Dynamics of a 2-DOF Planar Robot Arm

August 29, 2025

1 Project Overview

We derive the equations of motion for a 2-degree-of-freedom (2-DOF) planar robot arm using Lagrangian mechanics. The goal is to:

- Model the arm as two rigid links with point masses at the distal ends (neglecting moment of inertia, MOI)
- Compute total kinetic (KE) and potential energy (PE)
- Derive the Lagrangian $\mathcal{L} = T - V$
- Obtain the nonlinear ODEs governing the arm's dynamics

2 Robot Arm Description

- **Joints**: Revolute joints at shoulder (θ_1) and elbow (θ_2)
- **Links**: Massless rods of lengths L_1, L_2 with point masses m_1, m_2 at endpoints
- **Assumptions**:
  - Motion is constrained to the xy-plane
  - MOI is negligible (simplified model)
  - Gravity g acts downward

3 Why MOI is Neglected

We ignore moment of inertia because:

- The masses are modeled as *point masses* at the ends of massless links
- Rotational KE about the center of mass is zero (no mass distribution along links)
- This simplification is valid when:
  - Link masses are concentrated at endpoints
  - The arm moves slowly (minimal rotational effects)

4 Kinetic Energy (KE) Calculation

4.1 Link 1 (Shoulder to Elbow)

Position of m_1:

$$\mathbf{r}_1 = \begin{bmatrix} L_1 \cos\theta_1 \\ L_1 \sin\theta_1 \end{bmatrix}$$

Velocity:

$$\dot{\mathbf{r}}_1 = \begin{bmatrix} -L_1\dot{\theta}_1 \sin\theta_1 \\ L_1\dot{\theta}_1 \cos\theta_1 \end{bmatrix}$$

KE:

$$T_1 = \frac{1}{2} m_1 \|\dot{\mathbf{r}}_1\|^2 = \frac{1}{2} m_1 L_1^2 \dot{\theta}_1^2$$

4.2 Link 2 (Elbow to End-Effector)

Position of m_2:

$$\mathbf{r}_2 = \begin{bmatrix} L_1 \cos\theta_1 + L_2 \cos(\theta_1+\theta_2) \\ L_1 \sin\theta_1 + L_2 \sin(\theta_1+\theta_2) \end{bmatrix}$$

Velocity:

$$\dot{\mathbf{r}}_2 = \begin{bmatrix} -L_1\dot\theta_1 \sin\theta_1 - L_2(\dot\theta_1+\dot\theta_2)\sin(\theta_1+\theta_2) \\ L_1\dot\theta_1 \cos\theta_1 + L_2(\dot\theta_1+\dot\theta_2)\cos(\theta_1+\theta_2) \end{bmatrix}$$

KE:

$$T_2 = \frac{1}{2}m_2\|\dot{\mathbf{r}}_2\|^2 = \frac{1}{2}m_2\left[L_1^2\dot\theta_1^2 + L_2^2(\dot\theta_1+\dot\theta_2)^2 + 2L_1L_2\dot\theta_1(\dot\theta_1+\dot\theta_2)\cos\theta_2\right]$$

4.3 Total KE

$$T = T_1 + T_2 = \frac{1}{2}(m_1+m_2)L_1^2\dot\theta_1^2 + \frac{1}{2}m_2L_2^2(\dot\theta_1+\dot\theta_2)^2 + m_2L_1L_2\dot\theta_1(\dot\theta_1+\dot\theta_2)\cos\theta_2$$

5 Potential Energy (PE) Calculation

5.1 Link 1

$$V_1 = m_1 g y_1 = m_1 g L_1 \sin\theta_1$$

5.2 Link 2

$$V_2 = m_2 g y_2 = m_2 g\left(L_1\sin\theta_1 + L_2\sin(\theta_1+\theta_2)\right)$$

5.3 Total PE

$$V = V_1 + V_2 = (m_1+m_2)gL_1\sin\theta_1 + m_2gL_2\sin(\theta_1+\theta_2)$$

6 Lagrangian

$$\mathcal{L} = T - V = \frac{1}{2}(m_1+m_2)L_1^2\dot\theta_1^2 + \frac{1}{2}m_2L_2^2(\dot\theta_1+\dot\theta_2)^2 + m_2L_1L_2\dot\theta_1(\dot\theta_1+\dot\theta_2)\cos\theta_2 - (m_1+m_2)gL_1\sin\theta_1 - m_2gL_2\sin(\theta_1+\theta_2)$$

7 Equations of Motion

Apply the Euler-Lagrange equations for each generalized coordinate θ_i:

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot\theta_i}\right) - \frac{\partial\mathcal{L}}{\partial\theta_i} = \tau_i \quad \text{(where } \tau_i \text{ is the applied torque)}$$

7.1 For θ_1

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot\theta_1}\right) = (m_1+m_2)L_1^2\ddot\theta_1 + m_2L_2^2(\ddot\theta_1+\ddot\theta_2) + m_2L_1L_2(2\ddot\theta_1+\ddot\theta_2)\cos\theta_2 - m_2L_1L_2(2\dot\theta_1+\dot\theta_2)\dot\theta_2\sin\theta_2$$

$$\frac{\partial\mathcal{L}}{\partial\theta_1} = -(m_1+m_2)gL_1\cos\theta_1 - m_2gL_2\cos(\theta_1+\theta_2)$$

7.2 For θ_2

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot\theta_2}\right) = m_2L_2^2(\ddot\theta_1+\ddot\theta_2) + m_2L_1L_2\ddot\theta_1\cos\theta_2 - m_2L_1L_2\dot\theta_1\dot\theta_2\sin\theta_2$$

$$\frac{\partial\mathcal{L}}{\partial\theta_2} = -m_2L_1L_2\dot\theta_1(\dot\theta_1+\dot\theta_2)\sin\theta_2 - m_2gL_2\cos(\theta_1+\theta_2)$$

7.3 Final Nonlinear ODEs

The complete dynamics are described by:

$$\begin{aligned}\tau_1 = &\left[(m_1 + m_2)L_1^2 + m_2 L_2^2 + 2m_2 L_1 L_2 \cos\theta_2\right]\ddot{\theta}_1\\ &+ \left[m_2 L_2^2 + m_2 L_1 L_2 \cos\theta_2\right]\ddot{\theta}_2\\ &- m_2 L_1 L_2 (2\dot{\theta}_1 + \dot{\theta}_2)\dot{\theta}_2 \sin\theta_2\\ &+ (m_1 + m_2) g L_1 \cos\theta_1 + m_2 g L_2 \cos(\theta_1 + \theta_2)\end{aligned}$$

$$\begin{aligned}\tau_2 = &\left[m_2 L_2^2 + m_2 L_1 L_2 \cos\theta_2\right]\ddot{\theta}_1\\ &+ m_2 L_2^2 \ddot{\theta}_2\\ &+ m_2 L_1 L_2 \dot{\theta}_1^2 \sin\theta_2\\ &+ m_2 g L_2 \cos(\theta_1 + \theta_2)\end{aligned}$$

8 Constraints in the System

8.1 Holonomic Constraints

The 2-DOF arm has two **holonomic constraints**:

- **Rigid Link Constraint 1**: $x_1^2 + y_1^2 = L_1^2$ (fixes length of first link)
- **Rigid Link Constraint 2**: $(x_2 - x_1)^2 + (y_2 - y_1)^2 = L_2^2$ (fixes length of second link)

These are holonomic because:

- They depend only on coordinates (x_i, y_i) *not* velocities
- They reduce the system from 4 variables (x_1, y_1, x_2, y_2) to 2 DOF (θ_1, θ_2)

8.2 Why Only Holonomic?

The system has **no non-holonomic constraints** because:

- There are no velocity-dependent restrictions (e.g., no "no-slip" conditions)
- All constraints can be integrated to position-level equations

9 Torque as Non-Conservative Force

The RHS of Euler-Lagrange equations contains τ_i because:

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

- **Gravity** is conservative → already included in $\partial\mathcal{L}/\partial\theta_i$ via V
- **Motor Torques** (τ_1, τ_2) are non-conservative because:
  - They are externally applied forces not derivable from a potential
  - Their work depends on the path taken by the system

10 Kinetic Energy (KE) Calculation

[... Previous KE derivation sections ...]

11 Potential Energy (PE) Calculation

[... Previous PE derivation sections ...]

12 Lagrangian

[... Previous Lagrangian section ...]

13 Equations of Motion

[... Previous equations of motion ...]

$$\tau_1 = \text{Inertia terms} + \text{Coriolis/centripetal} + \text{Gravity}$$
$$\tau_2 = \text{Inertia terms} + \text{Coriolis/centripetal} + \text{Gravity}$$

13.1 Physical Interpretation

- τ_i appears on RHS because it's an *external non-conservative force*
- Gravity terms appear on LHS because they're *conservative forces* (embedded in $\mathcal{L}$)
- Coriolis terms arise from $\mathbf{C}(\theta, \dot{\theta})\dot{\theta}$ due to velocity coupling

14 Feedback Linearization

Feedback linearization is a nonlinear control technique that algebraically transforms a nonlinear system into a linear one through state feedback. For robotic systems, this allows us to leverage linear control tools for trajectory tracking.

15 Dynamic Equations

The dynamics of a 2-DOF planar robot arm are given by:

$$\mathbf{M}(\theta)\ddot{\theta} + \mathbf{C}(\theta, \dot{\theta})\dot{\theta} + \mathbf{G}(\theta) = \tau \tag{1}$$

where:

- $\theta = [\theta_1, \theta_2]^T$: joint angles
- $\mathbf{M}(\theta)$: inertia matrix (symmetric, positive-definite)
- $\mathbf{C}(\theta, \dot{\theta})$: Coriolis/centripetal matrix
- $\mathbf{G}(\theta)$: gravity vector
- τ: control torque

15.1 Why This Specific τ?

We select the control input τ as:

$$\tau = \mathbf{M}(\theta)\mathbf{u} + \mathbf{C}(\theta, \dot{\theta})\dot{\theta} + \mathbf{G}(\theta) \tag{2}$$

Substituting into (1) yields the *linearized* system:

$$\ddot{\theta} = \mathbf{u} \tag{3}$$

This transformation:

- Cancels all nonlinear terms ($\mathbf{C}\dot{\theta} + \mathbf{G}$)
- Reshapes the inertia to identity
- Decouples the joint dynamics

16 Control Design for the Linearized System

With the system now linear ($\ddot{\theta} = \mathbf{u}$), we can apply linear control techniques.

17 PD Control Law

For trajectory tracking, we define $\mathbf{u}$ as:

$$\mathbf{u} = \ddot{\theta}_d + \mathbf{K}_v(\dot{\theta}_d - \dot{\theta}) + \mathbf{K}_p(\theta_d - \theta) \tag{4}$$

where:

- $\theta_d(t)$: desired trajectory
- $\mathbf{K}_p$, $\mathbf{K}_v$: positive definite gain matrices

17.1 Closed-Loop Dynamics

Substituting (4) into (3) gives error dynamics ($e = \theta_d - \theta$):

$$\ddot{e} + \mathbf{K}_v\dot{e} + \mathbf{K}_p e = 0 \tag{5}$$

Properly chosen gains ensure exponential stability.

18 What is a Control Law?

A control law is a mathematical expression that determines the actuator inputs (τ) based on:

- Current state (θ, $\dot{\theta}$)
- Desired behavior (e.g., trajectory tracking)
- System dynamics

19 Why Not Just $\mathbf{u} = -\mathbf{K}\mathbf{x}$?

The intuitive answer is in this case, the theta or the angle of the robot joints is changing with time. Hence the theta dot and theta double dot terms. If we only use u=-kx, we are essentially disregarding these velocity and acceleration terms. The state feedback law $\mathbf{u} = -\mathbf{K}\mathbf{x}$ is insufficient because:

- It lacks the feedforward term $\ddot{\theta}_d$ needed for trajectory tracking
- It cannot compensate for initial errors in acceleration
- For our robot, we need:

  $$\mathbf{u} = \ddot{\theta}_d - \mathbf{K}_v\dot{e} - \mathbf{K}_p e$$

  to achieve $\ddot{e} + \mathbf{K}_v\dot{e} + \mathbf{K}_p e = 0$

20 Example: Trajectory Tracking

20.1 System Parameters

$$L_1 = 1.0\,\mathrm{m}, \quad L_2 = 0.8\,\mathrm{m}$$
$$m_1 = 2.0\,\mathrm{kg}, \quad m_2 = 1.5\,\mathrm{kg}$$
$$\theta(0) = [0,0]^T\ \mathrm{rad}, \quad \dot{\theta}(0) = [0,0]^T\ \mathrm{rad/s}$$

20.2 Desired Trajectory

$$\theta_d(t) = \begin{bmatrix} \sin(t) \\ 0.5\cos(t) \end{bmatrix} \tag{6}$$

20.3 Control Computation

At $t = 1\,\mathrm{s}$ with $\mathbf{K}_p = 100\mathbf{I}$, $\mathbf{K}_v = 20\mathbf{I}$:

$$\begin{aligned}
\theta(1) &= [0.5, 0.2]^T \\
\dot{\theta}(1) &= [0.8, -0.3]^T \\
\theta_d(1) &= [0.841, 0.270]^T \\
\dot{\theta}_d(1) &= [0.540, -0.420]^T \\
\ddot{\theta}_d(1) &= [-0.841, -0.270]^T \\
e &= [0.341, 0.070]^T \\
\dot{e} &= [-0.260, -0.120]^T \\
\mathbf{u} &= \begin{bmatrix} -0.841 \\ -0.270 \end{bmatrix} + 20 \begin{bmatrix} -0.260 \\ -0.120 \end{bmatrix} + 100 \begin{bmatrix} 0.341 \\ 0.070 \end{bmatrix} \\
&= \begin{bmatrix} 25.7 \\ 8.3 \end{bmatrix} \mathrm{rad/s^2}
\end{aligned}$$

21 Complete Control Law

The final control torque is:

$$\tau = \mathbf{M}(\theta) \begin{bmatrix} 25.7 \\ 8.3 \end{bmatrix} + \mathbf{C}(\theta, \dot{\theta})\dot{\theta} + \mathbf{G}(\theta) \tag{7}$$

22 Complete Torque Calculation

Using the computed $\mathbf{u}$ from (7), we now calculate the final control torque τ.

22.1 Compute Inertia Matrix $\mathbf{M}(\theta)$

At $t = 1\,s$ with $\theta(1) = [0.5, 0.2]^T$ rad:

$$\begin{aligned}
\mathbf{M}(\theta) &= \begin{bmatrix} (m_1 + m_2)L_1^2 + m_2 L_2^2 + 2m_2 L_1 L_2 c_2 & m_2 L_2^2 + m_2 L_1 L_2 c_2 \\ m_2 L_2^2 + m_2 L_1 L_2 c_2 & m_2 L_2^2 \end{bmatrix} \\
&= \begin{bmatrix} (3.5)(1) + 1.5(0.64) + 2(1.5)(1)(0.8)\cos(0.2) & 1.5(0.64) + 1.5(1)(0.8)\cos(0.2) \\ \text{Symmetric} & 1.5(0.64) \end{bmatrix} \\
&= \begin{bmatrix} 3.5 + 0.96 + 2.35 & 0.96 + 1.18 \\ 0.96 + 1.18 & 0.96 \end{bmatrix} \\
&= \begin{bmatrix} 6.81 & 2.14 \\ 2.14 & 0.96 \end{bmatrix} \mathrm{kg \cdot m^2}
\end{aligned}$$

22.2 Compute Coriolis Matrix $\mathbf{C}(\theta, \dot{\theta})$

With $\dot{\theta} = [0.8, -0.3]^T$ rad/s:

$$\begin{aligned}
\mathbf{C}(\theta, \dot{\theta}) &= \begin{bmatrix} -2m_2 L_1 L_2 \dot{\theta}_2 s_2 & -m_2 L_1 L_2 \dot{\theta}_2 s_2 \\ m_2 L_1 L_2 \dot{\theta}_1 s_2 & 0 \end{bmatrix} \\
&= \begin{bmatrix} -2(1.5)(1)(0.8)(-0.3)\sin(0.2) & -1.5(1)(0.8)(-0.3)\sin(0.2) \\ 1.5(1)(0.8)(0.8)\sin(0.2) & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0.144\sin(0.2) & 0.072\sin(0.2) \\ 0.96\sin(0.2) & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0.0285 & 0.0142 \\ 0.1904 & 0 \end{bmatrix} \mathrm{N \cdot m \cdot s}
\end{aligned}$$

22.3 Compute Gravity Vector $\mathbf{G}(\theta)$

$$\begin{aligned}
\mathbf{G}(\theta) &= \begin{bmatrix} (m_1+m_2)gL_1c_1 + m_2gL_2c_{12} \\ m_2gL_2c_{12} \end{bmatrix} \\
&= \begin{bmatrix} 3.5(9.81)(1)\cos(0.5) + 1.5(9.81)(0.8)\cos(0.7) \\ 1.5(9.81)(0.8)\cos(0.7) \end{bmatrix} \\
&= \begin{bmatrix} 34.34(0.8776) + 11.77(0.7648) \\ 11.77(0.7648) \end{bmatrix} \\
&= \begin{bmatrix} 30.14 + 9.00 \\ 9.00 \end{bmatrix} \\
&= \begin{bmatrix} 39.14 \\ 9.00 \end{bmatrix} \mathrm{N \cdot m}
\end{aligned}$$

22.4 Compute Final Torque τ

Using $\mathbf{u} = [25.7, 8.3]^T$ from (7):

$$\begin{aligned}
\tau &= \mathbf{M}(\theta)\mathbf{u} + \mathbf{C}(\theta,\dot{\theta})\dot{\theta} + \mathbf{G}(\theta) \\
&= \begin{bmatrix} 6.81 & 2.14 \\ 2.14 & 0.96 \end{bmatrix} \begin{bmatrix} 25.7 \\ 8.3 \end{bmatrix} + \begin{bmatrix} 0.0285 & 0.0142 \\ 0.1904 & 0 \end{bmatrix} \begin{bmatrix} 0.8 \\ -0.3 \end{bmatrix} + \begin{bmatrix} 39.14 \\ 9.00 \end{bmatrix} \\
&= \begin{bmatrix} 6.81(25.7) + 2.14(8.3) \\ 2.14(25.7) + 0.96(8.3) \end{bmatrix} + \begin{bmatrix} 0.0285(0.8) + 0.0142(-0.3) \\ 0.1904(0.8) + 0(-0.3) \end{bmatrix} + \begin{bmatrix} 39.14 \\ 9.00 \end{bmatrix} \\
&= \begin{bmatrix} 175.0 + 17.8 \\ 55.0 + 8.0 \end{bmatrix} + \begin{bmatrix} 0.0228 - 0.0043 \\ 0.1523 + 0 \end{bmatrix} + \begin{bmatrix} 39.14 \\ 9.00 \end{bmatrix} \\
&= \begin{bmatrix} 192.8 \\ 63.0 \end{bmatrix} + \begin{bmatrix} 0.0185 \\ 0.1523 \end{bmatrix} + \begin{bmatrix} 39.14 \\ 9.00 \end{bmatrix} \\
&= \begin{bmatrix} 231.96 \\ 72.15 \end{bmatrix} \mathrm{N \cdot m}
\end{aligned}$$

23 Interpretation

The computed torque values:

- $\tau_1 = 231.96\,\mathrm{N \cdot m}$
- $\tau_2 = 72.15\,\mathrm{N \cdot m}$

are the actual motor torques required at $t = 1\,s$ to:

- Cancel all nonlinear effects (Coriolis, gravity)
- Compensate for tracking errors
- Achieve the desired acceleration $\ddot{\theta}_d$

24 What if we linearize the system?

25 System Description

The system is a 2-link planar robotic arm with revolute joints. The dynamics are derived using the Euler-Lagrange formalism and are given by:

$$\mathbf{M}(\theta)\ddot{\theta} + \mathbf{C}(\theta,\dot{\theta})\dot{\theta} + \mathbf{G}(\theta) = \tau \tag{8}$$

where:

- $\theta = [\theta_1, \theta_2]^T$: Joint angles
- $\mathbf{M}(\theta)$: Inertia matrix (symmetric, positive-definite)
- $\mathbf{C}(\theta,\dot{\theta})$: Coriolis and centripetal matrix
- $\mathbf{G}(\theta)$: Gravity vector
- $\tau = [\tau_1, \tau_2]^T$: Control torques

26 Nonlinear State-Space Model

26.1 State Vector Definition

To convert the second-order differential equation into a first-order state-space form, we define the state vector $\mathbf{x}$ to include both positions and velocities:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

This choice is fundamental because:

- It captures the complete dynamic state of the system (position and momentum).
- It allows us to represent the 2^{nd}-order system as a set of 1^{st}-order differential equations.

26.2 Deriving the State Equations

The state equations are derived from the definitions and the original dynamics:

$$\dot{x}_1 = x_3 \tag{9}$$
$$\dot{x}_2 = x_4 \tag{10}$$
$$\dot{x}_3 = \ddot{\theta}_1 \tag{11}$$
$$\dot{x}_4 = \ddot{\theta}_2 \tag{12}$$

Equations (9) and (10) are trivial from the state definition. Equations (11) and (12) require solving the dynamics equation (1) for the accelerations. We rewrite (1) as:

$$\ddot{\theta} = \mathbf{M}^{-1}(\theta)\left[\tau - \mathbf{C}(\theta, \dot{\theta})\dot{\theta} - \mathbf{G}(\theta)\right]$$

Thus, the full nonlinear state-space model is:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \tau) = \begin{bmatrix} x_3 \\ x_4 \\ \left[\mathbf{M}^{-1}(\mathbf{x})\left(\tau - \mathbf{C}(\mathbf{x})\dot{\mathbf{x}} - \mathbf{G}(\mathbf{x})\right)\right]_1 \\ \left[\mathbf{M}^{-1}(\mathbf{x})\left(\tau - \mathbf{C}(\mathbf{x})\dot{\mathbf{x}} - \mathbf{G}(\mathbf{x})\right)\right]_2 \end{bmatrix}$$

where $\mathbf{C}(\mathbf{x})\dot{\mathbf{x}} = \mathbf{C}(\theta, \dot{\theta})\dot{\theta}$ and $\mathbf{G}(\mathbf{x}) = \mathbf{G}(\theta)$.

26.3 Consequences of the Nonlinear Model

- **Accuracy**: The model is exact and captures all nonlinear effects (inertial coupling, Coriolis forces, gravity).
- **Complexity**: The model is complex and requires real-time computation of $\mathbf{M}, \mathbf{C}, \mathbf{G}$ for simulation or control.
- **Control Design**: Designing a controller directly for this model is challenging and typically requires advanced nonlinear methods like feedback linearization or sliding mode control.

27 Equilibrium Points

This part is slightly more involved. We set the states to zero to find the equilibrium points. Because of that, theta double dot one and two will also be zero. Substituting theta one and two dots in the equation found when setting theta double dots to be zero, we get equation 13. The physical meaning of equation 13 is that the torque needed at the equilibrium point is the gravity matrix only, which makes sense. Now we can choose any eq point that satisfies this condition if we choose the hanging position in which both the links are hanging straight down, the torque required as calculated by equation 13 will be zeros which also makes sense as while hanging, no external torque is needed. Remember, that this is only a single equilibrium point as the system itself has infinitely many equilibrium points.

27.1 Definition

An equilibrium point $(\mathbf{x}^*, \tau^*)$ is a state-input pair where the system can remain at rest indefinitely:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}^*, \tau^*) = \mathbf{0}$$

27.2 Finding an Equilibrium

From the state equations (9-12), $\dot{\mathbf{x}} = \mathbf{0}$ implies:

$$\begin{aligned} x_3^* &= 0 \\ x_4^* &= 0 \\ \ddot{\theta}^* &= \mathbf{0} \end{aligned}$$

The acceleration condition $\ddot{\theta}^* = \mathbf{0}$, combined with $\dot{\theta}^* = 0$, simplifies the dynamics equation (1) to:

$$\mathbf{0} = \mathbf{M}(\theta^*)\mathbf{0} + \mathbf{C}(\theta^*, \mathbf{0})\mathbf{0} + \mathbf{G}(\theta^*) - \tau^*$$

Thus, the condition for equilibrium is:

$$\tau^* = \mathbf{G}(\theta^*) \tag{13}$$

This means at equilibrium, the control torque must exactly counteract the gravitational torque.

27.3 Choosing a Specific Equilibrium

Let us choose the arm to be at rest in a configuration where it is hanging straight down:

$$\theta^* = \begin{bmatrix} -\pi/2 \\ 0 \end{bmatrix}, \quad \dot{\theta}^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

The gravity vector is:

$$\mathbf{G}(\theta) = \begin{bmatrix} (m_1 + m_2)gL_1 \cos\theta_1 + m_2 g L_2 \cos(\theta_1 + \theta_2) \\ m_2 g L_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

Evaluating at θ^*:

$$\mathbf{G}(\theta^*) = \begin{bmatrix} (3.5)(9.81)(1)\cos(-\pi/2) + (1.5)(9.81)(0.8)\cos(-\pi/2) \\ (1.5)(9.81)(0.8)\cos(-\pi/2) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Thus, the required torque is:

$$\tau^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

This is a natural equilibrium where gravity does not exert a torque on the joints.

28 Jacobian Linearization

28.1 Purpose

Jacobian linearization approximates the nonlinear system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \tau)$ with a linear system $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\tau$ valid near the equilibrium point $(\mathbf{x}^*, \tau^*)$. This unlocks powerful linear control design tools.

28.2 Derivation

The linear approximation is given by the first-order Taylor expansion:

$$\dot{\mathbf{x}} \approx \mathbf{f}(\mathbf{x}^*, \tau^*) + \mathbf{A}(\mathbf{x} - \mathbf{x}^*) + \mathbf{B}(\tau - \tau^*)$$

where:

$$\mathbf{A} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}^*, \tau^*}$$

$$\mathbf{B} = \left.\frac{\partial \mathbf{f}}{\partial \tau}\right|_{\mathbf{x}^*, \tau^*}$$

Since $\mathbf{f}(\mathbf{x}^*, \tau^*) = \mathbf{0}$, the linearized system is:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\tau$$

28.3 Computing the Jacobian Matrices

The function $\mathbf{f}$ is:

$$\mathbf{f}(\mathbf{x}, \tau) = \begin{bmatrix} f_1(\mathbf{x}, \tau) \\ f_2(\mathbf{x}, \tau) \\ f_3(\mathbf{x}, \tau) \\ f_4(\mathbf{x}, \tau) \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \\ \left[\mathbf{M}^{-1}(\mathbf{x})(\tau - \mathbf{C}(\mathbf{x})\dot{\mathbf{x}} - \mathbf{G}(\mathbf{x}))\right]_1 \\ \left[\mathbf{M}^{-1}(\mathbf{x})(\tau - \mathbf{C}(\mathbf{x})\dot{\mathbf{x}} - \mathbf{G}(\mathbf{x}))\right]_2 \end{bmatrix}$$

28.3.1 Matrix A

The Jacobian $\mathbf{A} = \partial \mathbf{f}/\partial \mathbf{x}$ has elements $A_{ij} = \partial f_i/\partial x_j$. For $i = 1, 2$:

$$\frac{\partial f_1}{\partial x_3} = 1, \quad \text{others } \frac{\partial f_1}{\partial x_j} = 0$$
$$\frac{\partial f_2}{\partial x_4} = 1, \quad \text{others } \frac{\partial f_2}{\partial x_j} = 0$$

For $i = 3, 4$, the derivatives are complex. However, at equilibrium ($\dot{\theta}^* = 0$, $\tau^* = \mathbf{G}(\theta^*)$), many terms vanish or simplify. After derivation:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ A_{31} & A_{32} & 0 & 0 \\ A_{41} & A_{42} & 0 & 0 \end{bmatrix}$$

where:

$$A_{31} = \frac{\partial}{\partial \theta_1} \left[\mathbf{M}^{-1}(\tau - \mathbf{G})\right]_1 \bigg|_{\mathbf{x}^*, \tau^*}$$
$$A_{32} = \frac{\partial}{\partial \theta_2} \left[\mathbf{M}^{-1}(\tau - \mathbf{G})\right]_1 \bigg|_{\mathbf{x}^*, \tau^*}$$
$$A_{41} = \frac{\partial}{\partial \theta_1} \left[\mathbf{M}^{-1}(\tau - \mathbf{G})\right]_2 \bigg|_{\mathbf{x}^*, \tau^*}$$
$$A_{42} = \frac{\partial}{\partial \theta_2} \left[\mathbf{M}^{-1}(\tau - \mathbf{G})\right]_2 \bigg|_{\mathbf{x}^*, \tau^*}$$

These terms are evaluated at the equilibrium and depend on the derivatives of $\mathbf{M}^{-1}$ and $\mathbf{G}$.

28.3.2 Matrix B

The Jacobian $\mathbf{B} = \partial \mathbf{f}/\partial \tau$ is simpler. For $i = 1, 2$:

$$\frac{\partial f_1}{\partial \tau} = 0, \quad \frac{\partial f_2}{\partial \tau} = 0$$

For $i = 3, 4$:

$$\frac{\partial \ddot{\theta}}{\partial \tau} = \mathbf{M}^{-1}(\theta)$$

Thus:

$$\mathbf{B} = \left[\begin{array}{cc} 0 & 0 \\ 0 & 0 \\ \hline \multicolumn{2}{c}{\mathbf{M}^{-1}(\theta^*)} \end{array}\right]$$

28.4 Consequences of Linearization

- **Validity**: The linear model is only accurate in a small region around $(\mathbf{x}^*, \tau^*)$. Large deviations lead to significant errors.
- **Simplicity**: The model is linear time-invariant (LTI) and easier to analyze.
- **Control Design**: Enables the use of powerful linear design tools (pole placement, LQR, etc.) for local stabilization.
- **Performance**: A controller designed from the linear model may perform poorly or even become unstable if the arm moves too far from the equilibrium point.

29 Conclusion

This document has detailed the process of deriving the nonlinear state-space model, finding equilibrium points, and performing Jacobian linearization for a 2-DOF robot arm. The nonlinear model is essential for understanding the full dynamics and for designing controllers that operate over a wide range (e.g., via feedback linearization). The linearized model is a valuable tool for local control design and analysis, providing a simpler representation that facilitates the use of linear control techniques near a specific operating point.