Technical University of Dortmund

Nonlinear Model Predictive Control

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

Wayne Paul Martis

Matriculation no. 242844

Faculty : Prof.Dr.-Ing. Timm Faulwasser

Submission Date: 12 March 2024

CHAPTER 0

Contents

1 Introduction **2**
- 1.1 Problem statement . . . 2
- 1.2 Methodology . . . 3

2 Modeling **4**
- 2.1 Forward Kinematics . . . 4
- 2.2 Dynamics . . . 4
  - 2.2.1 Kinetic Energy . . . 5
  - 2.2.2 Potential Energy . . . 6
  - 2.2.3 Dynamic Model . . . 6
- 2.3 State Space . . . 7
  - 2.3.1 Reformulation as a first order system . . . 7

3 Open-loop Optimal Control **9**
- 3.1 Minimization of tracking error . . . 9
  - 3.1.1 Runge-Kutta Integration . . . 11
- 3.2 Linear Quadratic Regulator (LQR) problem . . . 13
- 3.3 Minimization of time . . . 14
  - 3.3.1 Minimization of time as a fixed end time problem . . . 15
- 3.4 Minimization of time and control input . . . 15

4 Model Predictive Control **17**
- 4.1 Uncertainties and Disturbances in the robot . . . 17
- 4.2 MPC Controller for the LQR Problem . . . 18
- 4.3 Comparison MPC (Closed loop) and Optimal Control (Open loop) for incorrect parameters . . . 19
- 4.4 MPC of the LQR problem in presence of Gaussian noise . . . 20

CHAPTER 1

Introduction

A robotic manipulator is a mechanical device that is used to control and move objects, in tasks which are difficult for humans. It finds diverse applications in fields such as welding, surgical arms and automobile industry.

Robotics manipulators can be stationary or mobile, and have multiple degrees of freedom (DoF) to facilitate flexible motion. A manipulator is built from a series of rigid bodies called links which are connected by joints. The joints house the motors which actuate the motion of robot. The base of the manipulator is usually fixed and the other end forms the end-effector for gripping etc.

1.1 Problem statement

In this project, a two degree of freedom (DoF) robot is studied for the purpose in control tasks to reach an end state from the start state. The objective is to steer the robot from an initial state of $q = (-5, -4)^T$, velocities $\dot{q} = (0, 0)^T$ to the end state $(= (\pi/2, 0)^T$, velocities $\dot{q} = (0, 0)^T$. In order to acheive this optimal control and model predictive control will be applied. The problems are explained in detail in the following chapters.

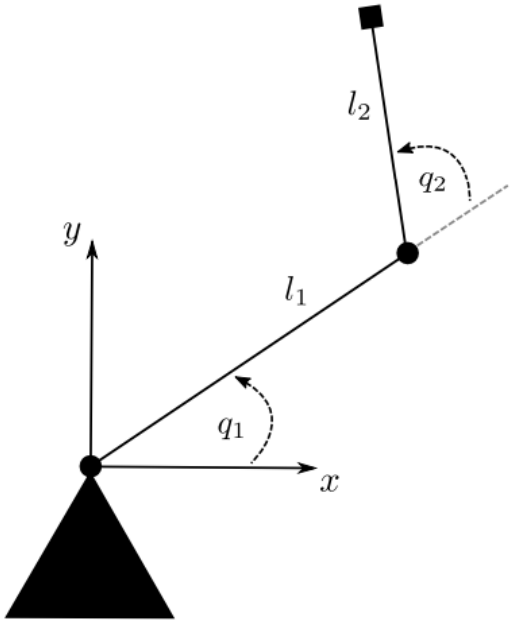


Figure 1.1: A 2-DoF robotic arm

As shown in fig. 1.1, the robot consists of two links of length of length l1 and l2; revolute joints with angles q1 and q2. The robot is fixed at the base.

1.2 Methodology

The objective of the problem is to arrive at a mathematical model of the robot with a an appropriate state space to serve as a basis for analysing the dynamics during application of the control.

- A dynamic model of the robot is derived in terms of the joint angles and the inputs. This model is formulated in a form appropriate for simulation.
- The simulation tool used with be Matlab owing to its ease of defining numerical algorithms and ODEs. It also vectorizes operations that allow efficient implementation of algorithms and provides powerful visualization tools.
- CasADi will be used as a solver in Matlab as it specialized in optimization and control. It has efficient tools for integration such as IPOPT which help to obtain accurate results.

CHAPTER 2

Modeling

Understanding the structure and motion of the 2-DoF robot is a key for applying any control algorithm to the robot. The robot consists of two links *l1* and *l2* and, two joint angles *q1* and *q2*. These two angles are actuated by motors, which effectively moves the end effector to the desired position *(x,y)*. In order to manipulate an object with the end effector, it is necessary to find a relationship between the join angles and the end effector position. This relation is given by the forward kinematics.

2.1 Forward Kinematics

The forward kinematics of the 2-DoF arm can be derived using trigonometry and Denavit-Hartenberg (DH) convention. The joint space of the robot formed by the joint angles is given by

$$q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \quad \in \mathbb{R}^2$$

The end effector coordinates, in which the manipulator task is specified, define the operational space of the robot, which are

$$p_e = \begin{bmatrix} x \\ y \end{bmatrix}$$

Hence, the name 2 degree of freedom (2-DoF) robot. From the manipulator geometry, the forward kinematics can be derived as shown in the equations below

$$x = l1 \cdot \cos(q_1) + l2 \cdot \cos(q_1 + q_2)$$
$$y = l1 \cdot \sin(q_1) + l2 \cdot \sin(q_1 + q_2)$$

2.2 Dynamics

The dynamic model of a manipulator plays an important role in simulating the control strategies and motion planning. An appropriate analogy for visualising the dynamics of this robot is a double pendula.

The dynamic model is derived based on the energy of the system using the Lagrange Formulation. The Lagrangian of a mechanical system is given by

$$L = T - U \tag{2.1}$$

where T and U denote the kinetic and potential energy of the system.

The Lagrange equations can be expressed as

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right)^T - (\frac{\partial L}{\partial q})^T = \xi \tag{2.2}$$

where ξ is the generalized force in the generalised coordinates q.

2.2.1 Kinetic Energy

Let us analyse the kinetic energy of the system. Let $m1$ and $m2$ be the masses of the links. The total kinetic energy T of the system is given by

$$T = \sum_{i=1}^{n}(T_{vi} + T_{\omega i})$$

where T_{vi} is the translational kinetic energy and $T_{\omega i}$ is the rotational kinetic energy of the i th links. Furthermore, the kinetic energy due to the motor could also be considered. The translational kinetic energy can be defined for each link as

$$T_{vi} = \frac{1}{2} m_i \dot{p_{\ell i}}^T \dot{p_{\ell i}}$$

where $\dot{p_{\ell i}}$ is the linear velocity of the center of mass of link i.

The rotational kinetic energy is given by the equation

$$T_{\omega i} = \frac{1}{2}\omega_i^T R_i^T I_{\ell i}^i R \omega_i$$

The inertia tensor of a link, $I_{\ell i}^i$ is the inertia expressed with its own frame and R_i is the rotational matrix with respect to the base. The linear and angular velocities can be re-written using their corresponding linear jacobian $J_P^{(\ell i)}$ and angular jacobians $J_O^{(\ell i)}$ as follows

$$\dot{p_i} = J_P^{(\ell i)}\dot{q}$$

$$w_i = J_O^{(\ell i)}\dot{q}$$

Substituting for the terms in the above equations we arrive at the equation for kinetic energy as

$$T = \frac{1}{2} m_i \dot{q}^T J_P^{(\ell i)T} J_P^{(\ell i)} \dot{q} + \frac{1}{2} m_i \dot{q}^T J_O^{(\ell i)T} R_i I_{\ell i}^i r_i^T J_O^{(\ell i)} \dot{q}$$

A similar analysis can be done for the motor kinetic energy. Finally, the total kinetic energy can be expressed in quadratic form as

$$\mathbf{T} = \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}(q)\dot{q}_i\dot{q}_j = \frac{1}{2}\dot{\mathbf{q}}^T \mathbf{B}(q)\dot{\mathbf{q}} \tag{2.3}$$

where $\mathbf{B}(\mathbf{q})$ is the (n x n) inertia matrix which is *symmetric, positive definite and configuration dependant.*

2.2.2 Potential Energy

In a manner similar to kinetic energy, the potential energy of a system is given by the sum of the contributions of each link and the rotor

$$U = -\sum_{i=1}^{n} (m_{\ell i} g_0^T p_{\ell i} + m_{mi} g_0^T p_{mi}) \tag{2.4}$$

For our 2-DoF robot, this can be written as

$$U = m_1 g l_1 \sin(q_1) + m_2 g (l_1 \sin(q_1) + l_2 \sin(q_1 + q_2))$$

2.2.3 Dynamic Model

A general equation for the dynamics of a robotic manipulator can be obtained by substituting 2.4 and 2.3 in the Lagrangian 2.2. Upon simplification, we arrive at the equation

$$\sum_{j=1}^{n} b_{ij}(q)\ddot{q_j} + \sum_{j=1}^{n}\sum_{k=1}^{n} h_{ijk}(q)\dot{q_k}\dot{q_j} + g_i(q) = \xi_i \qquad\qquad i = 1, 2, ...n \tag{2.5}$$

where

$$h_{ijk} = \frac{\partial b_{ij}}{\partial q_k} - \frac{1}{2}\frac{\partial b_{jk}}{\partial q_i}$$

In this equation, for the accelaration terms,

- the coefficients b_{ii} represents moment of inertia only considering the joint i axis in the current configuration.
- the coefficients b_{ij} accounts for the accelaration of joint i on j.

the quadratic velocity terms,

- $h_{ijj}q_j^2$ represents centrifugal effect of joint i on joint j.
- $h_{ijk}\dot{q_j}\dot{q_k}$ represents the coriolis effect induced by joint i on joints j and k.

the force applied ξ_i is the input applied which is considered as

$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad \in \mathbb{R}^2$$

The dynamic equation 2.5 can be represented as

$$\mathbf{B}(q)\ddot{\mathbf{q}} + \mathbf{C}(q, \dot{q}) + \mathbf{g}(q) = \mathbf{u} \tag{2.6}$$

For our 2-DoF manipulator, the coefficient matrices are given as

$$B(q) = \begin{bmatrix} b_1 + b_2\cos(q_2) & b_3 + b_4\cos(q_2) \\ b_3 + b_4\cos(q_2) & b_5 \end{bmatrix} \quad \in \mathbb{R}^{2*2}$$

$$C(q, \dot{q}) = -c_1 \sin(q_1) \begin{bmatrix} \dot{q_2} & \dot{q_1} + \dot{q_2} \\ -\dot{q_1} & 0 \end{bmatrix} \quad \in \mathbb{R}^{2*2}$$

$$g(q) = (g_1\cos(q_1) + g_2\cos(q_1 + q_2), g_2\cos(q_1 + q_2))^T \quad \in \mathbb{R}^{1*2}$$

2.3 State Space

A keen observation to be made is that the dynamics is in the joint space despite the objective being to control the end effector position (operational space), **the joint space is chosen as the state space** for the following reasons:

- The motion of the robot is controlled through the motion of joints. The joint motors apply direct torque to the joints which is observed in the joint angles. Hence, the joint space becomes a natural choice.
- The individual joint motion is decoupled from each other. On the other hand, the motion of the end effector is a result of motion of all joints. This makes the motion of the robot easier to analyse when expressed in the joint space.
- It is also easier to calculate the input required to execute a motion. The motion of the end effector can be easily calculated using forward kinematics and vice versa.

2.3.1 Reformulation as a first order system

The dynamic model is a second order equation which needs to be reformulated as a first order explicit state space model. The joint angles and the velocities are chosen as the states

$$q_1 = x_1 \qquad \dot{q_1} = x_3 = \dot{x_1} \qquad \ddot{q_1} = \dot{x_3}$$

$$q_2 = x_2 \qquad \dot{q_2} = x_4 = \dot{x_2} \qquad \ddot{q_2} = \dot{x_4}$$

State space of the system:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \in \mathbb{R}^4$$

Input:

$$U = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \in \mathbb{R}^2$$

Now, the dynamic equation 2.6 needs to be expressed explicitely in terms of this state

$$\begin{bmatrix} b_1 + b_2\cos(x_2) & b_3 + b_4\cos(x_2) \\ b_3 + b_4\cos(x_2) & b_5 \end{bmatrix} \begin{bmatrix} \dot{x_3} \\ \dot{x_4} \end{bmatrix} = c_1 \sin(x_1) \begin{bmatrix} x_4 & x_3 + x_4 \\ -x_3 & 0 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} - \begin{bmatrix} g_1\cos(x_1) + x_2\cos(x_1 + x_2) \\ g_2\cos(x_1 + x_2) \end{bmatrix} + \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Solving for $\dot{x_3}$ and $\dot{x_4}$, we get the final equation of the dynamic model as

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \\ g(x,u) \\ h(x,u) \end{bmatrix}$$

where

$$g(x,u) = \frac{-b_5 \cdot u_1 + b_3 \cdot u_2 - b_5 \cdot g_1 \cdot \cos(x_1) - b_4 \cdot u_2 \cdot \cos(x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2}$$
$$+\frac{b_3 \cdot g_2 \cdot \cos(x_1 + x_2) - b_5 \cdot g_2 \cdot \cos(x_1 + x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2}$$
$$+\frac{b_4 \cdot g_2 \cdot \cos(x_1 + x_2) \cdot \cos(x_2) + b_3 \cdot c_1 \cdot x_3^2 \cdot \sin(x_2) + b_5 \cdot c_1 \cdot x_4^2 \cdot \sin(x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2}$$
$$+\frac{b_4 \cdot c_1 \cdot x_3^2 \cdot \cos(x_2) \cdot \sin(x_2) + 2 \cdot b_5 \cdot c_1 \cdot x_3 \cdot x_4 \cdot \sin(x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2 - b_2 \cdot b_5 \cdot \cos(x_2) + 2 \cdot b_3 \cdot b_4 \cdot \cos(x_2)}$$

and

$$h(x,u) = -\frac{b_1 \cdot u_2 - b_3 \cdot u_1 - b_3 \cdot g_1 \cdot \cos(x_1) + b_2 \cdot u_2 \cdot \cos(x_2)}{-b_4 \cdot u_1 \cdot \cos(x_2) + b_1 \cdot g_2 \cdot \cos(x_1 + x_2))}$$
$$- \frac{-b_4 \cdot u_1 \cdot \cos(x_2) + b_1 \cdot g_2 \cdot \cos(x_1 + x_2)}{-b_4 \cdot u_1 \cdot \cos(x_2) + b_1 \cdot g_2 \cdot \cos(x_1 + x_2)} + \frac{b_3 \cdot g_2 \cdot \cos(x_1 + x_2) - b_3 \cdot g_2 \cdot \cos(x_1 + x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2}$$
$$+ \frac{+b_2 \cdot g_2 \cdot \cos(x_1 + x_2) \cdot \cos(x_2) - b_4 \cdot g_2 \cdot \cos(x_1 + x_2) \cdot \cos(x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2}$$
$$+ \frac{-b_4 \cdot g_1 \cdot \cos(x_1) \cdot \cos(x_2) + b_1 \cdot c_1 \cdot x_3^2 \cdot \sin(x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2}$$
$$+ \frac{b_3 \cdot c_1 \cdot x_4^2 \cdot \sin(x_2) + b_2 \cdot c_1 \cdot x_3^2 \cdot \cos(x_2) \cdot \sin(x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2} + \frac{b_4 \cdot c_1 \cdot x_4^2 \cdot \cos(x_2) \cdot \sin(x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2}$$
$$+ \frac{2 \cdot b_3 \cdot c_1 \cdot x_3 \cdot x_4 \cdot \sin(x_2) + 2 \cdot b_4 \cdot c_1 \cdot x_3 \cdot x_4 \cdot \cos(x_2) \cdot \sin(x_2)}{b_4^2 \cdot \cos(x_2)^2 - b_1 \cdot b_5 + b_3^2}$$

The parameters of the system are given as follows

Table 2.1: Parameters of the system

Parameter	Value	Unit
b_1	200.0	kg m^2/rad
b_2	50.0	kg m^2/rad
b_3	23.5	kg m^2/rad
b_4	25.0	kg m^2/rad
b_5	122.5	kg m^2/rad
c_1	-25.0	Nms^{-2}
g_1	784.8	Nm
g_2	245.3	Nm
l_1	0.5	m
l_2	0.5	m

CHAPTER 3

Open-loop Optimal Control

The objective of the Optimal Control problem is to drive the robot from $q = (-5, -4)^T$, $\dot{q} = (0, 0)^T$ to the position $q = (\pi/2, 0)^T$, $q = (0, 0)^T$. The robot begins from one rest configuration and stops at another.

The input constraints and the state constraints are

$$U \in [-1000, 1000]Nm \qquad\qquad \dot{q} \in [-\frac{3}{2}\pi, \frac{3}{2}\pi]rad/s$$

The optimal control problems chosen for analysis are *(1) Minimization of joint angles deviation, (2) Linear Quadratic Regulator problem.* The first problem is analysed for sampling steps of 50 and 100; also, the following integrators are compared.

Integrators	Euler Integrator	Runge-Kutta 4th order Integrator
Equation	$y_{n+1} = y_n + h \cdot f(:, k)$	$k_1 = h \cdot f(t_n, y_n)$ $k_2 = h \cdot f\left(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right)$ $k_3 = h \cdot f\left(t_n + \frac{h}{2}, y_n + \frac{k_2}{2}\right)$ $k_4 = h \cdot f(t_n + h, y_n + k_3)$ $y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$

Further, a minimum time objective is solved. Finally, this problem is analysed for minimum input and time simultaneously.

3.1 Minimization of tracking error

The objective of this problem is to minimize the error with the desired state and reach the same in 3s. The cost function is given as

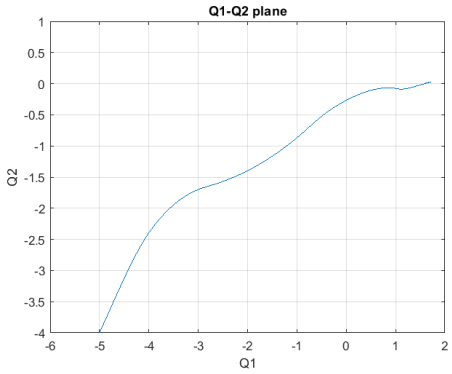
$$J = \int_{t_{\text{start}}}^{t_{\text{end}}} (q_i(t) - q_f(t))^2 \, dt$$

where, $q_f(t)$ is the final joint configuration
$q_i(t)$ is the current joint configuration

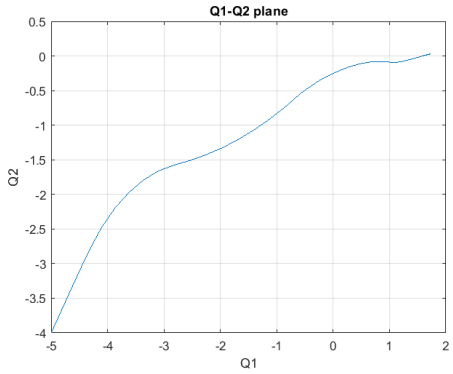
This integral cost function represents the accumulated squared differences between the desired and actual joint angles over the specified time interval. Minimizing this integral encourages the robot to closely follow the desired joint angle trajectory throughout the motion

Euler Integrator

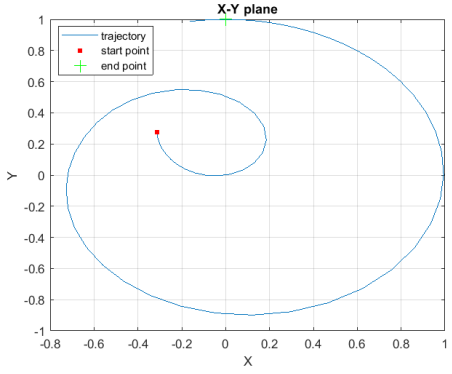
The results for the optimal control on the system, when Euler integration is applied, is shown below. The graphs on the left are for 100 sampling steps and on the right are for 50 sampling steps. It can be clearly observed that higher the number of sampling steps, smoother is the trajectory of the robot.



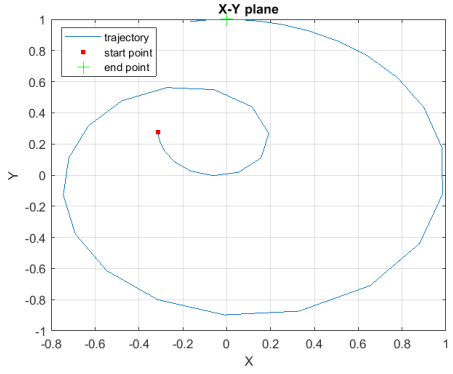
(a) q1-q2 curve (100 steps)



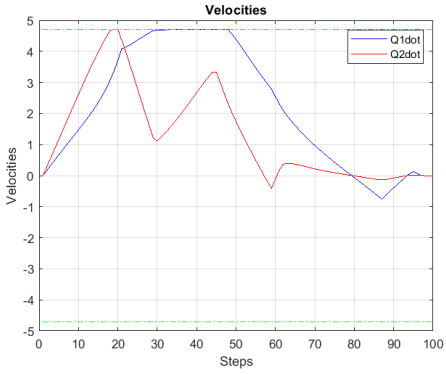
(b) q1-q2 curve (50 steps)



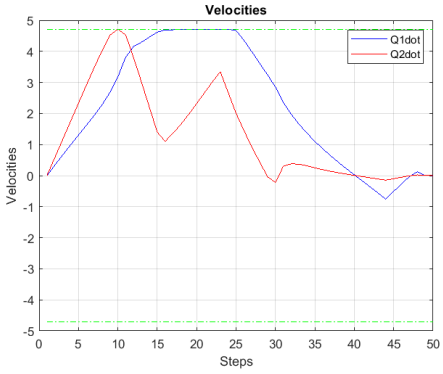
(c) XY curve (100 steps)



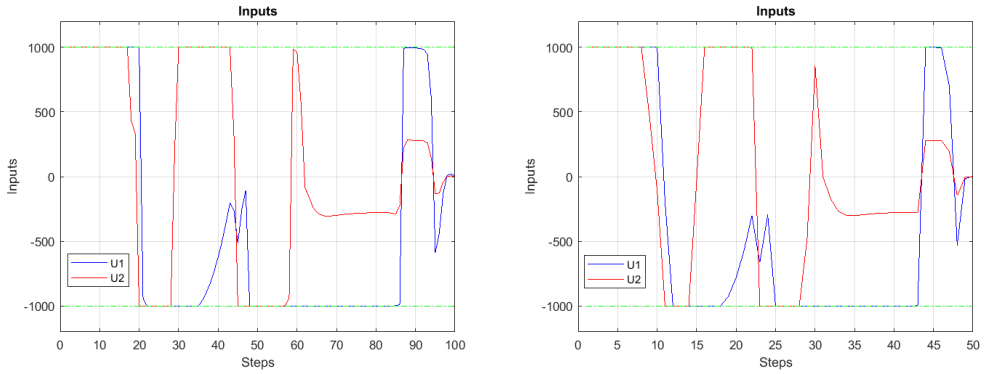
(d) XY curve (50 steps)



(e) Velocity curve (100 steps)



(f) Velocity curve (50 steps)



(a) Input curve (100 steps) (b) Input curve (50 steps)

Figure 3.2: A grid of plots compared the performace of sampling steps of 100 and 50, for Euler integration

Euler Integration		
Sampling Steps	100	50
Iterations to solve	98	77
Time in ipopt	15.833s	5.503s

- Euler integration for 100 samples takes more number of iterations to converge than for 50 sample. Consequently, the solver takes longer to solve the problem. This is directly caused by the number of samples.
- The trajectory becomes much smoother as the samples increase.

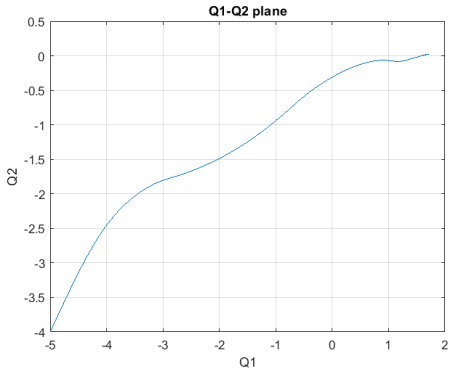
3.1.1 Runge-Kutta Integration

Since the Runge-Kutta method uses multiple slope estimates to calculate the value at the next time step, it is much more accurate than the Euler integration method. Similar to the former, the performance of the problem is compared for 100 and 50 sampling steps.

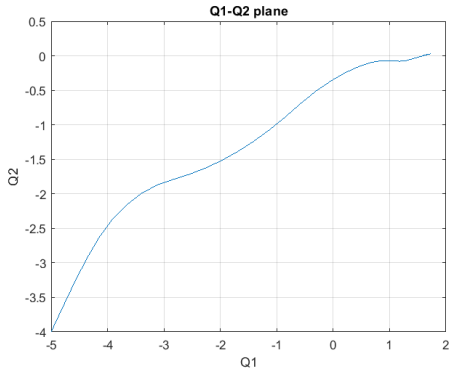
Runge-Kutta Integration		
Sampling Steps	100	50
Iterations to solve	126	72
Time in ipopt	55.718s	11.079s

- As for the Runge-Kutta method, the solver takes much longer to solve the problem due to the computational complexity of the method.
- The plots show that the system shows a similar trajectory compared to Euler method.

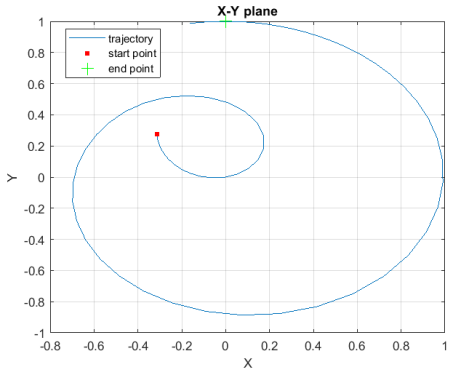
The plots are shown in the following page.



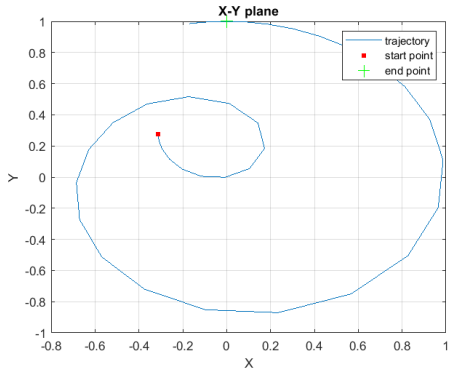
(a) q1-q2 curve (100 steps)



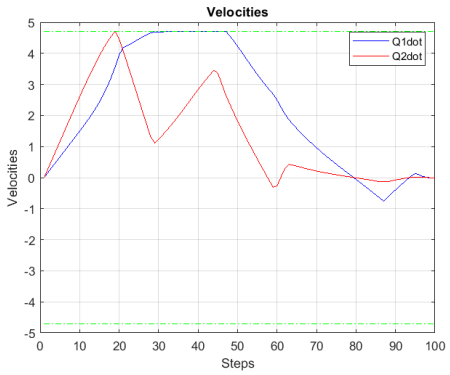
(b) q1-q2 curve (50 steps)



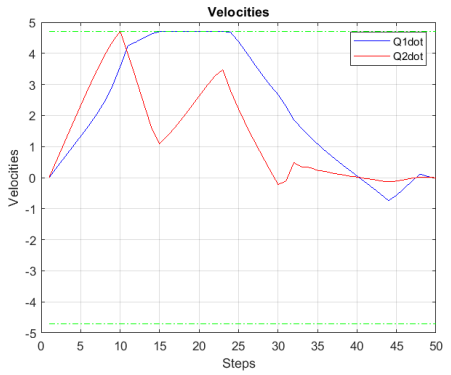
(c) XY curve (100 steps)



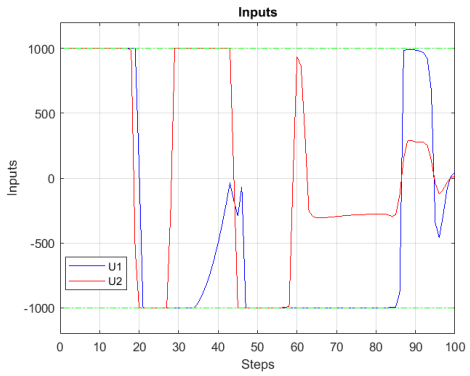
(d) XY curve (50 steps)



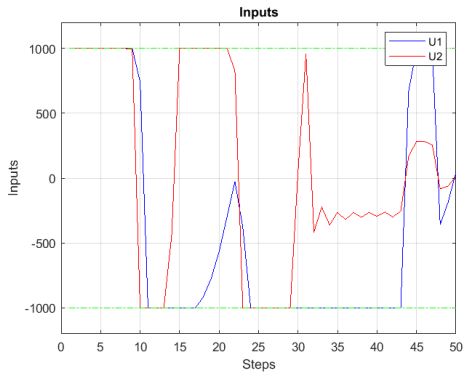
(e) Velocity curve (100 steps)



(f) Velocity curve (50 steps)



(g) Input curve (100 steps)



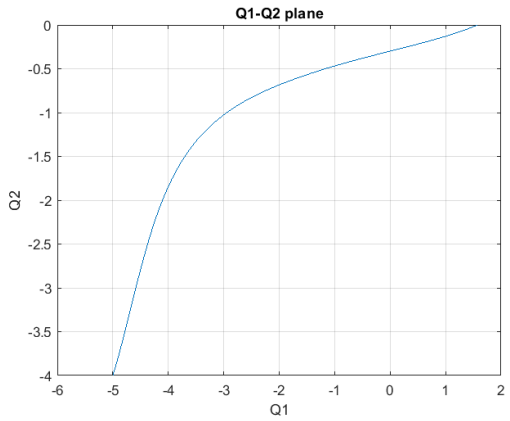
(h) Input curve (50 steps)

Figure 3.3: A grid of plots compared the performace of sampling steps of 100 and 50, for Runge-Kutta integration

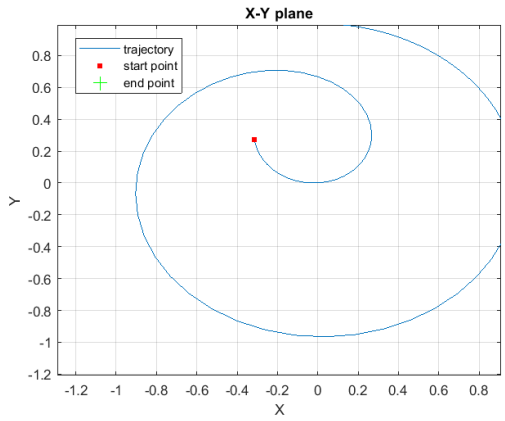
3.2 Linear Quadratic Regulator (LQR) problem

The Linear Quadratic Regulator (LQR) problem is a classic control problem in which the goal is to design a controller for a linear system that minimizes a quadratic cost function. The LQR problem cost function formulated as follows:

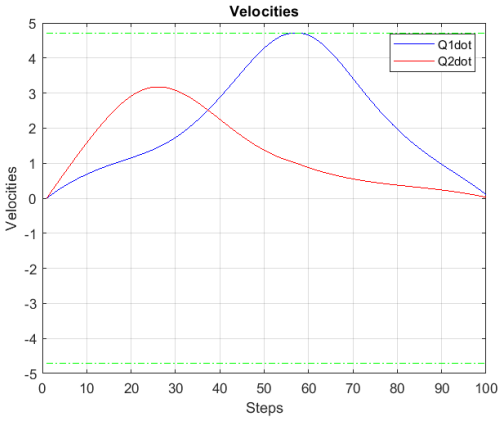
$$J = \int_{t_0}^{t_f} (x^T Q x + u^T R u)\, dt$$



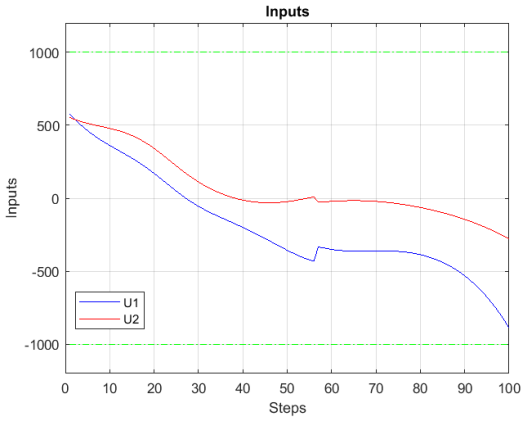
(a) Graph of joint angles



(b) Trajectory traced by end effector



(c) Graph of Velocities



(d) Graph of control inputs

Figure 3.4: A grid of graphs from the LQR problem.

The plots show that the input, velocity and angle curves are smooth and without any abrupt changes.
The optimization problem was solved for these parameters

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

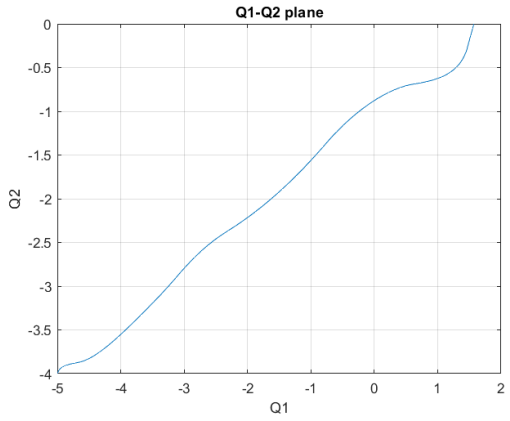
Sampling Steps	100
Integrator	Euler
Solver	ipopt
Iterations	72
Time in ipopt	12.056s

3.3 Minimization of time

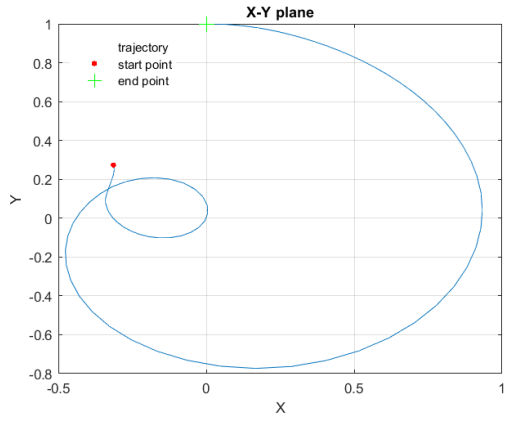
The optimal control cost function for minimum time is given by

$$J = \int_{t_{start}}^{t_{end}} 1 \, dt$$

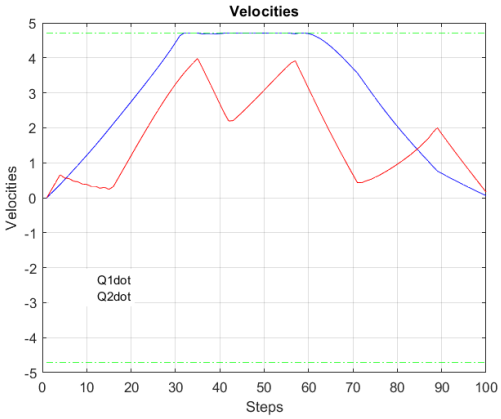
The objective of the problem is to get to the final state as quickly as possible within the constraint limits. The problem is formulated as free end time and the results obtained are shown below



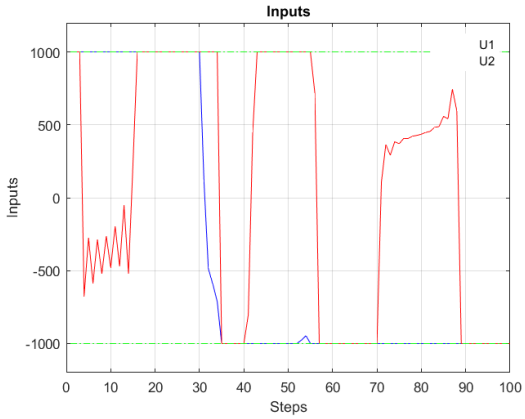
(a) Graph of joint angles



(b) Trajectory traced by end effector



(c) Graph of Velocities



(d) Graph of control inputs

Figure 3.5: A grid of graphs from minimum time optimization problem.

The optimization problem was solved for these parameters

Sampling Steps	100
Integrator	Euler
Solver	ipopt
Iterations	88
Time in ipopt	2.708s

The **minimum time** required to reach the desired state is **2.26s**

- The velocities exhibit abrupt changes thereby having large accelerations. Such behaviour can be damaging to the motors.
- The robot requires impulsive and large inputs to reach the final state as quickly as possible.

3.3.1 Minimization of time as a fixed end time problem

The free end time problem can be solved by applying a transformation to the time horizon. Consider the time horizon

$$t \in [0, t_1]$$

Taking t as function of a variable τ, we get

$$t = \tau t_1 \implies \tau = 1/t_1 \in [0, 1]$$

This also implies on the dynamics as

$$x(t) = x(t(\tau))$$

$$\dot{x} = \frac{dx}{dt} = f(t, x(t), u(t))$$

$$\frac{dx(t(\tau))}{d\tau} = \frac{\partial x}{\partial t}.\frac{\partial t}{\partial \tau}$$

the dynamic equation now becomes,

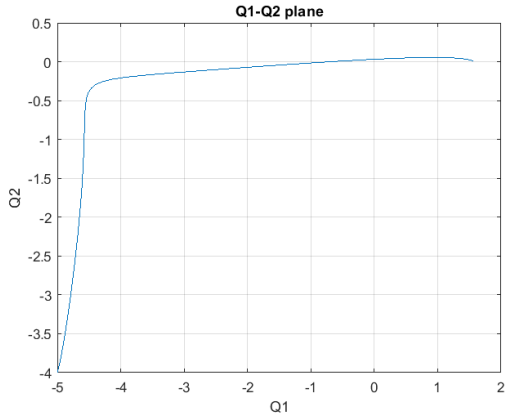
$$\frac{dx(t(\tau))}{d\tau} = t_1.f(\tau, x(\tau), u(\tau))$$

Now, the problem is transformed into a fixed end time problem. The solution of the system remains the same. Refer 3.5

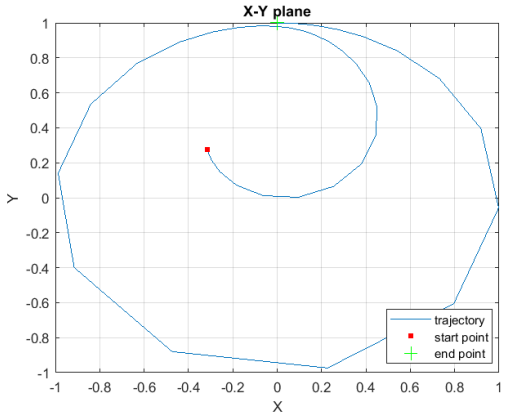
3.4 Minimization of time and control input

In this problem, the goal is to achieve the desired state as fast as possible while also utilising the least control energy. This can be formulated as a cost function as follows

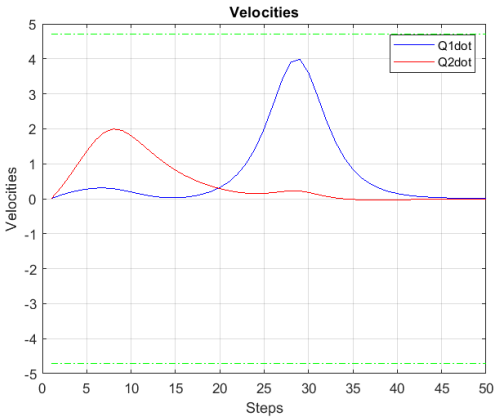
$$J = \int_{t_{start}}^{t_{end}} 1\, dt + u^T u\, dt$$



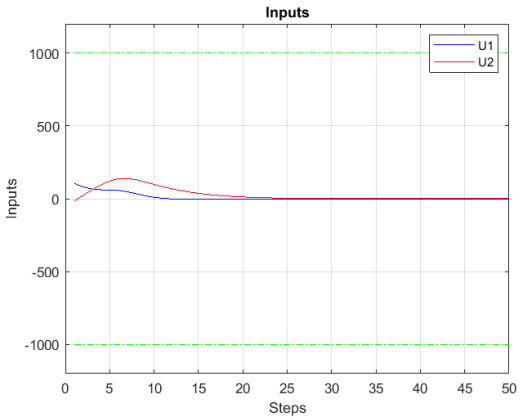
(a) Graph of joint angles



(b) Trajectory traced by end effector



(c) Graph of Velocities



(d) Graph of control inputs

Figure 3.6: A grid of graphs from the LQR problem.

The following parameter values were observed

Sampling Steps	50
Integrator	Runge-Kutta
Solver	ipopt
Iterations	377
Time in ipopt	39.985s

- From the q1-q2 plot, it is seen that the second joint is actuated first following by the first joint. This uses minimal input by exploiting gravity.
- In order to avoid large inputs, the system takes much longer to reach the desired state.

The minimum time taken to acheive the desired state with least control energy is **8.8s**.

CHAPTER 4

Model Predictive Control

Model Predictive Control (MPC) is an advanced and iterative control stratergy that is used to optimize the performance of systems subject to constraints. It particularly offers advantages when compared to optimal control due to its prediction horizon. It can predict the future behaviour of a system over a finite horizon and adjust the control accordingly.

MPC involves the process of *estimating the state, calculating the optimal control and applying the control input recursively*. On the designer's perspective, it is necessary to choose the Prediction horizon, sampling time and terminal penalty effectively.

The general cost function can be expressed as

$$V_T^{\beta}(x(t_k)) = \int_{tk}^{t_k+T} l(x(\tau|t_k), u(\tau|t_k))d\tau + \beta V_f(x(t_k+T|t_k))$$

the terminal penalty βV_f is positive definite.

4.1 Uncertainties and Disturbances in the robot

- **Viscuous friction torques** are occured due to the resistance from the joints in presence of fluids/oils etc.
$$F_v\dot{q}$$
- **Static friction torques** $f_s(q,\dot{q})$ arise as a resistence to move. Coulomb friction is a good model and is given by
$$F_s sgn(\dot{q})$$
- When the end effector is in contact with objects, a portion of the actuating torque is used to balance the torque by the contact forces h_e.
$$J^T(q)h_e$$
- The sensor noise and other measurement or random noise can be considered as gaussian noise
$$\mathcal{N}(\mu, \sigma^2)$$
with mean μ and variance σ^2

Including these in the final dynamic equation,

$$\mathbf{B}(q)\ddot{\mathbf{q}} + \mathbf{F_v}\dot{\mathbf{q}} + \mathbf{F_s sgn}(\dot{\mathbf{q}}) + \mathbf{C}(q,\dot{q}) + \mathbf{g}(q) + \mathcal{N}(\mu, \sigma^{\mathbf{2}}) = \mathbf{u} - \mathbf{J^T}(\mathbf{q})\mathbf{h_e}$$

4.2 MPC Controller for the LQR Problem

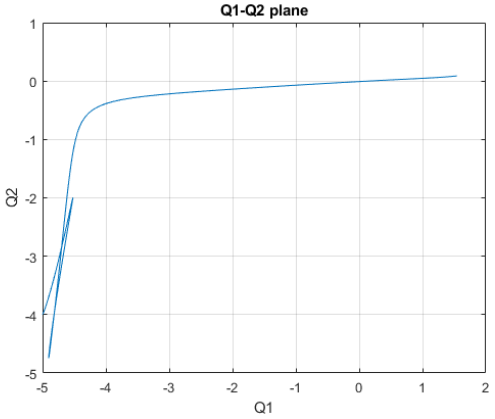
The LQR problem is used to obtain smoother state and control input trajectories. The cost function is formulated as a quasi-infinite horizon problem. The cost function now becomes

$$J = \int_{t_k}^{t_k+T} (x^T Q x + u^T R u)\, dt + \beta \cdot (x_f - x)' \cdot (x_f - x)$$

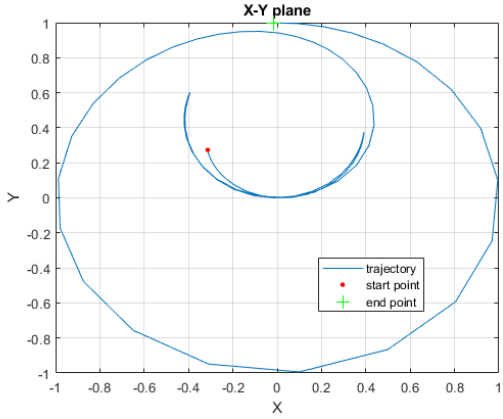
where x_f is the desired state, $\beta = 10000$ and,

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

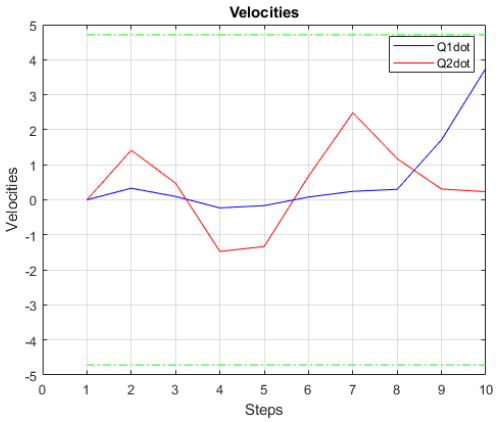
Notice that the order of magnitude of the control product in the cost function is in the millions as compared to that of the state in the 100s. As a result, the MPC scheme utilizes minimal control to acheive the desired state.



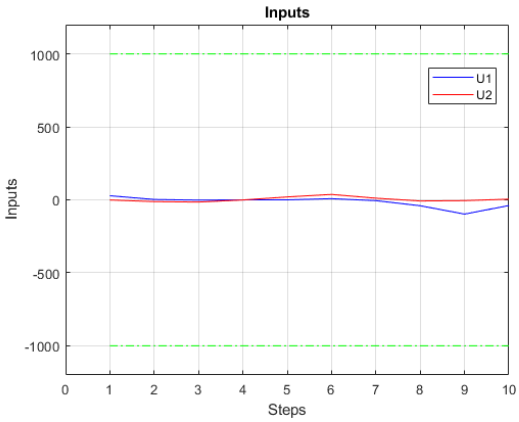
(a) Graph of joint angles



(b) Trajectory traced by end effector



(c) Graph of Velocities



(d) Graph of control inputs

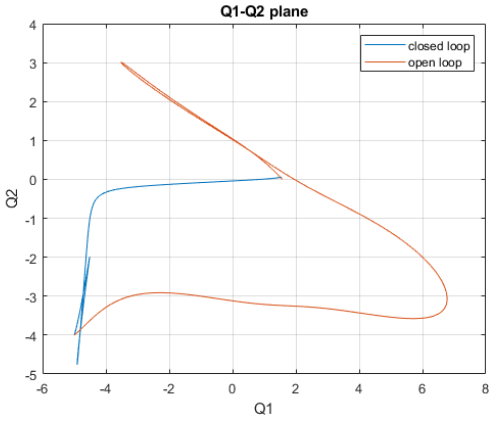
Figure 4.1: A grid of graphs from the LQR problem using Model Predictive Control.

Sampling size	0.1s
Time horizon	10s
Prediction horizon	1s

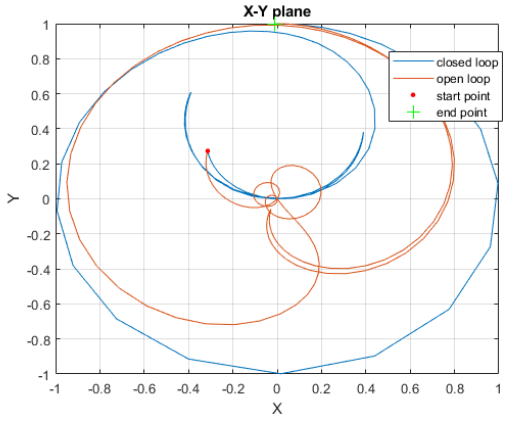
4.3 Comparison MPC (Closed loop) and Optimal Control (Open loop) for incorrect parameters

The parameters $b_1 = 180$ and $b_2 = 45$ are present in the the first term of the inertia matrix of the robot. They represent the moment of inertia of the first link in absence of the motion of the second link. Incorrect values of this can occur due to *a) incorrect measurement of mass/center of mass or the geometry of the robot link. b) When the robot link has a complex mass distribution c) due to measurement errors.*

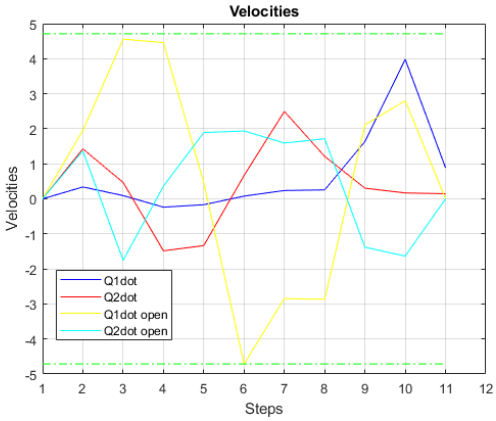
The results are shown below



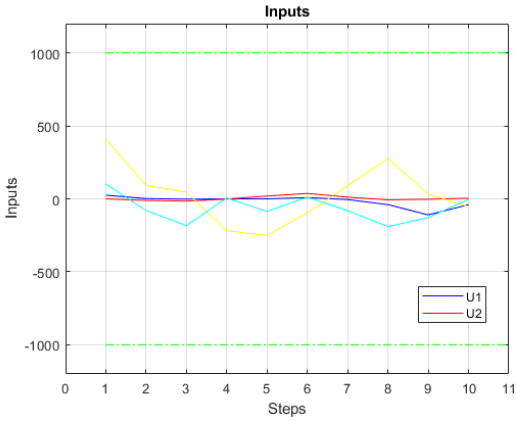
(a) Graph of joint angles



(b) Trajectory traced by end effector



(c) Graph of Velocities



(d) Graph of control inputs

Figure 4.2: A grid of graphs comparing the open loop and closed loop results for incorrect parameters.

From the q1-q2 graph, the open loop trajectory shows an overshoot before arriving at the target. Thus, the closed loop trajectory is more stable and accurate than the open loop. This

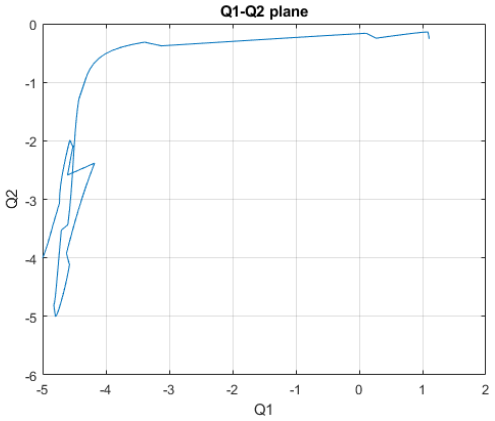
is due to the fact that MPC can adjust its controls according to the prediction horizon.

4.4 MPC of the LQR problem in presence of Gaussian noise

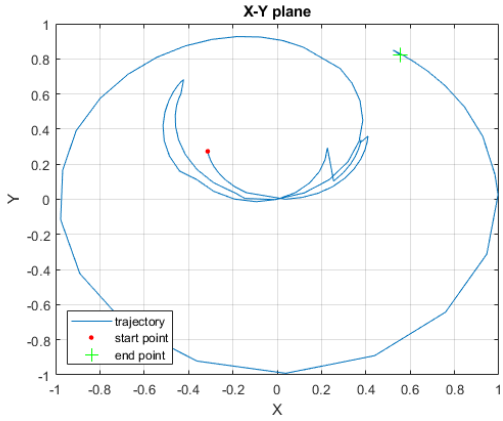
Often times, the real system contains additional noise due to various reasons like sensor/measurement errors, model inefficiencies etc. This can be expressed as gaussian noise.

$$\mathcal{N}(\mu, \sigma^2)$$

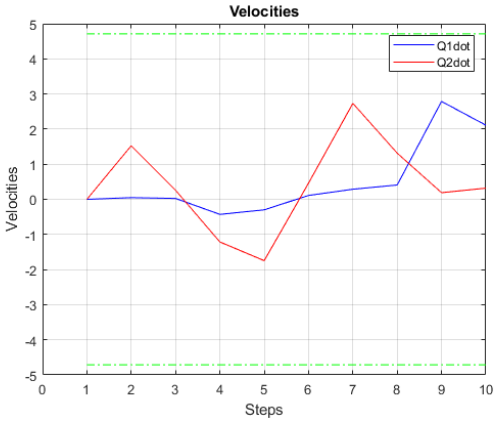
with mean $\mu = 0$ and variance $\sigma^2 = \pi/25$. The variance is chosen as a 16% of total angle range of 4π.



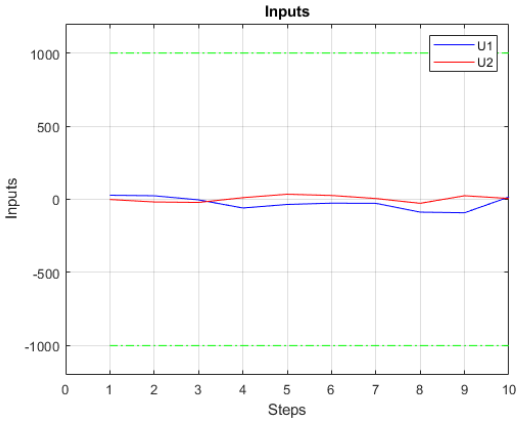
(a) Graph of joint angles



(b) Trajectory traced by end effector



(c) Graph of Velocities



(d) Graph of control inputs

Figure 4.3: A grid of graphs comparing the open loop and closed loop results for incorrect parameters.

The system converges at (q1, q2) = (-0.1, 1.2) due to the gaussian noise.

References

[1] Casadi. `https://web.casadi.org/docs/`, 2024.

[2] L. V. G. O. Bruno Siciliano, Lorenzo Sciavicco. Robotics: Modelling, Planning and Control. Springer London, 21 October 2010.

[3] B. Chachuat. Nonlinear and Dynamic Optimization: From Theory to Practice. 2007.

[4] T. Faulwasser. Nonlinear model predictive control – theory and applications. 2021. Lecture Notes, ie3, TU Dortmund University.

[2] [1] [4] [3]