Mechanics & Control Test

Martian Mindset Application

Tersoo Samuel Awai

July 2025

introduction

This submission is divided into two complementary artefacts:

- **Compiled report (this PDF).** Derivations, key results, and discussion are presented here. Wherever a result is generated by code, a small footnote that links directly to the corresponding script in the repository.
- **Open-source code base.** All source files are version-controlled on GitHub at `github.com/⟨awai005⟩/martian_mindset_test`. The repository contains three folders:
  1. `notebooks/` – Jupyter notebooks that reproduce Exercises 1–8 step-by-step, mirroring the order in the report.
  2. `matlab/` – MATLAB scripts for all pendulum simulations (energy-pumping, LQR stabilisation, and state-space visualisation). Git tags mark every major experiment so results in the PDF can be traced to the exact commit that produced them.

The LaTeX source uses the `hyperref` package, so every code reference in the text (e.g., Listing C.1) is a live link. Reviewers may therefore run or inspect any fragment of the solution without searching through the repository hierarchy.

Together, the PDF and the repository provide a completely reproducible record: mathematical development in print; numerics and graphics in executable code.

1 Mechanics

1.1 Topology and Mobility Analysis

Exercise 1 – Topology

A concise topological graph for the serial and parallel manipulators is shown in Fig. 1.

For background on graph-based classification of robot architectures, see Baron's survey [1].

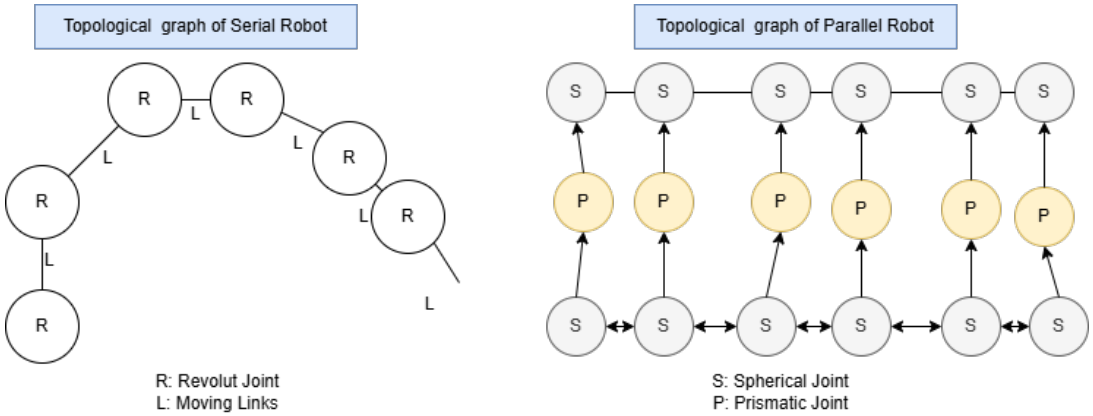

Figure 1: Topology diagram of the Serial Robot and Parallel Robot.

Exercise 2 – Mobility of Serial and Parallel Robots

Serial Robot: For a spatial mechanism with six motion parameters per link the Chebychev–Grübler–Kutzbach criterion can be implemented as follows:

$$m_s \;=\; 6\big(N-j-1\big) \;+\; \sum_{i=1}^{j} f_i \;=\; 6(-c) \;+\; \sum_{i=1}^{j} f_i, \tag{1}$$

where
N is the number of links (including the ground)
j the number of joints
f_i the DOF of joint i
$c = j + 1 - N$ counts *independent* closed loops.

$$N = 7 \;\rightarrow 6 \text{ moving} + \text{base}, \quad j = 6, \quad c = 0, \quad \sum f_i = 6 \times 1 = 6.$$

Hence

$$m_s = 6\big(7-6-1\big) + 6 \;=\; 6.$$

Hence, the arm possesses the expected six spatial DOF.

Simplified Formula: For an open n-joint serial chain there are *no* independent closed loops ($c = 0$) and the number of links is $N = j + 1$ see [2]. Substituting these relations into Equation (1)[1] gives

$$\boxed{m_s \;=\; \sum_{i=1}^{j} f_i},$$

i.e. the mobility of a serial manipulator equals the sum of its joint degrees of freedom. Parallel Robot: Each leg is an S–P–S chain (one prismatic actuator between two spherical joints):

$$\begin{aligned} N &= 14 \;\rightarrow \text{base} + \text{platform} + 6 \times 2 \text{ rods}) \\ j &= 18 \;\rightarrow 6 \times 2S + 6 \times 1P \\ \sum f_i &= 6(3+1+3) = 42. \end{aligned}$$

[1]CGK formula for spatial mechanisms: $m_s = 6(N-j-1) + \sum_{i=1}^{j} f_i$.

The mechanism closes $c = 6$ independent loops. Therefore

$$m_s^{\text{parallel}} \;=\; 6(-5) + 42 \;=\; 12.$$

This "12 DOF" value over-estimates the actual mobility because the platform is *statically over-constrained*.

Redundancy correction: As shown by Merlet [3, Sec. 2.2], the CGK formula must subtract the directions r_k that do not further restrict motion:

$$m_s \;=\; 6\,(N - j - 1) + \sum_{i=1}^{j} f_i \;-\; \sum_{k} r_k. \tag{2}$$

For the SPS[6] robot each leg contributes six redundant constraint directions, so $\sum_k r_k = 6$. Hence

$$m_s^{\text{Stewart}} = 12 - 6 = 6,$$

which matches physical reality.

Loop-count alternative (same result): Equivalently, one may count the six independent closed kinematic loops ($c = 6$—one per leg) and apply the loop form

$$m_s = 6(-c) + \sum f_i = 6(-6) + 42 = 6.$$

Both approaches confirm the Stewart–Gough platform's true mobility of six spatial degrees of freedom.

Exercise 3 — When does CGK fail? Classical counter-examples

The Chebychev–Grübler–Kutzbach count

$$m_s \;=\; 6\,(N - j - 1) \;+\; \sum_{i=1}^{j} f_i$$

assumes that every constraint introduced by every joint is *independent* and that the links behave as perfectly rigid bodies [4, Ch. 1]. When those assumptions break down, the formula can either under- or over-predict the true mobility.

(1) Over-constrained, paradoxical linkages. Bennett's spatial 4R linkage has four revolute joints whose axes satisfy specific skew-angle and offset conditions. CGK yields

$$m_s^{\text{CGK}} = 6\,(6 - 4 - 1) + 4 = -2,$$

yet the mechanism has *one* genuine degree of freedom [5, Sec. 3.3.3]. The "missing" mobility arises because two constraint directions are redundant; CGK counts them twice and predicts a negative value.

(2) Flexible-motion Sarrus linkage. Sarrus's 6R parallel linkage comprises three pairs of opposite links connected by revolute joints arranged in two parallel planes. With $N = 7,\ j = 6,\ f_i = 1$, CGK returns

$$m_s^{\mathrm{CGK}} = 6\,(7 - 6 - 1) + 6 = 0,$$

yet the structure folds with *one* DOF, forming a paradoxical "motion-platform" without a prismatic joint [6, p. 148]. Again, hidden constraint redundancies invalidate the independence assumption.

From the above examples, it can be infered that while CGK is a powerful first-pass tool, it does not always work. Classic spatial linkages like Bennett's and Sarrus's provide clear counter-examples where the naive count fails.

1.2 Geometry and Kinematics

Exercise 4 - Kinematics of 3DOF Robotic leg

Primitive Homogeneous Transforms Every rigid-body displacement can be written as a 4×4 homogeneous matrix appendix A.1

[2] DH Parameters for the 3-DOF Leg

Link i	α_{i-1}	a_{i-1} [m]	d_i [m]	θ_i
1	$+90^\circ$	0	0	q_1
2	-90°	0	0	q_2
3	0°	0	$\mathbf{q}_3$	0

This orthogonal frame choice follows from the pretext of exercise 4. Also the lengths between the joints are assumed to be zero since they arent explicitly mentioned in the problem.

Composite Transform to the End-Effector

Because all $a_{i-1} = 0$, the product simplifies to

$${}^0T_E = R_z(q_1)\,R_y(q_2)\,T_z(q_3) = \begin{bmatrix} c_1c_2 & s_1 & -c_1s_2 & c_1c_2\,q_3 \\ s_1c_2 & -c_1 & -s_1s_2 & s_1c_2\,q_3 \\ s_2 & 0 & c_2 & s_2\,q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad c_i = \cos q_i,\ s_i = \sin q_i.$$

Geometric object for E: For any fixed prismatic extension $q_3 = r$, the tip of the leg satisfies the equation $x^2 + y^2 + z^2 = r^2$; hence the end-effector point E lies on the surface of a **sphere** of radius r centred at the hip joint.

Forward Kinematics

The last column is $(x, y, z, 1)^{\mathsf{T}}$, hence

[2] The Denavit–Hartenberg (DH) geometry solution from [7] is used

$$x = q_3\, c_2 c_1, \qquad y = q_3\, c_2 s_1, \qquad z = q_3\, s_2$$

Inverse Kinematics

Given a non-zero Cartesian point (x, y, z) inside the prismatic limits, solve for (q_1, q_2, q_3).

Step 1: prismatic extension. Square and add the FK equations:

$$x^2 + y^2 + z^2 = q_3^2.$$

Thus

$$q_3 = \sqrt{x^2 + y^2 + z^2}$$

Step 2: yaw angle q_1. From the first two FK rows $\tan q_1 = y/x$. Use the two-argument arctangent for the correct quadrant:

$$q_1 = \text{atan2}(y, x)$$

Step 3: pitch angle q_2. Define the planar radius $\rho = \sqrt{x^2 + y^2} = q_3\, c_2$. Then $z = q_3\, s_2$. Hence

$$\tan q_2 = \frac{z}{\rho} = \frac{z}{\sqrt{x^2 + y^2}}.$$

Again using atan2 (non-negative denominator):

$$q_2 = \text{atan2}\big(z, \sqrt{x^2 + y^2}\big)$$

the collected inverse map is as follows:

$$\boxed{\begin{aligned} q_3 &= \sqrt{x^2 + y^2 + z^2}, \\ q_1 &= \text{atan2}(y, x), \\ q_2 &= \text{atan2}\big(z, \sqrt{x^2 + y^2}\big). \end{aligned}}$$

Fig. 2 is the reachable workspace + sample pose.

[3]

[3]Source code and executable notebook available at `https://github.com/Awai005/martian-mindset-mech-control-test/blob/main/code/notebooks/Exercise_4_Kinematics1.ipynb`

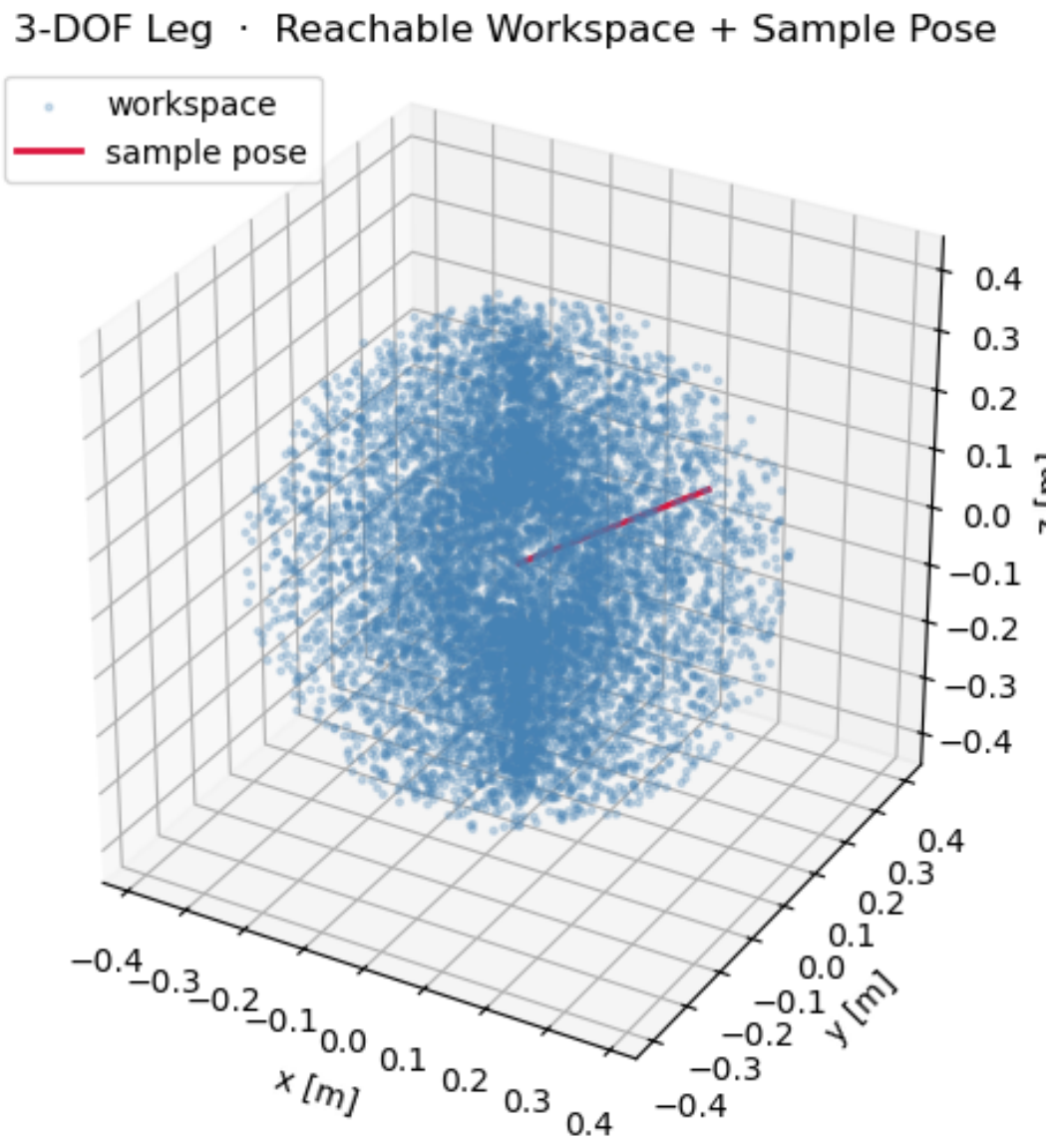

Figure 2: 3-DOF Leg. Reachable Workspace + Sample Pose

Exercise 5

Vector definitions

All points lie in the fixed xy-plane with ground pivot $O = (0, 0)$.

$$\mathbf{l}_1 = l_1 \begin{bmatrix} \cos q_1 \\ \sin q_1 \end{bmatrix}, \qquad \mathbf{l}_2 = l_2 \begin{bmatrix} \cos \phi \\ \sin \phi \end{bmatrix}, \qquad \mathbf{d} = \begin{bmatrix} d \\ 0 \end{bmatrix}, \quad d = q_3,$$

where

* l_1 = crank length OA (fixed), * l_2 = connecting-rod length AB (fixed), * d = slider displacement OB (prismatic joint), * ϕ = instantaneous orientation of the rod w.r.t. the x-axis.

Loop-closure equation is given as:

$$\mathbf{l}_1 + \mathbf{l}_2 = \mathbf{d}$$

In Coordinate form we have:

$$\begin{aligned} l_1 \cos q_1 + l_2 \cos \phi &= d, \\ l_1 \sin q_1 + l_2 \sin \phi &= 0. \end{aligned} \tag{1}$$

Eliminating the rod angle ϕ and deriving the scalar constraint see appendix A.2

Equation (19) is the fundamental relation linking the prismatic variable $d = q_3$ and the crank angle q_1; all forward and inverse-kinematics formulas follow directly from it.

Forward kinematics $q_1 = f(d)$

Start from the scalar loop-closure constraint (derived earlier):

$$(d - l_1 \cos q_1)^2 + l_1^2 \sin^2 q_1 = l_2^2 \tag{3}$$

Expand (3) and solve for $\cos q_1$:

$$d^2 - 2 l_1 d \cos q_1 + l_1^2 = l_2^2 \implies \cos q_1 = \frac{d^2 + l_1^2 - l_2^2}{2 l_1 d}$$

Principal (elbow-up) branch $q_1 \in [0, \pi]$:

$$q_1 = \arccos\!\left(\tfrac{d^2 + l_1^2 - l_2^2}{2 l_1 d} \right)$$

Reality condition: the arccos argument must lie in $[-1, 1]$, giving the admissible stroke range $\big| l_1 - l_2 \big| \le d \le l_1 + l_2$.

Inverse kinematics $d = f^{-1}(q_1)$

Re-interpret (3) as a quadratic in d:

$$d^2 - 2l_1 \cos q_1 \; d + (l_1^2 - l_2^2) = 0.$$

Positive (slider-extension) root:

$$d = l_1 \cos q_1 + \sqrt{l_2^2 - l_1^2 \sin^2 q_1}\,, \qquad |\sin q_1| \le \frac{l_2}{l_1} \text{ (real stroke).}$$

[4]

Jacobian $J(d) = \dfrac{\partial q_1}{\partial d}$

From the forward relation $\cos q_1 = C(d)$ with

$$C(d) = \frac{d^2 + l_1^2 - l_2^2}{2 l_1 d}, \qquad C'(d) = \frac{\partial C}{\partial d} = \frac{1}{2l_1}\Big(1 - \frac{l_1^2 - l_2^2}{d^2}\Big),$$

differentiate $\cos q_1 = \cos q_1(d)$:

$$-\sin q_1 \, \frac{\partial q_1}{\partial d} = C'(d) \quad \Longrightarrow \quad J(d) = \frac{\partial q_1}{\partial d} = -\frac{1 - \dfrac{l_1^2 - l_2^2}{d^2}}{2 l_1 \sin q_1}$$

Maximum angular velocity

With a prismatic speed limit $\dot{d}_{\max}$,

$$\boxed{\dot{q}_{1,\max} = \big|\, J(d) \big|\, \dot{d}_{\max} = \frac{\big|1 - (l_1^2 - l_2^2)/d^2\big|}{2 l_1 \,|\sin q_1|}\; \dot{d}_{\max}}.$$

Maximum output torque

Virtual–work principle $f\,\dot{d} = \tau\,\dot{q}_1$ gives $\tau = f/J(d)$. With actuator force limit $f_{\max}$:

$$\boxed{\tau_{\max} = \frac{f_{\max}}{|J(d)|} = \frac{2 l_1 \,|\sin q_1|}{\big|1 - (l_1^2 - l_2^2)/d^2\big|}\; f_{\max}}.$$

Singular configurations Dead-centre singularities occur at $q_1 = 0$ or π, where $\sin q_1 = 0$ and $|J| \to \infty$, so the mechanism cannot generate crank torque regardless of actuator force. [7].

[4] The Jupyter notebook to verify exercise$_5$: `github.com/⟨awai005⟩/martian_mindset_test`

Exercise 6

1 Yes, Angela can find a linear scaling factor to convert the 3D position from meters to inches. The scaling factor is:

$$k = 39.3701$$

She applies:

$$(x_{B,\text{inches}}, y_{B,\text{inches}}, z_{B,\text{inches}}) = k \cdot (x_{B,\text{meters}}, y_{B,\text{meters}}, z_{B,\text{meters}})$$

and inputs the resulting coordinates into Donald's model to obtain the actuator commands.

2 No, Donald cannot find a single linear scaling factor to convert the 6D pose from Angela's metric system (meters, quaternions) to his imperial system (inches, Euler angles in degrees). While the translational components can be scaled linearly by $k = 39.3701$, the rotational components require a nonlinear transformation from quaternions to Euler angles, followed by a different scaling factor ($\frac{180}{\pi}$) for degrees. Thus, the conversion is not a single linear scaling across all dimensions.

3 Yes, it is possible for Donald and Angela to use each other's models. Angela converts her base position from meters to inches using the scaling factor $k = 39.3701$ and inputs the result into Donald's model. Donald converts his end-effector position from inches to meters using $k^{-1} = \frac{1}{39.3701}$ then converts his Euler angles from degrees to radians using $\frac{\pi}{180}$ and finally transforms the angles to a quaternion using the above equations. The converted pose is then the input into Angela's model. These conversions are applied externally, respecting the black-box constraint.

1.3 Dynamics

Exercise 7

The pendulum is described as follows:

- A point mass m at the end of a massless rod of length l,
- Angular displacement θ from the vertical (positive counterclockwise),
- A revolute joint at the top applying torque τ.

then:

1. Torque τ in terms of $\theta, \dot{\theta}, \ddot{\theta}$ (inverse dynamics),
2. Angular acceleration $\ddot{\theta}$ in terms of τ (forward dynamics).

Inverse Dynamics: Expression for Torque

Lets begin with Newton's second law for rotational motion:

$$\tau = I\ddot{\theta} + \text{gravitational torque}$$

For a point mass m at distance l, the moment of inertia is:

$$I = ml^2$$

The gravitational torque acting on the mass (opposing motion) is:

$$\tau_g = -mgl\sin\theta$$

Substituting, we obtain the total torque:

$$\tau = ml^2\ddot{\theta} + mgl\sin\theta$$

Final expression:

$$\tau = ml^2\ddot{\theta} + mgl\sin\theta$$

Forward Dynamics: Expression for Acceleration

Rearranging the inverse dynamics equation to solve for $\ddot{\theta}$:

$$\begin{aligned} \tau &= ml^2\ddot{\theta} + mgl\sin\theta \\ ml^2\ddot{\theta} &= \tau - mgl\sin\theta \\ \ddot{\theta} &= \frac{\tau - mgl\sin\theta}{ml^2} \end{aligned}$$

Final expression:

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

Simulation In MATLAB output of angle of pendulum over time. with no damping the system conserves energy and amplitude of oscilation does not change. [5]

Exercise 8

Joint Space Dynamics by Lagrange's Method

The joint angles are $q = [\theta_1, \theta_2]^T$, with velocities $\dot{q} = [\dot{\theta}_1, \dot{\theta}_2]^T$ and accelerations $\ddot{q} = [\ddot{\theta}_1, \ddot{\theta}_2]^T$. Link 1 has length l_1, mass m_1, and moment of inertia I_1 about its center of mass (COM) at $l_1/2$. Link 2 has length l_2, mass m_2, and moment of inertia I_2 about its COM at $l_2/2$. Gravity acts along the negative y-axis with acceleration g. The coordinate system has the first joint at the origin $(0, 0)$, with the x-axis horizontal and y-axis vertical.

[6] Step 1: Kinematics. The COM positions are:

$$x_{c1} = \frac{l_1}{2}\cos\theta_1, \quad y_{c1} = \frac{l_1}{2}\sin\theta_1$$

[5] Source code and executable MATLAB code for exercise 7, 10, 11 `https://github.com/Awai005/martian-mindset-mech-control-test/tree/main/code/control/pendulum`

[6] This solution used below is presented by Asada [8, Eq. 7-64]

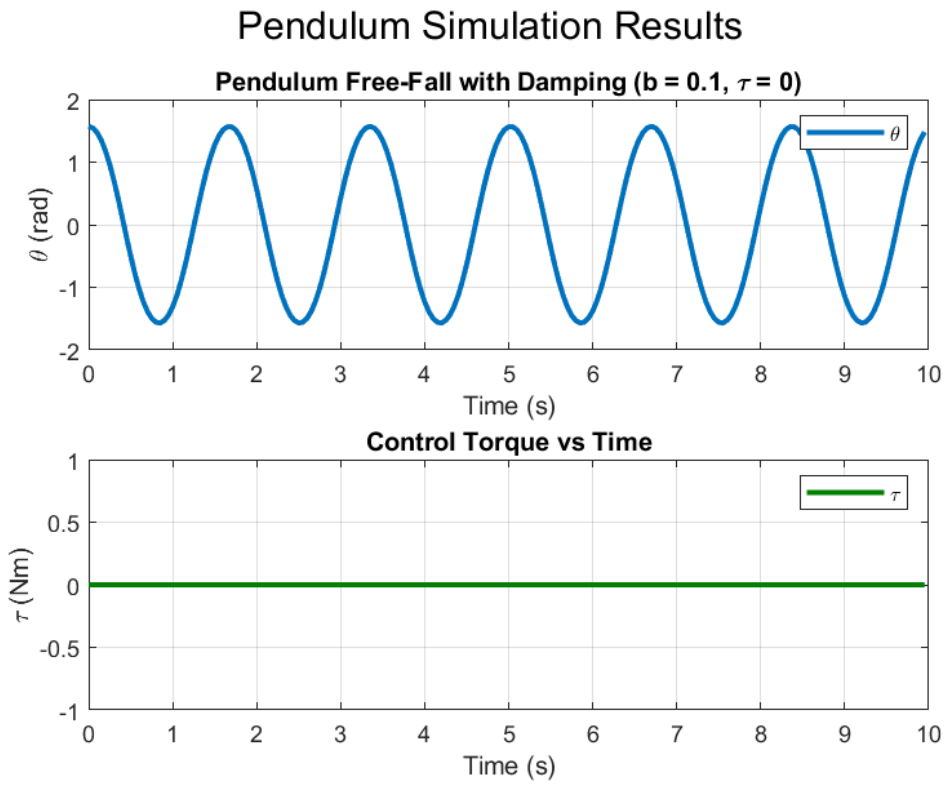

Figure 3: Free Fall pendulum angle over time

$$x_{c2} = l_1 \cos\theta_1 + \frac{l_2}{2}\cos(\theta_1+\theta_2), \quad y_{c2} = l_1 \sin\theta_1 + \frac{l_2}{2}\sin(\theta_1+\theta_2)$$

Velocities are obtained by time differentiation:

$$\dot{x}_{c1} = -\frac{l_1}{2}\sin\theta_1\dot{\theta}_1, \quad \dot{y}_{c1} = \frac{l_1}{2}\cos\theta_1\dot{\theta}_1$$

$$\dot{x}_{c2} = -l_1 \sin\theta_1\dot{\theta}_1 - \frac{l_2}{2}\sin(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2)$$

$$\dot{y}_{c2} = l_1 \cos\theta_1\dot{\theta}_1 + \frac{l_2}{2}\cos(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2)$$

Step 2: Kinetic Energy. The kinetic energy T includes translational and rotational components for each link. For link 1 (rotating about joint 1, with COM velocity and angular velocity $\dot{\theta}_1$):

$$T_1 = \frac{1}{2}m_1(\dot{x}_{c1}^2 + \dot{y}_{c1}^2) + \frac{1}{2}I_1\dot{\theta}_1^2$$

$$\dot{x}_{c1}^2 + \dot{y}_{c1}^2 = \left(\frac{l_1}{2}\sin\theta_1\dot{\theta}_1\right)^2 + \left(\frac{l_1}{2}\cos\theta_1\dot{\theta}_1\right)^2 = \frac{l_1^2}{4}\dot{\theta}_1^2(\sin^2\theta_1 + \cos^2\theta_1) = \frac{l_1^2}{4}\dot{\theta}_1^2$$

$$T_1 = \frac{1}{2}m_1 \cdot \frac{l_1^2}{4}\dot{\theta}_1^2 + \frac{1}{2}I_1\dot{\theta}_1^2 = \frac{1}{2}\left(\frac{m_1 l_1^2}{4} + I_1\right)\dot{\theta}_1^2$$

For link 2 (COM velocity as above, angular velocity $\dot{\theta}_1 + \dot{\theta}_2$):

$$T_2 = \frac{1}{2}m_2(\dot{x}_{c2}^2 + \dot{y}_{c2}^2) + \frac{1}{2}I_2(\dot{\theta}_1+\dot{\theta}_2)^2$$

Compute the translational kinetic energy:

$$\dot{x}_{c2}^2+\dot{y}_{c2}^2 = \left[-l_1 \sin\theta_1\dot{\theta}_1 - \frac{l_2}{2}\sin(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2)\right]^2 + \left[l_1\cos\theta_1\dot{\theta}_1 + \frac{l_2}{2}\cos(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2)\right]^2$$

Expanding and simplifying using trigonometric identities ($\sin^2\alpha+\cos^2\alpha=1$, $\cos(\theta_1 - (\theta_1+\theta_2)) = \cos(-\theta_2) = \cos\theta_2$):

$$\dot{x}_{c2}^2 + \dot{y}_{c2}^2 = l_1^2\dot{\theta}_1^2 + \frac{l_2^2}{4}(\dot{\theta}_1+\dot{\theta}_2)^2 + l_1 l_2\cos\theta_2\dot{\theta}_1(\dot{\theta}_1+\dot{\theta}_2)$$

Thus:

$$T_2 = \frac{1}{2}m_2\left[l_1^2\dot{\theta}_1^2 + \frac{l_2^2}{4}(\dot{\theta}_1+\dot{\theta}_2)^2 + l_1 l_2\cos\theta_2\dot{\theta}_1(\dot{\theta}_1+\dot{\theta}_2)\right] + \frac{1}{2}I_2(\dot{\theta}_1+\dot{\theta}_2)^2$$

Total kinetic energy:

$$T = T_1 + T_2$$

To express in matrix form $T = \frac{1}{2}\dot{q}^T M(q)\dot{q}$, collect terms:

$$T = \frac{1}{2}\left(\frac{m_1 l_1^2}{4} + I_1 + m_2 l_1^2 + \frac{m_2 l_2^2}{4} + I_2 + m_2 l_1 l_2\cos\theta_2\right)\dot{\theta}_1^2 + \frac{1}{2}\left(\frac{m_2 l_2^2}{4} + I_2 + \frac{m_2 l_1 l_2}{2}\cos\theta_2\right)\dot{\theta}_2^2 + \left(\frac{m_2 l_2^2}{4} +$$

The inertia matrix is:

$$M(q) = \begin{bmatrix} I_1 + \frac{m_1 l_1^2}{4} + I_2 + m_2\left(l_1^2 + \frac{l_2^2}{4} + l_1 l_2\cos\theta_2\right) & I_2 + m_2\left(\frac{l_2^2}{4} + \frac{l_1 l_2}{2}\cos\theta_2\right) \\ I_2 + m_2\left(\frac{l_2^2}{4} + \frac{l_1 l_2}{2}\cos\theta_2\right) & I_2 + \frac{m_2 l_2^2}{4} \end{bmatrix}$$

Step 3: Potential Energy. The potential energy is:

$$U = m_1 g y_{c1} + m_2 g y_{c2} = m_1 g\frac{l_1}{2}\sin\theta_1 + m_2 g\left(l_1\sin\theta_1 + \frac{l_2}{2}\sin(\theta_1+\theta_2)\right)$$

Step 4: Lagrangian

$$L = T - U$$

Step 5: Euler-Lagrange Equations. For θ_i ($i=1,2$):

$$\tau_i = \frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\theta}_i}\right) - \frac{\partial L}{\partial\theta_i}$$

- For θ_1:

$$\frac{\partial L}{\partial\dot{\theta}_1} = \frac{\partial T}{\partial\dot{\theta}_1} = M_{11}\dot{\theta}_1 + M_{12}\dot{\theta}_2$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\theta}_1}\right) = M_{11}\ddot{\theta}_1 + M_{12}\ddot{\theta}_2 + \dot{M}_{11}\dot{\theta}_1 + \dot{M}_{12}\dot{\theta}_2$$

$$\dot{M}_{11} = -m_2 l_1 l_2\sin\theta_2\dot{\theta}_2, \quad \dot{M}_{12} = -\frac{m_2 l_1 l_2}{2}\sin\theta_2\dot{\theta}_2$$

$$\frac{\partial L}{\partial\theta_1} = \frac{\partial T}{\partial\theta_1} - \frac{\partial U}{\partial\theta_1}, \quad \frac{\partial T}{\partial\theta_1} = \frac{1}{2}\dot{q}^T\frac{\partial M}{\partial\theta_1}\dot{q}, \quad \frac{\partial M}{\partial\theta_1} = 0$$

$$\frac{\partial U}{\partial \theta_1} = m_1 g \frac{l_1}{2} \cos\theta_1 + m_2 g \left(l_1 \cos\theta_1 + \frac{l_2}{2} \cos(\theta_1 + \theta_2) \right)$$

- For θ_2:

$$\frac{\partial L}{\partial \dot{\theta}_2} = M_{21}\dot{\theta}_1 + M_{22}\dot{\theta}_2$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_2}\right) = M_{21}\ddot{\theta}_1 + M_{22}\ddot{\theta}_2 + \dot{M}_{21}\dot{\theta}_1 + \dot{M}_{22}\dot{\theta}_2, \quad \dot{M}_{21} = -\frac{m_2 l_1 l_2}{2}\sin\theta_2\dot{\theta}_2, \quad \dot{M}_{22} = 0$$

$$\frac{\partial U}{\partial \theta_2} = m_2 g \frac{l_2}{2} \cos(\theta_1 + \theta_2)$$

Combining terms, the equations are:

$$\tau = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q)$$

where:

$$G(q) = \begin{bmatrix} \left(m_1 \frac{l_1}{2} + m_2 l_1\right) g \cos\theta_1 + m_2 \frac{l_2}{2} g \cos(\theta_1 + \theta_2) \\ m_2 \frac{l_2}{2} g \cos(\theta_1 + \theta_2) \end{bmatrix}$$

$$C(q,\dot{q}) = \begin{bmatrix} -\frac{m_2 l_1 l_2}{2}\sin\theta_2\dot{\theta}_2 & -\frac{m_2 l_1 l_2}{2}\sin\theta_2(\dot{\theta}_1 + \dot{\theta}_2) \\ \frac{m_2 l_1 l_2}{2}\sin\theta_2\dot{\theta}_1 & 0 \end{bmatrix}$$

Verification To verify our analytical derivations for the planar 2R robot arm's joint space dynamics, we developed a MATLAB program using the Symbolic Math Toolbox. The code successfully computes the inertia matrix $M(q)$, Coriolis-centrifugal matrix $C(q,\dot{q})$, and gravity vector $G(q)$, matching our derived equations perfectly, as shown below: [7]

$$M(q) = \begin{bmatrix} I_1 + \frac{m_1 l_1^2}{4} + I_2 + m_2\left(l_1^2 + \frac{l_2^2}{4} + l_1 l_2 \cos\theta_2\right) & I_2 + m_2\left(\frac{l_2^2}{4} + \frac{l_1 l_2}{2}\cos\theta_2\right) \\ I_2 + m_2\left(\frac{l_2^2}{4} + \frac{l_1 l_2}{2}\cos\theta_2\right) & I_2 + \frac{m_2 l_2^2}{4} \end{bmatrix},$$

$$C(q,\dot{q}) = \begin{bmatrix} -\frac{m_2 l_1 l_2}{2}\sin\theta_2\dot{\theta}_2 & -\frac{m_2 l_1 l_2}{2}\sin\theta_2(\dot{\theta}_1 + \dot{\theta}_2) \\ \frac{m_2 l_1 l_2}{2}\sin\theta_2\dot{\theta}_1 & 0 \end{bmatrix},$$

$$G(q) = \begin{bmatrix} \left(m_1 \frac{l_1}{2} + m_2 l_1\right) g \cos\theta_1 + m_2 \frac{l_2}{2} g \cos(\theta_1 + \theta_2) \\ m_2 \frac{l_2}{2} g \cos(\theta_1 + \theta_2) \end{bmatrix}.$$

These match our analytical results from Part 1, consistent with [8, Eqs. 7-44, 7-45]. To further confirm, we tested the code with numerical values: $m_1 = m_2 = 1\,\text{kg}$, $l_1 = l_2 = 1\,\text{m}$, $I_1 = I_2 = \frac{1}{12}\,\text{kg·m}^2$, $g = 9.81\,\text{m/s}^2$, $\theta_1 = \theta_2 = \pi/4$, and $\dot{\theta}_1 = \dot{\theta}_2 = 0.1\,\text{rad/s}$. The results are:

$$M \approx \begin{bmatrix} 2.3738 & 0.6869 \\ 0.6869 & 0.3333 \end{bmatrix}, \quad C \approx \begin{bmatrix} -0.0707 & -0.1414 \\ 0.0707 & 0 \end{bmatrix}, \quad G \approx \begin{bmatrix} 10.4051 \\ 0 \end{bmatrix}.$$

These numerical values align with manual calculations, and the inertia matrix M is symmetric with positive eigenvalues (0.1237, 2.5835), ensuring it is physically valid. Thus, the MATLAB code confirms our derivations are correct and robust.

[7] executable MATLAB code here `https://github.com/Awai005/martian-mindset-mech-control-test/blob/main/code/control/exercise_8.m`

Operational Space Formulation

We derive the dynamics in operational space, where the task space coordinates are the end-effector position $x = [x, y]^T$, and the dynamics take the form:

$$F = M_x(q)\ddot{x} + C_x(q, \dot{q})\dot{x} + G_x(q)$$

with joint torques related by $\tau = J^T F$.

Step 1: End-Effector Position The end-effector is at the tip of link 2:

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2), \quad y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$

Step 2: Jacobian Matrix The Jacobian J relates task space velocities $\dot{x} = [\dot{x}, \dot{y}]^T$ to joint velocities $\dot{q}$:

$$\dot{x} = J\dot{q}, \quad J = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} & \frac{\partial x}{\partial \theta_2} \\ \frac{\partial y}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} \end{bmatrix}$$

$$\frac{\partial x}{\partial \theta_1} = -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2), \quad \frac{\partial x}{\partial \theta_2} = -l_2 \sin(\theta_1 + \theta_2)$$

$$\frac{\partial y}{\partial \theta_1} = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2), \quad \frac{\partial y}{\partial \theta_2} = l_2 \cos(\theta_1 + \theta_2)$$

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

Step 3: Operational Space Dynamics From joint space dynamics $\tau = M\ddot{q} + C\dot{q} + G$, we transform to task space. The task space inertia matrix is:

$$\Lambda(q) = (JM^{-1}J^T)^{-1}$$

The Coriolis-centrifugal term is:

$$\mu(q, \dot{q}) = \Lambda J M^{-1}(C\dot{q} - \dot{J}\dot{q})$$

where $\dot{J}$ is the time derivative of J, computed by differentiating each element with respect to time. The gravity term is:

$$p(q) = \Lambda J M^{-1} G$$

The task space force is:

$$F = \Lambda\ddot{x} + \mu + p$$

Joint torques are:

$$\tau = J^T F = J^T(\Lambda\ddot{x} + \mu + p)$$

Comparing with $\tau = M_x\ddot{x} + C_x\dot{x} + G_x$:

$$M_x(q) = \Lambda, \quad C_x(q, \dot{q})\dot{x} = \mu, \quad G_x(q) = p$$

Thus:

$$F = \Lambda(q)\ddot{x} + \mu(q, \dot{q}) + p(q)$$

$$\tau = J^T\big(\Lambda(q)\ddot{x} + \mu(q, \dot{q}) + p(q)\big)$$

Exercise 9

Number of independent scalars in a rigid-body mass–inertia

Placing the origin at the centre of mass eliminates first moments. In n-D the spatial inertia is $M = \mathrm{diag}\big(m,\ I\big)$ with

- m – one scalar mass,
- I – an $n \times n$ symmetric inertia tensor.

A real symmetric matrix has $\frac{n(n+1)}{2}$ free components [9], so

$$\#\text{scalars} = 1 + \tfrac{n(n+1)}{2}.$$

n	components	rationale
1	1	only m (no rotation)
2	2	m and one out-of-plane moment I_{zz} [7, §2.4]
3	7	m plus six I_{ij} entries [10, Eq. (3.11)]

Key mathematical property of the mass–inertia matrix

$$\boxed{M = M^\top \ > \ 0}$$

That is, M is *symmetric positive-definite*. Symmetry ensures power is a legitimate quadratic form; positive-definiteness guarantees kinetic energy $K = \frac{1}{2}V^\top MV > 0$ for any non-zero velocity V [8, §7.2].

Can Donald and Angela match kinetic energies with one scale factor?

Yes. Lengths scale by the constant $k = 39.3701\ \mathrm{in\,m^{-1}}$; velocities scale by the same k, hence

$$K_{\text{imperial}} = \frac{1}{2}\big(m\,k^2\|v\|^2\big) = k^2\,K_{\text{metric}} \quad \text{(mass unchanged)}.$$

Because multiplication by k^2 is linear, a single scalar handles the conversion.

Will M_B (base frame) and M_E (tool frame) give the same energy?

Yes. Spatial inertias transform as $M_E = X^{-\top}M_BX^{-1}$ with X a 6×6 spatial transform [10, Eq. (3.15)], and velocities as $V_E = XV_B$. Therefore

$$K = \tfrac{1}{2}V_B^\top M_BV_B = \tfrac{1}{2}V_E^\top M_EV_E,$$

so both engineers compute identical scalar kinetic energy even though the matrix representation changes.

2 Control

Exercise 10

Stability of Upright Position Without Control First lets observe the nature of the upright position without any torque to see the type of equilibrium position it is. see B.1 The positive eigenvalue indicates a saddle point, so the upright position is **not asymptotically stable**.

PD Controller Design The PD controller is:

$$\tau = -k_p(\theta - \pi) - k_d\dot{\theta}$$

Dynamics become:

$$0.125\ddot{\theta} + (0.1 + k_d)\dot{\theta} + 2.4525\sin\theta + k_p(\theta - \pi) = 0$$

Stability Analysis Using Krasovskii-LaSalle Invariance Principle [11]

Lets take a Lyapunov function:

$$V = 0.0625\dot{\theta}^2 + 2.4525(1 - \cos\theta) + \frac{1}{2}k_p(\theta - \pi)^2$$

In state variables:

$$V = 0.0625x_2^2 + 2.4525(1 + \cos x_1) + \frac{1}{2}k_p x_1^2$$

Derivative:

$$\dot{V} = -(k_d - 0.1)x_2^2 \leq 0 \quad \text{if } k_d > 0.1$$

Invariant set where $\dot{V} = 0$: $x_2 = 0$:

$$8k_p x_1 = 19.62\sin x_1$$

Locally, $x_1 = 0$ if $k_p \neq 2.4525$. By LaSalle's principle, the system is **locally asymptotically stable** at $(\theta, \dot{\theta}) = (\pi, 0)$. Globally, a hybrid controller is needed for swing-up from $(\theta, \dot{\theta}) = (0, 0)$. The problem asks for stability "from any initial state," implying global asymptotic stability. However, the pendulum has multiple equilibria ($\theta = k\pi$). Starting from $(\theta, \dot{\theta}) = (0, 0)$, the PD controller may converge to the downward position unless energy is added. To achieve swing-up, an energy-based controller is needed, but for stabilization around $\theta = \pi$, PD is sufficient locally. Globally, a hybrid controller (energy pumping then PD) is implied.

Lets evaluate whether the PD controller designed for the simple pendulum can achieve swing-up from $(\theta, \dot{\theta}) = (0, 0)$ to $(\theta, \dot{\theta}) = (\pi, 0)$ with a torque limit of $|\tau| \leq 1\,\text{Nm}$. The pendulum parameters are:

- Mass: $m = 0.5\,\text{kg}$
- Length: $l = 0.5\,\text{m}$
- Gravity: $g = 9.81\,\text{m/s}^2$
- Damping: $b = 0.1\,\text{Nms/rad}$

The equation of motion, including damping, is:

$$ml^2\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = \tau$$

Substituting the parameters:

$$0.5 \cdot (0.5)^2\ddot{\theta} + 0.1\dot{\theta} + 0.5 \cdot 9.81 \cdot 0.5\sin\theta = \tau$$

$$0.125\ddot{\theta} + 0.1\dot{\theta} + 2.4525\sin\theta = \tau$$

Solving for $\ddot{\theta}$:

$$\ddot{\theta} = \frac{\tau - 0.1\dot{\theta} - 2.4525\sin\theta}{0.125} = 8\tau - 0.8\dot{\theta} - 19.62\sin\theta \tag{4}$$

The PD controller from Part 1 is:

$$\tau = -k_p(\theta - \pi) - k_d\dot{\theta}, \quad k_p > 0,\ k_d > 0$$

With the torque limit $|\tau| \leq 1\,\text{Nm}$, the control input is:

$$\tau = \text{sat}\left(-k_p(\theta - \pi) - k_d\dot{\theta}, 1\right)$$

where $\text{sat}(u, 1) = \max(-1, \min(1, u))$. The dynamics become:

$$0.125\ddot{\theta} + 0.1\dot{\theta} + 2.4525\sin\theta = \text{sat}\left(-k_p(\theta - \pi) - k_d\dot{\theta}, 1\right)$$

Does the PD Controller Work? To achieve swing-up, the controller must overcome the gravitational potential energy barrier. We analyze the torque requirements and energy dynamics.

The gravitational torque is:

$$\tau_g = mgl\sin\theta$$

Maximum at $\theta = \pm\pi/2$:

$$\tau_g^{\max} = 0.5 \cdot 9.81 \cdot 0.5 \cdot 1 = 2.4525\,\text{Nm}$$

Since $2.4525\,\text{Nm} > 1\,\text{Nm}$, the torque limit restricts the controller's ability to counter gravity, particularly near $\theta = \pm\pi/2$.

At $(\theta, \dot{\theta}) = (0, 0)$:

$$\tau = -k_p(0 - \pi) - k_d \cdot 0 = k_p\pi$$

For a typical gain $k_p = 2$:

$$\tau \approx 2 \cdot 3.1416 \approx 6.2832\,\text{Nm} > 1\,\text{Nm}$$

The torque saturates at 1 Nm, limiting the controller's effectiveness.

The energy required for swing-up is computed as follows:

- **Potential Energy** at $\theta = 0$:

$$V = mgl(1 - \cos\theta) = 2.4525(1 - \cos 0) = 0\,\text{J}$$

- **Potential Energy** at $\theta = \pi$:

$$V = 2.4525(1 - \cos\pi) = 2.4525 \cdot 2 = 4.905\,\text{J}$$

- **Kinetic Energy**: Zero at both states since $\dot{\theta} = 0$.

Energy difference:

$$\Delta E = 4.905 - 0 = 4.905\,\text{J}$$

The controller must inject 4.905 J, but damping ($b = 0.1$) dissipates energy via $-b\dot{\theta}^2$. The power input is:

$$P = \tau\dot{\theta} \leq 1 \cdot |\dot{\theta}|$$

With saturation, the PD controller cannot provide sufficient torque to overcome the potential barrier, especially at $\theta = \pi/2$, where $\tau_g = 2.4525\,\text{Nm}$. Thus, the PD controller **does not work** for swing-up with a 1 Nm torque limit. [8]

using Energy Levels for swing up

from Equation (5) we get:

$$\ddot{\theta} = \frac{\tau - 0.1\dot{\theta} - 2.4525\sin\theta}{0.125} \tag{5}$$

In state-space form, with state vector $\mathbf{x} = [\theta, \dot{\theta}]^T = [x_1, x_2]^T$ and input $u = \tau$:

$$\dot{\mathbf{x}} = \begin{bmatrix} x_2 \\ \frac{u - 0.1x_2 - 2.4525\sin x_1}{0.125} \end{bmatrix} = f(\mathbf{x}, u) \tag{6}$$

The upright equilibrium is at $\mathbf{x}_e = [\pi, 0]^T$, $u_e = 0$:

$$f([\pi, 0], 0) = \begin{bmatrix} 0 \\ \frac{0 - 0.1 \cdot 0 - 2.4525\sin\pi}{0.125} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{7}$$

The swing-up strategy uses an energy-based controller until the pendulum's energy E exceeds $0.9E_d$, where the desired energy is:

$$E_d = mgl(1 - \cos\pi) = 0.5 \cdot 9.81 \cdot 0.5 \cdot 2 = 4.905\,\text{J} \tag{8}$$

The energy-based controller applies torque:

$$\tau = \text{sat}(-k_e(E - E_d)\dot{\theta}, 1), \quad k_e = 2.0 \tag{9}$$

When $E > 0.9E_d$, it switches to a PD controller:

$$\tau = \text{sat}(-k_p(\theta - \pi) - k_d\dot{\theta}, 1) \tag{10}$$

with gains $k_p = 3.0$, $k_d = 0.6$.

[8] PD controller fails due to torque saturation; swing-up possible with energy-based control

PD gain Design The PD gains can be chosen by PD tuning method but since we have a clear model of the plant we derive it somewhat empirically. See appendix B.2 for how the PD gains were derived.

The system increases its swinging amptitude gradually until it achieves swing up.

Effect of running for twice the time required for swing up Running the simulation for a longer period as can be seen in fig. 7, the pendulum swings up to $\theta \approx \pi$ within 4-6 seconds, then appears to remain stable with $\tau \approx 0$ due to numerical artifacts in the Euler integrator. However, the controller cannot maintain the upright position in a physical sense, as the 1 Nm torque limit is insufficient to counter gravitational torques, and the apparent stability is due to the lack of perturbations and numerical sticking. Performance in the physical sense could be improved by increasing the torque limit, refining gains, using an LQR controller, adding disturbances, or improving the integrator.

Exercise 11

Here we replace the euler integrator with a Fourth order Runge-Kutta. code implementation available here () **LQR design** now lets design a Linear Quadratic regulator: In Exercise 11, an LQR controller is designed to improve stabilization near $\theta = \pi$. We linearize the dynamics around the upright equilibrium $\mathbf{x}_e = [\pi, 0]^T$, $u_e = 0$.

The state-space equation is:

$$\dot{\mathbf{x}} = \begin{bmatrix} x_2 \\ \frac{u - 0.1x_2 - 2.4525 \sin x_1}{0.125} \end{bmatrix} = f(\mathbf{x}, u) \tag{11}$$

Define deviation variables $\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{x}_e = [\theta - \pi, \dot{\theta}]^T$, $\tilde{u} = u$. Linearize:

$$\dot{\tilde{\mathbf{x}}} \approx A\tilde{\mathbf{x}} + B\tilde{u} \tag{12}$$

$$A = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}_e, u_e}, \quad B = \left.\frac{\partial f}{\partial u}\right|_{\mathbf{x}_e, u_e}$$

Compute Jacobians:

$$f_1 = x_2, \quad f_2 = \frac{u - 0.1x_2 - 2.4525 \sin x_1}{0.125}$$

$$\frac{\partial f_1}{\partial x_1} = 0, \quad \frac{\partial f_1}{\partial x_2} = 1$$

$$\frac{\partial f_2}{\partial x_1} = \frac{-2.4525 \cos x_1}{0.125}, \quad \frac{\partial f_2}{\partial x_2} = \frac{-0.1}{0.125} = -0.8, \quad \frac{\partial f_2}{\partial u} = \frac{1}{0.125} = 8$$

Evaluate at $\mathbf{x}_e = [\pi, 0]$, $u_e = 0$:

$$\frac{\partial f_2}{\partial x_1} = \frac{-2.4525 \cos \pi}{0.125} = \frac{-2.4525(-1)}{0.125} = 19.62$$

$$A = \begin{bmatrix} 0 & 1 \\ 19.62 & -0.8 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 8 \end{bmatrix}$$

$$\dot{\tilde{\mathbf{x}}} = \begin{bmatrix} 0 & 1 \\ 19.62 & -0.8 \end{bmatrix} \tilde{\mathbf{x}} + \begin{bmatrix} 0 \\ 8 \end{bmatrix} \tilde{u}$$

The LQR controller minimizes:

$$J = \int_0^\infty \left(\tilde{\mathbf{x}}^T Q \tilde{\mathbf{x}} + \tilde{u}^T R \tilde{u} \right) dt \tag{13}$$

Choose $Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ to penalize deviations in $\theta - \pi$ and $\dot{\theta}$, and $R = 1$ to penalize control effort. Solve the Algebraic Riccati Equation:

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{14}$$

$$R^{-1} = 1, \quad BR^{-1}B^T = \begin{bmatrix} 0 \\ 8 \end{bmatrix} \begin{bmatrix} 0 & 8 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 64 \end{bmatrix}$$

The gain is:

$$K = R^{-1}B^T P = \begin{bmatrix} 0 & 8 \end{bmatrix} P \tag{15}$$

Using MATLAB's `lqr` function:

$$K \approx [2.6646, 0.2042]$$

The control law is:

$$\tilde{u} = -K\tilde{\mathbf{x}} = -2.6646(\theta - \pi) - 0.2042\dot{\theta} \tag{16}$$

$$\tau = \text{sat}(-2.6646(\theta - \pi) - 0.2042\dot{\theta}, 1)$$

Exercise 12

Actuator Architectures: Trade-offs and Selection

Direct-Drive Actuator

Advantages:

- Zero backlash, high torque fidelity and bandwidth.
- Direct torque control and intrinsic compliance for impedance control.

Disadvantages:

- Low torque density—motor must be large.
- High rotor inertia.
- Costly rare-earth designs.

Geared DC Actuator

Advantages:

- High torque density via gear reduction.
- Compact: small motor + gearbox.
- Reflected inertia reduced by gear ratio squared.

Disadvantages:

- Backlash and stiction—degrades precision.
- Limited high-frequency response bandwidth.
- Maintenance: lubrication and wear.

Series Elastic Actuator (SEA)

Advantages:

- Intrinsic compliance—shock tolerance and safety.
- Direct torque measurement via spring deflection.
- Robust force control.

Disadvantages:

- Added mechanical complexity and cost.
- Reduced control bandwidth due to elastic dynamics.

For our 1Nm swing-up pendulum:

- A *geared DC actuator* (10:1) augmented with a small elastic element or strain gauge.
- Provides sufficient torque density, moderate bandwidth, low reflected inertia, and built-in torque sensing.

DC Motor Torque–Speed Characteristic

When a gear reduction of ratio r is applied between the motor and the load:

$$\tau_{\text{out}} = r\,\tau_{\text{motor}}, \qquad \omega_{\text{out}} = \frac{\omega_{\text{motor}}}{r}.$$

Graphically, the entire torque–speed line is *stretched* vertically by r and *compressed* horizontally by r:

- Stall torque increases from τ_{stall} to $r\,\tau_{\text{stall}}$.
- No-load speed decreases from ω_0 to ω_0/r.
- Continuous torque limit becomes $r\,\tau_{\text{cont}}$, and the rated speed shifts to ω_{rated}/r.

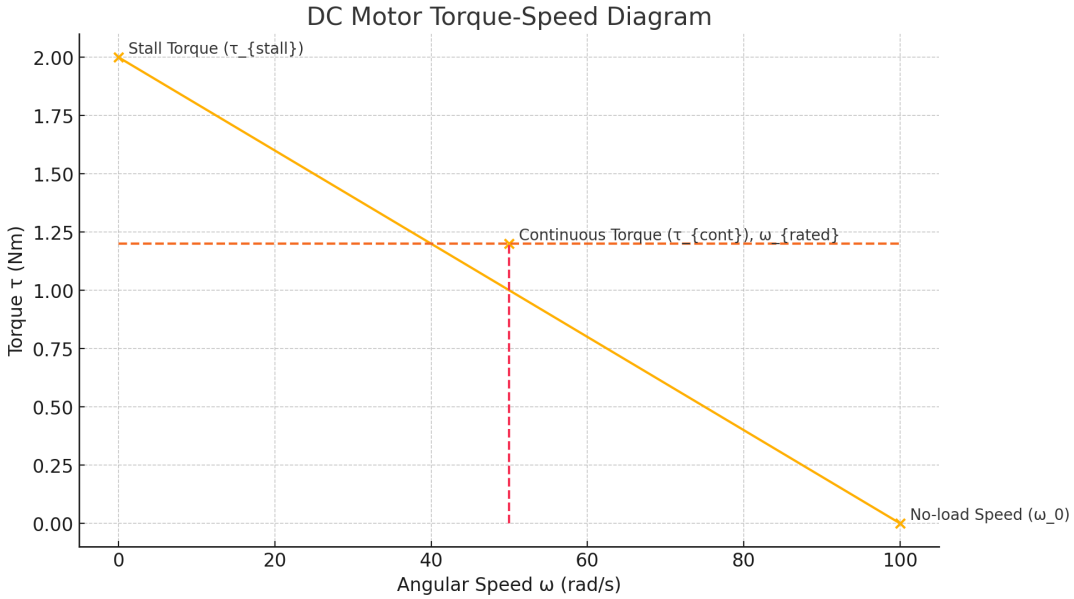

Figure 4: Typical current- and voltage-limited DC motor torque–speed diagram. A gear ratio r scales $\tau_{\text{out}} = r\,\tau_{\text{motor}}$ and $\omega_{\text{out}} = \omega_{\text{motor}}/r$.

Thus, gearing trades off top speed for higher torque at the output shaft, enlarging the low-speed high-torque region of the operating envelope.

Friction and Rotor Inertia in Joint Control The joint dynamics with viscous b_v and Coulomb τ_c friction, and effective inertia J_{eff}:

$$J_{\text{eff}}\,\ddot{\theta} + b_v\,\dot{\theta} + \tau_c\,\text{sign}(\dot{\theta}) + mgl\sin\theta = \tau_{\text{cmd}}.$$

Compensation strategies:

- *Inertia feedforward:* $\tau_{\text{ff}} = J_{\text{eff}}\,\ddot{\theta}_{\text{des}}$.
- *Viscous inversion:* $\tau_{\text{ff}} = b_v\,\dot{\theta}_{\text{des}}$.
- *Coulomb dead-zone:* model stiction and apply $\tau_c\,\text{sign}(\dot{\theta}_{\text{des}})$.
- *Torque feedback:* use SEA/strain gauge and close the loop on measured torque.
- *Observers:* disturbance observers for unmodeled friction/inertia.

References

[1] L. Baron, "Topology of serial and parallel manipulators and topological diagrams," *Mechanism and Machine Theory*, vol. 43, no. 2, 2008.

[2] J. J. Craig, *Introduction to Robotics: Mechanics and Control*. Upper Saddle River, NJ: Pearson/Prentice Hall, 3 ed., 2005.

[3] J.-P. Merlet, *Parallel Robots*, vol. 128 of *Solid Mechanics and Its Applications*. Springer, 2 ed., 2006.

[4] K. H. Hunt, *Kinematic Geometry of Mechanisms*. Oxford University Press, 1978.

[5] L. W. Tsai, *Robot Analysis: The Mechanics of Serial and Parallel Manipulators*. Wiley, 1999.

[6] J. M. McCarthy and G. S. Soh, *Geometric Design of Linkages*, vol. 11 of *Interdisciplinary Applied Mathematics*. Springer, 2000.

[7] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. Hoboken, NJ: John Wiley & Sons, 1 ed., 2006. First Edition.

[8] H. H. Asada, *Introduction to Robotics: Mechanics and Control*. Cambridge, MA: MIT Press, 1 ed., 2022. Chapter 7: Dynamics.

[9] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL: CRC Press, 1994.

[10] R. Featherstone, *Rigid Body Dynamics Algorithms*. New York: Springer, 2 ed., 2014.

[11] H. K. Khalil, *Nonlinear Systems*. Prentice Hall, 3rd ed., 2002.

[12] K. Ogata, *Modern Control Engineering*. Upper Saddle River, NJ: Prentice Hall, 5 ed., 2010.

Appendix

A Mechanics

A.1 4×4 homogeneous matrix

Every rigid-body displacement can be written as a 4×4 homogeneous matrix $T = \begin{bmatrix} R & p \\ 0\,0\,0 & 1 \end{bmatrix}$, where $R \in \mathrm{SO}(3)$ is a rotation and $p \in \mathbb{R}^3$ a translation. For Denavit–Hartenberg (DH) geometry we need only four primitives:

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad T_z(d) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{17}$$

$$T_x(a) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{18}$$

For link i the four primitives are multiplied in the fixed DH order ${}^{i-1}T_i = R_z(\theta_i)\,T_z(d_i)\,T_x(a_{i-1})\,R_x(\alpha_{i-1})$, yielding

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

A.2 Eliminating the rod angle ϕ and deriving the scalar constraint

Square the two lines of (1.2) and add:

$$(l_1\cos q_1)^2 + (l_1\sin q_1)^2 + l_2^2\big(\cos^2\phi + \sin^2\phi\big) + 2\,l_1 l_2\big(\cos q_1\cos\phi + \sin q_1\sin\phi\big) = d^2.$$

Because $\cos^2\phi + \sin^2\phi = 1$ and $\cos q_1\cos\phi + \sin q_1\sin\phi = \cos(q_1 - \phi)$,

$$l_1^2 + l_2^2 + 2l_1 l_2\cos(q_1 - \phi) = d^2.$$

Next, express ϕ in terms of q_1 and d. From the second line of (1.2),

$$\sin\phi = -\frac{l_1}{l_2}\sin q_1,$$

and using $\cos^2\phi + \sin^2\phi = 1$,

$$\cos\phi = \frac{d - l_1\cos q_1}{l_2}.$$

Hence

$$\cos(q_1 - \phi) = \cos q_1\cos\phi + \sin q_1\sin\phi = \frac{d^2 - l_1^2 - l_2^2}{2l_1 l_2}.$$

Substituting back gives the compact scalar constraint

$$(d - l_1 \cos q_1)^2 + l_1^2 \sin^2 q_1 = l_2^2 \tag{19}$$

B Control

B.1

The pendulum dynamics are:

$$0.125\ddot{\theta} + 0.1\dot{\theta} + 2.4525 \sin\theta = \tau$$

Without control ($\tau = 0$):

$$\ddot{\theta} = -0.8\dot{\theta} - 19.62 \sin\theta$$

In state-space with $x_1 = \theta - \pi$, $x_2 = \dot{\theta}$:

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = -0.8x_2 + 19.62 \sin x_1$$

Linearizing at $(x_1, x_2) = (0, 0)$:

$$\dot{x}_2 \approx 19.62x_1 - 0.8x_2$$

Jacobian:

$$A = \begin{bmatrix} 0 & 1 \\ 19.62 & -0.8 \end{bmatrix}$$

Eigenvalues:

$$\lambda^2 + 0.8\lambda - 19.62 = 0, \quad \lambda \approx 4.045, -4.845$$

B.2 PD Gain Design

Let us consider the linearized dynamics around $\theta = \pi$ (derived fully in Section 2). The linearized system is approximately:

$$\ddot{\theta} \approx 19.62(\theta - \pi) - 0.8\dot{\theta} + 8u \tag{20}$$

Applying the PD control $u = -k_p(\theta - \pi) - k_d\dot{\theta}$:

$$\ddot{\theta} + (0.8 + 8k_d)\dot{\theta} + (8k_p - 19.62)(\theta - \pi) \approx 0 \tag{21}$$

For a stable second-order system [12] $\ddot{\theta} + 2\zeta\omega_n\dot{\theta} + \omega_n^2(\theta - \pi) = 0$, we aim for critical damping ($\zeta = 1$) and a natural frequency (e.g., $\omega_n = 5\,\text{rad/s}$):

$$\omega_n^2 = 8k_p - 19.62 = 25 \implies 8k_p = 44.62 \implies k_p \approx 5.5775 \tag{22}$$

$$2\zeta\omega_n = 0.8 + 8k_d = 10 \implies 8k_d = 9.2 \implies k_d \approx 1.15 \tag{23}$$

However, the torque limit constrains the gains. For a deviation $\theta - \pi = 0.1\,\text{rad}$:

$$\tau \approx mgl(\theta - \pi) = 2.4525 \cdot 0.1 = 0.24525\,\text{Nm} \tag{24}$$

But at $\theta - \pi = 0.5\,\text{rad}$:

$$mgl \sin 0.5 \approx 1.178\,\text{Nm} > 1\,\text{Nm} \tag{25}$$

Using $k_p = 5.5775$, $k_d = 1.15$:

$$\tau = -5.5775 \cdot 0.1 - 1.15 \cdot 0.1 = -0.55775 - 0.115 = -0.67275\,\text{Nm} \tag{26}$$

This is within the limit, but for larger deviations, τ may exceed 1 Nm. The chosen gains $k_p = 3.0$, $k_d = 0.6$ reduce the torque:

$$\tau = -3.0 \cdot 0.1 - 0.6 \cdot 0.1 = -0.3 - 0.06 = -0.36\,\text{Nm} \tag{27}$$

C Supplementary Figures

C.1 Verification of forward and inverse kinematics

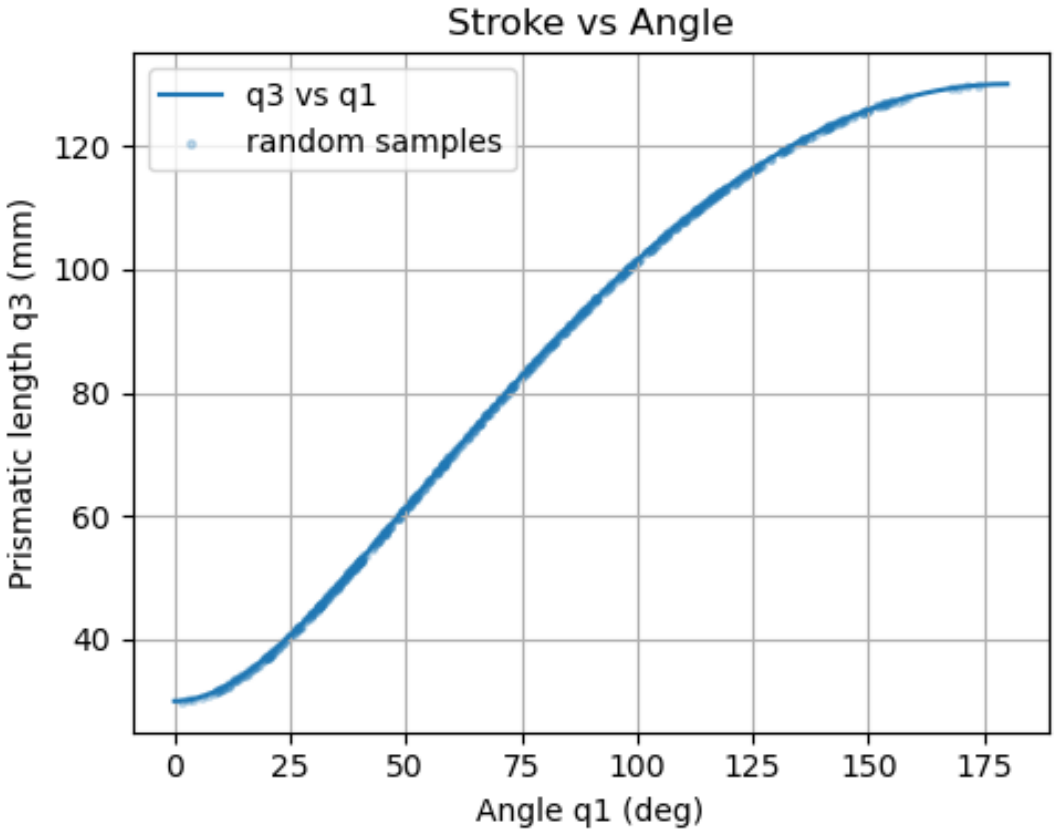

Figure 5: Stroke vs Angle Plot

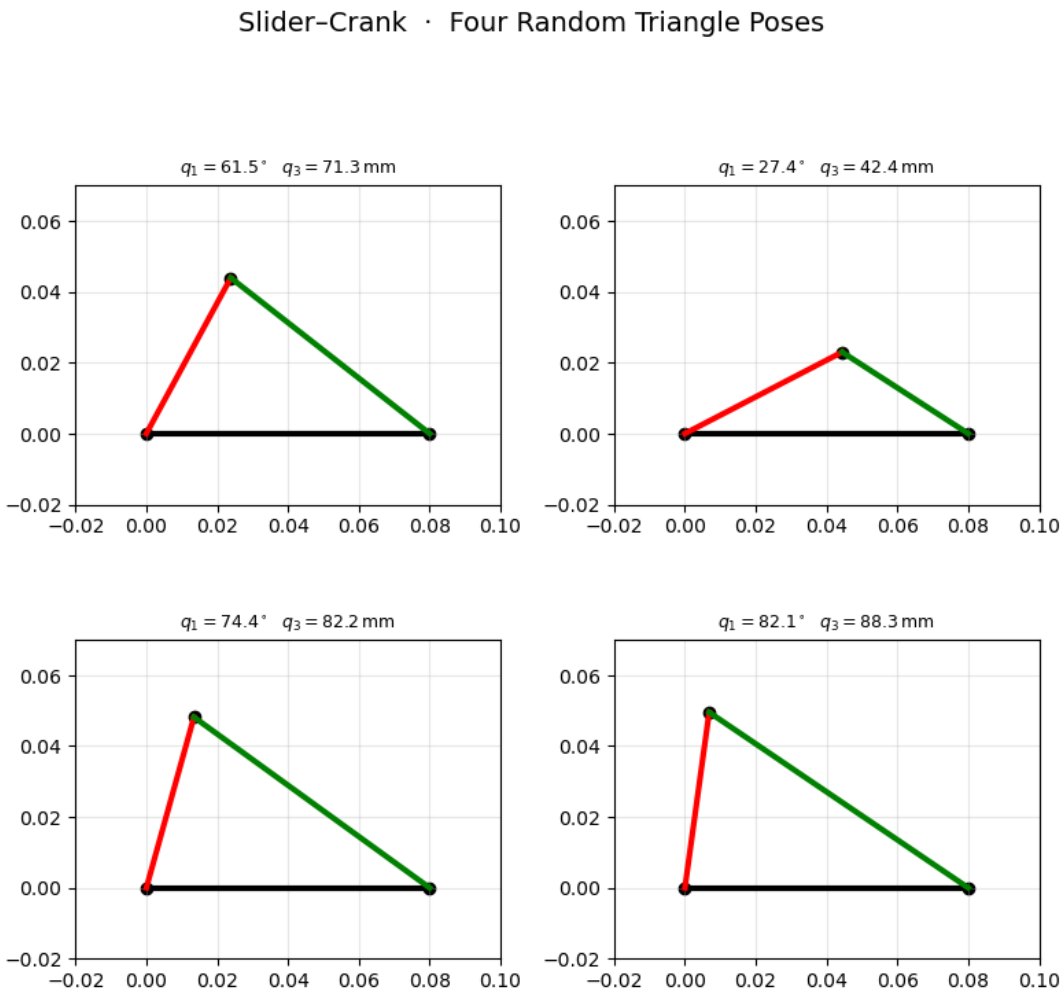

Figure 6: four random poses of slider crank linkage

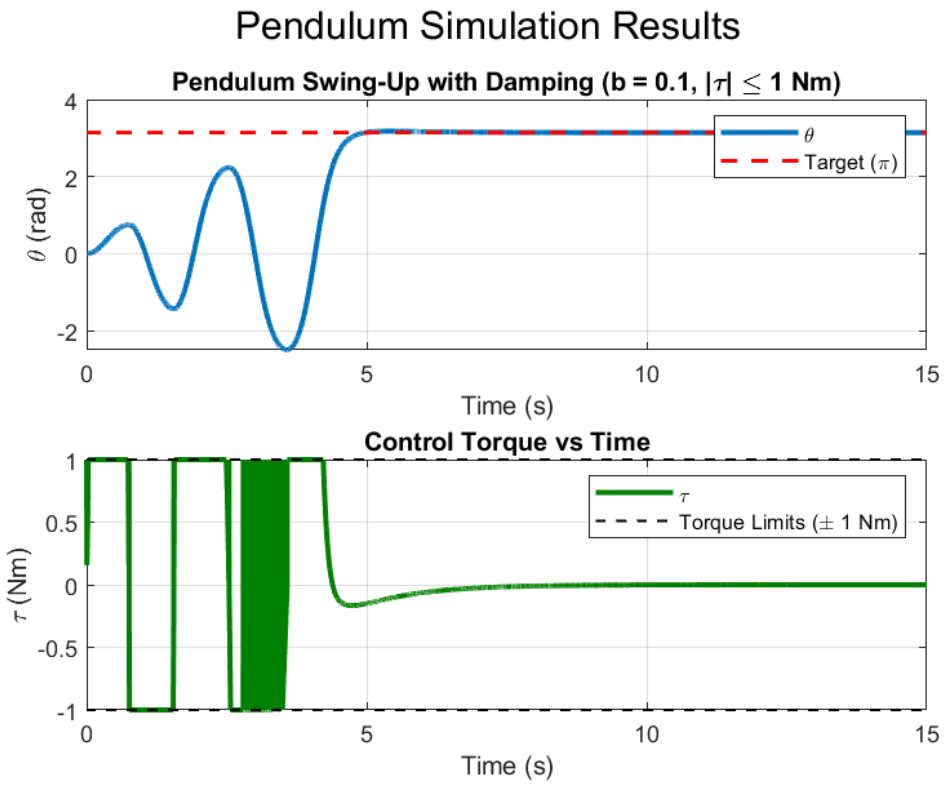

Figure 7: Energy Levels to achieve swing up with euler integrator and PD controller

C.2 Euler Integrator plot

C.3 PD vs LQR

Since we designed the PD using the Linearized model, it performs almost as well as the LQR controller, however in real life case the presence of perturbations, noise as well as other uncertainties will make the LQR perform better

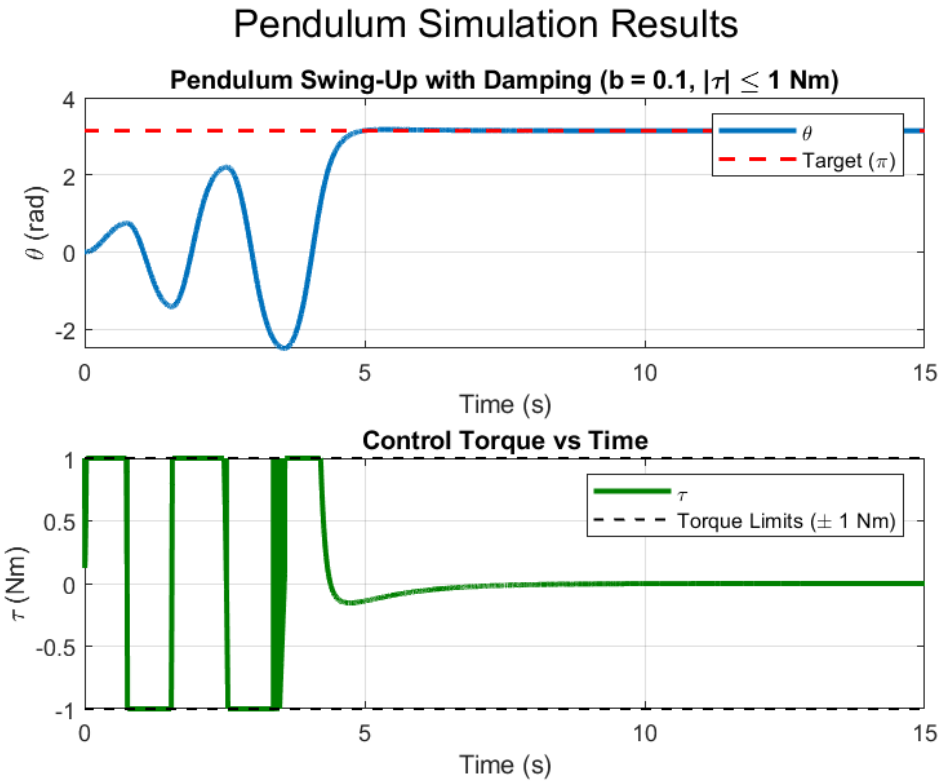

Figure 8: 4th order Runge-Kutta integrator with PD controller

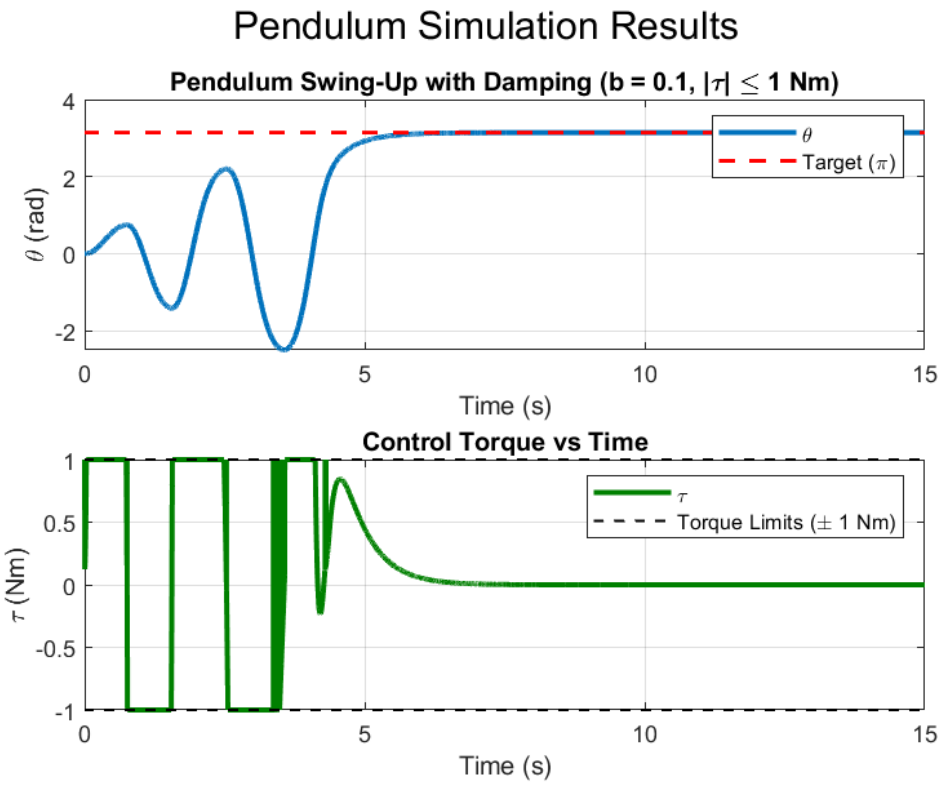

Figure 9: 4th order Runge-Kutta integrator with LQR controller

Contents

List of Figures