}$ is a scalar objective, $g : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R} \mapsto \mathbb{R}^{d_g}$ are inequality constraints at every time step t on the trajectory and t_0 and t_f are the initial and end time, respectively. $\Psi : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}^{d_\Psi}$ are the generalized boundary conditions. These could encode the initial and end state. The first constraint is the dynamic equation for the considered system.

For the KOMO framework the initial and end time are fix and use a grid with fixed step size τ on the position. Thus the continuous problem is transformed with finite differences into a discrete optimization problem with reduced optimization variables. Through the dynamic equation the position trajectory x can be mapped on a control trajectory u. For example for mechanical systems

$$u(x_{t-k:t}) = M(x)\frac{x_t - 2x_{t-1} + x_{t-2}}{\tau^2} + F(x, \frac{x_t - x_{t-1}}{\tau})$$

maps the position trajectory x onto a control trajectory u. If not the whole space is reachable, e. g. for box constraints on the control input u or under actuated systems, it is necessary to map these into the position space. The generalized boundary condition Ψ needed to be encoded in the prefix and as constraints on the last time step. Some boundary conditions might be not possible in the KOMO framework. For example a periodic boundary condition

$$\Psi(q(t_0), q(t_f)) = q(t_0) - q(t_f)$$

is not possible to express in this framework, since the condition depends on two time steps which lie not in a $k + 1$ neighborhood.

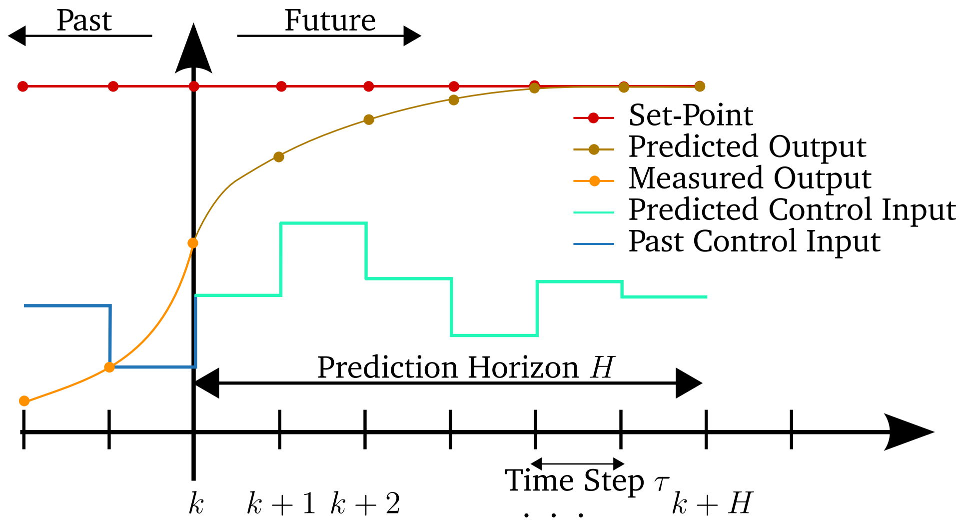


Figure 3.2: Illustration of the MPC scheme.

Adapted form: http://math.stackexchange.com/questions/1098070/model-predictive-control

3.2 Model Predictive Control

Model predictive control is an optimization based control scheme. In the literature it is also often referred to as receding horizon control. The key idea is to repetitively solve a small horizon optimal control problem and apply the first step to the real system. Thus it is easy to explicitly incorporate constraints on the state and control in the control scheme. This idea is for the time step k depicted in Figure 3.2, where H denotes the prediction horizon.

In order to state the MPC algorithm consider the system dynamics as nonlinear difference equation

$$x(t+1) = f(x(t), u(t)), \qquad x(0) = x_0,$$

where $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $x(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^m$ are the system dynamics, state and control input, respectively. The initial condition is given by $x_0 \in \mathbb{R}^n$. At every time step t the state and input constraints are denoted by

$$x(t) \in \mathbb{X},$$
$$u(t) \in \mathbb{U},$$

Algorithm 3.1 Model predictive control iteration

1: $\bar{x}(0) \leftarrow x(t)$	// Measure current state and set initial condition
2: $\bar{x}, \bar{u} \leftarrow$ solve (3.1)-(3.5)	// Solve the optimal control problem
3: $u(t) \leftarrow \bar{u}(0)$	// Apply the first control to the system
4: $i \leftarrow i + 1$ & goto 1	// Increment the iteration counter

where $\mathbb{X} \subseteq \mathbb{R}^n$ and $\mathbb{U} \subseteq \mathbb{R}^m$ are some constraint sets. Then the model predictive control optimization with horizon H at every time step t is given by

$$\min_{\bar{u}} \sum_{k=0}^{H-1} l(\bar{x}(k), \bar{u}(k)) + V^f(\bar{x}(H)) \tag{3.1}$$

$$\text{s.t. } \bar{x}(k+1) = f(\bar{x}(k), \bar{u}(k)) \qquad \forall_{k=0}^{H-1}, \tag{3.2}$$

$$\bar{x}(0) = x(t), \tag{3.3}$$

$$\bar{x}(k) \in \mathbb{X}, \qquad \bar{u}(k) \in \mathbb{U} \qquad \forall_{k=0}^{H-1}, \tag{3.4}$$

$$\bar{x}(H) \in \mathbb{X}^f, \tag{3.5}$$

where quantities with bar $\bar{\cdot}$ denote predicted state and input trajectories at time t. In the objective (3.1) $l : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is referred to as stage cost and $V^f : \mathbb{R}^n \to \mathbb{R}$ is the terminal cost. The constraint (3.2) represents a simulator for the predicted state and the control sequence. (3.3) sets the first state equal to the measurement/ estimation of the state. The sets $\mathbb{X}$ and $\mathbb{U}$ in (3.4) are the state and input constraints at every time step for the predicted trajectories and $\mathbb{X}^f \subseteq \mathbb{X}$ in (3.5) is a terminal set for the last state in the prediction horizon. The application of the model predictive control scheme can be found in Algorithm 3.1.

In the literature there exist several different theoretical results for different assumptions. There are mainly two big domains, the (stabilizing) MPC and economic MPC, where the latter is a generalization of the first domain. For stabilizing MPC the stage cost l, the terminal cost V^f and the terminal set $\mathbb{X}^f$ need to fulfill certain conditions. E.g. the stage cost function l needs to be positive definite w.r.t. a set-point and the terminal cost V^f form a local control-Lyapunov function w.r.t. an auxiliary local control law $u = \pi^{\text{loc}}(x)$ on the terminal set $\mathbb{X}^f$. Details on this specific approach can be found in [MA17] in section 2.1. The economic MPC scheme allows for more general objectives, especially the stage cost function l doesn't need to be positive definite w.r.t. a set-point. The optimal closed-loop behavior might also not be convergent to a set-point. A periodic solution might result in a lower cost and is thus the optimal strategy. For more details on different MPC schemes consider [MA17]. There, a good overview of different schemes is presented and also approaches to distributed MPC schemes are discussed.

4 Approximate Constrained Model Predictive Control

In this chapter the control strategy *approximate constrained model predictive control* will be derived from the k-order Bellman's principle. The derivation will be based on the fixed Lagrangian and a quadratic approximation of the cost-to-go. Special to this control scheme is, that it is designed to handle fixed horizon tasks in an approximate optimal way. Since model predictive control is a control scheme, which is based on optimization in the *real time* loop, the involved optimization problem needs to be structured and thus the solver can exploit this structure. Hence the KOMO framework will be used. For more details on KOMO framework consider section 3.1.

4.1 Model Predictive Control Optimization

The optimization problem at every time step t is

$$\begin{aligned} \pi_t : x_{t-k:t-1} \mapsto \underset{x_{t:t+H-1}}{\operatorname{argmin}} \quad & \left[\sum_{s=t}^{t+H-1} f_s(x_{s-k:s}) + \tilde{J}_{t+H}(x_{t+H-k:t+H-1}) \right. \\ & \left. \qquad + \rho \left|\left| x^*_{t+H-1} - x_{t+H-1} \right|\right|^2 \right] \\ \text{s. t.} \quad & g_s(x_{s-k:s}) \leq 0 \quad \forall_{s=t}^{t+H-1} \\ & h_s(x_{s-k:s}) = 0 \quad \forall_{s=t}^{t+H-1}, \end{aligned} \tag{4.1}$$

where f_s, g_s and h_s are the same functions as in the optimal control problem in (4.2). The scalar objective function f_s for every time step is called stage cost. The prediction horizon is H. $\tilde{J}_{t+H}$ is the cost-to-go approximation, which will be derived in detail in the following. The purpose of this cost-to-go approximation is to *transfer* the overall task encoded in the finite time optimal control problem into the MPC optimization. The last term in the objective is the tube term, which ensures that the cost-to-go approximation $\tilde{J}_{t+H}$ stays valid since the cost-to-go term is based on a approximation. Thus the cost-to-go approximation and the tube term form the terminal cost.

4.2 Approximate cost-to-go

The derivation of the approximate cost-to-go is based on a optimal control problem for the overall task. The cost-to-go is an approximation of the part which is not covered by the prediction horizon of the approximate constrained model predictive control optimization. Thus the optimization problem for optimal control with time horizon T is briefly restated in KOMO notation as

$$\begin{aligned} \min_x \quad & \sum_{t=0}^{T-1} f_t(x_{t-k:t}) \\ \text{s.t.} \quad & g_t(x_{t-k:t}) \leq 0 \quad \forall_{t=0}^{T-1} \\ & h_t(x_{t-k:t}) = 0 \quad \forall_{t=0}^{T-1}, \end{aligned} \tag{4.2}$$

where $f_t : \mathbb{R}^{n \times k+1} \to \mathbb{R}$ is the scalar objective and $g_t : \mathbb{R}^{n \times k+1} \to \mathbb{R}^{d_g}$ and $h_t : \mathbb{R}^{n \times T} \to \mathbb{R}^{d_h}$ are the inequality and equality constraints, respectively.

Derivation cost-to-go: Motivated by the tutorial paper [Tou16] we use the k-order Bellman's principle on the fixed Lagrangian. Therefore the fixed Lagrangian for every time step t is defined as

$$\tilde{f}_t(x_{t-k:t}) = f_t(x_{t-k:t}) + (\lambda_t^*)^\top g_t(x_{t-k:t}) + (\kappa_t^*)^\top h_t(x_{t-k:t}),$$

where λ_t^* and κ_t^* denote the fixed Lagrange parameter for the optimal solution to the optimal control problem (4.2). Defining then the cost-to-go

$$\tilde{J}_t(x_{t-k:t-1}) = \min_{x_{t:T-1}} \sum_{s=t}^{T-1} \tilde{f}_s(x_{s-k:s})$$

for the fixed Lagrangian over a k-order separator. A separator means the future becomes independent from the past conditional to the separator in Markovian settings. Then Bellman's principle imply

$$\tilde{J}_t(x_{t-k:t-1}) = \min_{x_t} \left[\tilde{f}_t(x_{t-k:t}) + \tilde{J}_{t+1}(x_{t-k+1:t}) \right], \qquad \tilde{J}_T(x_{T-k:T-1}) = 0.$$

In order to solve this analytically a quadratic approximation of the cost-to-go and the fixed Lagrangian

$$\begin{aligned} \tilde{J}_t(x_{t-k:t-1}) &= x_{t-k:t-1}^\top V_t x_{t-k:t-1} + 2 v_t^\top x_{t-k:t-1} + \vartheta_t \\ \tilde{f}_t(x_{t-k:t}) &= \frac{1}{2}(x_{t-k:t} - x_{t-k:t}^*)^\top \nabla^2 \tilde{f}_t(x_{t-k:t}^*)(x_{t-k:t} - x_{t-k:t}^*) \\ &\quad + \nabla \tilde{f}_t(x_{t-k:t}^*)^\top (x_{t-k:t} - x_{t-k:t}^*) + \tilde{f}_t(x_{t-k:t}^*) \end{aligned}$$

is used, where the approximation of the fixed Lagrangian is a Taylor series expansion around the solution x^* to the optimal control problem (4.2). Using the quadratic approximation to rewrite the optimization as an unconstrained quadratic program in block matrix form

$$\tilde{J}_t(x_{t-k:t-1}) = \min_{x_t} \left[\begin{pmatrix} x_{t-k:t-1} \\ x_t \end{pmatrix}^\top \begin{pmatrix} D_t & C_t \\ C_t^\top & E_t \end{pmatrix} \begin{pmatrix} x_{t-k:t-1} \\ x_t \end{pmatrix} + 2 \begin{pmatrix} d_t \\ e_t \end{pmatrix}^\top \begin{pmatrix} x_{t-k:t-1} \\ x_t \end{pmatrix} + c_t \right], \quad (4.3)$$

where the matrices are defined as

$$\begin{pmatrix} D_t & C_t \\ C_t^\top & E_t \end{pmatrix} =$$

$$\begin{pmatrix} \left(\frac{1}{2}\nabla^2 \tilde{f}_t(x_{t-k:t})\right)_{11} & \left(\frac{1}{2}\nabla^2 \tilde{f}_t(x_{t-k:t})\right)_{12} & \left(\frac{1}{2}\nabla^2 \tilde{f}_t(x_{t-k:t})\right)_{13} \\ \left(\frac{1}{2}\nabla^2 \tilde{f}_t(x_{t-k:t})\right)_{21} & \left(\frac{1}{2}\nabla^2 \tilde{f}_t(x_{t-k:t})\right)_{22} + \left(V_{t+1}\right)_{11} & \left(\frac{1}{2}\nabla^2 \tilde{f}_t(x_{t-k:t})\right)_{23} + \left(V_{t+1}\right)_{12} \\ \left(\frac{1}{2}\nabla^2 \tilde{f}_t(x_{t-k:t})\right)_{31} & \left(\frac{1}{2}\nabla^2 \tilde{f}_t(x_{t-k:t})\right)_{32} + \left(V_{t+1}\right)_{21} & \left(\frac{1}{2}\nabla^2 \tilde{f}_t(x_{t-k:t})\right)_{33} + \left(V_{t+1}\right)_{22} \end{pmatrix}$$

$$\begin{pmatrix} d_t \\ e_t \end{pmatrix} = \begin{pmatrix} \left(\frac{1}{2}\nabla \tilde{f}_t(x_{t-k:t}) - \frac{1}{2}\nabla^2 \tilde{f}_t(x^*)x^*\right)_1 \\ \left(\frac{1}{2}\nabla \tilde{f}_t(x_{t-k:t}) - \frac{1}{2}\nabla^2 \tilde{f}_t(x^*)x^*\right)_2 + \left(v_{t+1}\right)_1 \\ \left(\frac{1}{2}\nabla \tilde{f}_t(x_{t-k:t}) - \frac{1}{2}\nabla^2 \tilde{f}_t(x^*)x^*\right)_3 + \left(v_{t+1}\right)_2 \end{pmatrix}$$

$$c_t = \frac{1}{2}(x^*)^\top \nabla^2 \tilde{f}_t(x^*)x^* - \nabla \tilde{f}_t(x^*)^\top x^* + \tilde{f}_t(x^*) + \vartheta_{t+1},$$

where the $(\cdot)_{ij}$ is the sub block matrix and $(\cdot)_k$ is the sub block vector notation according to the dimensions of $x_{t-k:t-1}$, x_t, V_{t+1} and v_{t+1}. The explicit minimizer of the unconstrained QP (4.3) is

$$\bar{x}_t = \operatorname*{argmin}_{x_t} \left[\tilde{f}_t(x_{t-k:t}) + \tilde{J}_{t+1}(x_{t-k+1:t})\right] = -E_t^{-1}\left(C_t^\top x_{t-k:t-1} + e_t\right).$$

Plugging in the minimizer $\bar{x}_t$ and the quadratic approximation of $\tilde{J}_t$ in the quadratic Bellman equation (4.3) and a coefficient comparison yields in the update rules

$$\begin{aligned} V_t &= D_t - C_t E_t^{-1} C_t^\top, \quad & V_T &= 0 \\ v_t &= d_t - C_t E_t^{-1} e_t, \quad & v_T &= 0 \\ \vartheta_t &= c_t - e_t^\top E_t^{-1} e_t, \quad & \vartheta_T &= 0. \end{aligned} \quad (4.4)$$

With these update rules it is possible to compute the cost-to-go approximation around x^* for every time step t as a backwards in time recursion.

Algorithm 4.1 Application of the approximate model predictive control scheme

solve optimal control problem (4.2) // preparation phase
for $k \leftarrow T-1, T-2, \ldots, 0$ **do**
    apply update rule (4.4)
end for
for $k \leftarrow 0, 1, \ldots, T-1$ **do** // main control loop
    if $k + H < T - 1$ **then**
        solve approximate constrained MPC optimization (4.1)
    else
        solve optimal control optimization (4.2) with reducing horizon
    end if
    $u \leftarrow$ solve dynamics for control at time step k
    apply control u
end for

4.3 Design Scheme

The derived approximate constrained model predictive control scheme is applied in four steps to a real system:

1. Solve the optimal control problem (4.2) for generating an optimal trajectory
2. Compute the local approximations of the cost-to-go for every time step by repetitively applying the update rules (4.4) backwards in time
3. Repetitively solve the approximate constrained model predictive control optimization problem (4.1) and apply the first step as control input until the prediction horizon hits the end point
4. Repetitively solve the optimal control problem (4.2) with a reducing prediction horizon and apply the first step as control input until the end time is reached

Here the first two steps are computed offline and are not time critical. Only the last two steps are computed in the *real time* loop. But the involving optimization has a reduced size and is structured. Therefore these MPC optimizations are fast to solve.

This design procedure is summarized in a structured way in Algorithm 4.1.

4.4 Stability investigation

Stability in the sense of Lyapunov is defined as:

> Let $\dot{x} = f(x)$, $f : \mathbb{R}^n \to \mathbb{R}^n$, f locally Lipschitz be a time invariant nonlinear dynamical system with the initial condition $x_0 \in \mathbb{R}^n$. Further let $\bar{x}$ be a equilibrium point for the system. If for any $\varepsilon > 0$ there exists a $\delta > 0$ such that
>
> $$\forall x_0 \in B_\delta(\bar{x}) \quad \Rightarrow \quad x(t) \in B_\varepsilon(\bar{x}) \qquad \forall t > 0,$$
>
> then the equilibrium point is stable.

This definition clearly shows that stability is a property for dynamical systems, which says that system stays close to the equilibrium if we start close for all future time. For asymptotic stability the property $\lim_{t\to\infty} x(t) = \bar{x}$ is added. Since we consider a finite time task with the approximate constrained model predictive control scheme classical asymptotic stability does not make sense to investigate. Because we allow general tasks, like in the economic MPC setting, it is not clear to consider stability against what. E.g. for a stable error between planned reference and system trajectory it would be necessary that the target of the controller is a reference tracking (error stabilization). But exactly this is not the purpose of the proposed control scheme, at least not in this work, since a controller for executing compliant motion tasks is desired.

5 Evaluation

Here the derived method *approximate constrained model predictive control* is evaluated. First a simple one dimensional trajectory optimization example is considered. This is done in order to get insights into the method and represents a proof of concept. The second example system is a two link serial robot. It can be seen as a simple practical application of the control scheme and enables further investigations. Especially the contact phase can be evaluated.

5.1 1D Trajectory Optimization

This is the simple 1D example, which is used as a first evaluation of the method. Here a trajectory is discretized in T points and then the acceleration profile is minimized between two points x^*_{-1} and x^*_{T-1}. The initial and end velocity should be zero. This yields in KOMO notation the optimization problem

$$
\begin{aligned}
\min_{x_{0:T-1}} \quad & \sum_{s=0}^{T-1} ||x_{s-2} - 2x_{s-1} + x_s||^2 \\
\text{s.t.} \quad & x_s - x^*_s = 0 \qquad \text{for } s \in \{-1, T-1\} \\
& x_{s-1} - x_s = 0 \quad \text{for } s \in \{-1, T-1\},
\end{aligned}
\tag{5.1}
$$

where $s = -1$ define the prefix and $s = T - 1$ the target state.

The current implementation of the KOMO framework supports only sum-of-squares cost features and the quadratic approximation can be passed as sum-of-squares via a quadratic factor. For this a Cholesky decomposition is used and thus the quadratic approximation needs to have full rank ($\det(V_t) \neq 0$). The cost in this example has only a local effect in every time step. Thus cancels out in the update law (4.4) and has no influence on V_t. Since linear constraints have a linear impact on the approximation of the cost-to-go, see (4.4) a method is needed to ensure that the cost-to-go approximation is a full rank quadratic function. An easy and simple way is to base the approximation on the fixed augmented Lagrangian

$$
\tilde{L}_t(x, \lambda_t, \kappa_t, \nu, \mu) = f_t(x) + \lambda_t^\top g_t(x) + \kappa_t h_t(x) + \mu g_t(x)^\top I_{\lambda_t}(x) g_t(x) + \nu h_t(x)^\top h_t(x), \tag{5.2}
$$

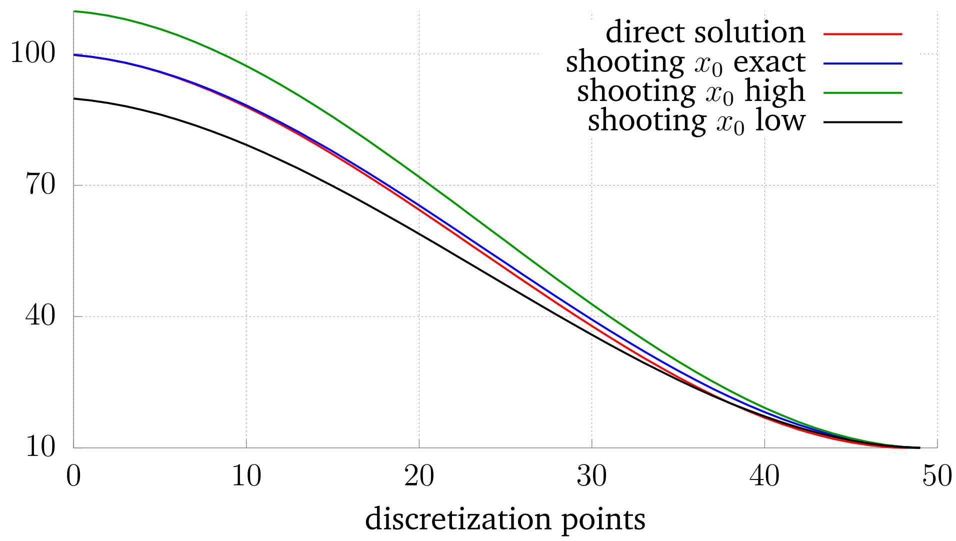


Figure 5.1: Comparison between the forward shooting and the direct solution of a 1D trajectory optimization for different initial conditions x_0.

which is also used in the solver of the optimization problem, instead of the fixed Lagrangian for every time step t. The indicator matrix I_λ is given by $I_\lambda(x) = \mathrm{diag}([\lambda_{t,i} > 0 \vee g_{t,i}(x) \geq 0]_+)$ and i denotes the index of the corresponding inequality $g_{t,i}$. The notation $[a]_+$ is 1 if a is true and 0 if a is false. The parameters μ and ν are chosen to be the same as the last run in the augmented Lagrangian solver for the optimal control problem (4.2). For details on the augmented Lagrangian and the solver consider [Tou14a].

As first evaluation of the cost-to-go approximation we consider a forward shooting. This means the cost-to-go approximation is used to find an optimal path by iteratively solving one step optimization problems starting from the initial point. The optimization for every time step t is

$$\begin{aligned} \min_{x_t} \quad & ||x_{t-2} - 2x_{t-1} + x_t||^2 + \tilde{J}_{t+1}(x_{t-2:t-1}) \\ \mathrm{s.\,t.} \quad & x_s - x_s^* = 0 \qquad s \in \{t-1, T-1\} \\ & x_{s-1} - x_s = 0 \quad s \in \{t-1, T-1\}, \end{aligned} \tag{5.3}$$

where the constraints are only active in the time step $T-1$. The prefix in the first time step is the same as in (5.1) and then it is chosen based on the results of the optimization of the previous time steps. Only in the last time step the constraint of the target point x^*_{T-1} affects the optimization. The solution to the forward shooting problem (5.3) compared to a direct solution (5.1) is given in Figure 5.1. Also a variation of the initial condition showed a good behavior. A slight deviation between the direct solution and the forward shooting with the same initial condition is observed. Based on the observation that the cost of the forward shooting trajectory is lower than the direct solution one, leads to the hypothesis that the KOMO solver could not converge exactly to the minimum and the shooting could further improve the quality of the trajectory.

In the numerical experiments it turns out that the quadratic approximation has its center at the target point and gets steeper if the time gets closer to the end point. This makes sense since via the squared penalty of the fixed augmented Lagrangian (5.2) a squared potential on the end point is introduced. The closer the end time gets, the harder the cost-to-go should pull to the target point. If the end time is hit then the larger eigenvalue jumps to a high value (from 5.8 to 524290). This shows that an equality constraint pops up.

5.2 Two link serial robot

In this section the second example is presented. Here a practical scenario is considered and simulation experiments are evaluated. The considered example system is a two link serial robot, which is schematically illustrated in Figure 5.2.

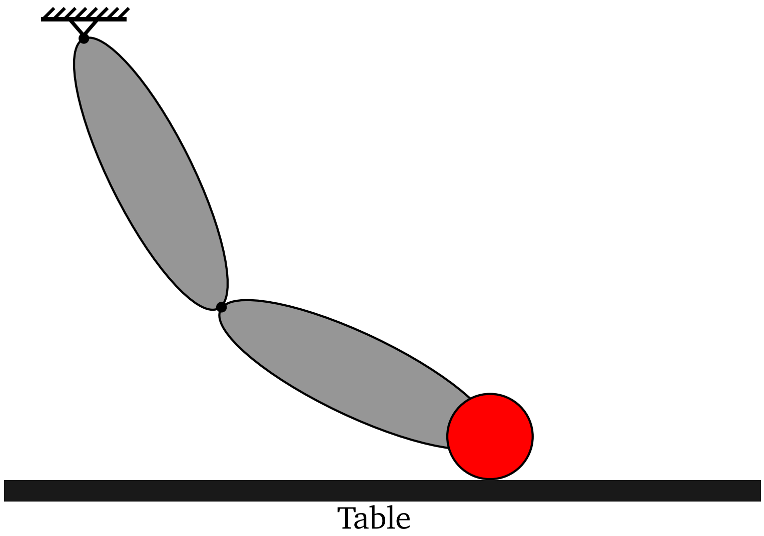


Figure 5.2: Schematic illustration of the two link serial robot with contact to the table.

5.2.1 Setup and Simulation Environment

The robot consists of two cylindrical rigid links of the length $l = 0.5\,\text{m}$, radius $r = 0.05\,\text{m}$ and is made from aluminium which result in a mass of $m \approx 10.6\,\text{kg}$ per link. The end-effector is a PVC sphere with radius $r = 0.1\,\text{m}$ and mass of $m \approx 5.4\,\text{kg}$. In the zero position of the angle coordinates x_t the arm is pointing downwards and the positive direction is counter clockwise rotation. The dynamic equation is derived with Lagrange's second method in the form

$$M(x)\ddot{x} + F(x, \dot{x}) = u, \tag{5.4}$$

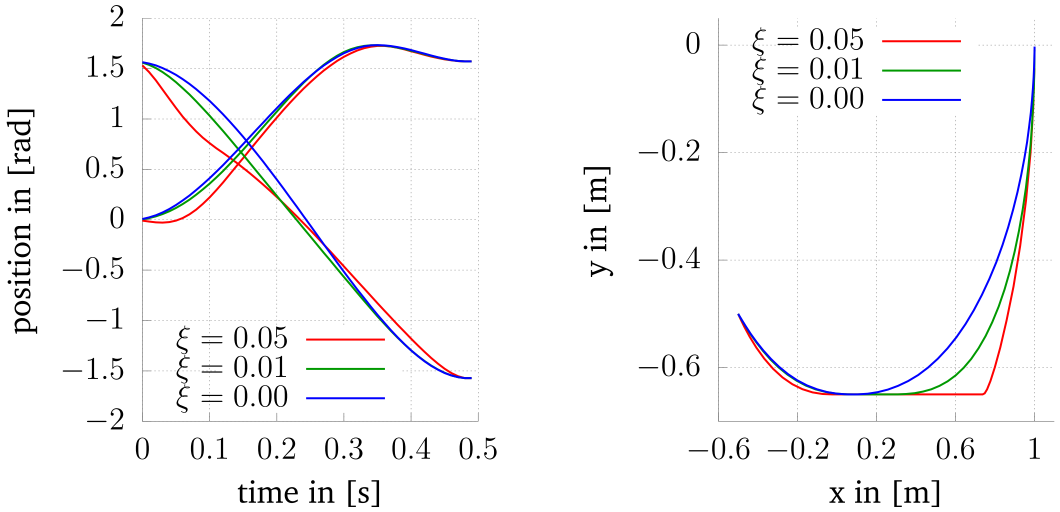
where $M(x)$ is the generalized mass matrix and $F(x, \dot{x})$ the generalized forces consisting of Coriolis and gravitational forces. The robot is actuated via motors in the joints and thus the control input u is the motor torque measured in Nm. Further we assume that the control input is updated every $\tau = 10\,$ms. For simulation a Runge-Kutta method of 4. order is used. To improve the accuracy of the integration 10 intermediate steps between each control update are introduced. Thus the numerical integration runs with a time step of $1\,$ms.

Since the evaluation scenario involves a contact phase the simulation environment needs to handle contacts. This can be achieved via a spring damper model acting normal to the constraint surface on the end-effector. The spring and damper coefficients are chosen high to approximate a rigid body contact. This means, that if a contact in the simulation is detected (interpenetration into a constraint) a generalized spring damper force is added to the dynamic equation (5.4) used for simulation. In this case the dynamic of the simulated system has a switching behavior and the accuracy of the numerical integration may drop. This is e.g. the case when the time step for integration is too high and thus the contact time is not covered precisely enough. When an adaptive step size in the integration is used it is important to set a good lower bound of the step size since at the discontinuity in the dynamics the integration step will be the lower bound.

5.2.2 Task

As task we consider a movement from a state A to state B using T time steps τ. Further, the resulting movement should be smooth and should keep a constraint active — e.g. sliding on the table.

In the proposed test scenario the arm should move from completely stretched to the right to first segment pointing left and second segment pointing downwards, where the velocities in the start and end point are zero. During this motion a sliding phase on the table surface, which is located at $y = -0.65\,$m for the end-effector center point, should take place. Thus, the initial position of the robot is $x^*_{-1} = \begin{pmatrix} \frac{1}{2}\pi & 0 \end{pmatrix}^\top$ and the target position is $x^*_{T-1} = \begin{pmatrix} -\frac{1}{2}\pi & \frac{1}{2}\pi \end{pmatrix}^\top$. The smoothness is expressed via an acceleration penalty $||x_{s-2} - 2x_{s-1} + x_s||^2$. The sliding phase can be controlled via a pull-to-table penalty $||D_c(x_s)||^2$ for every time step s, where D_c is the difference function between end-effector



(a) Position trajectories in joint space.

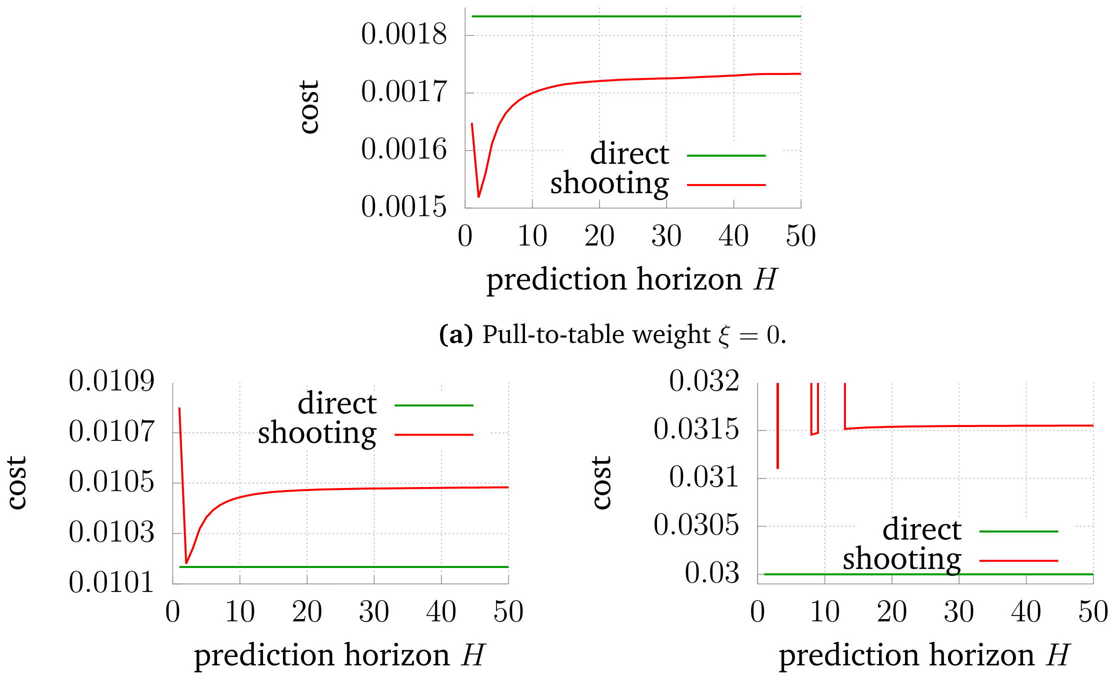
(b) End-effector position in cartesian space.

Figure 5.3: Solution of the optimal control problem (5.5) for different values of ξ.

and the table y-coordinates and is negative when the end-effector is interpenetrating the table. The time the motion should take is $0.5\,\mathrm{s}$ and with a time step of $\tau = 10\,\mathrm{ms}$ the number of discretization points is $T = 50$. The initial and end state are encoded as prefix and position constraints in terms of x_s^* and x_{s-1}^* for $s = -1$ and $s = T-1$, respectively. Modeling the table position as inequality constraint for every time step yields the optimal control problem

$$
\begin{aligned}
\underset{x_{0:T-1}}{\operatorname{argmin}} \quad & \left[\sum_{s=0}^{T-1} ||x_{s-2} - 2x_{s-1} + x_s||^2 + \xi\,||D_c(x_s)||^2\right] & \\
\text{s.t.} \quad & -D_c(x_s) \leq 0 & \forall_{s=0}^{T-1} \\
& x_s^* - x_s = 0 & \text{for } s \in \{-1, T-1\} \\
& x_{s-1}^* - x_{s-1} = 0 & \text{for } s \in \{-1, T-1\}
\end{aligned}
\tag{5.5}
$$

for trajectory generation, where ξ is a trade off parameter between a smooth trajectory and how long the sliding phase should be. The resulting angle and end-effector trajectories for different values of ξ can be found in Figure 5.3. In the left diagram 5.3(a) the trajectories starting at $\frac{1}{2}\pi$ correspond to the first angle and starting a 0 corresponds to the second angle.



(a) Pull-to-table weight $\xi = 0$.

(b) Pull-to-table weight $\xi = 0.025$.

(c) Pull-to-table weight $\xi = 0.05$.

Figure 5.4: Forward shooting cost over all possible horizons H for different pull-to-table weights ξ compared to the cost of the direct solution from trajectory generation.

5.2.3 Forward shooting

For the evaluation of the proposed approximate constrained MPC scheme the effect of the MPC horizon is investigated in a forward shooting evaluation. This excludes several possibilities introducing numerical errors like the integration with a Runge-Kutta method. Also the tube-term is neglected because the effect of the cost-to-go approximation $\tilde{J}_{t+H}$ and the prediction horizon H should be evaluated. The forward shooting is tested for all possibilities of different horizons H ranging from a 1-step optimization to a 50-step optimization. To stay close to the proposed MPC scheme the fading out strategy is also included in the forward shooting. Thus the 50-step optimization is a complete replanning of the trajectory. Running the shooting for different different values of the pull-to-table parameter ξ yields in the results shown in Figure 5.4.

The result in Figure 5.4(a) is close to the result of the 1D trajectory optimization. As expected the shooting cost is lower than the cost for the direct solution. The spiking down between prediction horizon $H = 1$ and $H = 2$ could be the effect of the fact that the optimization is able to go in two steps towards the minimum of the cost-to-go approximation instead of going in one step towards the minimum and getting on this

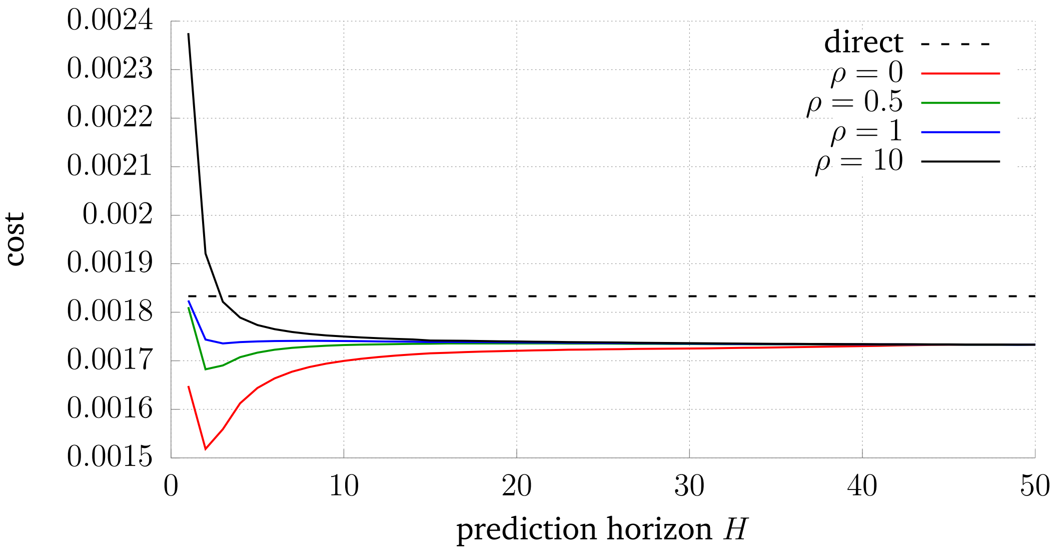


Figure 5.5: Varying the parameter ρ of the tube-term.

one step a larger penalty since the movement is to aggressive. The getting worse with a larger prediction horizon behavior could be due to the precision of the solver for different large optimization variables. If the solver returns a better solution for low dimensional optimization variables than for high dimensional optimization variables then using an *exact* cost-to-go and solving the problem iteratively is a more precise way than solving the optimization directly. In this case the cost-to-go approximation could be good enough to cover this behavior. But these hypothesizes definitively need some further evaluations.

In the two lower plots of Figure 5.4 the effect of the pull-to-table weight is evaluated. My hypothesis is that the pull-to-table penalty makes the objective harder. Hence the quality of the cost-to-go approximation $\tilde{J}_{t+H}$ gets worse and thus the result of the shooting also gets worse. The spikes in Figure 5.4(c) are outliers with a really high cost (about 30), where the solution between different time steps converged to different minima.

The effect of the tube term is evaluated in the same way. A simulation of different values of ρ over all possible horizons H with a pull-to-table weight ξ fixed to zero is done. The result is shown in Figure 5.5. Here it is visible that the parameter affects mainly the small horizons, which is intuitively clear since the tube term is a penalty pulling the last point of the prediction horizon back to the planned trajectory. For a horizon larger than 10 the cost of the shooting trajectory settles to its *stationary* value. This leads to the assumption that the tube term does not significantly affect the quality of the trajectory optimization in the nominal case for large enough horizons.

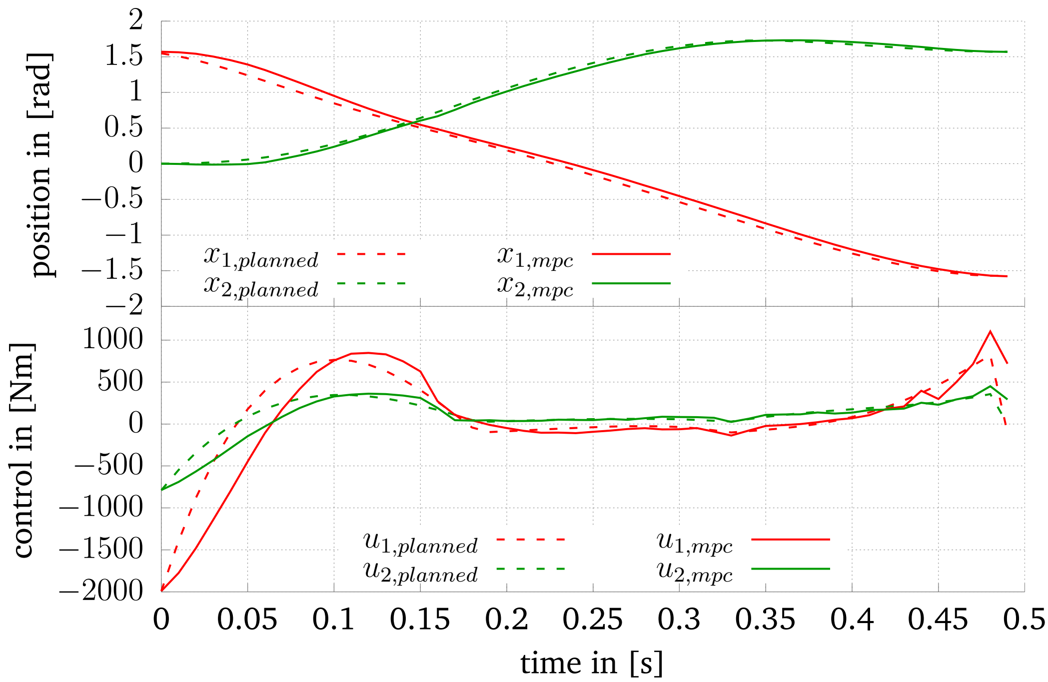
5.2.4 Simulation results

For evaluation of the closed-loop system different scenarios are considered. The first shows the basic scenario as described above and shows that the control strategy is generally working. Then in the second case the initial condition of the controller will be different than in the setup phase. Then a force will disturb the end-effector over a short time and thus the reaction of the controller will be evaluated. This will show whether the approximate constraint MPC scheme is capable of performing a compliant motion task or not. In the last scenario a position error in the table constraint is introduced and thus the results will show how uncertain constraint will be handled by the MPC controller.

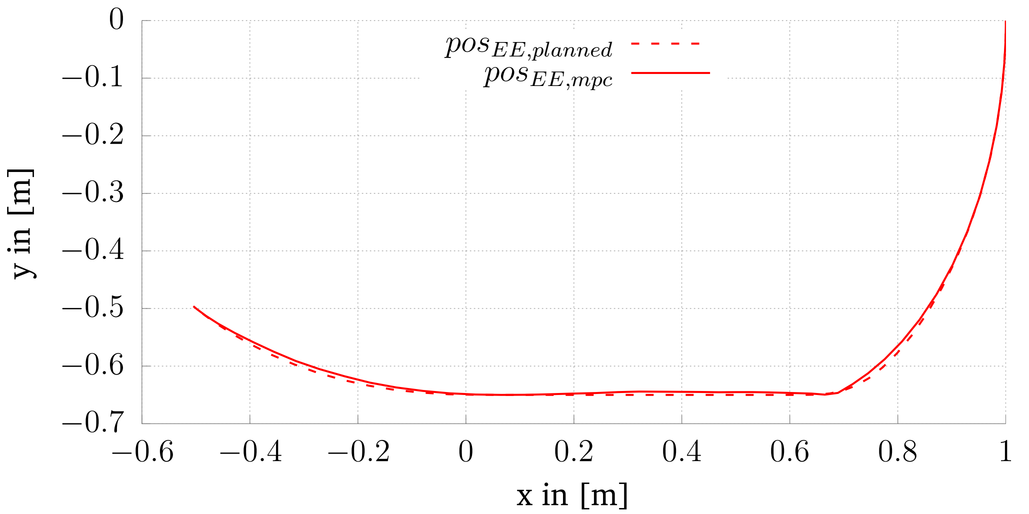
First scenario: In order to define a good test scenario the parameter for the pull-to-table weight is chosen to $\xi = 0.025$. This leads to a long sliding phase and is not pulling too aggressive to the table. For showing a *robust* execution an additive mean free noise with a standard deviation of $25\,\mathrm{Nm}$ is added to the control signal before simulating one step. This noise makes the above testing case, where the cost is evaluated over all possible prediction horizons useless, because the noise makes the cost randomly varying in a band. Thus the controller parameters are chosen to $H = 5$ for the prediction horizon and $\rho = 1$ for the tube term based on the knowledge of the forward shooting results.

The position and control trajectories for the closed-loop simulation run compared to the planned trajectories can be found in Figure 5.6(a). Here the position trajectory deviates slightly from the planned one due to noise, but the controller does not force the system back to reference. Instead, it tries to smoothly complete the motion. Thus the control trajectories are also smooth and have no peak. The cost for the whole trajectory is 0.012674. In Figure 5.6(b) the according trajectory for the center point of the robot's end-effector is shown. The first contact with the table takes place after $0.17\,\mathrm{s}$. This is also the time at which the trajectories in Figure 5.6(a) are really close to the planned ones. Starting from this time step the control trajectories stay close to the planned ones, because the constraint acts as a guide.

Second scenario: For evaluating the compliance of the control strategy a second test scenario is evaluated. Under compliance we understand the reaction behavior to disturbances during execution of the motion. A compliant controller should aim to fulfill the motion task rather than trying to execute stiffly the position profile. Therefore the initial condition for the simulation is changed to $\tilde{x}_{-1} = \begin{pmatrix} 1.3 & -0.26 \end{pmatrix}^\top$ and thus the controller will base the trajectory planning on an assumption with $x_{-1} = \begin{pmatrix} \frac{1}{2}\pi & 0 \end{pmatrix}^\top$. This is indeed a large offset of the initial position, but it clearly shows the compliant behavior if the prediction horizon is chosen large enough. For further investigations the tube term weight ρ is chosen to be 1.



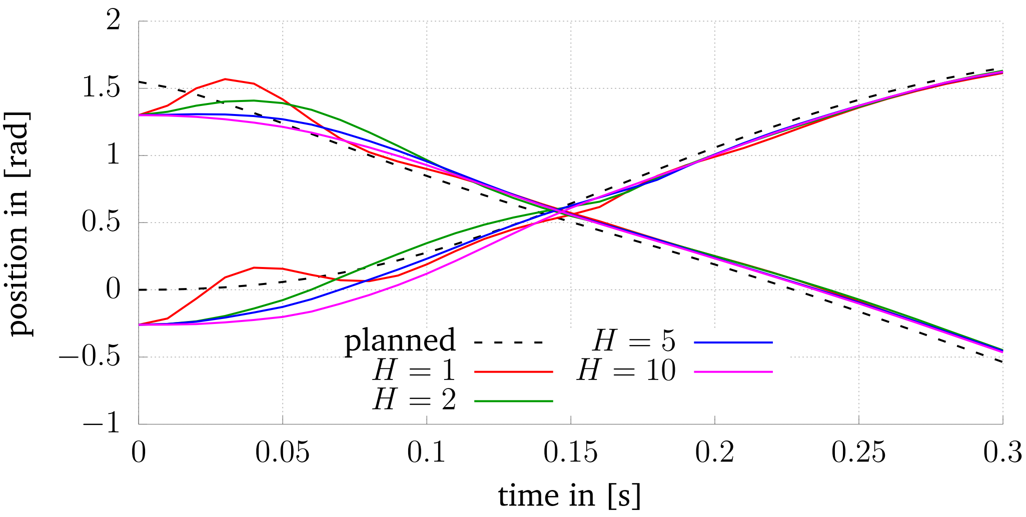
(a) Position and control trajectories.



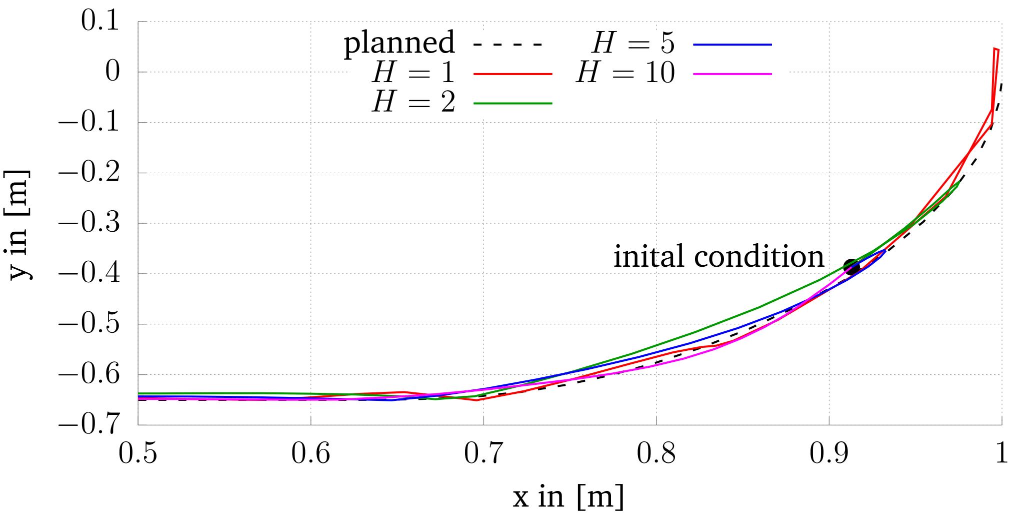
(b) End-effector trajectory.

Figure 5.6: Trajectories of the two link robot in closed-loop with the approximate constrained MPC compared to the optimal control problem solution.

In Figure 5.7 the first part of the position trajectory and a detail view of the end-effector center around the initial condition are given. The shown behavior matches with the expected behavior, because if the prediction horizon is too short, then the tube term will pull the last point in the prediction horizon to the planned trajectory. Thus for the case where the prediction horizon H is 1, the end-effector trajectory will first go towards



(a) First part of the position trajectories.



(b) Detail view of the end-effector center around the initial condition $\tilde{x}_{-1}$.

Figure 5.7: Different initial condition for simulation and planning.

the initial point of the planned trajectory and then move along it. In Figure 5.7(b) the trajectory for the prediction horizon $H = 1$ (red) will go close to x_{-1} and track the planned trajectory. This is not what we understand under compliance. But if the prediction horizon gets larger, the MPC optimization can handle such deviations in a smooth way. In this scenario the MPC controller uses the mismatch of the initial conditions if the last point of the prediction horizon is on the opposite side of the initial condition of the simulator. Hence the MPC will use it as a *benefit*. This behavior is visible for the case where the prediction horizon H is 10 or larger and in Figure 5.7 it looks like the closed-loop trajectory is close to optimal. This is then a compliant execution of the planned motion.

scheme	compliance η_c	cost
complete replanning	2.50	$0.054\,314\,9$
approximate constrained MPC ($H = 10$)	4.76	$0.043\,348\,2$
approximate constrained MPC ($H = 2$)	6.67	$0.043\,032\,1$
approximate constrained LQR	8.33	$0.082\,747\,6$
PID	11.11	$0.073\,283\,4$

Table 5.1: Compliance and cost values for the different schemes.

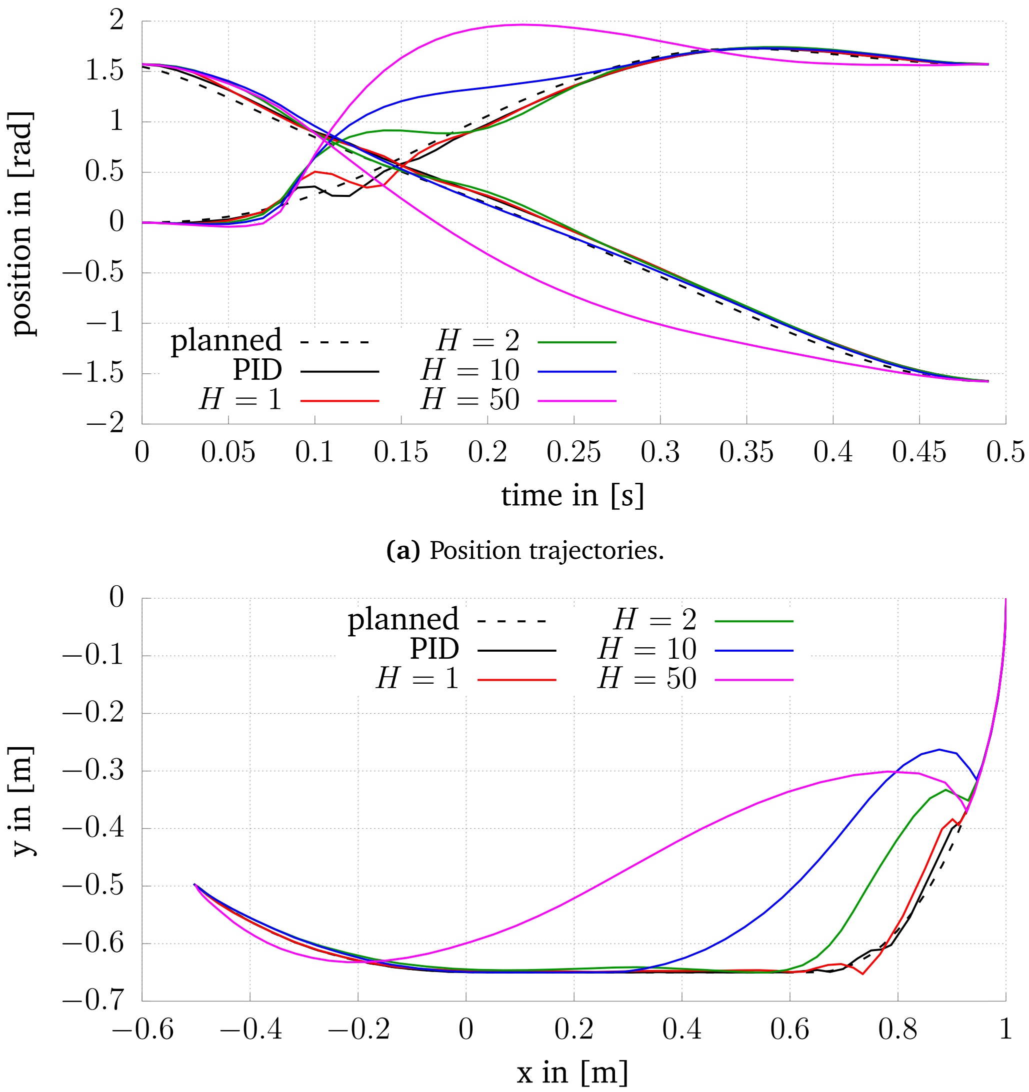
When varying the tube term, simulations have shown similar results as for the variation of the prediction horizon. This is intuitively clear since the tube term pulls the last point of the MPC optimization to the trajectory. Thus if the weight is low the last point is able to move more freely. But since the cost-to-go is based on quadratic approximations without crude bounds, it is better to keep the weight of the tube term active. Hence the overall objective is correctly transferred to the MPC optimization.

Third scenario: The second test for evaluating the compliant behavior of the proposed control scheme is a force disturbance. Here a strong force disturbance $f_{dist} = \begin{pmatrix} 3000 & 4000 \end{pmatrix}^\top$ N is given on the end-effector center during the time 0.07 s to 0.08 s. Then the reaction of the approximate constrained MPC is compared for different horizons similar to the second scenario and to a PID controller stabilizing the error between the planned trajectory and the current state of the robot, hence in joint space. For measuring the compliance of the controllers a measure

$$\eta_c = \frac{1}{t_b}$$

is proposed, where t_b is the time the controller needs to steer the systems output back to the planned trajectory. Figuratively speaking the longer the controller needs to steer the system back to the planned trajectory the more compliant the controller is. This makes only sense if the controller *knows* that not steering back to the planned trajectory is a good/optimal strategy. In our setting this is the case, because the MPC optimization is a local approximation of the optimal control problem for trajectory planning. For a simple error stabilizing PID controller as in the above case, such a measure is not a valid criteria.

The simulation results are shown in Figure 5.8. The disturbance in Figure 5.8(b) does not always occur at the same point, because the disturbance force is active at a certain time interval and the controller does not track the trajectory exactly due to noise. Through the forward kinematic a small position error in joint space translates into a larger error in cartesian space. Thus the center of the end-effector is at the disturbance time at different



(a) Position trajectories.

(b) End-effector center trajectory

Figure 5.8: Force disturbance on the end-effector during execution of different control schemes.

points. Based on the gained intuition the complete replanning strategy ($H = 50$) is the most compliant scheme. Then the approximate constrained MPC scheme with long horizon ($H = 10$) follows before the shorter horizon ($H = 2$). The least compliant controllers are the approximate constrained LQR ($H = 1$) and the PID controller. It is remarkable that the compliant execution of the motion already starts with a prediction horizon of $H = 2$. The resulting compliance numbers and cost value of the closed-loop trajectory are given in Table 5.1 and prove the intuitive behavior. Interestingly the cost for the complete replanning scheme is higher than the approximate constrained MPC schemes as in subsection 5.2.3.

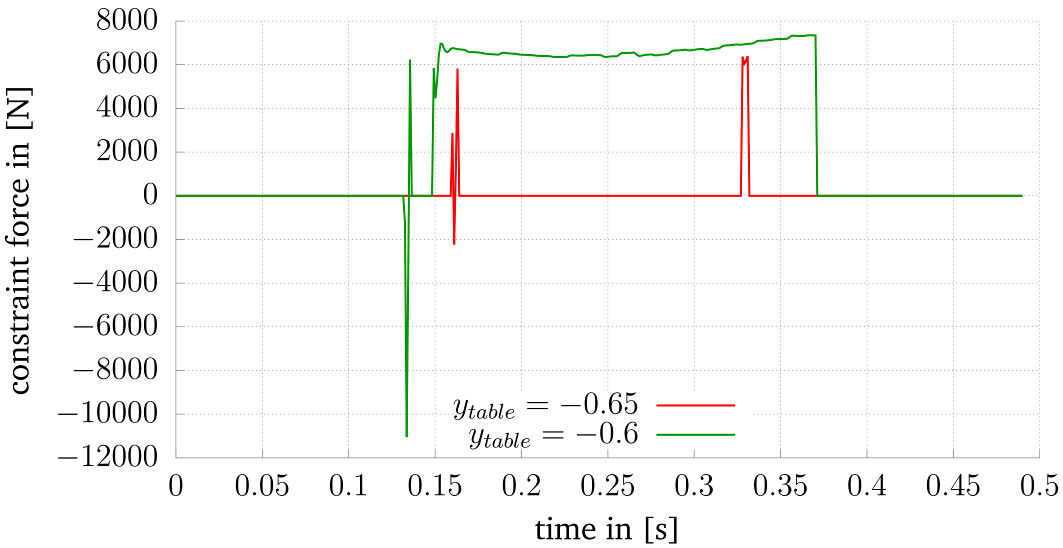


Figure 5.9: Contact force profile of the table constraint moved 0.05 m upwards.

Fourth scenario: For evaluating the contact behavior the table constraint is moved. Therefore consider again the parametrization of the first scenario and use the results as baseline, see Figure 5.6. Here only a small contact at the beginning and at the end of the sliding phase takes place. The maximal contact force is 6402.51 N and the mean is 4093.21 N. Thus the desired sliding behavior does not take place. First consider the case where the constraint is moved up by 0.05 m. Then the desired sliding takes place with a nearly constant contact force with mean and maximal value of 6499.16 N and 7355.07 N, respectively. The contact force for these two cases is plotted in Figure 5.9. The negative peaks in the contact force happen because the contact model consists of a spring and a damper. Since the damper acts in the opposite direction to the velocity part of the end-effector normal to the constraint surface, and the spring force is always pointing outwards and orthogonal to the constraint surface,it may happen that the sum of these two constraint forces is negative. This is the case when the end-effector is moving outwards of the constraint surface and being close to the boundary.

For smaller upwards deviations of the table constraints the contact is *bouncy* and for larger deviations, the contact force in the sliding phase increases. If the table constraint is moved downwards up to 0.03 m the sliding also takes place and for larger deviations the resulting closed-loop trajectory is not close to optimal. This behavior is expected, because in the MPC optimization the initial position in the prefix and the wrongly assumed table constraint contradict. Hence the optimization pushes the first optimization variables for the first time step in the prediction out of the constraint, but the imprecise simulation is not able to perform this action. Thus the closed-loop stays below the expected table position. For larger downwards deviations the closed-loop could get chaotic.

6 Discussion

6.1 Results

As in the evaluations shown have we achieved a compliant execution controller based on the trajectory optimization framework KOMO. This specifically means that we were able to encode a more general objective directly in the control structure. Thus a general design scheme for the execution controller based on the trajectory optimization is derived.

Therefore two different testing setups were considered. The first test example is a simple 1D trajectory optimization, which is a synthetic test of the derived cost-to-go approximation. Here the first technical insight into the method was achieved and showed that the cost-to-go approximation works good for such a simple case. The second test example is a simulation of a two link serial robot, where the motion of the end-effector is constrained by a table. This is a practical test of the control scheme, where especially the compliance of the resulting closed-loop behavior can be evaluated. The first test scenario showed that the approximate constrained MPC scheme is able to execute the motion task. Then the next two scenarios showed that a long enough prediction horizon is needed in order to compliantly execute the motion if the system is disturbed. Therefore a disturbance on the initial condition in the simulation and a disturbance force on the end-effector are introduced. A comparison between different parametrizations of the proposed MPC controller and a stabilizing PID showed that a good compliance can be achieved for medium large prediction horizons. The variation of the prediction horizon to the extreme case recovers the approximate constrained LQR and complete replanning of the trajectory. In the last test scenario the planned contact phase is evaluated. For a precise model-world matching, the controller is not sliding and only a small contact in the beginning and end of the planned sliding phase take place. Thus the controller does not correctly execute the desired sliding motion. For further evaluations the table constrained in the simulation is changed. For changing the table position upwards a sliding motion with a constant contact force is achieved and changing the table constrained slightly downwards the sliding takes place. But for larger downwards deviations resulting closed-loop behavior is chaotic.

6.2 Alternatives

As alternative a possible extension for ensuring the sliding motion for the proposed approximate constrained MPC scheme is discussed. Motivated by the approach used for the dual execution controller proposed in [TRB+14] a separate controller could handle the sliding. Hence during the planned sliding phase a drift force and a force controller as an additional controller are added to the control signal. If the real position would contradict to the constrained model of the controller then either the position seen by the MPC controller could be moved to the next valid point on the boundary of the constraint or the constrained model of the MPC could be adapted to satisfy the measured position. In the two link serial robot example the first would move the y-coordinate of end-effector position to the corresponding y-coordinate of the table position and the latter would be a height adaption of the table constraint.

This extension should fix the sliding issue, but will not be the best strategy if the disturbances are large as for the second and third scenario in the evaluations, because then the planned sliding might not be the desired behavior anymore. Also an important point is that the parametrization of the force controller will be highly sensitive to the parametrization of the used contact force model.

6.3 Insights

The main insight during this work is the way people with different backgrounds approach the problem of deriving a model predictive controller. In control theory (background based on [Mül15] and [MA17]) a MPC controller is defined and then implications on the closed-loop and thus on the underlying infinite horizon optimal control problem are made. People from trajectory optimization background define the task in terms of an optimal control problem and then try to derive an execution controller (MPC) based on an approximation of the optimal control problem. An interesting research problem could be to find bounds on the error introduced by the approximations and thus have a guarantee for e.g. recursive feasibility of the approximate constrained MPC scheme when constraints pop up. Also a nice insight is how optimal control problems are defined as trajectory optimization and what people do to reduce the complexity of the involving optimization problems by introducing structure.

Aside from these insights several technical ones are gained during this work. Mostly dominated by insights into numerics, optimization and C++ programming, because a simple simulation framework was built up from scratch. There everything from numerical integration of the dynamical equation with a Runge-Kutta method over contact handling up to visualization with OpenGl of the progress is needed to setup properly. Hence I gained lots of C++ experience and learned a lot about numerical optimization and interfacing to Lapack through the *self made* array class (by the mlr).

6.4 Conclusion

In this work the control strategy *approximate constrained model predictive control* is derived in detail and a design procedure based on the solution of a trajectory optimization for planning is provided. This method is based on the fixed Lagrangian and uses as optimization backend the k-order constrained path optimization framework for a fast and reliable solution of the involving optimization problem. The evaluations have shown that this method is capable of the execution of a compliant motion task like executing a planned trajectory from different initial condition or with disturbances without redesigning the cost-to-go approximation. Hence the approximate constrained MPC scheme will outperform error stabilization based approaches like a PID for tracking of the trajectory with respect to compliance. The main drawback is that sliding can not be ensure during the execution. But this issue may be resolves as discussed in section 6.2 if sliding is desired.

Bibliography

[Arn16] E. Arnold. *Lecture: Numerische Methoden der Optimierung und Optimalen Steuerung*. University of Stuttgart, Institute for System Dynamics. 2016 (cit. on p. 17).

[Ebe14] C. Ebenbauer. *Lecture: Optimal Control*. University of Stuttgart, Institute for Systems Theory and Automatic Control. 2014 (cit. on p. 17).

[ELT+13] T. Erez, K. Lowrey, Y. Tassa, V. Kumar, S. Kolev, E. Todorov. “An integrated system for real-time model predictive control of humanoid robots.” In: *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. Oct. 2013, pp. 292–299. DOI: 10.1109/HUMANOIDS.2013.7029990 (cit. on p. 13).

[MA17] M. A. Müller, F. Allgöwer. “Economic and Distributed Model Predictive Control: Recent Developments in Optimization-Based Control.” In: *SICE Journal of Control, Measurement, and System Integration* 10.2 (2017), pp. 39–52. DOI: 10.9746/jcmsi.10.39 (cit. on pp. 13, 19, 42).

[Mül15] M. A. Müller. *Lecture: Model Predictive Control*. University of Stuttgart, Institute for Systems Theory and Automatic Control. 2015 (cit. on pp. 13, 42).

[PKT16] M. Posa, S. Kuindersma, R. Tedrake. “Optimization and stabilization of trajectories for constrained dynamical systems.” In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. May 2016, pp. 1366–1373. DOI: 10.1109/ICRA.2016.7487270 (cit. on p. 11).

[TET12] Y. Tassa, T. Erez, E. Todorov. “Synthesis and stabilization of complex behaviors through online trajectory optimization.” In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Oct. 2012, pp. 4906–4913. DOI: 10.1109/IROS.2012.6386025 (cit. on p. 13).

[TL05] E. Todorov, W. Li. “A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems.” In: *Proceedings of the 2005, American Control Conference, 2005.* June 2005, 300–306 vol. 1. DOI: 10.1109/ACC.2005.1469949 (cit. on p. 13).

[Tou14a] M. Toussaint. *A Novel Augmented Lagrangian Approach for Inequalities and Convergent Any-Time Non-Central Updates*. e-Print arXiv:1412.4329. 2014 (cit. on pp. 16, 28).

[Tou14b] M. Toussaint. *KOMO: Newton methods for k-order Markov Constrained Motion Problems*. e-Print arXiv:1407.0414. 2014 (cit. on p. 16).

[Tou16] M. Toussaint. "A tutorial on Newton methods for constrained trajectory optimization and relations to SLAM, Gaussian Process smoothing, optimal control, and probabilistic inference." In: *Geometric and Numerical Foundations of Movements*. Ed. by J.-P. Laumond. Springer, 2016 (cit. on pp. 10, 13, 14, 16, 22).

[TRB+14] M. Toussaint, N. Ratliff, J. Bohg, L. Righetti, P. Englert, S. Schaal. "Dual execution of optimized contact interaction trajectories." In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Sept. 2014, pp. 47–54. DOI: 10.1109/IROS.2014.6942539 (cit. on pp. 12, 42).

All links were last followed on May 2, 2017.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

---

place, date, signature