Systems Theory Project Report

Nonlinear Model Predictive Control Design

Elias Niepötter (3684096)

Abstract

1 Introduction

The goal of the project is to combine the methods of Sum of Squares (SOS) Programming and nonlinear Model Predictive Control (MPC). Based on an exemplary nonlinear dynamical system an MPC is designed in such a way that it acts outside a specified region in the state space and drives it towards this region. The region is called *terminal region*. Inside the terminal region a linear controller takes over, this concept is called *Dual Mode Control*. SOS methods are used to estimate the terminal region which is a *region of attraction* of the closed loop dynamics of the linearly controlled system. The report first introduces the basics of MPC in 2. Continuing with the concept of dual mode controller design focussing on SOS methods in 3. Lastly, the methods presented are applied to the ball beam system. The results are shown and discussed in 4.

reformulate Introduction

2 Model Predictive Control

Model Predicitve Control is a flexible and powerful control strategy. It naturally handles nonlinearities constraints as well as multiple inputs and outputs. The underlying idea is based on optimal control theory. MPC is characterized by two key features. First, an iterative online optimization is performed to compute the next control input. Second, at each instance of time an Optimal Control Problem (OCP) is solved, therefore the time horizon of the OCP moves forward. [3]

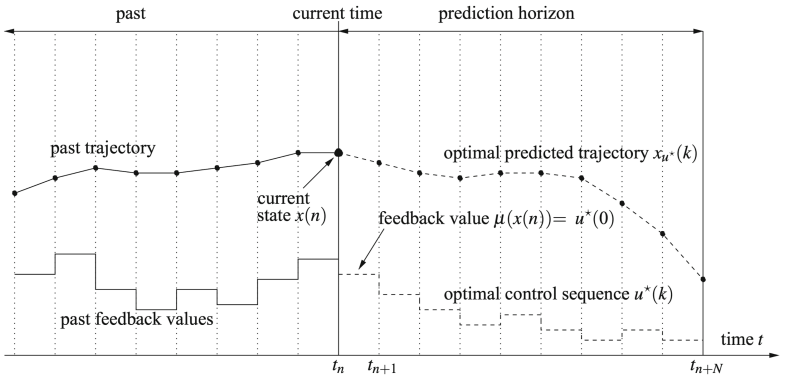

Figure 1: Moving horizon and optimization of an MPC in discrete time [3]

Figure 1 illustrates the two key feastures of MPC in discrete time. At time instance t_n the OCP for the next N time steps is solved. Given that the OCP is feasible, the first optimal control input u_0^* is applied to the system. The system evolves to the next time instance t_{n+1} and the process repeats.

2.1 Problem Formulation

The discrete time finite horizon optimal control problem can be formulated as follows:

$$\begin{aligned} \min_{\nu} \quad & E\left(x_{k+N|k}\right) + \sum_{i=k}^{k+N-1} F\left(x_{i|k}, u_{i|k}\right) \\ \text{s.t.} \quad & x_{i+1|k} = f\left(x_{i|k}, u_{i|k}\right), \quad x_{k|k} = x_k \\ & u_{i|k} \in \mathcal{U}, \quad i \in [k, k+N-1] \\ & x_{i|k} \in \mathcal{X}, \quad i \in [k, k+N-1] \\ & x_{k+N|k} \in \mathcal{E} \end{aligned} \tag{1}$$

where

$$J\left(\nu, x_k\right) = \underbrace{E\left(x_{k+N|k}\right)}_{\text{terminal costs}} + \underbrace{\sum_{i=k}^{k+N-1} F\left(x_{i|k}, u_{i|k}\right)}_{\text{stage costs}} \tag{2}$$

with $\mathcal{U}$ input constraints, $\mathcal{X}$ state constraints and $\mathcal{E}$ terminal region/set as the constraining sets. The OCP is solved for the prediction horizon N, the initial condition x_k at the current time step t_k, the dynamics of the system given by $f\left(x_{i|k}, u_{i|k}\right)$ and the terminal state $x_{k+N|k}$. The cost function (2) consists of the terminal costs at last time instance t_N of the optimization and the stage costs for all other time instances starting from the initial condition. The optimal control sequence is defined as

$$\nu^* = [u^*_{k|k}, \ldots, u^*_{k+N-1|k}]^T. \tag{3}$$

The control input μ at each discrete time step t_k is the first element of the optimal control sequence ν^*

$$\mu(x(t_k)) \coloneqq \nu^*(0) \tag{4}$$

See [3] and [4] for details on the formulation of the OCP resp. the MPC formulation.

2.2 Terminal Conditions for MPCs

The shown MPC includes terminal conditions. These conditions are part of the design of the MPC and are used to guarantee certain stability properties of the closed loop system. The followong Theorem connects terminal conditions and stability of the closed loop system. [4]

Theorem 1

The closed loop system is asymptotically stable if the optimal control problem is feasible at the first time instant and the following assumptions are satisfied for a terminal cost E, a terminal region $\mathcal{E}$, and a locally stabilizing control law $u_k = \varphi\left(x_k\right)$:
[A1] $E\left(x_k\right) > 0, \forall x_k \in \mathbb{R}^n \backslash \{0\}$
[A2] $\mathcal{E} \subseteq \mathcal{X}, 0 \in \mathcal{E}$
[A3] $\varphi\left(x_k\right) \in \mathcal{U}, \forall x_k \in \mathcal{E}$
[A4] $f\left(x_k, \varphi\left(x_k\right)\right) \in \mathcal{E}, \forall x_k \in \mathcal{E}$
[A5] $E\left(f\left(x_k, \varphi\left(x_k\right)\right)\right) - E\left(x_k\right) \leq -F\left(x_k, \varphi\left(x_k\right)\right), \forall x_k \in \mathcal{E}.$

Several approaches based on Theorem 1 exist to ensure closed loop stability. Those investigated in this work are the following:

Zero Terminal State Constraint
The idea is to use $E(x_k) = 0$, $\mathcal{E} = \{0\}$ and $\varphi(x_k) = 0$. No linear controller is used. The stability of the MPC is guaranteed through $x_{k+N|k} = 0$. When assuming an equlibrium at the origin of the system, this ensures stability at the last step of the OCP.

Terminal Region
No terminal costs are used, but a terminal region $\mathcal{E} = \{x_k \in \mathbb{R}^n \mid x_k^T P x_k \leq \alpha\}$ is choosen as well as a locally stabilizing linear controller $\varphi(x_k) = Kx_k$. This approach is also *dual mode control*.

Terminal Region and Terminal Cost
The so-called quasi-infinite horizon MPC uses a terminal region $\mathcal{E} = \{x_k \in \mathbb{R}^n \mid x_k^T P x_k \leq \alpha\}$ and a terminal cost $E(x_k) = x_k^T P x_k$ as well as locally stabilizing linear controller $\varphi(x_k) = Kx_k$. The terminal cost E, terminal region $\mathcal{E}$ and locally stabilizing linear controller $\varphi(x_k)$ are calculated off-line. The procedure is described in section 3.

2.3 Variations of MPC

There are several variations of the MPC formulation. Two important distinctions have to be made regarding the length of the prediction horiton. One differentiate between *finite* and *infinite* horizon optimal control. presented above is the finite horizon MPC. The infinite horizon MPC, as the name implies, has an infinite prediction horizon $N \to \infty$. It can be shown that the inifinte horizon MPC asymptotically stabilizes the system, without need of any terminal conditions. Hence the name *quasi-infinite horizon* is motivated by this property. Furthermore, it should be differentiated between *stabilizing* and *tracking* MPC. The stabilizing MPC is used to stabilize the system around an equilibrium point. The tracking MPC is used to follow a reference trajectory as good as possible. For the stabilizing MPC the reference can assumed by constant zero, when the system is stable around the origin. Given the definitions above, it is possible to introduce the tracking MPC by utilizing the difference, or error, of the current state to the reference $e_k = x_k - x_{\text{ref},k}$ as a new state. In other words, the tracking MPC is a stabilizing MPC for the error dynamics of the system. Tracking MPC won't be considered in the follwing. For more details see [3].

3 Synthesis of Terminal Conditions

Followed by the introduction of terminal conditions and their application in MPC, the synthesis of terminal cost E, the terminal region $\mathcal{E}$ and the locally stabilizing linear controller $\varphi(x_k)$ is presented. The procedures are based on [1]. This section draws the connection to SOS programming. The procedure is as follows:

Step 1. Solve the linear control problem based on the jacobian linearization of the nonlinear system and obtain the feedback matrix K.

Step 2. Choose a constant $\kappa \in [0, \infty)$ satisfying

$$\kappa < -\lambda_{\max}(A_K) \tag{5}$$

where $A_K \coloneqq A + BK$ expressing the closed loop dynamics of the linearized and locally controlled system. The operator $\lambda_{\max}(\cdot)$ denotes the maximum eigenvalue of a matrix. Furthermore, solve the lyapunov equation

$$(A_K + \kappa I)^{\mathrm{T}} P + P (A_K + \kappa I) = -Q^* \tag{6}$$

where $Q^* = Q + K^T R K \in \mathbb{R}^{n \times n}$ itself is positive definite and symmetric. The obtained matrix P is positive definite and symmetric. The matrices Q and R might be used for *Step 1* to obatain the linear feedback matrix K based on LQ-regulators.

Step 3. Find the largest possible α_1 s.t $Kx \in \mathcal{U} \quad \forall x \in \Omega_{\alpha_1}$.

Step 4. Find the largest possible $\alpha \in (0, \alpha_1]$ s.t. $\{x_k \in \mathbb{R}^n \mid x_k^T P x_k \leq \alpha\}$ is an inner approximation of the region of attraction of the closed loop dynamics of the linearly controlled system.

3.1 Lyapunov's second method for stability

Source?

In order to estimate a region of attraction, Lyapunov's second method for stability is used. The method makes use of the concept of *dissipativity*. Consider the dynamical system $f(x) = \dot{x}$ and a *storage function* $V : \mathbb{R}^n \to \mathbb{R}$ where $\mathbb{R}^n$ denotes the state space. The system is asymptotically stable in the sense of Lyapunov if the following conditions hold:

$$V(x) = 0 \text{ if and only if } x = 0 \tag{7}$$
$$V(x) > 0 \text{ if and only if } x \neq 0 \tag{8}$$
$$\dot{V}(x) = \nabla V \cdot f(x) < 0 \quad \forall x \neq 0 \tag{9}$$

In that case, $V(x)$ is calles a *Lyapunov function*. Note that these conditions are not sufficient to prove global asymptotical stability. For that to be the case, $V(x)$ also has to be radially unbounded.

3.2 SOS Programming [2]

SOS programming is used to *verify* wether a given, potentially multivariate, polynomial $p(x) \in \mathbb{R}[x]$, where $\mathbb{R}[x]$ denotes the ring of polynomial in $x \in \mathbb{R}^n$ with real coefficients, is a sum-of-squares polynomial. A polynomial $p \in \mathbb{R}^n$ is sum-of-squares if there exists $q_1, \ldots, q_m$ s.t.

$$p(x) = \sum_{i=1}^{m} q_i^2(x) \tag{10}$$

The set of SOS polynomials will be denoted by $\Sigma[x]$. If a polynomial is sum-of-squares, it is also a nonnegativ polynomial $p(x) \geq 0$. Subsequently, if we want to proof wether a polynomial is nonnegative or not, we can now relax this problem to the question, wether the the polynomial is sum-of-squares or not. The advantage of this relaxation over the orginal problem is that solving the problem to verify if a polynomial is sum-of-squares can be done in polynomial time. Proofing if a polynomial is nonnegative is a NP-hard problem. The verification on whether a polynomial is sum-of-squares or not is done by solving a Semidefinite Program (SDP).

Lemma 1. Let $p(x) \in \mathbb{R}[x]$ be a polynomial of degree d. Then, $p(x) \in \Sigma[x]$ if and only if there exists a symmetric matrix $Q \in \mathbb{S}^l$ being positive semidefinite (all eigenvalues of Q are nonnegative) such that

$$p(x) = z(x)^\top Q z(x) \tag{11}$$

for some monomial vector $z(x) \in \mathbb{R}[x]^l$ with $\deg(z(x)) \leq \frac{d}{2}$.

Using Lemma 1, we further relaxed the problem of verifying if a polynomial is sum-of-squares to the problem of finding a positive semidefinite matrix Q. This task is done via a SDP.

More details in SDP?

The following procedure, called *Generalized S-procedure*, is used to solve the set containment problem:

Generalized S-procedure [2]

Given $h, f_0, \ldots, f_r \in \mathbb{R}[x]$, if there exist $p \in \mathbb{R}[x]$, and $s_1, \ldots, s_f \in \Sigma[x]$ s.t.

$$\text{ph} \ -\sum_{j=1}^{r} s_j f_j + f_0 \in \Sigma[x] \tag{12}$$

then the following set containment holds:

$$\{x \in \mathbb{R}^n \mid h(x) = 0, f_1(x) \geq 0, \ldots, f_r(x) \geq 0\} \subseteq \{x \in \mathbb{R}^n \mid f_0(x) \geq 0\} \tag{13}$$

3.3 Sublevel set optimization

In *Step 3* and *4* one tries to find the largest sublevel set of a given function satisfying certain constraints. These optimization problems can be formulated to one single problem. The idea is to setup a nonlinear root search for the parameter α. An iterative algorithm, e.g. a bisection algorithm , is used to maximize the parameter. The problem can be formulated as follows:

short description of the algorithm

$$\max_{\alpha} \quad \mathrm{V}\left(\{x_k \in \mathbb{R}^n \mid x_k^T P x_k \leq \alpha\}\right) \tag{14}$$

The resulting constraints for the optimiztion are

Finish full problem formulation

$$s_i \in \Sigma[x] \tag{15}$$

$$-s_i(-x^T P x + \alpha_1) + (Kx + u_{\max,i}) \in \Sigma[x] \tag{16}$$

$$-s_{N_u+i}(-x^T P x + \alpha_1) + (-Kx - u_{\min,i}) \in \Sigma[x] \tag{17}$$

4 Exemplary Application

Implementation details? Multiple Shooting?

Based on an exemplary system a MPC with terminal conditions is designed using the procedures described above. The exemplary system is the ball beam system. It consists of a ball rolling on a beam. The beams angle is controlled by a motor. Gravity acting on the ball accelerates it accoding to the angle of the beam. The system is described by the following equations of motion:

$$0 = \left(\frac{J}{R^2} + m\right)\ddot{r} + mg\sin(\alpha) = -mr\dot{\alpha}^2 \tag{18}$$

States are chosen as

$$x = \begin{bmatrix} r & \dot{r} & \alpha & \dot{\alpha} \end{bmatrix}^T \tag{19}$$

and the input as

$$u = \ddot{\alpha} \tag{20}$$

$$\dot{x} = f(x,u) = \begin{bmatrix} x_2 \\ mx_1x_4^2 - mg\left(\frac{x_3^5}{120} - \frac{x_3^3}{6} + x_3\right)\cdot k \\ x_4 \\ u \end{bmatrix}$$

Note that the sin() term is approximated by a Taylor series epansion of degree five in order to obtain a polynomial system. The constant k combines the physical parameters of the system.

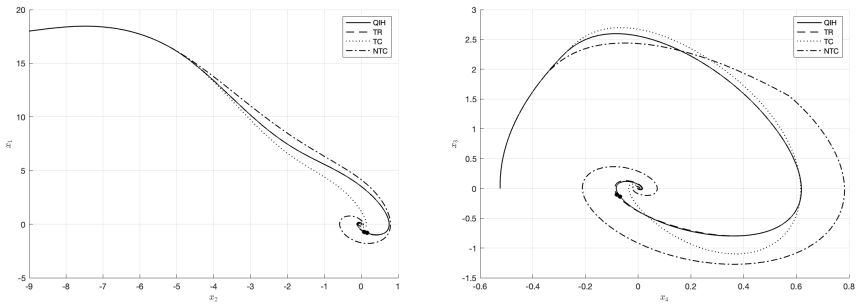

Figure 2: state space trajectories for defined control setups

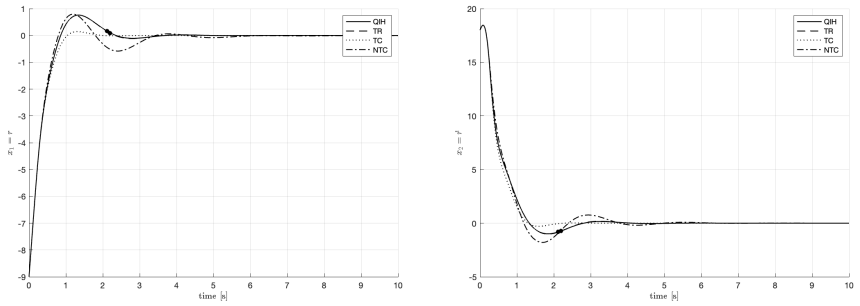

Figure 3: trajectories for states x_1 and x_2

5 Conclusion & Outlook

Briefly introduce the other usage of SOS covered by

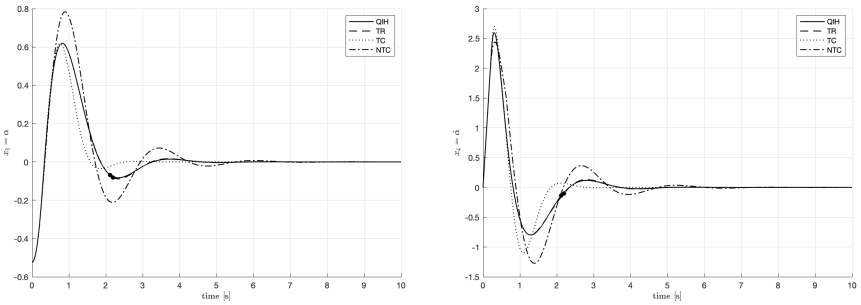

Figure 4: trajectories for states x_3 and x_4

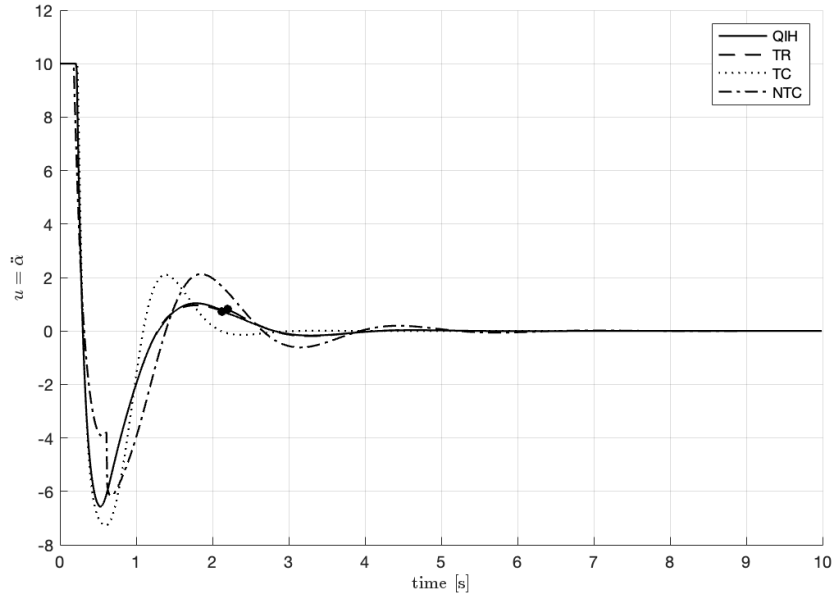

Figure 5: trajectories for input u

References

[1] H. Chen and F. Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1217, 1998.

[2] Torbjørn Cunis and Renato Loureiro. Systems theoretical methods for flight control, 2023.

[3] Jürgen Pannek Lars Grüne. *Nonlinear Model Predicitve Control*. Springer Cham, 2 edition, 2016.

[4] T. Raff, C. Ebenbauer, R. Findeisen, and F. Allgöwer. *Nonlinear Model Predictive Control and Sum of Squares Techniques*, volume 10, pages 325–344. 07 2007.

Todo list