Exercise Sheet - PhD Position

Coursework

Ashutosh Mukherjee

26.08.2024

Contents

1 Topology and Mobility Analysis

Exercise 1

Serial Robot

The topology tree will contain seven (including the base) nodes depicting the links and six edges depicting the joints. The base link is depicted by B.

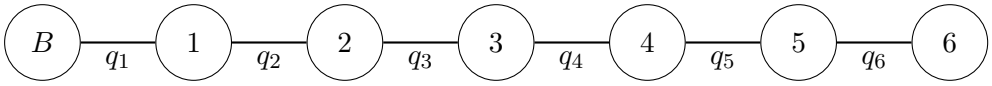

Figure 1.1: Graph Topology of Serial Robot

Parallel Robot

The topology tree will contain fourteen (including the top-platform and base) nodes depicting the links and eighteen edges depiciting the joints (six of which are the prismatic joints). The prismatic joint edges are colored green and the spherical joints are colored blue. The base link is depicted by B and the top platform is depicted by EE.

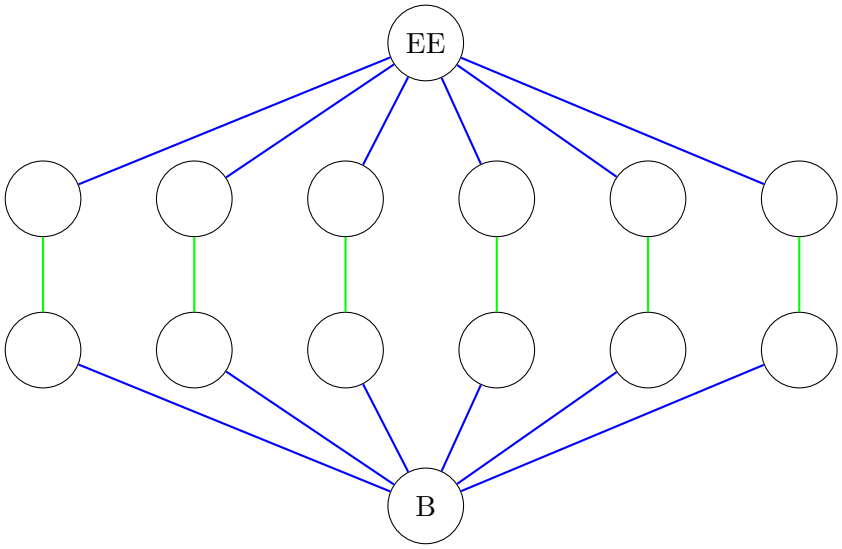

Figure 1.2: Graph Topology of Parallel Robot

Exercise 2

Both the serial and parallel robots are considered spatial mechanisms hence for both $s = 6$.

For the serial robot which has six revolute joints (one degree of freedom each),

$$N = 7, j = 6, f = 6$$
$$m_s = 6$$

The parallel robot has twelve spherical joints (three degrees of freedom each) and six prismatic joints (one degree of freedom each) with fourteen links.

$$N = 14, j = 18, f = 42$$
$$m_s = 12$$

Simplification for Serial Robots

The Chebychev-Grübler-Kutzbach formula can be simplified for serial manipulators which consist of no redundant degrees of freedom (eg. due to multiple closed kinematic chains). The simplification is given by

$$m_s = \sum_{i=1}^{j} f_i \tag{1.1}$$

where f_i is the degrees of freedom allowed by the i^{th} joint.

Modification for Parallel Robots

It is clear that (1.1) doesn't apply to parallel mechanisms which have multiple kinematic loops and hence redundant degrees of freedom. Moreover, the original Chebychev-Grübler-Kutzbach formula also sometimes doesn't work for all parallel mechanisms. For instance, the parallel mechanism given in this question is the *Stewart Platform*, which always provides six degrees of freedom to the top platform, hence the formula should ideally give 6 as an answer (which infact happens for the configuration where universal joints connect the legs and the base). But the redundant degrees of freedom introduced due to spherical joints on both sides of the legs are not considered. This redundant degree of freedom is the rotation of the leg about its own axis, which essentially has no influence on the resultant mechanism motion. Hence, this redundancy needs to be compensated. For six legs, the redundant degrees of freedom will be six. The modified Chebychev-Grübler-Kutzbach becomes

$$m_s = s(N - j - 1) + \sum_{i=1}^{j} f_i - C \tag{1.2}$$

where $C = 6$ and we get the correct answer of six for the given parallel mechanism.

Exercise 3

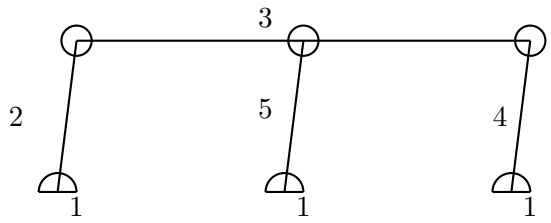

Figure 1.3: Four-bar linkage with one additional link and two additional revolute joints

Figure 1.3 depicts a two-dimensional four-bar linkage with an additional link (5) and two additional revolute joints connecting link 5 to link 3 and the base. According to the Chebychev-Grübler-Kutzbach formula

$$N = 5, j = 6, f = 6, s = 3$$
$$m_s = 0$$

But in reality, the described mechanism will have a single degree of freedom. Hence, in this case the formula fails. The reason is that in the above mechanism, the constraint introduced by the additional joints is redundant, and the instrinsic assumption of the Chebychev-Grübler-Kutzbach formula is that the constraints provided by the joints are independent.

2 Geometry and Kinematics

Exercise 4

The geometric object in which the end-effector lives is called the *Workspace*.

Forward Kinematics

The formulation of the forward kinematics is based on the book *Introduction to Robotic, Mechanics and Control* by John J. Craig, which makes use of the *Denavit-Hartenberg* parameters.

For the given formulation, it is assumed that $q_1 \in [0, \pi]$ and $q_2 \in \left[\frac{-\pi}{2}, \frac{\pi}{2}\right]$.

For brevity, only the Z and X axes of the joint frames are drawn in 2D, axis coming out of the plane is denoted by $\oplus$ and axis going into the plane is denoted by $\otimes$.

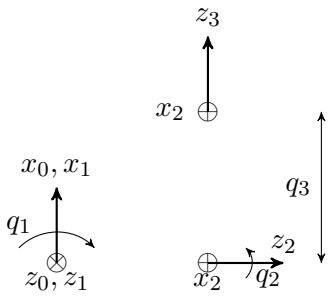

Figure 2.1: Joint Frame Assignment

Figure 2.1 illustrates the joint frame allocation along with the base frame $\{0\}$.

Frame (i)	α_{i-1} [deg]	a_{i-1} [m]	d_i [m]	θ_i [deg]
1	0	0	0	q_1
2	-90	0	0	q_2
3	90	0	q_3	0

Table 2.1: Denavit-Hartenberg parameters

Table 2.1 summarizes the DH parameters according to the frame allocation of the manipulator. The homogenous transformation matrix is used to represent the pose. The transformation matrix between two arbritary frames $\{A\}$ and $\{B\}$ is given by

$$\boldsymbol{T}_B^A = \left[\begin{array}{c|c} \boldsymbol{R}_B^A & \boldsymbol{r}_B^A \\ \hline \boldsymbol{0}_{1\times 3} & 1 \end{array}\right]$$

where $\boldsymbol{R}_B^A$ describes the orientation of $\{B\}$ with respect to $\{A\}$ while $\boldsymbol{r}_B^A$ represents the position of origin of $\{B\}$ with respect to the origin of $\{A\}$. Assuming $E\,(x, y, z)$ directly lies on the origin of $\{3\}$, the forward kinematics are given by

$$\boldsymbol{T}_3^0 = \left[\begin{array}{c|c} \boldsymbol{R}_3^0 & \boldsymbol{r}_3^0 \\ \hline \boldsymbol{0}_{1\times 3} & 1 \end{array}\right]$$

where $\boldsymbol{r}_3^0 = (x\ y\ z)^\intercal$ and

$$\boldsymbol{T}_3^0 = \boldsymbol{T}_1^0 \boldsymbol{T}_2^1 \boldsymbol{T}_3^2$$

where

$$\boldsymbol{T}_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ -\sin\theta_i \cos\alpha_{i-1} & \cos\theta_i \cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1} d_i \\ \sin\theta_i \sin\alpha_{i-1} & \cos\theta_i \sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Hence,

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} q_3 \cos(q_1) \sin(q_2) \\ q_3 \sin(q_1) \sin(q_2) \\ q_3 \cos(q_2) \end{bmatrix} \tag{2.1}$$

Inverse Kinematics

Inverse kinematics can be computed with the help of the obtained forward kinematics expression. We know from (2.1)

$$\begin{aligned} x^2 + y^2 + z^2 &= q_3^2 \\ \implies q_3 &= \sqrt{x^2 + y^2 + z^2} \end{aligned}$$

Moreover,

$$\begin{aligned} \cos(q_2) &= \frac{z}{q_3} = \frac{z}{\sqrt{x^2 + y^2 + z^2}} \\ \sin(q_2) &= \sqrt{1 - \cos(q_2)^2} = \frac{\sqrt{x^2 + y^2}}{\sqrt{x^2 + y^2 + z^2}} \\ \implies \tan(q_2) &= \frac{\sqrt{x^2 + y^2}}{z} \\ \implies q_2 &= \arctan 2\left(\sqrt{x^2 + y^2}, z\right) \end{aligned}$$

The negative version of $\sin(q_2)$ is not considered since it is assumed that $q_2 \in \left[\frac{-\pi}{2}, \frac{\pi}{2}\right]$ and hence $\arctan 2$ is used. Finally the expression for q_1 is obtained trivially

$$q_1 = \arctan\left(\frac{y}{x}\right)$$

Since $q_1 \in [0, \pi]$, hence arctan is used and zero values of x represents arbritary values of q_1. Hence, the inverse kinematics can be summarized as

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \arctan\left(\frac{y}{x}\right) \\ \arctan 2\left(\sqrt{x^2+y^2}, z\right) \\ \sqrt{x^2+y^2+z^2} \end{bmatrix} \tag{2.2}$$

Program Verification

The verification of kinematics is done in the MATLAB® script *Kinematics.m*. The input DH parameters are read from the file *dhParam.txt*. The script first computes the forward kinematics according to (2.1) for the input DH parameters and then the inverse kinematics according to (2.2). The output of the inverse kinematics matches the inputs in the the file *dhParam.txt*.

Workspace

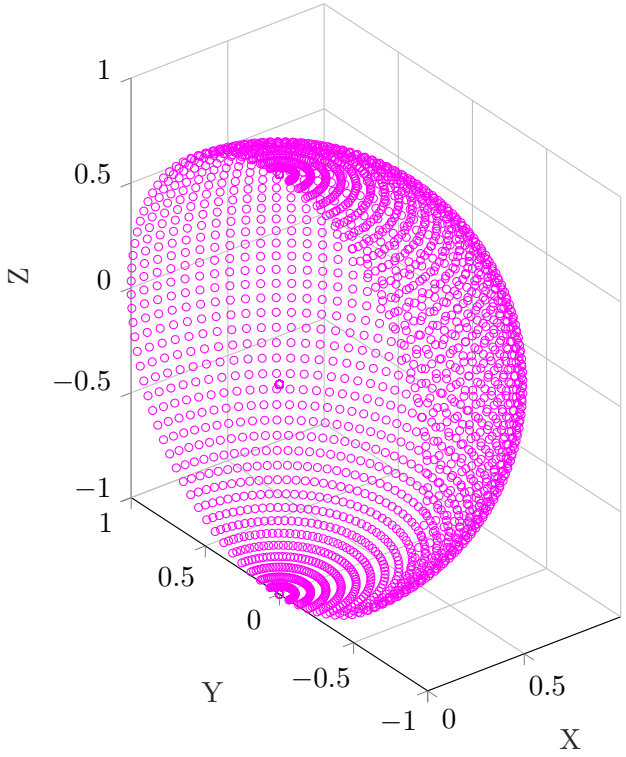

Figure 2.2: 3D Workspace of the manipulator

Figure 2.2 illustrates the three-dimensional workspace of the manipulator for the given joint constraints. It should be noted that the workspace is plotted with respect to the

assumed base frame orientation and the present exposition of the workspace figure may not be optimal for illustration.

Exercise 5

Forward Kinematics

The frame of reference is assumed to originate at the input revolute joint (q_1), with the Z-axis coming out of the plane and the Y axis pointing upwards. It is clear from the given loop geometry that

$$\begin{aligned} l_1 &= l_2 \cos(q_1) + q_3 \cos(\pi - q_2) \\ &= l_2 \cos(q_1) - q_3 \cos(q_2) \\ \implies \cos(q_1) &= \frac{q_3 \cos(q_2) + l_1}{l_2} \\ \implies q_1 &= \arccos\left(\frac{q_3 \cos(q_2) + l_1}{l_2}\right) = f(q_3 = d) \end{aligned}$$

Inverse Kinematics

Now, we know

$$\begin{aligned} l_1 &= l_2 \cos(q_1) - q_3 \cos(q_2) \\ \implies q_3 &= \frac{l_2 \cos(q_1) - l_1}{\cos(q_2)} \qquad \text{given } q_2 > \frac{\pi}{2} \\ &= f^{-1}(q_1) = d \end{aligned}$$

Maximum Output Angular Velocity

The forward kinematics is given by

$$\begin{aligned} q_1 &= f(d) \\ \dot{q}_1 &= \frac{\partial f}{\partial d}\dot{d} \\ \implies \dot{q}_{1,\max} &= \frac{\partial f}{\partial d}\dot{d}_{\max} \end{aligned}$$

Variable substitution is done in order to obtain $\frac{\partial f}{\partial d}$

$$\begin{aligned} u &= \frac{d}{l_2}\cos(q_2) + \frac{l_1}{l_2} \\ \implies \frac{\partial f}{\partial d} &= \frac{\partial f}{\partial u}\frac{\partial u}{\partial d} && \text{where } f = \arccos(u) \\ \implies \frac{\partial f}{\partial d} &= \frac{-1}{\sqrt{1-u^2}}\frac{1}{l_2}\cos(q_2) \\ &= \frac{-\cos(q_2)}{\sqrt{l_2^2 - (l_1 + d\cos q_2)^2}} \end{aligned}$$

Hence, the maximum output angular velocity for a given maximum actuator velocity is given by

$$\dot{q}_{1,\max} = \frac{-\cos(q_2)\,\dot{d}_{\max}}{\sqrt{l_2^2 - (l_1 + d\cos q_2)^2}}$$

Maximum Output Torque

Let f and τ be the input actuator force and output torque respectively. By the principle of virtual work we know,

$$f\delta d = \tau\delta q_1 \tag{2.3}$$

δd and δq_1 are related by the Jacobian (in this case scalar value) which is nothing but

$$J = \frac{-\cos(q_2)}{\sqrt{l_2^2 - (l_1 + d\cos q_2)^2}}$$

Hence, (2.3) can be formulated as

$$\begin{aligned} f\delta d &= \tau J\delta d \\ \implies \tau &= \frac{1}{J}f \\ \implies \tau_{\max} &= \frac{1}{J}f_{\max} \\ \implies \tau_{\max} &= -\frac{\sqrt{l_2^2 - (l_1 + d\cos q_2)^2}}{\cos(q_2)}f_{\max} \end{aligned}$$

Singularities

The mechanism becomes singular whenever its Jacobian loses rank i.e. a degree of freedom is lost. In this case, it corresponds to the set $\{(q_1, q_2, d)^\intercal \,|J = 0\}$ which is given

by

$$\begin{aligned} &\cos(q_2) = 0 \\ &\implies q_2 = \frac{\pi}{2} \qquad\qquad \text{assuming } q_2 \in [0, \pi] \end{aligned}$$

Exercise 6

For the given scenario, it is assumed that the manipulator model from Angela makes use of the base coordinates (x_B, y_B, z_B) to compute the end-effector coordinates (x_E, y_E, z_E) and the orientation quaternion $\mathbf{q}$ which are then fed back to Donald's crane model. Figure 2.3 illustrates this assumed workflow.

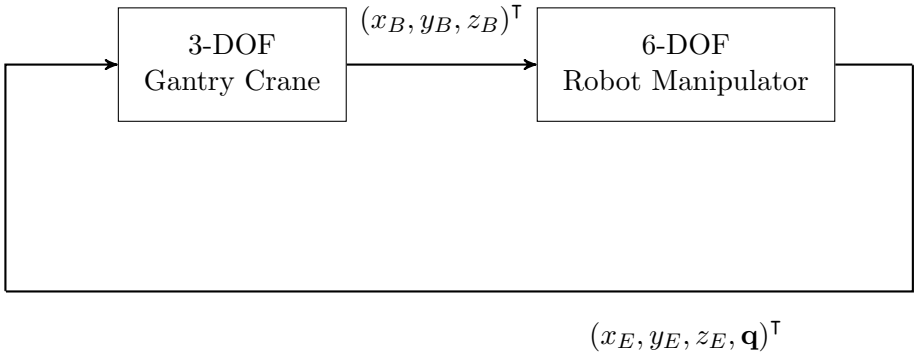

Figure 2.3: Workflow between Donald and Angela

1. It will be possible for Angela to make use of just a single linear scaling factor which is $\frac{1}{39.3701}$, that converts the base coordinates from inches into meters.
2. While Donald can use a single linear scaling factor for converting end-effector coordinates from meters to inches (39.3701) and the euler angles from radians to degrees $\left(\frac{180}{\pi}\right)$, he will first have to convert the output quaternions from Angela's model into the equivalent euler angles which is a non-linear tranformation and is given by

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} a\tan 2\left(2q_2q_3 + 2q_0q_1, q_3^2 - q_2^2 - q_1^2 + q_0^2\right) \\ -a\sin\left(2q_1q_3 - 2q_0q_2\right) \\ a\tan 2\left(2q_1q_2 + 2q_0q_3, q_1^2 + q_0^2 - q_3^2 - q_2^2\right) \end{bmatrix} = \mathcal{F}(\mathbf{q})$$

   assuming $\mathbf{q} = (q_1, q_2, q_3, q_4)^\mathsf{T}$. Hence, there exists no single linear scaling factor that Donald can use to convert the 6D pose to imperial system.

Figure 2.4 illustrates the way Donald and Angela can use each other's models. In the figure, $\mathcal{F}$ refers to the non-linear transformation between quaternions and euler angles, and $m = 39.3701$

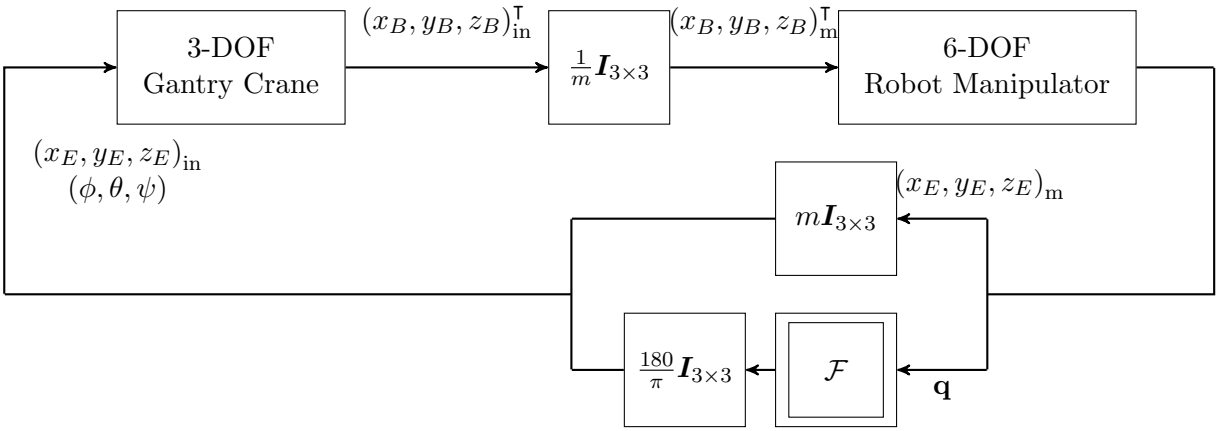

Figure 2.4: Modified Workflow between Donald and Angela

3 Dynamics

Exercise 7

Inverse Dynamics

Inverse dynamics is done by balancing the moments about the hinge

$$\begin{aligned} ml^2\ddot{\theta} &= \tau - mgl\sin\theta \\ \implies \tau &= ml\left(l\ddot{\theta} + g\sin\theta\right) \end{aligned}$$

Forward Dynamics

The forward dynamics is given by

$$\ddot{\theta} = \frac{\tau}{ml^2} - \frac{g\sin\theta}{l}$$

Euler Numerical Integration

A forward Euler numerical scheme is used to approximate the solution of the unforced system, whose state-space can be written as

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ -\frac{g\sin\theta}{l} \end{bmatrix} = f(\boldsymbol{x}) = \dot{\boldsymbol{x}}$$

Then, the euler numerical integration scheme is given by

$$\boldsymbol{x}(t_{i+1}) = \boldsymbol{x}(t_i) + f(\boldsymbol{x}(t_i))\,\Delta t$$

where $\Delta t = t_{i+1} - t_i$.

The euler-integrator is implemented for the unforced system in the python file *eulerIntergrator.py* and the resultant visualization of the pendulum can be accessed from *pendulum.gif*.

Exercise 8

Kinetic Energy

Since the system contains rigid bodies, there will be both rotational and linear kinetic energies. The linear kinetic energies are given by

$$K_{\text{Lin},1} = \frac{1}{2} m_1 \|\boldsymbol{v}_{c1}\|^2$$
$$K_{\text{Lin},2} = \frac{1}{2} m_2 \|\boldsymbol{v}_{c2}\|^2$$

where $\boldsymbol{v}_{c1}, \boldsymbol{v}_{c2}$ are the velocities of the center of masses of links 1 and 2 respectively.

$$\|\boldsymbol{v}_{c1}\|^2 = \dot{x}_{c1}^2 + \dot{y}_{c1}^2$$
$$\|\boldsymbol{v}_{c2}\|^2 = \dot{x}_{c2}^2 + \dot{y}_{c2}^2$$

where

$$x_{c1} = \frac{l_1}{2} \cos\theta_1$$
$$y_{c1} = \frac{l_1}{2} \sin\theta_1$$
$$x_{c2} = l_1 \cos\theta_1 + \frac{l_2}{2} \cos\left(\theta_1 + \theta_2\right)$$
$$y_{c2} = l_1 \sin\theta_1 + \frac{l_2}{2} \sin\left(\theta_1 + \theta_2\right)$$

The rotational kinetic energies are given by

$$K_{\text{Rot},1} = \frac{1}{2} I_{c1} \|$$
$$K_{\text{Rot},2} = \frac{1}{2} I_{c2} \|\boldsymbol{\omega}_2\|^2$$

where

$$I_{c1} = \frac{1}{12} m_1 l_1^2$$
$$I_{c2} = \frac{1}{12} m_2 l_2^2$$
$$\|\boldsymbol{\omega}_1\|^2 = \dot{\theta}_1^2$$
$$\|\boldsymbol{\omega}_2\|^2 = \left(\dot{\theta}_1 + \dot{\theta}_2\right)^2$$

The kinetic energy of the system is given by

$$\begin{aligned} K_{\text{sys}} &= K_{\text{Rot},1} + K_{\text{Rot},2} + K_{\text{Lin},1} + K_{\text{Lin},2} \\ &= \frac{1}{2} l_1^2 \dot{\theta}_1^2 \left(\frac{1}{3} m_1 + m_2\right) + \frac{1}{6} m_2 {l_2}^2 \left(\dot{\theta}_1 + \dot{\theta}_2\right)^2 + \frac{1}{2} m_2 l_1 l_2 \cos\left(\theta_2\right) \dot{\theta}_1 \left(\dot{\theta} + \dot{\theta}_2\right) \end{aligned}$$

Potential Energy

Considering the base joint to be the datum, the potential energy of the system is given by

$$P_{\text{sys}} = P_1 + P_2 = m_1 g \frac{l_1}{2} \sin\theta_1 + m_2 g \left[l_1 \sin\theta_1 + \frac{l_2}{2} \sin\left(\theta_1 + \theta_2\right) \right]$$

Lagrangian 2nd Kind

The equations of motion can be derived from the system lagrangian

$$L_{\text{sys}} = K_{\text{sys}} - P_{\text{sys}}$$
$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial L}{\partial \boldsymbol{q}} = 0$$

where $\boldsymbol{q} = (\theta_1 \ \theta_2)^\intercal$ The individual terms are

$$\frac{\partial L}{\partial \theta} = -m_1 g \frac{l_1}{2} \cos\theta_1 - m_2 g \left[l_1 \cos\theta_1 + \frac{l_2}{2} \cos\left(\theta_1 + \theta_2\right) \right]$$
$$\frac{\partial L}{\partial \theta_2} = -\frac{1}{2} m_2 l_1 l_2 \sin\left(\theta_2\right) \dot{\theta}_1 \left(\dot{\theta}_1 + \dot{\theta}_2\right) - \frac{1}{2} m_2 g l_1 \cos\left(\theta_1 + \theta_2\right)$$

$$\begin{aligned}
\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) &= \left(\frac{1}{3} m_1 {l_1}^2 + m_2 {l_1}^2 + \frac{1}{3} m_2 {l_2}^2 + m_2 l_1 l_2 \cos\theta_1\right) \ddot{\theta}_1 \\
&+ \left(\frac{1}{3} m_2 {l_2}^2 + \frac{1}{2} m_2 l_1 l_2 \cos\theta_1\right) \ddot{\theta}_2 + \left(-m_2 l_1 l_2 \sin\theta_1 \dot{\theta}_2\right) \dot{\theta}_1 \\
&+ \left(-\frac{1}{2} m_2 l_1 l_2 \sin\theta_1 \dot{\theta}_2\right) \dot{\theta}_2 \\
\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_2}\right) &= \left(\frac{1}{3} m_2 l_2^2 + \frac{1}{2} m_2 l_1 l_2 \cos\theta_2\right) \ddot{\theta}_1 + \frac{1}{3} m_2 l_2^2 \ddot{\theta}_2 - \frac{1}{2} m_2 l_1 l_2 \sin\theta_2 \dot{\theta}_1 \dot{\theta}_2
\end{aligned}$$

The final equations of motion are represented by

$$\begin{bmatrix} \frac{1}{3} m_1 {l_1}^2 + m_2 {l_1}^2 + \frac{1}{3} m_2 {l_2}^2 + m_2 l_1 l_2 \cos\theta_1 & \frac{1}{3} m_2 {l_2}^2 + \frac{1}{2} m_2 l_1 l_2 \cos\theta_2 \\ \frac{1}{3} m_2 {l_2}^2 + \frac{1}{2} m_2 l_1 l_2 \cos\theta_2 & \frac{1}{3} m_2 {l_2}^2 \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix}$$
$$\begin{bmatrix} -m_2 l_1 l_2 \sin\theta_1 \dot{\theta}_2 & -\frac{1}{2} m_2 l_1 l_2 \sin\theta_2 \dot{\theta}_2 \\ \frac{1}{2} m_2 l_1 l_2 \sin\theta_2 \dot{\theta}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$
$$\begin{bmatrix} m_1 g \frac{l_1}{2} \cos\theta_1 + m_2 g \left(l_1 \cos\theta_1 + \frac{l_2}{2} \cos\left(\theta_1 + \theta_2\right)\right) \\ \frac{1}{2} m_2 g l_1 \cos\left(\theta_1 + \theta_2\right) \end{bmatrix} = 0$$

The matrices $\boldsymbol{M}, \boldsymbol{C}$ and the vector $\boldsymbol{G}$ can be trivially found from the above equations.

The equations of motion have been symbolically derived and verified in the Jupyter Notebook file *2RRobotDynamics.ipynb*.

Formulation using Task-space Coordinates

The jacobian information can be used to obtain the equations of motion in task-space coordinates. We have,

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \\ l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \end{bmatrix}$$

$$\implies \boldsymbol{J} = \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

$$\implies \dot{\boldsymbol{J}} = \begin{bmatrix} -l_1 \cos\theta_1\dot{\theta}_1 - l_2 \cos(\theta_1 + \theta_2)\left(\dot{\theta}_1 + \dot{\theta}_1\right) & -l_2 \cos(\theta_1 + \theta_2)\left(\dot{\theta}_1 + \dot{\theta}_1\right) \\ -l_1 \sin\theta_1\dot{\theta}_1 - l_2 \sin(\theta_1 + \theta_2)\left(\dot{\theta}_1 + \dot{\theta}_1\right) & -l_2 \sin(\theta_1 + \theta_2)\left(\dot{\theta}_1 + \dot{\theta}_1\right) \end{bmatrix}$$

Now

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}} \qquad \ddot{\boldsymbol{x}} = \dot{\boldsymbol{J}}\dot{\boldsymbol{q}} + \boldsymbol{J}\ddot{\boldsymbol{q}}$$

$$\implies \dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}\dot{\boldsymbol{x}} \qquad \ddot{\boldsymbol{q}} = \boldsymbol{J}^{-1}\left(\ddot{\boldsymbol{x}} - \dot{\boldsymbol{J}}\boldsymbol{J}^{-1}\dot{\boldsymbol{x}}\right)$$

The equations of motions in task-space coordinates are hence obtained

$$\boldsymbol{M}\boldsymbol{J}^{-1}\ddot{\boldsymbol{x}} + \left(\boldsymbol{C}\boldsymbol{J}^{-1} - \boldsymbol{M}\boldsymbol{J}^{-1}\dot{\boldsymbol{J}}\boldsymbol{J}^{-1}\right)\dot{\boldsymbol{x}} + \boldsymbol{G} = \boldsymbol{\tau}$$

Exercise 9

1. For $n = 1$ one parameter is required, for $n = 2$ three parameters are required, for $n = 3$ six parameters are required.

2. The mass-inertia matrix is atleast semi-positive definite and symmetric. This can be explained with the help of the kinetic energy

   $$K = \frac{1}{2}\boldsymbol{v}^\intercal \boldsymbol{M}\boldsymbol{v} \in \mathbb{R}$$

   Since kinetic energy is a scalar quantity

   $$\frac{1}{2}\boldsymbol{v}^\intercal \boldsymbol{M}\boldsymbol{v} = \frac{1}{2}\boldsymbol{v}^\intercal \boldsymbol{M}^\intercal \boldsymbol{v}$$
   $$\implies \boldsymbol{M} = \boldsymbol{M}^\intercal \qquad \text{Symmetry}$$

   Since kinetic energy can not be negative

   $$\frac{1}{2}\boldsymbol{v}^\intercal \boldsymbol{M}\boldsymbol{v} \geq 0 \qquad \text{Semi-Positive Definite}$$

3. Donald can use the linear scalar factor $\frac{1}{1.355818}$ which converts the energy from *Joule* (Metric) to *Feet-Pound* (Imperial).

4. Yes, they will get the same values since the mass-inertia matrix is invariant to coordinate transformations for rigid bodies.

4 Control

Exercise 10

The forced system equation of the pendulum is given by

$$ml^2\ddot{\theta} + bl\dot{\theta} + mgl\sin\theta = \tau \tag{4.1}$$

For the PD-controller, the actuator torque will be

$$\tau = K_p\left(\pi - \theta\right) - K_d\left(\dot{\theta}\right)$$

Substituting the actuator torque in (4.1), we get

$$\ddot{\theta} + \left(\frac{bl + K_d}{ml^2}\right)\dot{\theta} + \frac{mgl\sin\theta + K_p\left(\theta - \pi\right)}{ml^2} = 0$$

LaSalle's Invariance Principle

Consider an energy function $V\left(\boldsymbol{x}\right)$ of the system which is radially unbounded and positive definite. Consider a subspace Ω where $\dot{V}\left(x\right) \leq 0$. If the initial state $\boldsymbol{x}_0$ lies in this subspace, then it converges to the invariant set $S \subset \Omega$ where $\dot{V}\left(x\right) = 0$. If the set S contains only the equilibrium point $\boldsymbol{x}_E$, then $\boldsymbol{x}_E$ is globally asymptotically periodic.

Proving Asymptotic Stability

Let the total energy of the system be the energy function candidate. Assume the lower equilibrium point $(0, 0)$ is the datum. Consider $\boldsymbol{x} = \left(\theta\ \dot{\theta}\right)^{\mathsf{T}}$

$$V\left(\boldsymbol{x}\right) = \frac{1}{2}ml^2\dot{\theta}^2 + mgl\left(1 - \cos\theta\right)$$

$$\lim_{\|\boldsymbol{x}\|\to\infty} V\left(\boldsymbol{x}\right) = \infty \qquad \text{Radially Unbounded}$$

$$V > 0 \qquad \text{Positive Definite}$$

$$\dot{V} = \left(\tau - bl\dot{\theta}\right)\dot{\theta}$$

If $\tau = bl\dot{\theta} - k\dot{\theta}$ with $k > 0$

$$\dot{V} = -k\dot{\theta}^2 \leq 0 \qquad \text{Semi-definite negative}$$

$$\dot{V} = 0 \Leftrightarrow \dot{\theta} = 0$$

Hence, only the two equilibrium points $(0, 0)$ and $(\pi, 0)$ correspond to the invariant set which means $(\pi, 0)$ is asymptotically globally stable in the presence of an actuation input.

PD Controller Response

The PD controller along with the euler integrator has been implemented in *eulerIntegrator.py* and *PD_Pendulum.py*, The PD controller parameters are set as $K_p = 20$ and $K_d = 1$.

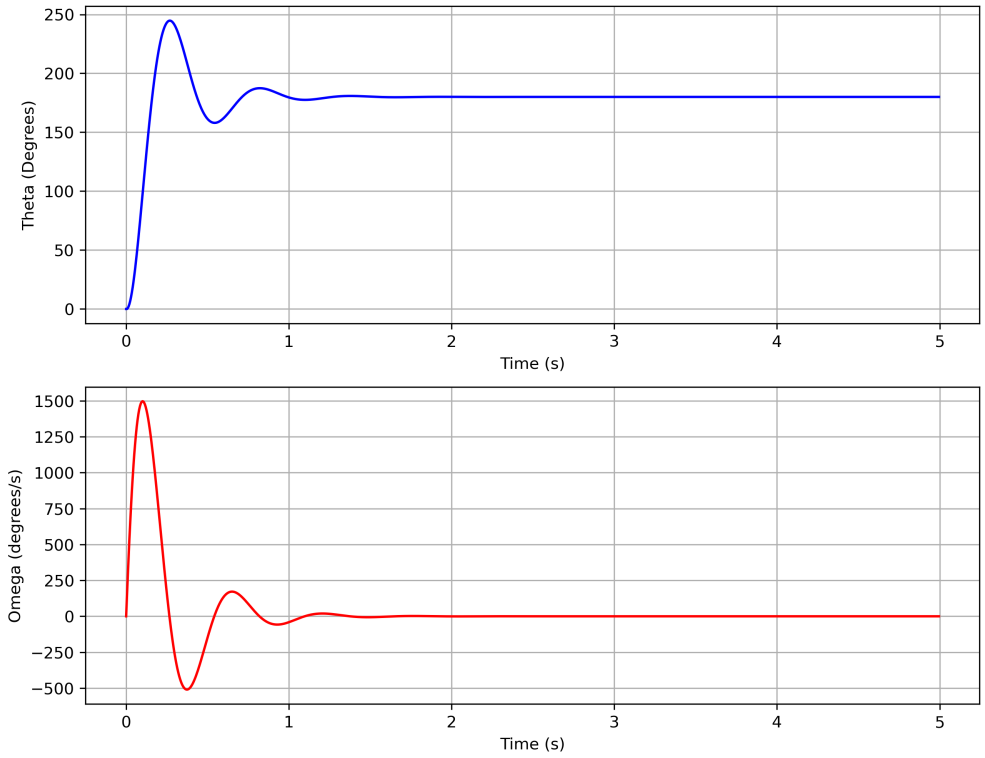

Figure 4.1: Closed-Loop System Response of PD Controller

Figure 4.1 and Figure 4.2 illustrate the acceptable behaviour of the implemented PD Controller considering there are no actuator limits.

PD Controller Response with actuator limits

For the same controller parameters, a hard limit of $\tau_{\max} = 1Nm$ is implemented on the PD Controller.

Figure 4.3 clearly shows that the implemented PD controller fails in presence of the actuator constraint. A possible reason for a PD controller failing to work in presence of a actuator limit (or requiring so much actuation effort in the first place) maybe due to the fact that a PD controller is essentially a linear controller, i.e. designed for a linear system, whereas the swing-up motion falls under the non-linear behaviour of the pendulum. This means that the PD controller may be more effective in balancing the pendulum around the equilibrium points (around which the system can be linearized),

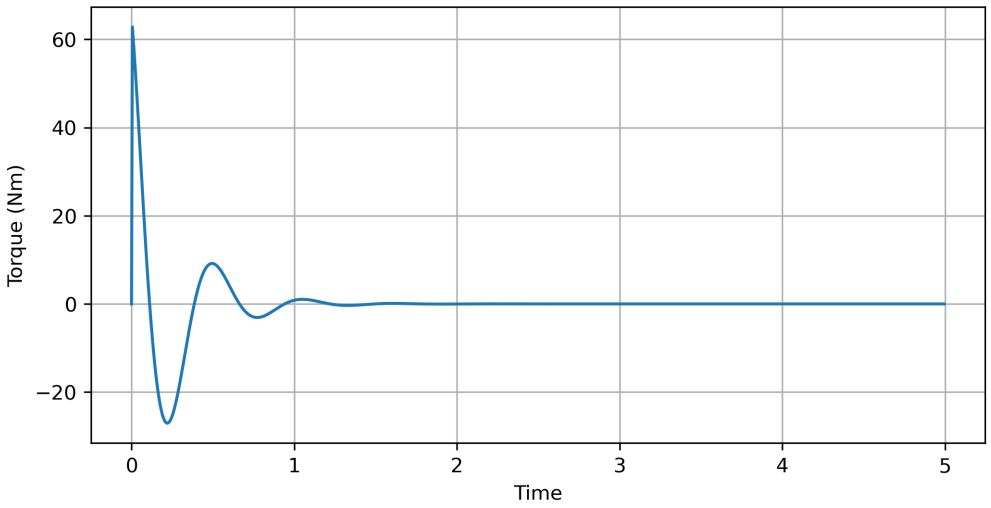

Figure 4.2: Actuator Torque (PD Controller)

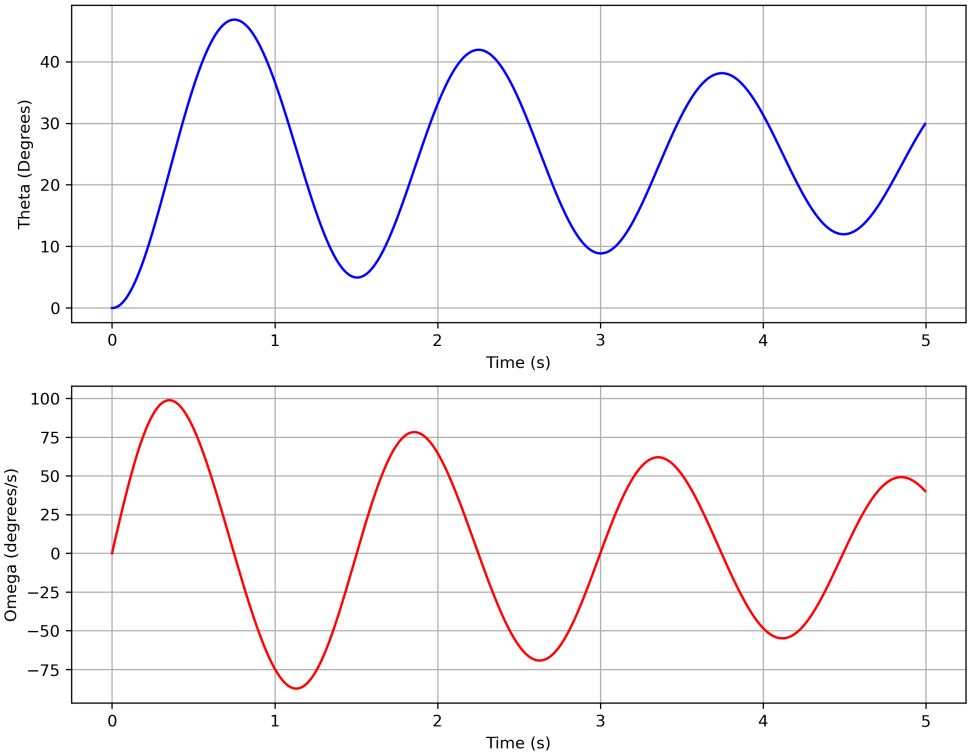

Figure 4.3: Closed-Loop System Response of PD Controller with actuator limit

rather than bringing the pendulum from the downright to upright position.

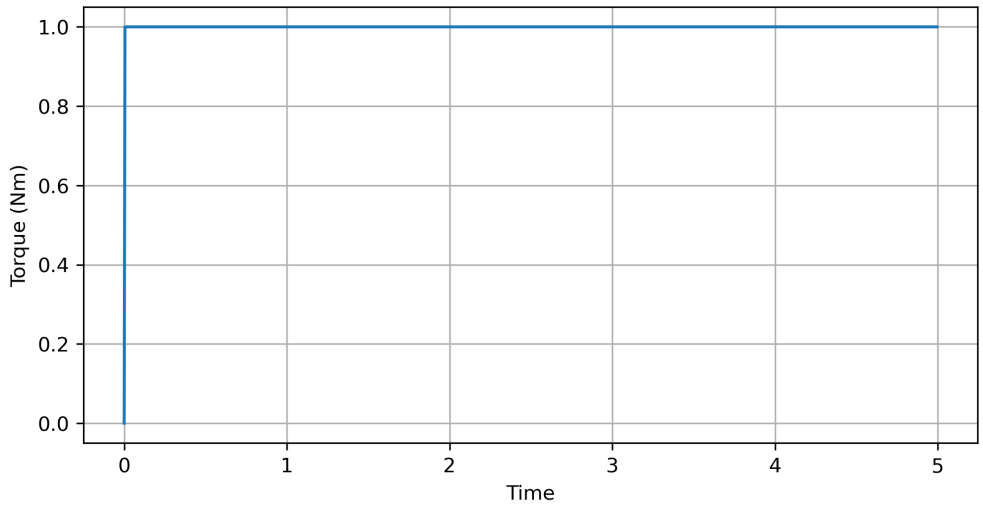

Figure 4.4: Actuator Torque (PD Controller with actuator limit)

Energy-based Controller for Swing-Up

The idea behind the control objective is to add energy to the pendulum until the desired energy is achieved i.e. the control set-point is not the angle and angular velocity, rather the energy of the pendulum at the upper equilibrium point. Assuming the potential energy datum to be at the lower equilibrium point,

$$E_{\text{des}} = 2mgl$$
$$E_{\text{act}} = \frac{1}{2}ml^2\dot{\theta}^2 + mgl\,(1 - \cos\theta)$$

The actuator effort is then given by

$$\tau = K\,(E_{\text{des}} - E_{\text{act}})$$

In order to ensure the direction of the input torque is in the current direction, the direction of $\dot{\theta}$ is monitored such that

$$\tau = K\,(E_{\text{des}} - E_{\text{act}})\,\text{sgn}\left(\dot{\theta}\right)$$

Limiting the torque within $[-\tau_{\max}, \tau_{\max}]$ gives us the final form of the actuator effort

$$\tau = \text{sat}\left(K\,(E_{\text{des}} - E_{\text{act}})\,\text{sgn}\left(\dot{\theta}\right), \tau_{\max}\right)$$

For the closed-loop response $K = 10$

Figure 4.5 illusrates that while the desired setpoint is reached (a peak of about 177 degrees is reached) at about three seconds, the position is not maintained. On the other hand, Figure 4.6 proves that the energy-based controller performs while maintaining

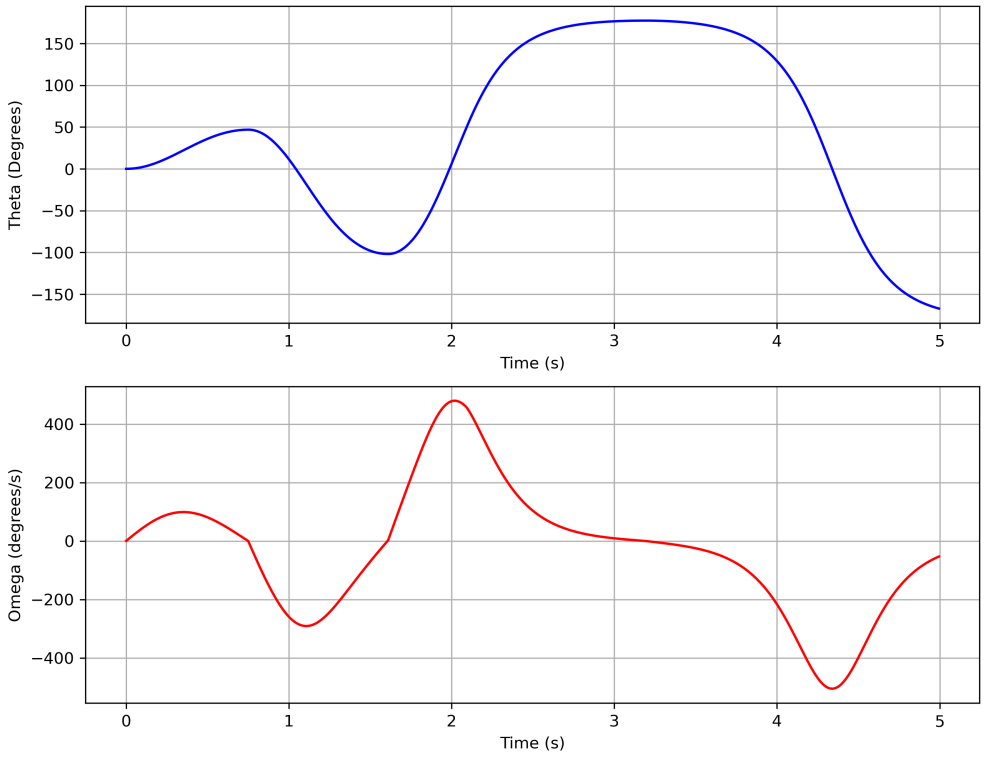

Figure 4.5: Closed-Loop System Response of Energy based controller for swing-up

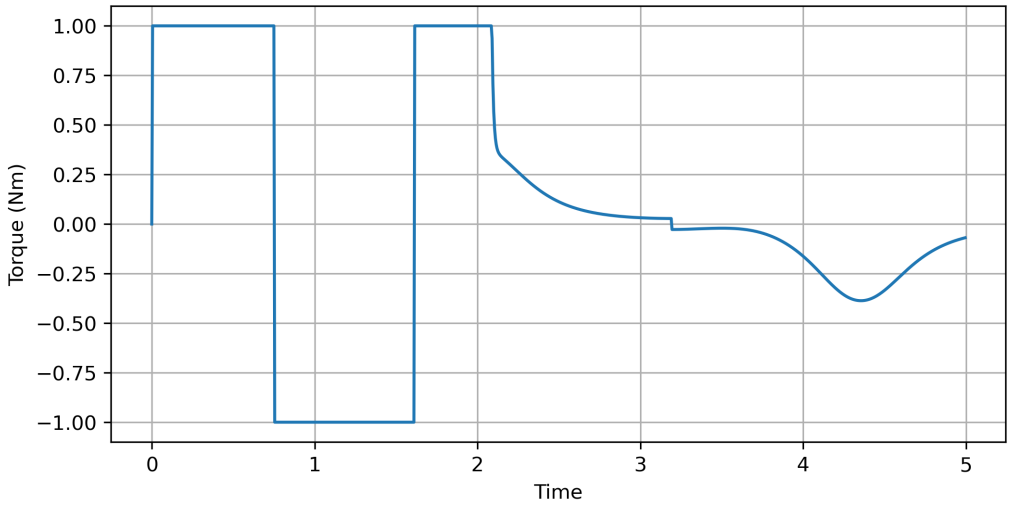

Figure 4.6: Actuator Torque (Energy based controller with actuator limit)

the actuator limits. Furthermore it is observed for double the simulation time that the

controller response essentially is cyclic i.e. it reaches the upper equilibrium point, then returns back to it from the other side (-180 degrees) and then the cycle continues.

The performance may be improved by having a linear quadratic regulator take-over the energy controller when the pendulum approaches the upper equilibrium point. This is advantageous since the system can be linearized as it approaches the upper equilibrium point and the actuator limit can also be maintained since the LQR minimizes actuation energy as well.

Exercise 11

In order to make use of the convenience of inbuilt tools already available, the *ode45* solver from MATLAB® is made use of to emulate the fourth-order Runge-Kutta scheme. The implementation of the same along with the euler integrator can be found in *Pendulum.m*.

Trade-offs between Euler Integrator and Fourth-order Runge-Kutta

1. The euler integrator implemented is the forward euler method which is an explicit method and is highly sensitive to the time step-size. Very large step-sizes can lead to numerical instabilities. On the other hand, the fourth-order Runge-Kutta handles smaller step-sizes much better.

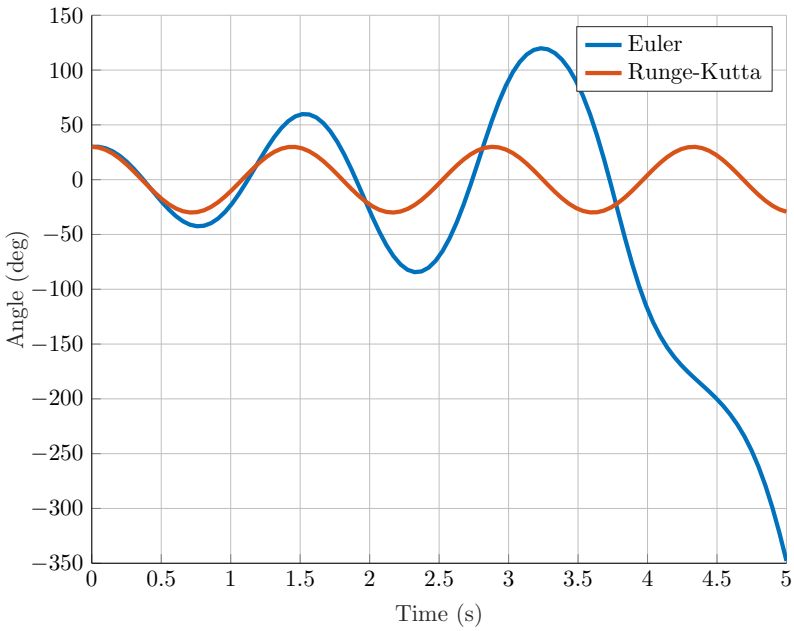

Figure 4.7: Solution of the open-loop pendulum from Euler-Integrator and Runge-Kutta for time step-size = 0.05

Figure 4.7 demonstrates the diverging behavior of the euler integrator for step-size of 0.05 seconds. On the other hand, the two solutions are a lot closer for 0.005

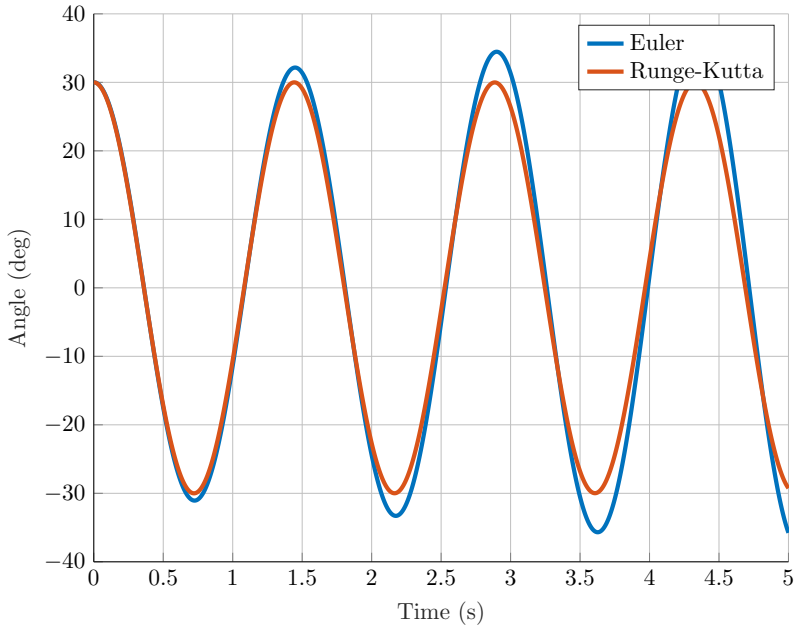

Figure 4.8: Solution of the open-loop pendulum from Euler-Integrator and Runge-Kutta for time step-size = 0.005

seconds as demonstrated by Figure 4.8.

2. Euler integrator is easier to implement compared to the fourth-order Runge-Kutta since the fourth-order Runge-Kutta requires four approximations of the derivative and Euler integrator requires only one.

3. Error accumulation over time is faster in Euler as already demonstrated in Figure 4.7.

Linearization of the System about the upper equilibrium point

The non-linear state-space is given by

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ \frac{u}{ml^2} - \frac{b\dot{\theta}}{ml} - \frac{g\sin\theta}{l} \end{bmatrix} = \boldsymbol{f}\left(\boldsymbol{x}, u\right)$$

The linear states will be given by

$$\begin{bmatrix} z \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \pi - \theta \\ -\dot{\theta} \end{bmatrix} = \begin{bmatrix} \delta\theta \\ \delta\dot{\theta} \end{bmatrix}$$

The first-order taylor series approximation around $\boldsymbol{x}_E = (\pi, 0)$ is given by

$$\begin{bmatrix} \delta\dot{\theta} \\ \delta\ddot{\theta} \end{bmatrix} \approx \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=(\pi,0)} \begin{bmatrix} \delta\theta \\ \delta\dot{\theta} \end{bmatrix}$$

$$\implies \begin{bmatrix} \dot{z} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & -\frac{b}{ml} \end{bmatrix} \begin{bmatrix} \pi - \theta \\ -\dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{ml^2} \end{bmatrix} u$$

$$\implies \begin{bmatrix} \dot{z} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & -\frac{b}{ml} \end{bmatrix} \begin{bmatrix} z \\ \dot{z} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{ml^2} \end{bmatrix} u$$

Hence, for the linear system around the upper equilibrium point,

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & -\frac{b}{ml} \end{bmatrix}$$

$$\boldsymbol{b} = \begin{bmatrix} 0 \\ \frac{1}{ml^2} \end{bmatrix}$$

Linear Quadratic Regulator Design

A LQR is designed assuming full-state feedback. Assuming an infinite-time horizon, the algebraic Riccati equation is solved in order to obtain the LQR gain matrix $\boldsymbol{K}_{\text{LQR}} \in \mathbb{R}^{1\times 2}$. Positive definite and symmetric matrices $\boldsymbol{Q} \in \mathbb{R}^{2\times 2}$ and $\boldsymbol{R} \in \mathbb{R}$ are used to prioritize the minimization of the transient energy (states reach reference quicker) and the minimization of the actuator energy (actuator limits are prioritized) respectively.

The closed-loop linear system with LQR is formulated as

$$\dot{\boldsymbol{z}} = \boldsymbol{A}\boldsymbol{z} + \boldsymbol{b}u \qquad u = -\boldsymbol{K}_{\text{LQR}}\boldsymbol{z}$$

$$\dot{\boldsymbol{z}} = (\boldsymbol{A} - \boldsymbol{b}\boldsymbol{K}_{\text{LQR}})\,\boldsymbol{z} \qquad \boldsymbol{A}_{\text{CL}} = \boldsymbol{A} - \boldsymbol{b}\boldsymbol{K}_{\text{LQR}}$$

Strategy for Swing-up and Balancing

1. Perform swing-up using energy error till a close value to 180 degrees is reached.
2. Define a threshold (ϵ) for linear behaviour of the pendulum about its upright equilibrium point. For this implementation, $\epsilon = 10$ degrees.
3. Switch to the LQR Controller once the linear threshold is reached in order to balance the pendulum at its upright position.

For this implementation, $\boldsymbol{Q} = 0.1\boldsymbol{I}_{2\times 2}$ and $R = 1$. The values of $\boldsymbol{Q}$ and R can naturally be fine-tuned in order to ensure a balance between faster response time and the actuator limit of $1Nm$.

This strategy implementation can be found in *swingUp_Balance_Pend.m*.

Figure 4.9 clearly shows the decent performance of the implemented strategy where the upright position is maintained as compared to when only the energy based controller is used.

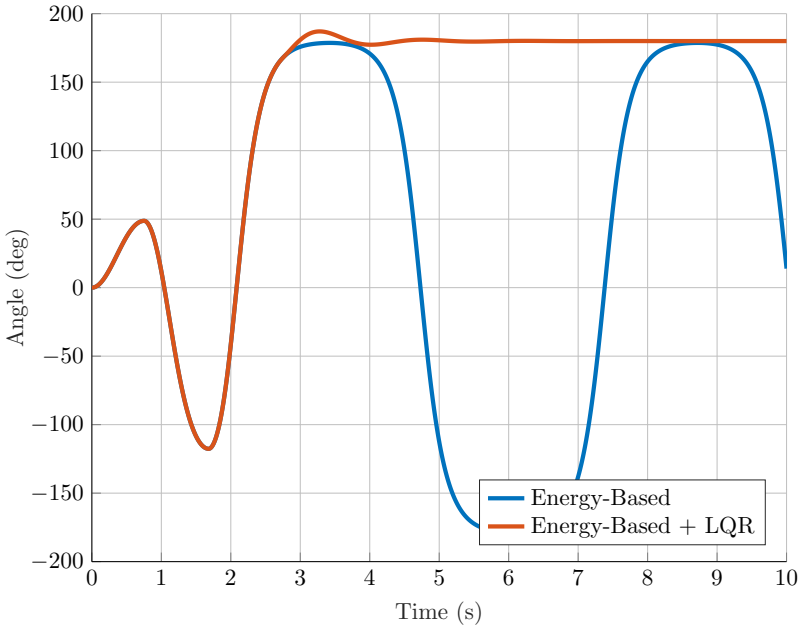

Figure 4.9: Swing-up and Balance Response

5 Questionnaire

German grade scale: 1.0 is highest, 4.0 is lowest

1. Mechanics (Bachelor), Mechanism Theory (Bachelor), Machine Design (Bachelor), Multi-body Dynamics (Master, 1.0), Linear Control Theory (Master, 1.0), Non-Linear Control Theory (Master, 1.0), Machine Learning (Master, 1.7), Introduction to Robotics (Bachelor), Linear Algebra (Bachelor), Advanced Calculus (Bachelor), Object Oriented Programming (Master, 1.3)
2. Medium
3. Textbooks, Research Paper
4. Solved without any external reference: Exercise 6, 7, 8, 9 ,Solved with the help of known textbooks: Exercise 1 to Exercise 5, Solved after reading research papers and online content: The control part
5. C/C++ (Decent), Python (Advanced), Matlab(Advanced)
6. Matlab Symbolic toolbox, Sympy
7. 3-axis SCARA.
8. Involved with UAVs (aerial robot) during master thesis (both software and hardware)
9. Git
10. Advanced Latex Skills