KARLSRUHE INSTITUTE OF TECHNOLOGY

Project Rest-to-Rest of a 2-DoF Robot

Hesham Hendy

October 18, 2019

Abstract

In industrial manufacturing , the use of robots is progressively growing. According to that, optimized control of robots becomes a very interesting research field. Dependent on the robot degrees of freedom and the required target we find ourselves confronted with a main issue, which is an optimal trajectory for the robots joints and its configuration. Many aspects should be considered by designing the robot like cost-effectiveness, low power consumption or high production speeds.

Key words: NMPC, Energy Efficiency, Robots, Optimization, Sustainable manufacturing, Research activities

CONTENTS

1 INTRODUCTION

Nonlinear Model Predictive Control (NMPC) becomes one of the interesting possibilities to provide an optimized solution regarding the previous mentioned aspects. This technical documentation explains many tested approaches for solving Rest-to-Rest control of a 2 DoF Robot problem. By designing such a problem one should considered five main pillar . First, we should find an admissible controls. Second, Mathematical Description of the plant should be available. Third, Performance criterion should be specified. Forth, Plant constrains are very crucial and limiting our solution and must be precisely defined and satisfied by solving. At the end comes the optimal criteria . Three Main Issues arise by designing an NMPC namely, Feasibility, Convergence and Stability. Theses issues decide if we are going to find an optimal solution or not.

2 MODELING

2.1 BASIC MODEL

This project aims showing an optimal transformation of grip hand from a given start position to a defined end position as energy- and time-efficiently as possible using 2 joints. Using a mathematical model we can calculate states of the system by substituting in the model.

$$x_1 = l_1 * cos(q_1) \qquad y_1 = l_1 * sin(q_1) \tag{2.1}$$

$$x_2 = l_1 * cos(q_1) + l_2 * cos(q_1 + q_2) \qquad y_2 = l_1 * sin(q_1) + l_2 * sin(q_1 + q_2) \tag{2.2}$$

The Newton-Euler formulation is based on a balance of all effected forces and counteracting forces (if available) and counteracting mass inertia. This approach uses the mathematical implementation for transnational and rotational motion of a rigid body. Their motions are often referred to a Cartesian coordinate system. The equations can be read easily from the graph. Notice, choosing the coordinates is very critical as we try to pick only minimum number of coordinates by using generalized angle coordinates.

Lagrange function 2.3 describes the conservation of Energy in a system, where T is the kinetic energy.and V is the potential energy of the considered system. The Lagrange formulation works with generalized coordinates effectively describing the link positions of the manipulator in relation to a fixed reference coordinate system. Its solution gives directly Motion equations of the joints. It should be noted that the generalized coordinates are a minimal set of independent coordinates that provide a clear description of the system. The number of generalized coordinates was introduced to be the number of coordinated, which are necessary to decide the configuration of a set of points. [1].

In order to obtain the desired equations of motion from the Lagrangian function, we should choose first the generalized coordinates. Deflection angles of every joint should be chosen as generalized coordinates. For a clear and Full description of the grip in the space,Two joint angles q_1 and q_2 are needed. Throughout this choice the generalized forces in the form of moments directly in the q_1 - $q_2$coordinate system can be represented without another transformation. The Lagrange formulation comprises of two basic formulas which represent the energy conservation as following

$$\mathscr{L} = T - V \tag{2.3}$$

and

$$\frac{d}{dt}\left(\frac{\partial}{\partial \dot{q}_i}\right) - \frac{\partial}{\partial q_i} = Q_i \tag{2.4}$$

Where Q_i is the generalized force, which can be contributed to the input torque u in the exercise description. T represents the kinetic energy and V is the potential energy the whole system possesses. The kinetic energy and the potential energy can be calculated by derivation of 2.1 and 2.2 and inserting in

$$T = \frac{1}{2} * m_1 * \dot{x}_1^2 + \frac{1}{2} * m_1 * \dot{y}_1^2 + \frac{1}{2} * m_2 * \dot{x}_2^2 + \frac{1}{2} * m_2 * \dot{y}_2^2 \tag{2.5}$$

$$V = m_1 * g * l_1 * sin(q_1) + m_2 * g * (l_1 * sin(q_1) + l_2 * sin(q_1 + q_2) \tag{2.6}$$

m_1 and m_2 are the masses of the rigid arms. By solving 2.3 and 2.4, we come to model (1) from the exercise.

2.2 Reformulation in state space representation

Reformulation the resulted second order differential equation

$$B(q) * \ddot{q} + C(q, \dot{q}) * \dot{q} + g(q) = u \tag{2.7}$$

into a first order system in state space form $\dot{x} = f(x, u)$. Picking the angles and its differentiation the angular velocity as state vector x.

$$x = \begin{bmatrix} q_1 \\ q_2 \\ \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \tag{2.8}$$

Differentiating the equation 2.10 w.r.t time and introducing a state vector x in relation to the joint angles qi and their derivatives q leads to:

$$\dot{x} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \\ \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} \tag{2.9}$$

with

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} = B(q)^{-1} * u - B(q)^{-1} * (C(q, \dot{q}) * \dot{q} + g(q)) \tag{2.10}$$

The state space representation in the form $x = f(x; u)$ can now be implemented in Matlab as a function, which forms the basis for model calculations.

3 Open Loop Optimal Control

In this chapter, two open-loop control problems should be designed with respect to the given constraints. The optimal control problems (OCP) should reach out an end-state $q = (\pi/2, 0)^{\mathrm{T}}$ rad from a starting-state $q = (-5, -4)^{\mathrm{T}}$ rad in a fixed time of 3s. Standard NLPs, which are used to solve OCPs, have the form as

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq b_i,\ i = 1, \ldots, m. \end{aligned}$$

where $f_0(x)$ is the objective function to minimize to and their constraints $f_i(x) \leq b_i$ delimiting the feasible solution area.

3.1 Problem formulation

There are three problem formulations of OCPs with fixed start and End time,and start and end position set up. The first objective function (OCP1) only minimizes the energy that the robot needs for the movement process. through minimizing the Amplitude of the input over the entire time interval.

$$\underset{u(*)}{\text{minimize}} \quad \int_0^{t_e} u^{\mathrm{T}} * R * u \quad dt$$

The second objective function (OCP2) minimizes the deviation of the current state vector from the target state. Since the state vector x considers the angular velocities in x_3 and x_4 and they become zero at the final state x_e, a high angular velocity is punished and this is not necessarily, so we changed corresponding entries in the weighting matrix Q to avoid that. Therefore, states 3 and 4 should be weighted much less than the angular positions.

$$\underset{u(*)}{\text{minimize}} \quad \int_0^{t_e} x^{\mathrm{T}} * Q * x + u^{\mathrm{T}} * R * u \quad dt$$

An important aspect would be in consider the time as well, so that we can have the shortest time possible in running.OCP3 in 3.1 describes that:

$$\underset{u(*)}{\text{minimize}} \quad \int_0^{t_e} 1 \quad dt$$

The constraints stay the same, but the optimization focus changes and previous OCPs becomes a new optimal control problem. At some point of optimization it is needed to discretize the state trajectory, because the input trajectory given to the engines is also in a close discrete way. So in this project the state space model the integration is fullled with a Euler forward,midpoint, a Heun's method or optionally Runge-Kutta scheme for multiple small time steps in an bordered iterations loop.

$$
\begin{aligned}
\text{Constrains:} \quad & \dot{x} = f(x,u) \\
& t_e = 3 \\
& x(0) = [-5,0,-4,0]^{\mathrm{T}} \\
& x(t_e) = [\frac{\pi}{2},0,0,0]^{\mathrm{T}} \\
& x_3(t), x_4(t) \in [-\frac{3}{2}\pi, \frac{3}{2}\pi] \\
& u_1(\bullet), u_2(\bullet) \in [-1000, 1000]
\end{aligned}
$$

3.2 Comparison different sampling points and integrators

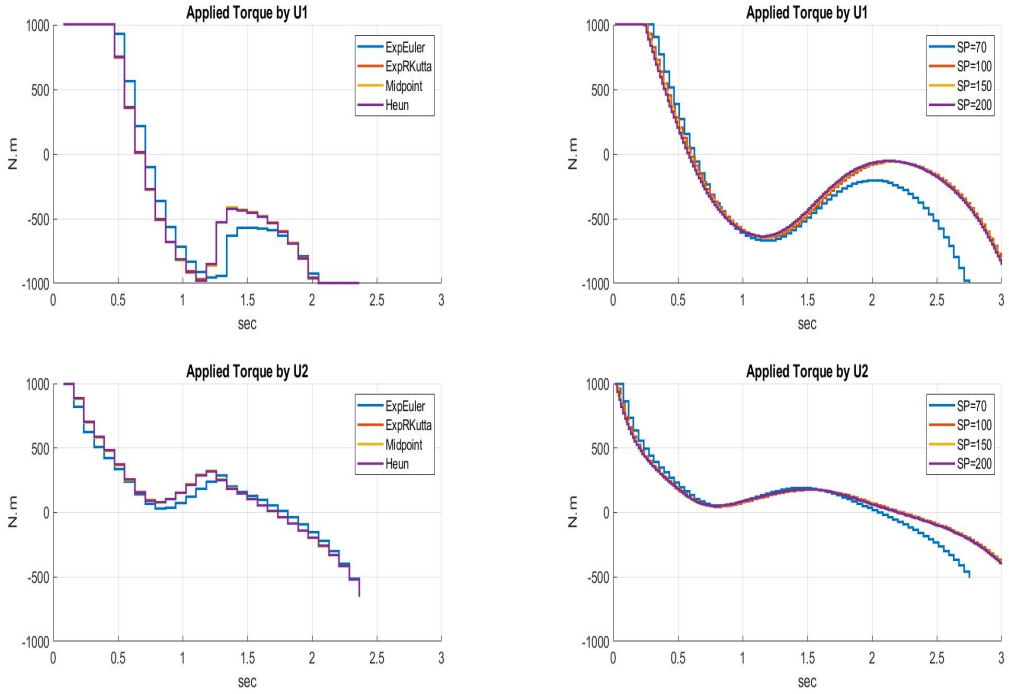
For the numerical integration different methods are available. In my implementation results will be shown with respect to the objective function of OCP3 on different integrators like 'Midpoint' 'Euler-forward', 'Heun's method' 'Runge-Kutta schemes'.

The simplest method is integration according to Euler explicit method. Here, the area under a curve f in Interval of the selected step size h is approximated as a rectangle with the height $f(t_k; x(t_k))$. As the exact method depends on the selected step size h (which at a fixed end time directly related to the sampling points number) and from the slope of the curve f as well. For large gradients and large integration error as a consequence,the curve can be described as purely approximated by a rectangle. A much more precise method is the fourth-order Runge-Kutta method. Here, it is described in simplified form through combination of different difference quotients in the interval h.

The selection of sampling points always cares for a compromise between computational effort and accuracy. 3.1b shows a variation of sampling points N = 70 to N = 200 using the same integrator, which is explicit Euler. It can be clearly seen that from N = 100 the curves take a similar course. Notice less that 70 Sp has showed infeasibility as matter of convergence. It should be noted that the cost functional and its complexity decide significantly more or less sampling points are needed to to get an optimal solution.

3.3 Adapting weighting matrix

The task in the last OCP concerning exercise part is to simulate the previous problem formulation numerically, detecting the systems minimal time needed and solving a problem reformulation for a compromise of minimal time and minimal control energy. After implementation of the minimal time problems formulation the solver accepts a minimal time of 2.39. As explained before, the solver tries to find a solution according to the feasible region of the constraints. So the result of the input trajectory of the minimal time problem has a similarity to a bang-bang control. The engine torques are frequently switched from the positive maximum torque to the negative maximum torque with very steep edges.



(a) Different Ints

(b) Different SPs

Figure 3.1: Generated Input Torque

To find a compromise between minimal time costs and minimal needed control energy this particular project exchanges the objective function. So from here the function to minimize to is described by

$$J = x(t_e) * G * x(t_e) + \int_0^\tau x * Q * x + u^{\mathrm{T}} * R * u dt \tag{3.1}$$

Where G gives a weight on the time optimality and $Q = 1$ for path optimality and $R = 100$ for input optimality .

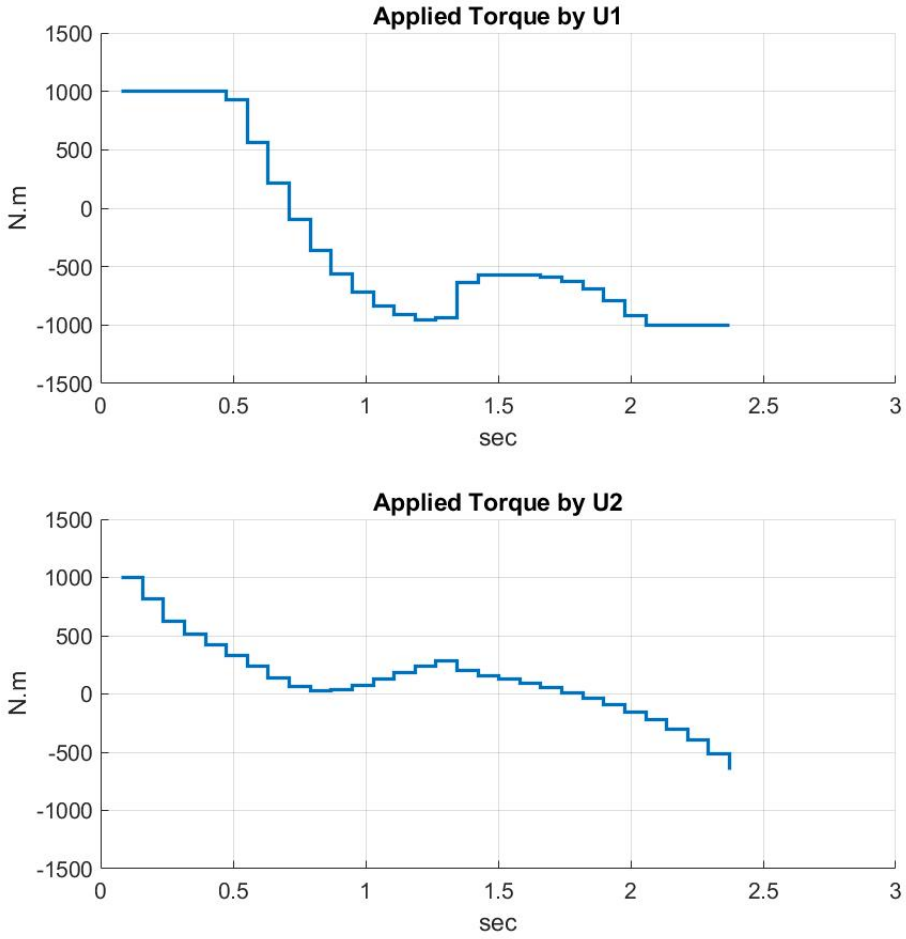


Figure 3.2: Quadratic Cost Function Performance

4 Model Predictive Control

4.1 Model uncertainties and disturbances

The mathematical modeling of the robot can lead to deviations from reality.

Uncertainties could be the following:

- varying starting and end points
- sensors have disturbances noise
- grasping or loosing of parts may change the inertia and accordingly the of the model
- not fully reachable working space, bending joints or floors in the space
- nonlinear engine behaviour
- Simplified assumptions in the parameters

Therefore, a compromise must always be found between model quality and the effort to create the corresponding model. For example Dissipative effects can often only be represented by a great deal of effort, but they can mostly neglected. A flawed model, by means of a changed dynamics be modeled. This error can be eliminated by a control (MPC) for some extent.

It is possible to describe the deviation in actual state vector in sensor input signals x caused by noise or tolerances like

$$x = x_{sensor} + x_{noise} \tag{4.1}$$

The equation 4.1 is only useful, if the sensor noise is involved before the controls design.We use Gauss disturbance as a noise model in our simulation.

Notice, Grasping or loosing of parts will lead to differing inertia and mass inertia. Due to the fact, that while modeling already most inertia are considered in the Lagrangian formulation it could lead to unstable behaviour of the control.

4.2 MPC

A infinite horizon suboptimal controller can be designed by repeatedly solving finite time optimal control problem in a receding horizon fashion. The only difference in this OCP is the initial state variable x_{nmpc0} which is modified by the controller in each loop to be considered as the new initial state at each sampling point, i.e an open-loop supoptimal control problem is solved over a finite horizon. The resulting controller is referred to as a Receding Horizon Controller (RHC). A Receding Horizon Controller where the finite time optimal control law is computed by solving an optimization problem on-line is called Model Predictive Control (MPC). For MPC,the cost functional should be changed compared to the OCP2 and a penalty matrix from the deviation should be included in order to achieve the target quickly as possible. Moreover, high amplitudes of the input signals are penalized to find a compromise between speed and energy consumption. Since a closed-loop time horizon of t_{hor} = 1s

is selected, the number of sampling points can be set to 50 be reduced without increasing the interval steps.

$$\underset{u(*)}{\text{minimize}} \quad \int_0^{t_e} (x(t) - x(t_e))^{\mathrm{T}} * Q * (x(t) - x(t_e)) + u^{\mathrm{T}} * R * u \quad dt$$

The first entry will be initialized in the code with the vector $x(0) = [-5, 0, -4, 0]^{\mathrm{T}}$. For completion of the MPC the OCP is solved in a loop until the end-state x_{end} is reached. At each iteration the first entry of the input trajectory $U(1)$ for the current initial state x_{nmpc0} is applied to a Runge Kutta 4th order integrator, which adopts the input state vector and calculates the state trajectory x_{nmpc} of the joints.

4.3 Comparin MPC with OCP considering incorrect parameters

Through simplified assumptions in the parameters, such as a friction-less system or a faulty system identification, errors in model identification can be examined. 4.1 recognizes that the system without feedback (OCP) has not been reaching the desired final state and deviating from OCP with correct parameters, because calculation of the optimal control trajectory is not based on precise model. With the closed loop (MPC error) it is clear that the controller has been achieving the desired target. By MPC the faulty model consider as dictation of the value, that should be taken at the beginng but after that the controller continues building up on itself. This will ensure that the desired destination state is reached.

4.4 Comparison MPC considering correct and incorrect parameters

In this part of the exercise an additive gaussian measurement error has to be applied to the simulation. Since measurement technology is needed in reality for the determination of the state vector and this could be noisy, the influence of noise should be examined separately to understand its effect. Here, white Gaussian noise has been added to the state vector determined for us. In figure 4.2 and 4.3 it can be seen that, despite noise, the target state can be reached. On the basis of the input signals one can see clearly that the system reacts actively to the changes. Thus the regulation can suppress the noise up to a certain extent.

5 Stability check

Throughout an optimal solution can be figured out if the system is stable or not, thus separate stability analysis must be carried out.
For this first there are three standing assumptions from the lecture, that should be available:

A1 -> $SteadyState$.
That means that the system can keep the equilibrium at given using and the result of $x = f(x;u)$ becomes 0 due to the equilibrium, In our case it's the (,) =(0,0). One can see that the starting state x_s = [π / 2; 0; 0; 0] is already in equilibrium and for that u = [0,0]

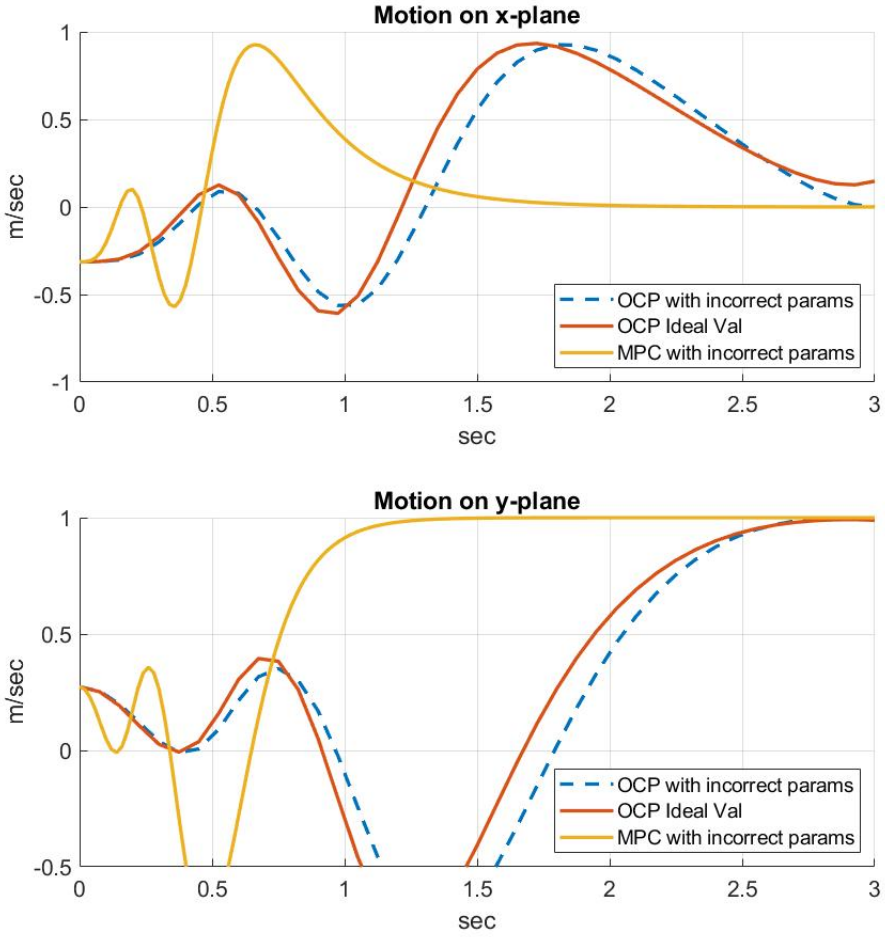


Figure 4.1: Comparing MPC with OCP considering the disturbance

A2 → *LowerBoundness L.*
There are class where a class - K function with α that satisfies $L(x,u) \geq \alpha(\| x - \bar{x} \|)$ and L(0,0)=0. The used cost functional is postive and equal zero if the inputs are zero as well.

A3 → *AbsoluteContinuity L.*
For all $x_0 \in X$ and u ∈ C([0,T],U), the solution exists in the same period.That has been satisfied through the dynamics of the system, which is continuous.
Note, The assumption (SA) guarantees the existence of an optimal trajectory.

If we consider our value function $V_N(\hat{x}_i)$ as a MPC Feedback of a discrete-time Optimal

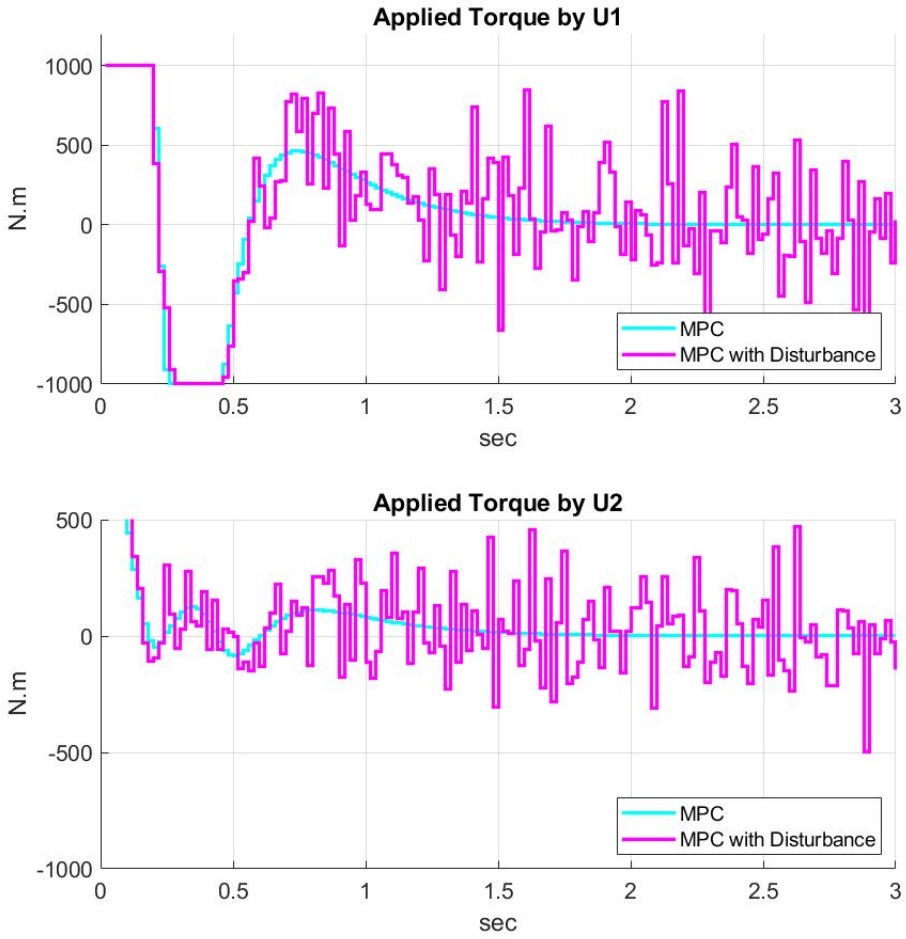


Figure 4.2: Comparing MPC inputs considering the disturbance

Control Problem (OCP) as following:

$$
\begin{aligned}
\underset{u(*)}{\text{minimize}} \quad & V_N(\hat{x}_i) := min \sum_{k=0}^{N-1} l(x_k, u_k) + V_f(x_N) \\
\text{subject to} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = \hat{x}_i \\
& x_k \in \mathbb{X}, u_k \in \mathbb{U} \forall k \in 0, ..., N \\
& x_N \in \mathbb{X}_f
\end{aligned}
$$

where
$l_(x_k, u_k) \rightarrow$ Cost function
$V_f(x_N) \rightarrow$ Terminal cost.

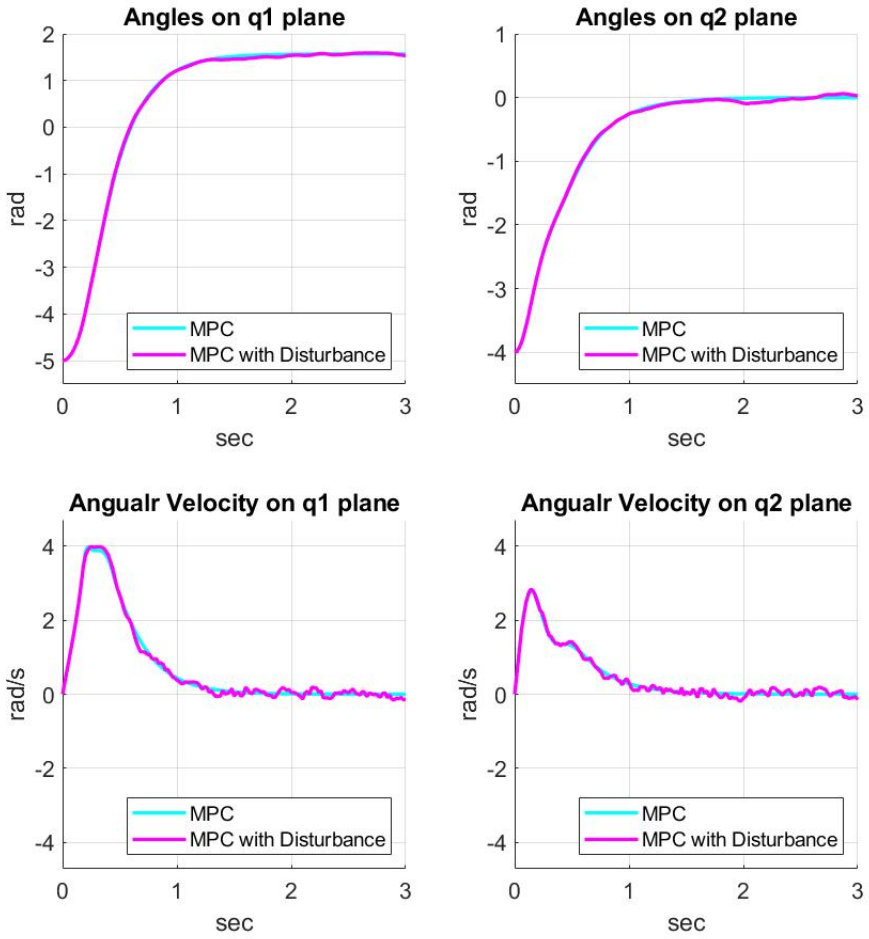


Figure 4.3: Comparing MPC Outputs with considering the disturbance

If the Assumptions are completely fulfilled and the value function $V_N(\hat{x}_i)$ is continuously diffrentiable and its differential is bounded,then our MPC controller is able to achieve uniformly stability at x=0.[2] [3]. Notice our value function has a decreasing property on the terminal set $\mathbb{X}_f$ and through that it becomes a Lyaponov function candidate which satisfies stability of instantaneous infinite-horizon and the convergence of sampled data is granted.[4]. Stated differently:

$$\forall x \in \mathbb{X}_f \exists u \in \mathbb{U} \longrightarrow V_f(f(x,u)) - V_f(x) \leq -l_(x,u)$$

From the output behaviour our robot, it can be identified that the terminal cost shows a descrease property on terminal set as Lyapunov function like.

REFERENCES

[1] Prof. Dr.-Ing. Wolfgang Seemann,Lecture Multi-body dynamics,Script,KIT.

[2] Nonlinear Systems. Third Edition. HASSAN K. KHALIL. Department of Electrical and Computer Engineering. Michigan State University. Prentice. Hall.

[3] A. Jadbabaie, J. Yu, J. Hauser. Unconstrained receding-horizon control of nonlinear systems. In: IEEE Trans. Automat. Contr. 46.5 (2001), pp. 776-783.

[4] H. Chen and F. Allgoewer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. In: Automatica 34.10 (1998), pp. 1205 - 1217