








Advanced Robotics Project

Objective: Control of a 2-link manipulator and comparing three control algorithms namely PD controller, PD controller with computed torque controller and robust controller (i.e. Adaptive controller).

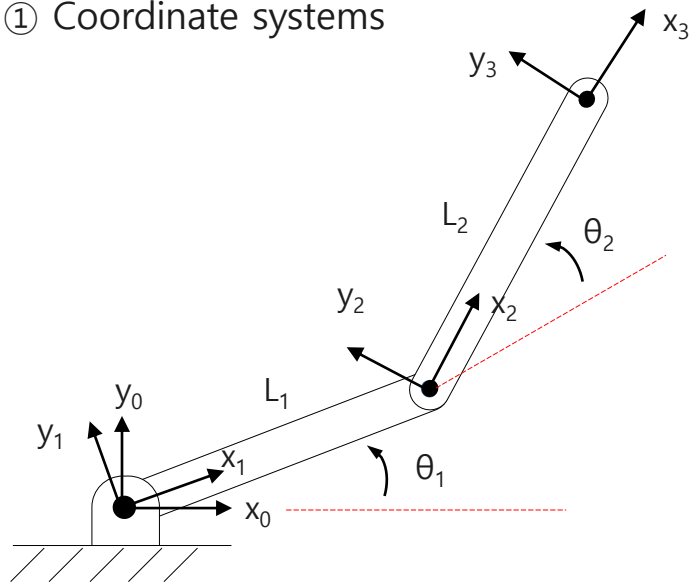
Contents

-  1 Kinematics of Manipulator●
-  2 Dynamic Equations●
-  3 Desired Trajectory●
-  4 PD Controller Design●
-  5 PD + Computed Torque Controller Design●
-  6 Robust Controller Design●
-  7 Performance Comparison of Controllers●

1. Kinematics of a Manipulator(1)

■ Forward Kinematics

① Coordinate systems



② D-H Parameter Table

	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	L_1	0	0	θ_2
3	L_2	0	0	0

$${}^{i+1}_iT = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ c\alpha_{i-1}s\theta_i & c\alpha_{i-1}c\theta_i & -s\alpha_{i-1} & -d_is\alpha_{i-1} \\ s\alpha_{i-1}s\theta_i & s\alpha_{i-1}c\theta_i & c\alpha_{i-1} & d_ic\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

③ Transformation Matrix

$${}^0_1T = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & 0 \\ s\theta_1 & c\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^1_2T = \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 & L_1 \\ s\theta_2 & c\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^2_3T = \begin{bmatrix} 1 & 0 & 0 & L_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \text{constant}$$

1. Kinematics of a Manipulator(2)

$${}^0_2T = {}^0_1T \cdot {}^1_2T = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & 0 \\ s\theta_1 & c\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 & L_1 \\ s\theta_2 & c\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c\theta_1c\theta_2 - s\theta_1s\theta_2 & -c\theta_1s\theta_2 - s\theta_1c\theta_2 & 0 & L_1c\theta_1 \\ s\theta_1c\theta_2 + c\theta_1s\theta_2 & -s\theta_1s\theta_2 + c\theta_1c\theta_2 & 0 & L_1s\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c\theta_{12} & -s\theta_{12} & 0 & L_1c\theta_1 \\ s\theta_{12} & c\theta_{12} & 0 & L_1s\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where, $\theta_{12} = \theta_1 + \theta_2$ and

$${}^0_3T = {}^0_2T \cdot {}^2_3T = \begin{bmatrix} c\theta_{12} & -s\theta_{12} & 0 & L_1c\theta_1 \\ s\theta_{12} & c\theta_{12} & 0 & L_1s\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & L_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c\theta_{12} & -s\theta_{12} & 0 & L_1c\theta_1 + L_2c\theta_{12} \\ s\theta_{12} & c\theta_{12} & 0 & L_1s\theta_1 + L_2s\theta_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

④ Kinematic Coordinates

$${}^0P = {}^0_3T \cdot {}^3P \quad \begin{Bmatrix} x_d \\ y_d \\ z_d \\ 1 \end{Bmatrix} = \begin{bmatrix} c\theta_{12} & -s\theta_{12} & 0 & L_1c\theta_1 + L_2c\theta_{12} \\ s\theta_{12} & c\theta_{12} & 0 & L_1s\theta_1 + L_2s\theta_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{Bmatrix} = \begin{Bmatrix} L_1c\theta_1 + L_2c\theta_{12} \\ L_1s\theta_1 + L_2s\theta_{12} \\ 0 \\ 1 \end{Bmatrix}$$

$$\phi_d = \theta_1 + \theta_2$$

1. Kinematics of a Manipulator(3)

■ Inverse Kinematics

① Desired Position & Orientation

we can define only position's of the end effector, except its orientation, because the system has just 2-dof for the planar motion.

② Inverse Kinematic equations – by Algebraic method

$$\begin{aligned} x_d &= L_1 c\theta_1 + L_2 c\theta_2 \\ y_d &= L_1 s\theta_1 + L_2 s\theta_2 \end{aligned} \quad \longrightarrow \quad \begin{aligned} x_d^2 + y_d^2 &= L_1^2 + L_2^2 + 2L_1L_2c\theta_2 \end{aligned}$$

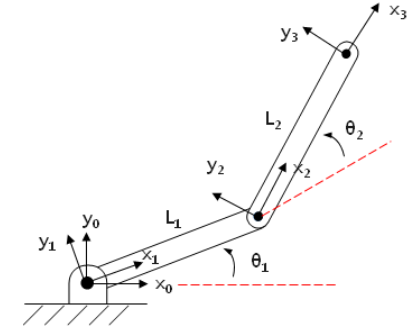
and then, we can compute θ_2

$$c\theta_2 = \frac{x_d^2 + y_d^2 - L_1^2 - L_2^2}{2L_1L_2} \quad s\theta_2 = \pm\sqrt{1 - c^2\theta_2}$$

$$\underline{\theta_2 = \text{Atan } 2(s\theta_2, c\theta_2)}$$

There exist two values for the θ_2 .

The two solutions are the 'elbow up' and elbow down', respectively



1. Kinematics of a Manipulator(4)

for the θ_1

$$x_d = L_1 c\theta_1 + L_2 c\theta_{12} = L_1 c\theta_1 + L_2 c\theta_1 c\theta_2 - L_2 s\theta_1 s\theta_2 = (L_1 + L_2 c\theta_2) c\theta_1 - (L_2 s\theta_2) s\theta_1$$

$$y_d = L_1 s\theta_1 + L_2 s\theta_{12} = L_1 s\theta_1 + L_2 s\theta_1 c\theta_2 + L_2 c\theta_1 s\theta_2 = (L_2 s\theta_2) c\theta_1 + (L_1 + L_2 c\theta_1) s\theta_1$$



$$\begin{aligned} x_d &= k_1 c\theta_1 - k_2 s\theta_1 \\ y_d &= k_2 c\theta_1 + k_1 s\theta_1 \end{aligned} \quad \text{where,} \quad \begin{aligned} k_1 &= L_1 + L_2 c\theta_2 \\ k_2 &= L_2 s\theta_2 \end{aligned}$$

for the calculation of above equations, we can make matrix form

$$\begin{aligned} x &= k_1 c_1 - k_2 s_1 \\ y &= k_1 s_1 + k_2 c_1 \end{aligned} \quad \longrightarrow \quad \begin{bmatrix} k_1 & -k_2 \\ k_2 & k_1 \end{bmatrix} \begin{Bmatrix} c_1 \\ s_1 \end{Bmatrix} = \begin{Bmatrix} x \\ y \end{Bmatrix} \quad \longrightarrow \quad \text{by Cramer's rule}$$

$$\theta_1 = \text{Atan } 2 \left(\frac{k_1 y - k_2 x}{k_1^2 + k_2^2}, \frac{k_1 x + k_2 y}{k_1^2 + k_2^2} \right) = \underline{\text{Atan } 2(k_1 y - k_2 x, k_1 x + k_2 y)}$$

$$c_1 = \frac{\begin{vmatrix} x & -k_2 \\ y & k_1 \end{vmatrix}}{\begin{vmatrix} k_1 & -k_2 \\ k_2 & k_1 \end{vmatrix}} = \frac{k_1 x + k_2 y}{k_1^2 + k_2^2}$$

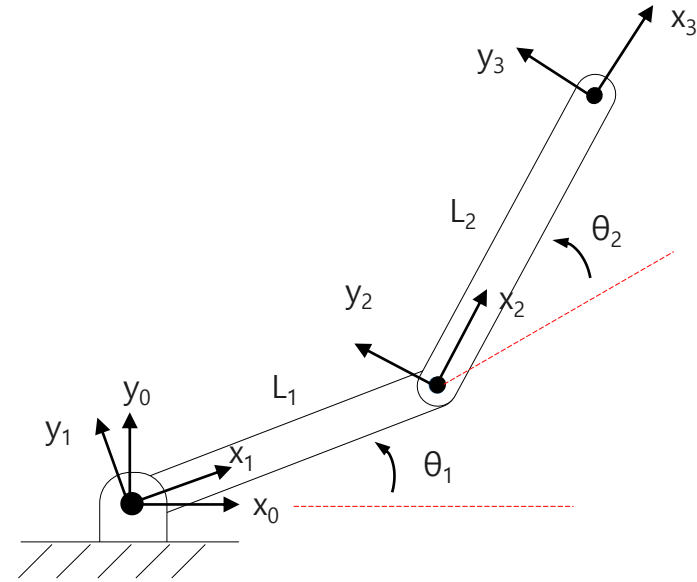
$$s_1 = \frac{\begin{vmatrix} k_1 & x \\ k_2 & y \end{vmatrix}}{\begin{vmatrix} k_1 & -k_2 \\ k_2 & k_1 \end{vmatrix}} = \frac{k_1 y - k_2 x}{k_1^2 + k_2^2}$$

1. Kinematics of a Manipulator(5)

■ System Parameters

we can choose the parameter as a following table.

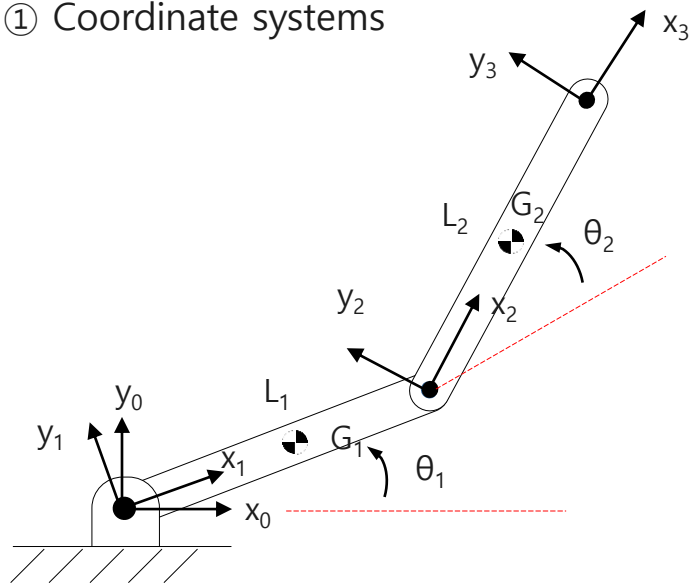
m1	m2	L1	L2
1 kg	1 kg	1 m	1 m



2. Dynamic Equations.(1)

■ Equations of Motion

① Coordinate systems



$$\begin{aligned} x_{G2} &= L_1 c_1 + 0.5L_2 c_{12} \\ y_{G2} &= L_1 s_1 + 0.5L_2 s_{12} \end{aligned} \xrightarrow{\text{differentiation}} \begin{aligned} \dot{x}_{G2} &= -L_1 s_1 \dot{\theta}_1 - 0.5L_2 s_{12} (\dot{\theta}_1 + \dot{\theta}_2) \\ \dot{y}_{G2} &= L_1 c_1 \dot{\theta}_1 + 0.5L_2 c_{12} (\dot{\theta}_1 + \dot{\theta}_2) \end{aligned}$$

then,

$$\begin{aligned} V_{G2}^2 &= \dot{x}_{G2}^2 + \dot{y}_{G2}^2 \\ &= (L_1^2 + 0.25L_2^2 + L_1L_2c_2)\dot{\theta}_1^2 + (0.25L_2^2)\dot{\theta}_2^2 + (0.5L_2^2 + L_1L_2c_2)\dot{\theta}_1\dot{\theta}_2 \end{aligned}$$

② Kinetic Energy

$$K_1 = \frac{1}{2} I_{G1} \dot{\theta}_1^2 + \frac{1}{2} m_1 V_{G1}^2 = \frac{1}{2} \left(\frac{1}{12} m_1 L_1^2 \right) \dot{\theta}_1^2 + \frac{1}{2} m_1 V_{G1}^2$$

$$K = K_1 + K_2$$

or

$$K_1 = \frac{1}{2} I_{L1} \dot{\theta}_1^2 = \frac{1}{2} \left(\frac{1}{3} m_1 L_1^2 \right) \dot{\theta}_1^2 = \frac{1}{6} m_1 L_1^2 \dot{\theta}_1^2$$

we can get the same result.

2. Dynamic Equations.(2)

$$\begin{aligned} K_2 &= \frac{1}{2} I_{G2} (\dot{\theta}_1^2 + \dot{\theta}_2^2) + \frac{1}{2} m_2 V_{G2}^2 = \frac{1}{2} \left(\frac{1}{12} m_2 L_2^2 \right) (\dot{\theta}_1 + \dot{\theta}_2)^2 + \frac{1}{2} m_2 V_{G2}^2 \\ &= \frac{1}{2} \left(\frac{1}{12} m_2 L_2^2 \right) (\dot{\theta}_1^2 + \dot{\theta}_2^2 + 2\dot{\theta}_1 \dot{\theta}_2) + \frac{1}{2} m_2 \{ (L_1^2 + 0.25L_2^2 + L_1 L_2 c_2) \dot{\theta}_1^2 + (0.25L_2^2) \dot{\theta}_2^2 + (0.5L_2^2 + L_1 L_2 c_2) \dot{\theta}_1 \dot{\theta}_2 \} \\ &= \left(\frac{1}{24} m_2 L_2^2 + \frac{1}{2} m_2 (L_1^2 + 0.25L_2^2 + L_1 L_2 c_2) \right) \dot{\theta}_1^2 + \left(\frac{1}{24} m_2 L_2^2 + \frac{1}{2} m_2 (0.25L_2^2) \right) \dot{\theta}_2^2 + \frac{1}{2} m_2 \left(\frac{1}{6} L_2^2 + 0.5L_2^2 + L_1 L_2 c_2 \right) \dot{\theta}_1 \dot{\theta}_2 \\ &= \left(\frac{1}{24} m_2 L_2^2 + \frac{1}{2} m_2 L_1^2 + \frac{1}{8} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_1^2 + \left(\frac{1}{24} m_2 L_2^2 + \frac{1}{8} m_2 L_2^2 \right) \dot{\theta}_2^2 + \left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_1 \dot{\theta}_2 \\ &= \left(\frac{1}{2} m_2 L_1^2 + \frac{1}{6} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_1^2 + \left(\frac{1}{6} m_2 L_2^2 \right) \dot{\theta}_2^2 + \left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_1 \dot{\theta}_2 \end{aligned}$$

thus, the kinetic energy is

$$\begin{aligned} K &= K_1 + K_2 = \frac{1}{6} m_1 L_1^2 \dot{\theta}_1^2 + \left(\frac{1}{2} m_2 L_1^2 + \frac{1}{6} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_1^2 + \left(\frac{1}{6} m_2 L_2^2 \right) \dot{\theta}_2^2 + \left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_1 \dot{\theta}_2 \\ &= \left(\frac{1}{6} m_1 L_1^2 + \frac{1}{2} m_2 L_1^2 + \frac{1}{6} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_1^2 + \left(\frac{1}{6} m_2 L_2^2 \right) \dot{\theta}_2^2 + \left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_1 \dot{\theta}_2 \end{aligned}$$

2. Dynamic Equations.(3)

③ Potential Energy

$$P = P_1 + P_2$$
$$P_1 = \frac{1}{2}m_1gL_1s_1$$
$$P_2 = m_2g\left(L_1s_1 + \frac{1}{2}L_2s_{12}\right)$$

thus, the potential energy is

$$P = P_1 + P_2 = \frac{1}{2}m_1gL_1s_1 + m_2g\left(L_1s_1 + \frac{1}{2}L_2s_{12}\right)$$

④ Lagrangian

$$L = K - P$$

$$L = K - P$$
$$= \left(\frac{1}{6}m_1L_1^2 + \frac{1}{2}m_2L_1^2 + \frac{1}{6}m_2L_2^2 + \frac{1}{2}m_2L_1L_2c_2\right)\dot{\theta}_1^2 + \left(\frac{1}{6}m_2L_2^2\right)\dot{\theta}_2^2 + \left(\frac{1}{3}m_2L_2^2 + \frac{1}{2}m_2L_1L_2c_2\right)\dot{\theta}_1\dot{\theta}_2 - \frac{1}{2}m_1gL_1s_1 - m_2g\left(L_1s_1 + \frac{1}{2}L_2s_{12}\right)$$

2. Dynamic Equations.(4)

④ Differentiation of Lagrangian

$$\frac{\partial L}{\partial \dot{\theta}_1} = \left(\frac{1}{6} m_1 L_1^2 + \frac{1}{2} m_2 L_1^2 + \frac{1}{6} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) 2\dot{\theta}_1 + \left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_2$$

$$\frac{\partial L}{\partial \dot{\theta}_2} = \left(\frac{1}{6} m_2 L_2^2 \right) 2\dot{\theta}_2 + \left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \dot{\theta}_1$$

$$\frac{\partial L}{\partial \theta_1} = - \left(\frac{1}{2} m_1 g L_1 c_1 \right) - m_2 g \left(L_1 c_1 + \frac{1}{2} L_2 c_{12} \right) = - \left(\frac{1}{2} m_1 + m_2 \right) g L_1 c_1 - \frac{1}{2} m_2 g L_2 c_{12}$$

$$\frac{\partial L}{\partial \theta_2} = \left(-\frac{1}{2} m_2 L_1 L_2 s_2 \right) \dot{\theta}_1^2 + \left(-\frac{1}{2} m_2 L_1 L_2 s_2 \right) \dot{\theta}_1 \dot{\theta}_2 - \frac{1}{2} m_2 g L_2 c_{12}$$

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\theta}_1} \right) = \left(\frac{1}{3} m_1 L_1^2 + m_2 L_1^2 + \frac{1}{3} m_2 L_2^2 + m_2 L_1 L_2 c_2 \right) \ddot{\theta}_1 + \left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \ddot{\theta}_2 + (-m_2 L_1 L_2 s_2) \dot{\theta}_1 \dot{\theta}_2 + \left(-\frac{1}{2} m_2 L_1 L_2 s_2 \right) \theta_2^2$$

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\theta}_2} \right) = \left(\frac{1}{3} m_2 L_2^2 \right) \ddot{\theta}_2 + \left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \ddot{\theta}_1 + \left(-\frac{1}{2} m_2 L_1 L_2 s_2 \right) \dot{\theta}_1 \dot{\theta}_2$$

2. Dynamic Equations.(5)

⑤ Equations of motion

$$\text{for } \theta_1 \quad \frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\theta}_1} \right) - \frac{\partial L}{\partial \theta_1} = T_1$$

$$\begin{aligned} \left(\frac{1}{3} m_1 L_1^2 + m_2 L_1^2 + \frac{1}{3} m_2 L_2^2 + m_2 L_1 L_2 c_2 \right) \ddot{\theta}_1 + \left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \ddot{\theta}_2 + (-m_2 L_1 L_2 s_2) \dot{\theta}_1 \dot{\theta}_2 + \left(-\frac{1}{2} m_2 L_1 L_2 s_2 \right) \theta_2^2 \\ + \left(\frac{1}{2} m_1 + m_2 \right) g L_1 c_1 + \frac{1}{2} m_2 g L_2 c_{12} = T_1 \end{aligned}$$

$$\text{for } \theta_2 \quad \frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\theta}_2} \right) - \frac{\partial L}{\partial \theta_2} = T_2$$

$$\left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \ddot{\theta}_1 + \left(\frac{1}{3} m_2 L_2^2 \right) \ddot{\theta}_2 + \left(-\frac{1}{2} m_2 L_1 L_2 s_2 \right) \dot{\theta}_1 \dot{\theta}_2 + \left(\frac{1}{2} m_2 L_1 L_2 s_2 \right) \theta_1^2 + \left(\frac{1}{2} m_2 L_1 L_2 s_2 \right) \dot{\theta}_1 \dot{\theta}_2 + \frac{1}{2} m_2 g L_2 c_{12} = T_2$$

$$\left(\frac{1}{3} m_2 L_2^2 + \frac{1}{2} m_2 L_1 L_2 c_2 \right) \ddot{\theta}_1 + \left(\frac{1}{3} m_2 L_2^2 \right) \ddot{\theta}_2 + \left(\frac{1}{2} m_2 L_1 L_2 s_2 \right) \dot{\theta}_1^2 + \frac{1}{2} m_2 g L_2 c_{12} = T_2$$

2. Dynamic Equations.(6)

matrix form
$$\begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{Bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{Bmatrix} + \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \end{Bmatrix} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1 \dot{\theta}_2 \\ \dot{\theta}_2 \dot{\theta}_1 \end{Bmatrix} + \begin{Bmatrix} G_1 \\ G_2 \end{Bmatrix} = \begin{Bmatrix} T_1 \\ T_2 \end{Bmatrix}$$

$$J_{11} = \frac{1}{3}m_1L_1^2 + m_2L_1^2 + \frac{1}{3}m_2L_2^2 + m_2L_1L_2c_2$$

$$J_{12} = \frac{1}{3}m_2L_2^2 + \frac{1}{2}m_2L_1L_2c_2$$

$$J_{21} = \frac{1}{3}m_2L_2^2 + \frac{1}{2}m_2L_1L_2c_2$$

$$J_{22} = \frac{1}{3}m_2L_2^2$$

$$D_{11} = 0 \quad D_{12} = -\frac{1}{2}m_2L_1L_2s_2$$

$$D_{21} = \frac{1}{2}m_2L_1L_2s_2 \quad D_{22} = 0$$

$$C_{11} = -\frac{m_2}{2}L_1L_2s_2$$

$$C_{12} = -\frac{m_2}{2}L_1L_2s_2$$

$$C_{21} = 0$$

$$C_{22} = 0$$

$$G_1 = \left(\frac{1}{2}m_1 + m_2 \right) gL_1c_1 + \frac{1}{2}m_2gL_2c_{12}$$

$$G_2 = \frac{1}{2}m_2gL_2c_{12}$$

2. Dynamic Equations.(7)

⑥ State Equations for a simulation

$$J_{11}\ddot{\theta}_1 + J_{12}\ddot{\theta}_2 + D_{12}\dot{\theta}_2^2 + C_{11}\dot{\theta}_1\dot{\theta}_2 + C_{12}\dot{\theta}_2\dot{\theta}_1 + G_1 = T_1$$

$$J_{21}\ddot{\theta}_1 + J_{22}\ddot{\theta}_2 + D_{21}\dot{\theta}_1^2 + G_2 = T_2$$

state variables

$$\begin{aligned} x_1 &= \theta_1 \\ x_2 &= \dot{\theta}_1 \\ x_3 &= \theta_2 \\ x_4 &= \dot{\theta}_2 \end{aligned}$$

$$\ddot{\theta}_1 = -\frac{J_{12}}{J_{11}}\ddot{\theta}_2 - \frac{D_{12}}{J_{11}}\dot{\theta}_2^2 - \frac{C_{11}}{J_{11}}\dot{\theta}_1\dot{\theta}_2 - \frac{C_{12}}{J_{11}}\dot{\theta}_2\dot{\theta}_1 - \frac{G_1}{J_{11}} + \frac{1}{J_{11}}T_1$$

$$\ddot{\theta}_2 = -\frac{J_{21}}{J_{22}}\ddot{\theta}_1 - \frac{D_{21}}{J_{22}}\dot{\theta}_1^2 - \frac{G_2}{J_{22}} - \frac{1}{J_{22}}T_2$$

$$\dot{x}_1 = x_2 = 0x_1 + 1x_2 + 0x_3 + 0x_4$$

$$\dot{x}_3 = x_4 = 0x_1 + 0x_2 + 0x_3 + 1x_4$$

$$\begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{Bmatrix} \dot{x}_2 \\ \dot{x}_4 \end{Bmatrix} = - \begin{Bmatrix} D_{12}x_4^2 + C_{11}x_2x_4 + C_{12}x_4x_2 + G_1 \\ D_{21}x_2^2 + G_2 \end{Bmatrix} + \begin{Bmatrix} T_1 \\ T_2 \end{Bmatrix}$$

$$J_{11}\dot{x}_2 + J_{12}\dot{x}_4 + D_{12}x_4^2 + C_{11}x_2x_4 + C_{12}x_4x_2 + G_1 = T_1$$

$$J_{21}\dot{x}_2 + J_{22}\dot{x}_4 + D_{21}x_2^2 + G_2 = T_2$$

$$\begin{Bmatrix} \dot{x}_2 \\ \dot{x}_4 \end{Bmatrix} = - \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}^{-1} \begin{Bmatrix} D_{12}x_4^2 + C_{11}x_2x_4 + C_{12}x_4x_2 + G_1 \\ D_{21}x_2^2 + G_2 \end{Bmatrix} + \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}^{-1} \begin{Bmatrix} T_1 \\ T_2 \end{Bmatrix}$$

2. Dynamic Equations.(8)

■ Simulation with matlab – ode45()

for Dynamic function

$$\ddot{X} = -J^{-1}D(\dot{\theta}_1^2, \dot{\theta}_2^2) - J^{-1}C(\dot{\theta}_1, \dot{\theta}_2) - J^{-1}G(\theta_1, \theta_2) + J^{-1}T$$

$$\dot{x}_3 = x_4 = 0x_1 + 0x_2 + 0x_3 + 1x_4$$

$$\begin{Bmatrix} \dot{x}_2 \\ \dot{x}_4 \end{Bmatrix} = - \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}^{-1} \begin{Bmatrix} D_{12}x_3^2 + C_{11}x_2x_4 + C_{12}x_4x_2 + G_1 \\ D_{21}x_1^2 + G_2 \end{Bmatrix} + \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}^{-1} \begin{Bmatrix} T_1 \\ T_2 \end{Bmatrix}$$

$$m_1 = 1 \text{ kg} \quad L_1 = 1 \text{ m} \quad g = 9.81 \text{ m/sec}^2$$

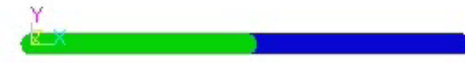
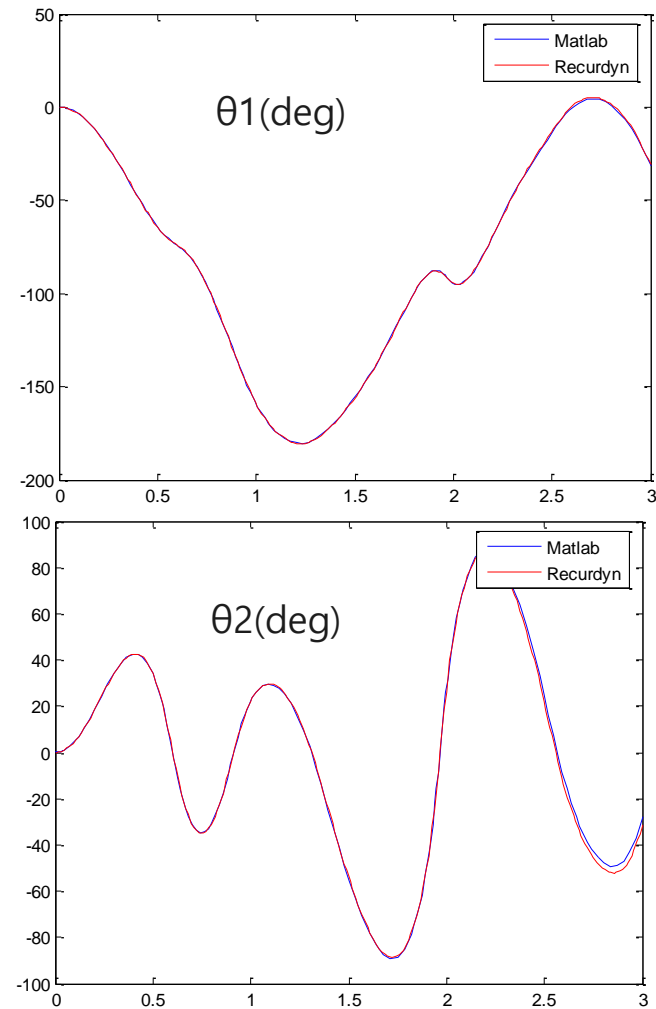
$$m_2 = 1 \text{ kg} \quad L_2 = 1 \text{ m}$$

initial conditions are zero, that is free fall problem

for the verification of the result, we compared the result of commercial software, Recurdyn.

2. Dynamic Equations.(9)

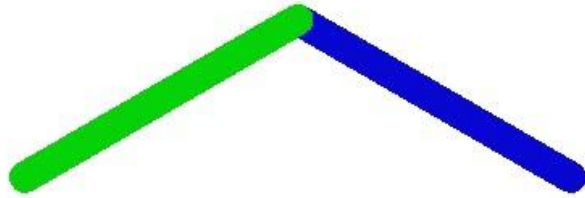
■ Result Comparison with Commercial S/W



< Animation of a Double Pendulum for free fall >

3. Desired Trajectory Simulation

■ Desired Motion

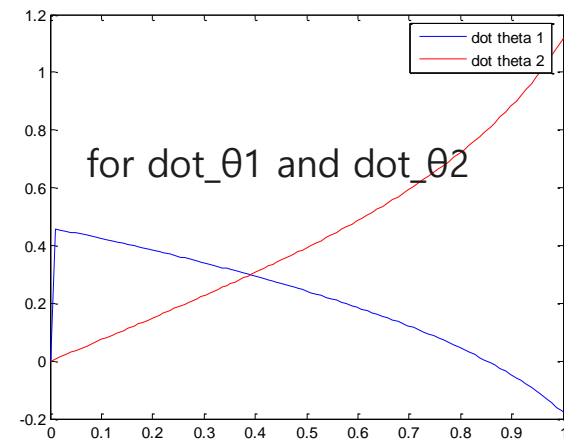
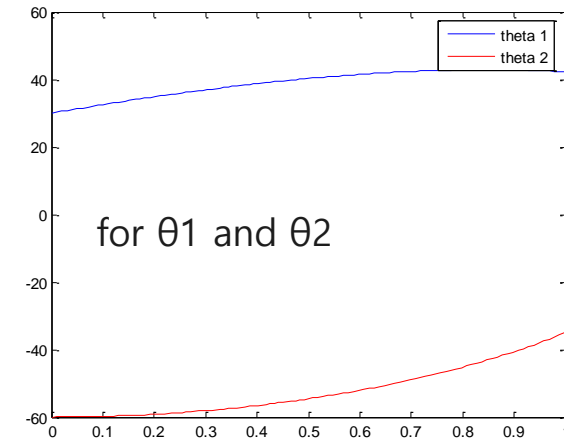


< Animation of a Double Pendulum for Desired Motion >

$V_{up} = 800\text{mm/sec}$

Initial condition : $\theta_1 = 30$, $\theta_2 = -60$

In order to acquire desired motion,
a commercial S/W, Recurdyn was used.



5. PD Controller Design(1)

■ Equations of motion for the system

$$\begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{Bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{Bmatrix} + \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \end{Bmatrix} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1 \dot{\theta}_2 \\ \dot{\theta}_2 \dot{\theta}_1 \end{Bmatrix} + \begin{Bmatrix} G_1 \\ G_2 \end{Bmatrix} = \begin{Bmatrix} T_1 \\ T_2 \end{Bmatrix} \longrightarrow J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1 \dot{\theta}_2) + G = T$$

In order to apply PD controller, we have to make linearized E.O.M.s because PD control is for a linear system

< Assumption of Linearization >

- operating points are zero.
- operating ranges are very small



$$\theta = \theta_0 + \delta\theta$$



$$\theta = \delta\theta$$

$$\cos \theta \approx 1$$

$$\sin \theta \approx \theta$$

$$\tan \theta \approx \theta$$

$$\theta_i \cdot \theta_j \approx 0$$

$$J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1 \dot{\theta}_2) + G = T \longrightarrow J\ddot{\theta} + \cancel{D(\dot{\theta}^2)} + \cancel{C(\dot{\theta}_1 \dot{\theta}_2)} + G_{linearized} = T$$

where,

$$G_{linearized} = \begin{Bmatrix} G_{1,linearized} \\ G_{2,linearize} \end{Bmatrix} = \begin{Bmatrix} \left(\frac{1}{2} m_1 + m_2 \right) g L_1 + \frac{1}{2} m_2 g L_2 \\ \frac{1}{2} m_2 g L_2 \end{Bmatrix}$$

5. PD Controller Design(2)

■ PD Control Input

For the pure PD control, we neglect the effect of gravity.

$$J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1\dot{\theta}_2) + G = T$$

we can select the input Torques as following

$$T = \alpha T' \quad \text{where,} \quad \alpha = J = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}$$

then, the E.O.M. can be

$$J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1\dot{\theta}_2) + G = J \cdot T'$$

we choose the input torque T'

$$\ddot{\theta}_d = \begin{Bmatrix} \ddot{\theta}_{1,d} \\ \ddot{\theta}_{2,d} \end{Bmatrix} \quad \text{desired angles}$$

$$e = \begin{Bmatrix} \theta_{1,d} - \theta_1 \\ \theta_{2,d} - \theta_2 \end{Bmatrix} \quad \text{errors between desired angle and real angle}$$

$$T' = \ddot{\theta}_d + K_v \dot{e} + K_p e \quad \text{where,} \quad K_v = \begin{bmatrix} k_{v11} & k_{v12} \\ k_{v21} & k_{v22} \end{bmatrix} \quad \begin{matrix} \text{gain of D.} \\ \text{(for velocity)} \end{matrix}$$

$$K_p = \begin{bmatrix} k_{p11} & k_{p12} \\ k_{p21} & k_{p22} \end{bmatrix} \quad \begin{matrix} \text{gain of P.} \\ \text{(for position)} \end{matrix}$$

5. PD Controller Design(3)

then,

$$J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1\dot{\theta}_2) + G = J \cdot (\ddot{\theta}_d + K_v\dot{e} + K_p e)$$

$$J(\ddot{\theta}_d - \ddot{\theta}) + J \cdot K_v\dot{e} + J \cdot K_p e - D(\dot{\theta}^2) - C(\dot{\theta}_1\dot{\theta}_2) - G = 0$$

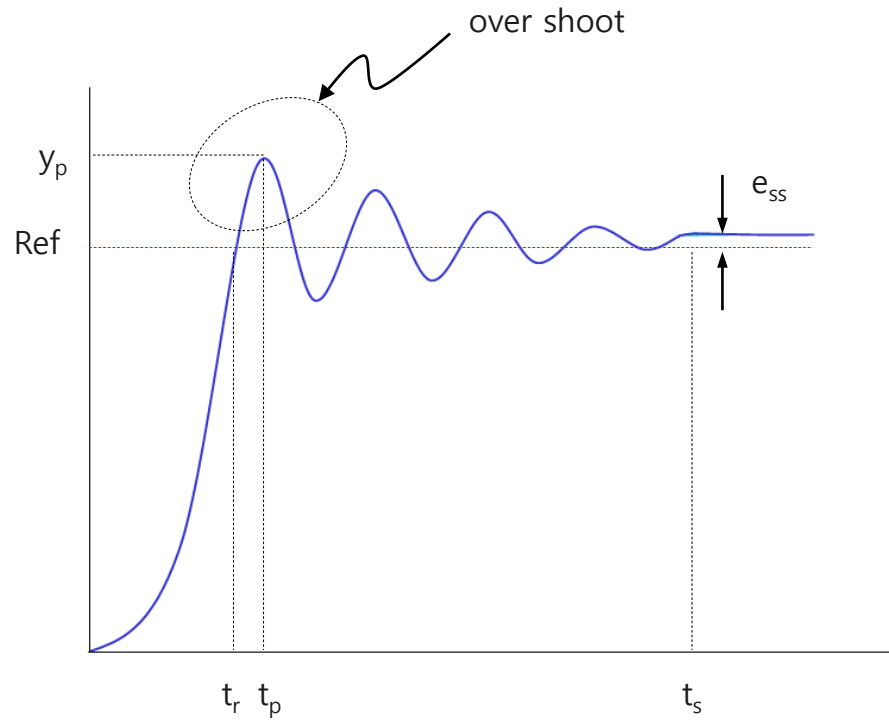
$$J(\ddot{e} + K_v\dot{e} + K_p e) - D(\dot{\theta}^2) - C(\dot{\theta}_1\dot{\theta}_2) - G = 0$$

finally, we can get the equations

$$\ddot{e} + K_v\dot{e} + K_p e - J^{-1}D(\dot{\theta}^2) - J^{-1}C(\dot{\theta}_1\dot{\theta}_2) - J^{-1}G = 0$$

5. PD Controller Design(4)

■ System's Performance for 1-dof model



$$P.O.(%) = \left(\frac{y_p - y_{ss}}{y_{ss}} \right) \times 100 = 100 \cdot e^{-\left(\zeta \pi / \sqrt{1-\zeta^2} \right)}$$

$$\zeta = \frac{-\ln(P.O./100)}{\sqrt{\pi^2 + \ln^2(P.O./100)}}$$

$$t_p = \frac{\pi}{\omega_d} = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}}$$

$$y_p = 1 + e^{-\left(\zeta \pi / \sqrt{1-\zeta^2} \right)}$$

$$t_r = \frac{\pi - \theta}{\omega_d}$$

$$\theta = \tan^{-1} \left(\sqrt{1-\zeta^2} / \zeta \right)$$

$$t_s = \frac{4}{\zeta \omega_n}$$

5. PD Controller Design(5)

■ Desired Performance of the System

Settling time	t_s	0.3 sec
Percent Overshoot	P.O.(%)	20 %

for θ_1 and θ_2

$$\zeta = \frac{-\ln(P.O./100)}{\sqrt{\pi^2 + \ln^2(P.O./100)}} = \frac{-\ln(10/100)}{\sqrt{\pi^2 + \ln^2(10/100)}} = 0.6523$$

$$\omega_n = \frac{4}{\zeta t_s} = 20.44$$

$$\ddot{e} + 26.6667\dot{e} + 417.7923e = 0$$

then, desired system equations are

$$\ddot{\theta} + 2\zeta\omega_n\dot{\theta} + \omega_n^2\theta = 0 \quad \longrightarrow \quad \ddot{\theta} + 26.6667\dot{\theta} + 417.7923\theta = 0$$

$$\ddot{e} + 2\zeta\omega_n\dot{e} + \omega_n^2e = 0 \quad \longrightarrow \quad \ddot{e} + 26.6667\dot{e} + 417.7923e = 0$$

from the response of system's error

$$\ddot{e} + J^{-1}K_v\dot{e} + J^{-1}K_p e = 0 \quad \text{then, we can compare the coefficient matrices.}$$

5. PD Controller Design(6)

■ Determination of PD Gains

$$\ddot{e} + J^{-1}K_v\dot{e} + J^{-1}K_p e = 0$$

$$K_v' = J^{-1}K_v = \begin{bmatrix} k_{v11}' & k_{v12}' \\ k_{v21}' & k_{v22}' \end{bmatrix}$$

$$K_p' = J^{-1}K_p = \begin{bmatrix} k_{p11}' & k_{p12}' \\ k_{p21}' & k_{p22}' \end{bmatrix}$$



$$\ddot{e} + 2\zeta\omega_n\dot{e} + \omega_n^2 e = 0$$

$$K_v' = \begin{bmatrix} 2\zeta\omega_n & 0 \\ 0 & 2\zeta\omega_n \end{bmatrix}$$

$$K_p' = \begin{bmatrix} \omega_n^2 & 0 \\ 0 & \omega_n^2 \end{bmatrix}$$

we just consider diagonal terms and then, computed the gains of controller

$$K_v = J \cdot K_v' = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} 2\zeta\omega_n & 0 \\ 0 & 2\zeta\omega_n \end{bmatrix}$$

$$K_p = J \cdot K_p' = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} \omega_n^2 & 0 \\ 0 & \omega_n^2 \end{bmatrix}$$

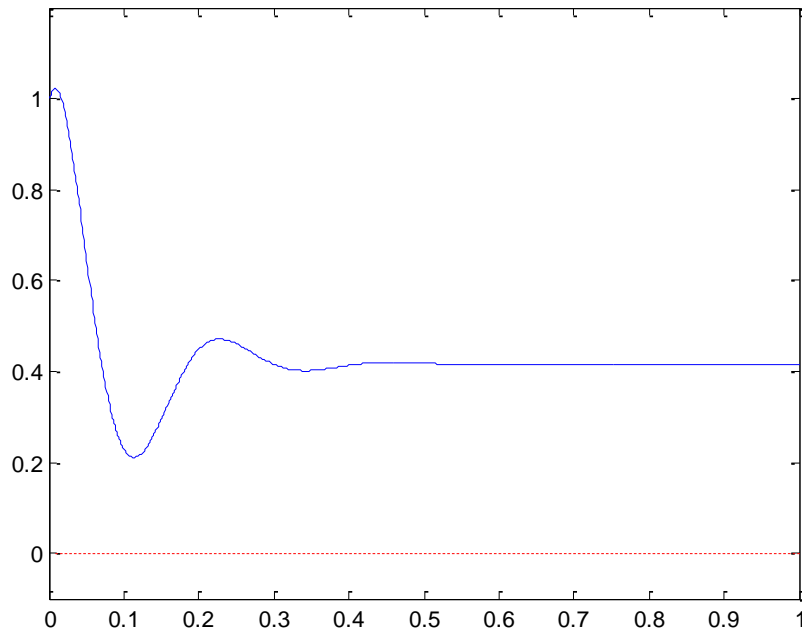
5. PD Controller Design(7)

■ Results of simulation for PD Control Algorithms

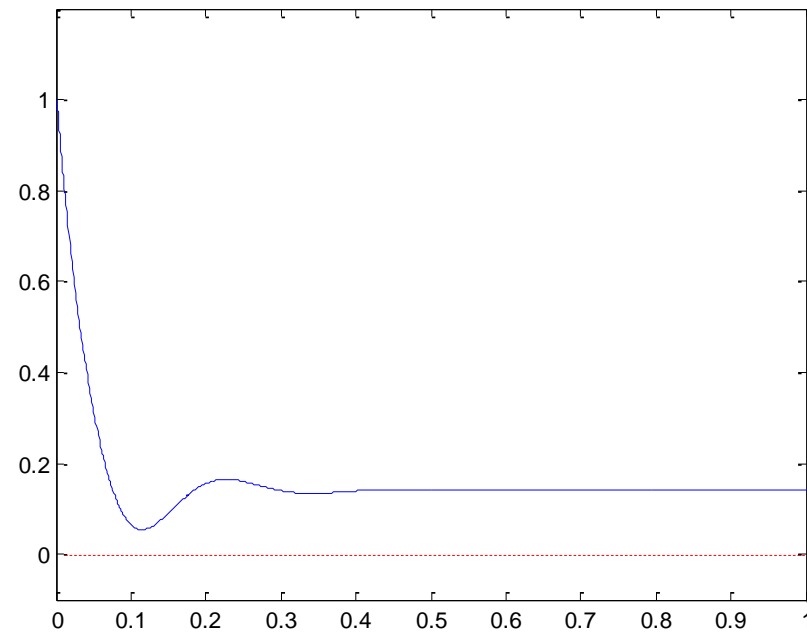
for the error response, initial error values were assumed as

$$e_1 = 1 \text{ (deg)} \quad \dot{e}_1 = 0$$

$$e_2 = 1 \text{ (deg)} \quad \dot{e}_2 = 0$$



< the result of error for θ_1 (deg) >



< the result of error for θ_2 (deg) >

✘ The final steady state error is bigger than initial error.
That is, Only PD controller cannot compensate the error

5. PD Controller Design(8)

■ matlab codes – PD_Control.m

```
function dx = PD_Control(t, x)

%% x - x(1):error1, x(2):error2, x(3):dot_error1, x(4):dot_error2

global m1 m2 L1 L2 g
global Kv Kp
theta = zeros(2,1);
theta(1) = x(1); theta(2) = x(2);
c1 = cos(theta(1)); s1 = sin(theta(1));
c2 = cos(theta(2)); s2 = sin(theta(2));
c12 = cos(theta(1)+theta(2)); s12 = sin(theta(1)+theta(2));

% J matrix
J_11 = (m1*L1^2)/3 + m2*L1^2 + (m2*L2^2)/3 + m2*L1*L2*c2;
J_12 = (m2*L2^2)/3 + (m2*L1*L2*c2)/2;
J_21 = (m2*L2^2)/3 + (m2*L1*L2*c2)/2;
J_22 = (m2*L2^2)/3;
J = [ J_11, J_12;
      J_21, J_22];
```

```
% D matrix
D_11 = 0;
D_12 = -(m2*L1*L2*s2)/2;
D_21 = (m2*L1*L2*s2)/2;
D_22 = 0;
D = [D_11, D_12;
      D_21, D_22]*[x(3)^2; x(4)^2];

% C matrix
C_11 = -m2*L1*L2*s2/2;
C_12 = -m2*L1*L2*s2/2;
C_21 = 0;
C_22 = 0;
C = [C_11, C_12;
      C_21, C_22]*[x(1)*x(2); x(2)*x(1)];

% G matrix
G_1 = (m1/2 + m2)*g*L1*c1 + (m2*g*L2*c12)/2;
G_2 = (m2*g*L2*c12)/2;
G = [G_1;
      G_2];
```

5. PD Controller Design(9)

■ matlab codes – PD_Control.m

```
X = -JWKv*[x(3); x(4)] - JWKp*[x(1); x(2)] + JWG + JWD + JWC;
```

```
dx = zeros(4,1); % A column vector  
dx(1) = x(3);  
dx(2) = x(4);  
dx(3) = X(1);  
dx(4) = X(2);
```

5. PD Controller Design(10)

■ matlab codes – Response_PD_Control.m

```
clear all;
clc

global m1 m2 L1 L2 g
global Kv Kp
m1 = 1; m2 = 1;
L1 = 1; L2 = 1;
g = 9.81;

%% x - x(1):error1, x(2):error2, x(3):dot_error1, x(4):dot_error2
% Initial Conditions
x_0 = [ deg2rad(1); deg2rad(1); 0; 0];
time = [0 1];
Kv = [ 70, 0;
      0, 50];
Kp = [ 2700, 0;
      0, 2000];

[t, x] = ode45(@PD_Control, time, x_0);
```

```
figure(1)
plot(t,rad2deg(x(:,1)))
axis([0,1,-0.1,1.2]);

figure(2)
plot(t,rad2deg(x(:,2)))
axis([0,1,-0.1,1.2]);
```

6. PD + Computed Torques(1)

■ Equations of motion for the system

$$\begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{Bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{Bmatrix} + \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \end{Bmatrix} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1 \dot{\theta}_2 \\ \dot{\theta}_2 \dot{\theta}_1 \end{Bmatrix} + \begin{Bmatrix} G_1 \\ G_2 \end{Bmatrix} = \begin{Bmatrix} T_1 \\ T_2 \end{Bmatrix} \quad \longrightarrow \quad J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1 \dot{\theta}_2) + G = T$$

we can select the input Torques as following

$$T = \alpha T' + \beta \quad \text{where,} \quad \alpha = J = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}$$

$$\beta = D(\dot{\theta}^2) + C(\dot{\theta}_1 \dot{\theta}_2) + G = \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \end{Bmatrix} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1 \dot{\theta}_2 \\ \dot{\theta}_2 \dot{\theta}_1 \end{Bmatrix} + \begin{Bmatrix} G_1 \\ G_2 \end{Bmatrix}$$

then, the E.O.M. can be simplified

$$J\ddot{\theta} + \cancel{D(\dot{\theta}^2)} + \cancel{C(\dot{\theta}_1 \dot{\theta}_2)} + \cancel{G} = \alpha T' + \beta = JT' + \cancel{D(\dot{\theta}^2)} + \cancel{C(\dot{\theta}_1 \dot{\theta}_2)} + \cancel{G}$$

$$J\ddot{\theta} = JT'$$

6. PD + Computed Torques(2)

simplified E.O.M. is $J\ddot{\theta} = JT'$

we choose the input torque T'

$$\ddot{\theta}_d = \begin{Bmatrix} \ddot{\theta}_{1,d} \\ \ddot{\theta}_{2,d} \end{Bmatrix} \quad \text{desired angles}$$

$$e = \begin{Bmatrix} \theta_{1,d} - \theta_1 \\ \theta_{2,d} - \theta_2 \end{Bmatrix} \quad \text{errors between} \\ \text{desired angle and real angle}$$

$$T' = \ddot{\theta}_d + K_v \dot{e} + K_p e$$

where,

$$K_v = \begin{Bmatrix} k_{v11} & k_{v12} \\ k_{v21} & k_{v22} \end{Bmatrix} \quad \begin{array}{l} \text{gain of D.} \\ \text{(for velocity)} \end{array}$$

$$K_p = \begin{Bmatrix} k_{p11} & k_{p12} \\ k_{p21} & k_{p22} \end{Bmatrix} \quad \begin{array}{l} \text{gain of P.} \\ \text{(for position)} \end{array}$$

then,

$$J\ddot{\theta} = J\ddot{\theta}_d + K_v \dot{e} + K_p e \quad \longrightarrow \quad J\ddot{\theta}_d - J\ddot{\theta} + K_v \dot{e} + K_p e = 0$$

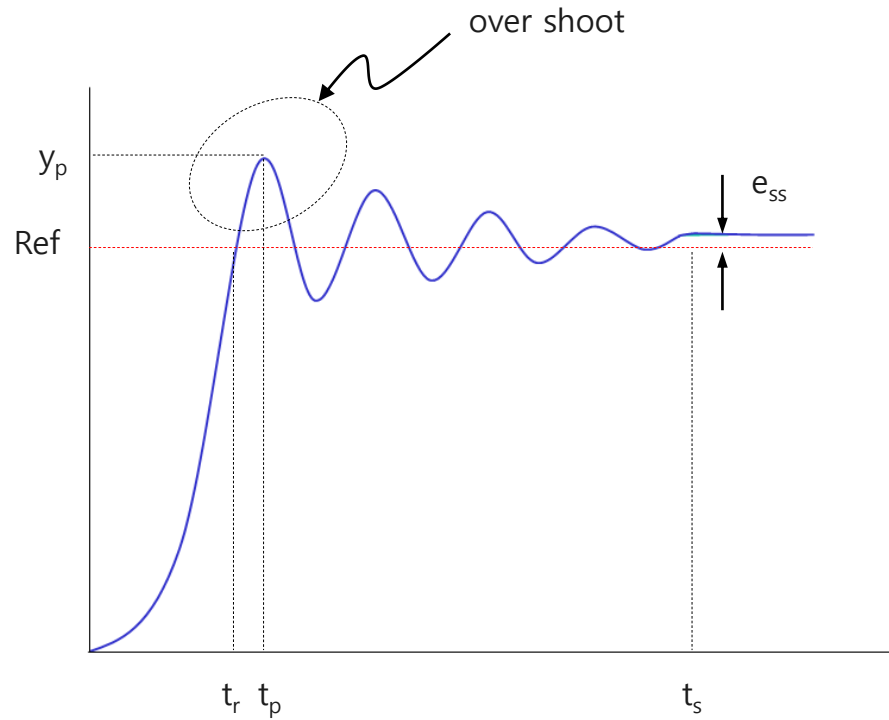
finally, we can get the equations

$$J\ddot{e} + K_v \dot{e} + K_p e = 0$$

$$\ddot{e} = -J^{-1}K_v \dot{e} - J^{-1}K_p e$$

6. PD + Computed Torques(3)

■ System's Performance for 1-dof model



$$P.O.(%) = \left(\frac{y_p - y_{ss}}{y_{ss}} \right) \times 100 = 100 \cdot e^{-\left(\frac{\zeta \pi}{\sqrt{1-\zeta^2}} \right)}$$

$$\zeta = \frac{-\ln(P.O./100)}{\sqrt{\pi^2 + \ln^2(P.O./100)}}$$

$$t_p = \frac{\pi}{\omega_d} = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}}$$

$$y_p = 1 + e^{-\left(\frac{\zeta \pi}{\sqrt{1-\zeta^2}} \right)}$$

$$t_r = \frac{\pi - \theta}{\omega_d}$$

$$\theta = \tan^{-1} \left(\sqrt{1-\zeta^2} / \zeta \right)$$

$$t_s = \frac{4}{\zeta \omega_n}$$

6. PD + Computed Torques(4)

■ Desired Performance of the System

Settling time	t_s	0.3 sec
Percent Overshoot	P.O.(%)	20 %

for θ_1 and θ_2

$$\zeta = \frac{-\ln(P.O./100)}{\sqrt{\pi^2 + \ln^2(P.O./100)}} = \frac{-\ln(10/100)}{\sqrt{\pi^2 + \ln^2(10/100)}} = 0.6523$$

$$\omega_n = \frac{4}{\zeta t_s} = 20.44$$

$$\ddot{e} + 26.6667\dot{e} + 417.7923e = 0$$

then, desired system equations are

$$\ddot{\theta} + 2\zeta\omega_n\dot{\theta} + \omega_n^2\theta = 0 \quad \longrightarrow \quad \ddot{\theta} + 26.6667\dot{\theta} + 417.7923\theta = 0$$

$$\ddot{e} + 2\zeta\omega_n\dot{e} + \omega_n^2e = 0 \quad \longrightarrow \quad \ddot{e} + 26.6667\dot{e} + 417.7923e = 0$$

from the response of system's error

$$\ddot{e} + J^{-1}K_v\dot{e} + J^{-1}K_p e = 0 \quad \text{then, we can compare the coefficient matrices.}$$

6. PD + Computed Torques(5)

■ Determination of PD Gains

$$\ddot{e} + J^{-1}K_v\dot{e} + J^{-1}K_p e = 0$$

$$K_v' = J^{-1}K_v = \begin{bmatrix} k_{v11}' & k_{v12}' \\ k_{v21}' & k_{v22}' \end{bmatrix}$$

$$K_p' = J^{-1}K_p = \begin{bmatrix} k_{p11}' & k_{p12}' \\ k_{p21}' & k_{p22}' \end{bmatrix}$$



$$\ddot{e} + 2\zeta\omega_n\dot{e} + \omega_n^2 e = 0$$

$$K_v' = \begin{bmatrix} 2\zeta\omega_n & 0 \\ 0 & 2\zeta\omega_n \end{bmatrix}$$

$$K_p' = \begin{bmatrix} \omega_n^2 & 0 \\ 0 & \omega_n^2 \end{bmatrix}$$

we just compare diagonal terms and then, computed the gains of controller

$$K_v = J \cdot K_v' = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} 2\zeta\omega_n & 0 \\ 0 & 2\zeta\omega_n \end{bmatrix}$$

$$K_p = J \cdot K_p' = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} \omega_n^2 & 0 \\ 0 & \omega_n^2 \end{bmatrix}$$

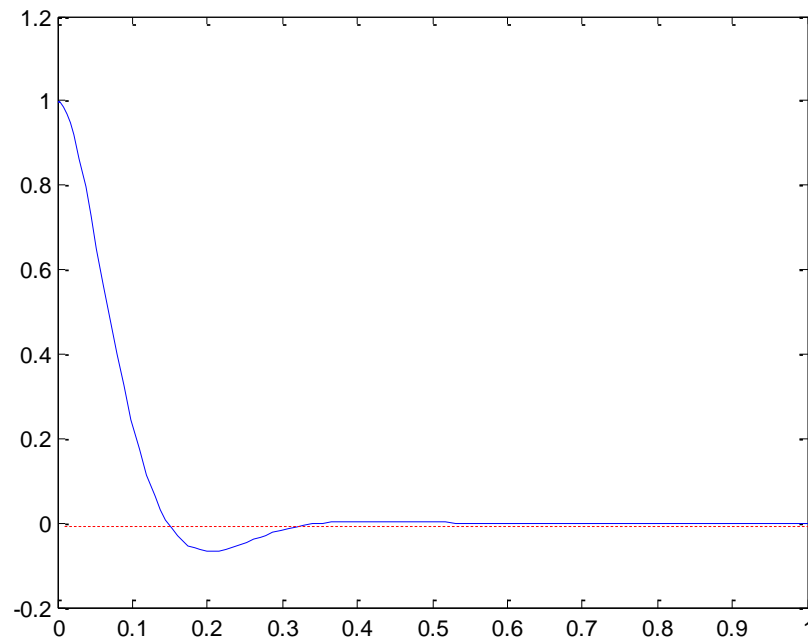
6. PD + Computed Torques(6)

■ Results of simulation for PD+Computed Torques Algorithms

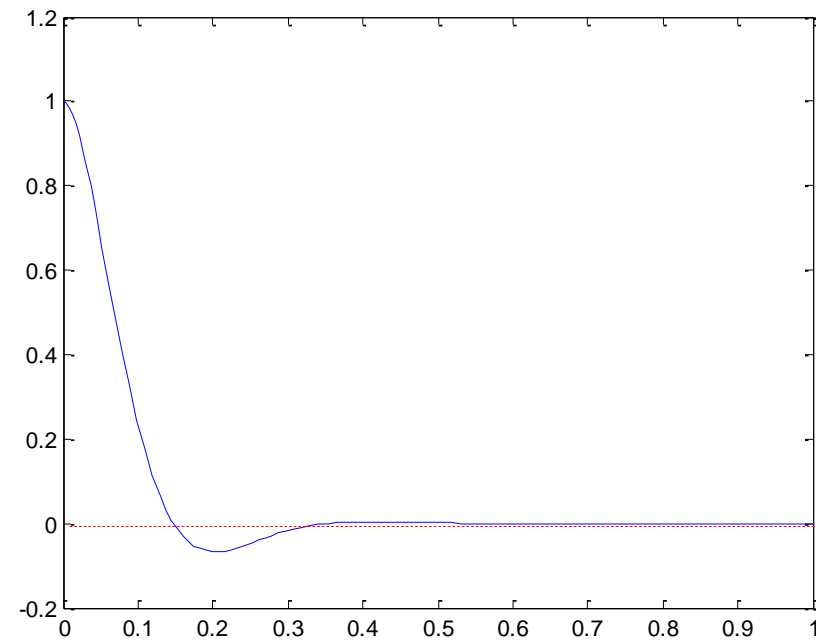
for the error response, initial error values were assumed as

$$e_1 = 1 \text{ (deg)} \quad \dot{e}_1 = 0$$

$$e_2 = 1 \text{ (deg)} \quad \dot{e}_2 = 0$$



< the result of error for θ_1 (deg) >



< the result of error for θ_2 (deg) >

✂ System's behavior meets the desired performance.

6. PD + Computed Torques(7)

■ matlab codes – PD_Computed_Torques.m

```
function dx = PD_Computed_Torques (t, x)

%% x - x(1):error1, x(2):dot_error1, x(3):error2, x(4):dot_error2
global m1 m2 L1 L2 g

theta = zeros(2,1);
theta(1) = x(1); theta(2) = x(3);
c1 = cos(theta(1)); s1 = sin(theta(1));
c2 = cos(theta(2)); s2 = sin(theta(2));
c12 = cos(theta(1)+theta(2)); s12 = sin(theta(1)+theta(2));

J_11 = (m1*L1^2)/3 + m2*L1^2 + (m2*L2^2)/3 + m2*L1*L2*c2;
J_12 = (m2*L2^2)/3 + (m2*L1*L2*c2)/2;
J_21 = (m2*L2^2)/3 + (m2*L1*L2*c2)/2;
J_22 = (m2*L2^2)/3;

dx = zeros(4,1); % a column vector
dx(1) = x(2); dx(3) = x(4);

J = [ J_11, J_12;
      J_21, J_22];
```

```
zeta = 0.6523;
w_n = 20.44

Kv = J*[ 2*zeta*w_n, 0;
         0, 2*zeta*w_n];

Kp = J*[ w_n^2, 0;
         0, w_n^2];

X = -JWKv*[x(2); x(4)] - JWKp*[x(1); x(3)];

dx(2) = X(1);
dx(4) = X(2);
```

6. PD + Computed Torques(8)

■ matlab codes – Response_PD_Computed_Torques.m

```
clear all;
clc

global m1 m2 L1 L2 g

m1 = 1; m2 = 1;
L1 = 1; L2 = 1;
g = 9.81;

%% x – x(1):error1, x(2):dot_error1, x(3):error2, x(4):dot_error2

% Initial Conditions
x_0 = [ deg2rad(1); 0; deg2rad(1); 0];
time = [0 1];
[t, x] = ode45(@PD_Computed_Torques, time, x_0);

figure(1)
plot(t, rad2deg(x(:,1)))

figure(2)
plot(t, rad2deg(x(:,3)))
```

6. PD + Computed Torques with Uncertainty(1)

■ Consideration of uncertainty

we consider uncertainty of mass and inertia terms and check the system's response.

6-cases can be assumed for uncertainty

case ①	$m = m + 0.1m$	$L = L + 0.1 L$
case ②	$m = m - 0.1m$	$L = L - 0.1 L$
case ③	$m = m + 0.2m$	$L = L + 0.2 L$
case ④	$m = m - 0.2m$	$L = L - 0.2 L$
case ⑤	$m = m + 0.3m$	$L = L + 0.3L$
case ⑥	$m = m - 0.3m$	$L = L - 0.3 L$

for the all 6 cases, the gains of P.D. are not changed.

6. PD + Computed Torques with Uncertainty(2)

■ System's dynamics

$$J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1\dot{\theta}_2) + G = T$$

uncertainty
→

J : real system inertia	\tilde{J} : estimated inertia
$D(\dot{\theta}^2)$: real system centrifugal	$\tilde{D}(\dot{\theta}^2)$: estimated centrifugal
$C(\dot{\theta}_1\dot{\theta}_2)$: real system Coriolis	$\tilde{C}(\dot{\theta}_1\dot{\theta}_2)$: estimated Coriolis

$$T = \alpha T' + \beta$$

where,

$$\alpha = \tilde{J} = \begin{bmatrix} \tilde{J}_{11} & \tilde{J}_{12} \\ \tilde{J}_{21} & \tilde{J}_{22} \end{bmatrix}$$

then, the E.O.M. can be simplified

$$\beta = \tilde{D}(\dot{\theta}^2) + \tilde{C}(\dot{\theta}_1\dot{\theta}_2) + \tilde{G} = \begin{bmatrix} \tilde{D}_{11} & \tilde{D}_{12} \\ \tilde{D}_{21} & \tilde{D}_{22} \end{bmatrix} \begin{Bmatrix} \theta_1^2 \\ \theta_2^2 \end{Bmatrix} + \begin{bmatrix} \tilde{C}_{11} & \tilde{C}_{12} \\ \tilde{C}_{21} & \tilde{C}_{22} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1\dot{\theta}_2 \\ \dot{\theta}_2\dot{\theta}_1 \end{Bmatrix} + \begin{Bmatrix} \tilde{G}_1 \\ \tilde{G}_2 \end{Bmatrix}$$

$$J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1\dot{\theta}_2) + G = \alpha T' + \beta = \tilde{J}T' + \tilde{D}(\dot{\theta}^2) + \tilde{C}(\dot{\theta}_1\dot{\theta}_2) + \tilde{G}$$

$$\tilde{J}T' - J\ddot{\theta} + [\tilde{D}(\dot{\theta}^2) - D(\dot{\theta}^2)] + [\tilde{C}(\dot{\theta}_1\dot{\theta}_2) - C(\dot{\theta}_1\dot{\theta}_2)] + [\tilde{G} - G] = 0$$

if we consider that inertia uncertainty can be neglected.

$$T' = \ddot{\theta}_d + K_v \dot{e} + K_p e$$

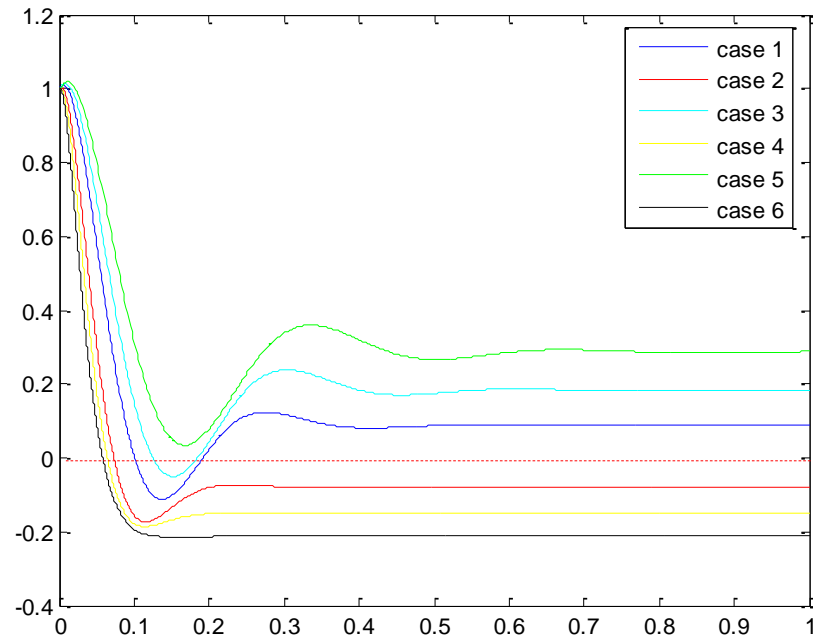
$$\tilde{J}\ddot{e} + K_v \dot{e} + K_p e + [\tilde{D}(\dot{\theta}^2) - D(\dot{\theta}^2)] + [\tilde{C}(\dot{\theta}_1\dot{\theta}_2) - C(\dot{\theta}_1\dot{\theta}_2)] + [\tilde{G} - G] = 0$$

thus, final eqn. of uncertainty system is

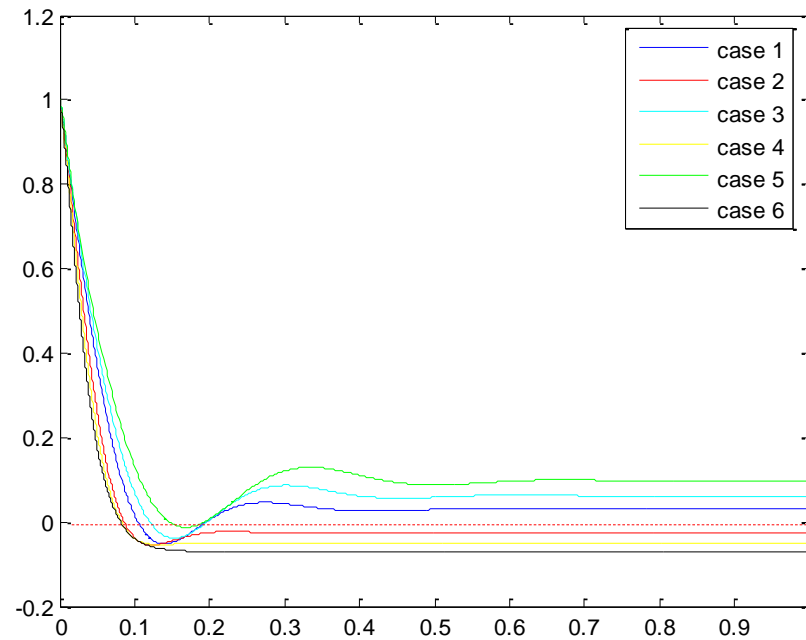
$$\ddot{e}_d + \tilde{J}^{-1}K_v \dot{e} + \tilde{J}^{-1}K_p e + \tilde{J}^{-1}[\tilde{D}(\dot{\theta}^2) - D(\dot{\theta}^2)] + \tilde{J}^{-1}[\tilde{C}(\dot{\theta}_1\dot{\theta}_2) - C(\dot{\theta}_1\dot{\theta}_2)] + \tilde{J}^{-1}[\tilde{G} - G] = 0$$

6. PD + Computed Torques with Uncertainty(3)

■ Results of simulation for PD + Computed Torques Algorithms + Uncertainty



< the result of error for θ_1 (deg) >



< the result of error for θ_2 (deg) >

✂ If parameter uncertainty exists, steady state error cannot be compensated with computed torque algorithm.

6. PD + Computed Torques with Uncertainty(4)

■ matlab codes – PD_Computed_Torques_Uncertainty.m

```
function dx = PD_Computed_Torques_Uncertainty(t, x)
%% x - x(1):error1, x(2):error2, x(3):dot_error1, x(4):dot_error2
global m1 m2 L1 L2 g
global m1_U m2_U L1_U L2_U
global Kv Kp

theta = zeros(2,1);
theta(1) = 0;%x(1);
theta(2) = 0;%x(3);
c1 = cos(theta(1));    s1 = sin(theta(1));
c2 = cos(theta(2));    s2 = sin(theta(2));
c12 = cos(theta(1)+theta(2)); s12 = sin(theta(1)+theta(2));

%% Calculated Parameters %%
J_11 = (m1*L1^2)/3 + m2*L1^2 + (m2*L2^2)/3 + m2*L1*L2*c2;
J_12 = (m2*L2^2)/3 + (m2*L1*L2*c2)/2;
J_21 = (m2*L2^2)/3 + (m2*L1*L2*c2)/2;
J_22 = (m2*L2^2)/3;
D_11 = 0;    D_12 = -(m2*L1*L2*s2)/2;
D_21 = (m2*L1*L2*s2)/2;    D_22 = 0;
C_11 = -(m2*L1*L2*s2)/2;    C_12 = -(m2*L1*L2*s2)/2;
C_21 = 0;    C_22 = 0;
G_1 = (m1/2 + m2)*g*L1*c1 + (m2*g*L2*c12)/2;
G_2 = (m2*g*L2*c12)/2;
```

```
%% Uncertainty included Parameters %%
J_11_U = (m1_U*L1_U^2)/3 + m2_U*L1_U^2 + (m2_U*L2_U^2)/3 + m2_U*L1_U*L2_U*c2;
J_12_U = (m2_U*L2_U^2)/3 + (m2_U*L1_U*L2_U*c2)/2;
J_21_U = J_12_U;
J_22_U = (m2_U*L2_U^2)/3;
D_11_U = 0;    D_12_U = -(m2_U*L1_U*L2_U*s2)/2;
D_21_U = (m2_U*L1_U*L2_U*s2)/2;    D_22_U = 0;
C_11_U = -(m2_U*L1_U*L2_U*s2)/2;    C_12_U = -(m2_U*L1_U*L2_U*s2)/2;
C_21_U = 0;    C_22_U = 0;
G_1_U = (m1_U/2 + m2_U)*g*L1_U*c1 + (m2_U*g*L2_U*c12)/2;
G_2_U = (m2_U*g*L2_U*c12)/2;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
J = [ J_11, J_12;
      J_21, J_22];
D = [ D_11, D_12;
      D_21, D_22]*[x(3)^2; x(4)^2];
C = [ C_11, C_12;
      C_21, C_22]*[x(1)*x(2); x(2)*x(1)];
G = [ G_1; G_2];

J_U = [ J_11_U, J_12_U;
        J_21_U, J_22_U];
```

6. PD + Computed Torques with Uncertainty(5)

■ matlab codes – PD_Computed_Torques_Uncertainty.m

```
D_U = [ D_11_U, D_12_U;  
        D_21_U, D_22_U]*[x(3)^2; x(4)^2];  
  
C_U = [ C_11_U, C_12_U;  
        C_21_U, C_22_U]*[x(1)*x(2); x(2)*x(1)];  
  
G_U = [ G_1_U; G_2_U];  
  
X = -J_UWKv*[x(3); x(4)] - J_UWKp*[x(1); x(2)]-J_UW(D-D_U)...  
    - J_UW(C-C_U) - J_UW(G-G_U);  
  
dx = zeros(4,1); % a column vector  
  
dx(1) = x(3);  
dx(2) = x(4);  
dx(3) = X(1);  
dx(4) = X(2);
```


6. PD + Computed Torques with Uncertainty(6)

■ matlab codes – Response_Comparison_Uncertainty_PD_Computed_Torques.m

```
clear all;
clc

global m1 m2 L1 L2 g
global m1_U m2_U L1_U L2_U
global Kv Kp

% Calculated Parameter
m1 = 1; m2 = 1; L1 = 1; L2 = 1; g = 9.81;

%% x - x(1):error1, x(2):dot_error1, x(3):error2, x(4):dot_error2
% Initial Conditions
x_0 = [ deg2rad(1); deg2rad(1); 0; 0];
time = [0 1];

Kv = [ 100, 0;
      0, 70];
Kp = [ 2700, 0;
      0, 2000];

% for case1의 경우
m1_U = 1 + 0.1; m2_U = 1 + 0.1;
L1_U = 1 + 0.1; L2_U = 1 + 0.1;
[t1, x_case1] = ode45(@PD_Computed_Torques_Uncertainty, time, x_0);
```

```
% for case2의 경우
m1_U = 1 - 0.1; m2_U = 1 - 0.1;
L1_U = 1 - 0.1; L2_U = 1 - 0.1;
[t2, x_case2] = ode45(@PD_Computed_Torques_Uncertainty, time, x_0);

% for case3의 경우
m1_U = 1 + 0.2; m2_U = 1 + 0.2;
L1_U = 1 + 0.2; L2_U = 1 + 0.2;

[t3, x_case3] = ode45(@PD_Computed_Torques_Uncertainty, time, x_0);

% for case4의 경우
m1_U = 1 - 0.2; m2_U = 1 - 0.2;
L1_U = 1 - 0.2; L2_U = 1 - 0.2;
g = 9.81;
[t4, x_case4] = ode45(@PD_Computed_Torques_Uncertainty, time, x_0);

% for case5의 경우
m1_U = 1 + 0.3; m2_U = 1 + 0.3;
L1_U = 1 + 0.3; L2_U = 1 + 0.3;
[t5, x_case5] = ode45(@PD_Computed_Torques_Uncertainty, time, x_0);
```

6. PD + Computed Torques with Uncertainty(7)

■ matlab codes – Response_Comparison_Uncertainty_PD_Computed_Torques.m

```
% for case6의 경우
m1_U = 1 - 0.3; m2_U = 1 - 0.3;
L1_U = 1 - 0.3; L2_U = 1 - 0.3;
[t6, x_case6] = ode45(@PD_Computed_Torques_Uncertainty, time, x_0);

figure(1)
plot(t1,rad2deg(x_case1(:,1)), 'b', t2,rad2deg(x_case2(:,1)), 'r'...
     , t3,rad2deg(x_case3(:,1)), 'c', t4,rad2deg(x_case4(:,1)), 'y'...
     , t5,rad2deg(x_case5(:,1)), 'g', t6,rad2deg(x_case6(:,1)), 'k')
legend('case 1', 'case 2', 'case 3', 'case 4', 'case 5', 'case 6');

figure(2)
plot(t1,rad2deg(x_case1(:,2)), 'b', t2,rad2deg(x_case2(:,2)), 'r'...
     , t3,rad2deg(x_case3(:,2)), 'c', t4,rad2deg(x_case4(:,2)), 'y'...
     , t5,rad2deg(x_case5(:,2)), 'g', t6,rad2deg(x_case6(:,2)), 'k')
legend('case 1', 'case 2', 'case 3', 'case 4', 'case 5', 'case 6');
```

6. Adaptive Control(1)

■ Equations of motion for the system

When the system parameter have uncertainty, Computed Torque algorithm cannot compensate the error completely.

Adaptive Computed Torque Control shows how to compensate the error as follows

$$J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1\dot{\theta}_2) + G = T \quad \text{by computed Torques method} \quad T = \alpha T' + \beta \quad \alpha = \tilde{J}$$

where, $\beta = \tilde{D}(\dot{\theta}^2) + \tilde{C}(\dot{\theta}_1\dot{\theta}_2) + \tilde{G}$

J : real system inertia \tilde{J} : estimated inertia

$D(\dot{\theta}^2)$: real system centrifugal $\tilde{D}(\dot{\theta}^2)$: estimated centrifugal

$C(\dot{\theta}_1\dot{\theta}_2)$: real system Coriolis $\tilde{C}(\dot{\theta}_1\dot{\theta}_2)$: estimated Coriolis

then, the E.O.M. can be simplified

$$J\ddot{\theta} + D(\dot{\theta}^2) + C(\dot{\theta}_1\dot{\theta}_2) + G = \alpha T' + \beta = \tilde{J}T' + \tilde{D}(\dot{\theta}^2) + \tilde{C}(\dot{\theta}_1\dot{\theta}_2) + \tilde{G}$$

$$T' = \ddot{\theta}_d + K_v\dot{e} + K_p e$$

thus, final eqn. of uncertainty system is

$$\ddot{e}_d + \tilde{J}^{-1}K_v\dot{e} + \tilde{J}^{-1}K_p e + \tilde{J}^{-1}[\tilde{D}(\dot{\theta}^2) - D(\dot{\theta}^2)] + \tilde{J}^{-1}[\tilde{C}(\dot{\theta}_1\dot{\theta}_2) - C(\dot{\theta}_1\dot{\theta}_2)] + \tilde{J}^{-1}[\tilde{G} - G] = 0$$

6. Adaptive Control(2)

the eqn. can be modified

$$\ddot{e}_d + \tilde{J}^{-1} K_v \dot{e} + \tilde{J}^{-1} K_p e = -\tilde{J}^{-1} \left\{ \left[\tilde{D}(\dot{\theta}^2) - D(\dot{\theta}^2) \right] + \left[\tilde{C}(\dot{\theta}_1 \dot{\theta}_2) - C(\dot{\theta}_1 \dot{\theta}_2) \right] + \left[\tilde{G} - G \right] \right\}$$



$$\ddot{e}_d + K_v \dot{e} + K_p e = \tilde{J}^{-1} \cdot W(\theta, \dot{\theta}, \ddot{\theta}) \cdot \tilde{\varphi} \quad \text{where,} \quad \tilde{\varphi} = \begin{bmatrix} \tilde{\varphi}_1 \\ \tilde{\varphi}_2 \end{bmatrix} = \begin{bmatrix} m_1 - \tilde{m}_1 \\ m_2 - \tilde{m}_2 \end{bmatrix}$$

then, $W(\cdot)$ is defined as

$$\begin{aligned} W(\theta, \dot{\theta}, \ddot{\theta}) \cdot \tilde{\varphi} &= -\left[\tilde{D}(\dot{\theta}^2) - D(\dot{\theta}^2) \right] - \left[\tilde{C}(\dot{\theta}_1 \dot{\theta}_2) - C(\dot{\theta}_1 \dot{\theta}_2) \right] - \left[\tilde{G} - G \right] \\ &= \left[D(\dot{\theta}^2) - \tilde{D}(\dot{\theta}^2) \right] + \left[C(\dot{\theta}_1 \dot{\theta}_2) - \tilde{C}(\dot{\theta}_1 \dot{\theta}_2) \right] + \left[G - \tilde{G} \right] \end{aligned}$$

here, for matrix D

$$D(\dot{\theta}^2) - \tilde{D}(\dot{\theta}^2) = \begin{bmatrix} \left(-\frac{1}{2} m_2 L_1 L_2 s_2 \right) \dot{\theta}_2^2 \\ \left(\frac{1}{2} m_2 L_1 L_2 s_2 \right) \dot{\theta}_1^2 \end{bmatrix} - \begin{bmatrix} \left(-\frac{1}{2} \tilde{m}_2 L_1 L_2 s_2 \right) \dot{\theta}_2^2 \\ \left(\frac{1}{2} \tilde{m}_2 L_1 L_2 s_2 \right) \dot{\theta}_1^2 \end{bmatrix} = \begin{bmatrix} \left(-\frac{1}{2} L_1 L_2 s_2 \dot{\theta}_2^2 \right) (m_2 - \tilde{m}_2) \\ \left(\frac{1}{2} L_1 L_2 s_2 \dot{\theta}_1^2 \right) (m_2 - \tilde{m}_2) \end{bmatrix} = \begin{bmatrix} 0 & \left(-\frac{1}{2} L_1 L_2 s_2 \right) \dot{\theta}_2^2 \\ 0 & \left(\frac{1}{2} L_1 L_2 s_2 \right) \dot{\theta}_1^2 \end{bmatrix} \begin{bmatrix} m_1 - \tilde{m}_1 \\ m_2 - \tilde{m}_2 \end{bmatrix}$$

6. Adaptive Control(3)

for matrix C

$$C(\dot{\theta}_1, \dot{\theta}_2) - \tilde{C}(\dot{\theta}_1, \dot{\theta}_2) = \begin{bmatrix} -m_2 L_1 L_2 s_2 \dot{\theta}_1 \dot{\theta}_2 \\ 0 \end{bmatrix} - \begin{bmatrix} -\tilde{m}_2 L_1 L_2 s_2 \dot{\theta}_1 \dot{\theta}_2 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & -L_1 L_2 s_2 \dot{\theta}_1 \dot{\theta}_2 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} m_1 - \tilde{m}_1 \\ m_2 - \tilde{m}_2 \end{bmatrix}$$

for matrix G

$$\begin{aligned} G(\theta_1, \theta_2) - \tilde{G}(\theta_1, \theta_2) &= \begin{bmatrix} \left(\frac{1}{2} m_1 + m_2 \right) g L_1 c_1 + \frac{1}{2} m_2 g L_2 c_{12} \\ \frac{1}{2} m_2 g L_2 c_{12} \end{bmatrix} - \begin{bmatrix} \left(\frac{1}{2} \tilde{m}_1 + \tilde{m}_2 \right) g L_1 c_1 + \frac{1}{2} \tilde{m}_2 g L_2 c_{12} \\ \frac{1}{2} \tilde{m}_2 g L_2 c_{12} \end{bmatrix} \\ &= \begin{bmatrix} \left(\frac{1}{2} (m_1 - \tilde{m}_1) + (m_2 - \tilde{m}_2) \right) g L_1 c_1 + \frac{1}{2} (m_2 - \tilde{m}_2) g L_2 c_{12} \\ \frac{1}{2} (m_2 - \tilde{m}_2) g L_2 c_{12} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} g L_1 c_1 & \frac{1}{2} g L_1 c_1 + \frac{1}{2} g L_2 c_{12} \\ 0 & \frac{1}{2} g L_2 c_{12} \end{bmatrix} \begin{bmatrix} m_1 - \tilde{m}_1 \\ m_2 - \tilde{m}_2 \end{bmatrix} \end{aligned}$$

6. Adaptive Control(4)

then, matrix $W(\cdot)$ is

$$W(\theta, \dot{\theta}, \ddot{\theta}) \cdot \tilde{\varphi} = [D(\dot{\theta}^2) - \tilde{D}(\dot{\theta}^2)] + [C(\dot{\theta}_1 \dot{\theta}_2) - \tilde{C}(\dot{\theta}_1 \dot{\theta}_2)] + [G - \tilde{G}]$$

$$W(\theta, \dot{\theta}, \ddot{\theta}) = \begin{bmatrix} 0 & \left(-\frac{1}{2} L_1 L_2 s_2\right) \dot{\theta}_2^2 \\ 0 & \left(\frac{1}{2} L_1 L_2 s_2\right) \dot{\theta}_1^2 \end{bmatrix} + \begin{bmatrix} 0 & -L_1 L_2 s_2 \dot{\theta}_1 \dot{\theta}_2 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} \frac{1}{2} g L_1 c_1 & \frac{1}{2} g L_1 c_1 + \frac{1}{2} g L_2 c_{12} \\ 0 & \frac{1}{2} g L_2 c_{12} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{2} g L_1 c_1 & \left(-\frac{1}{2} L_1 L_2 s_2\right) \dot{\theta}_2^2 - L_1 L_2 s_2 \dot{\theta}_1 \dot{\theta}_2 + \frac{1}{2} g L_1 c_1 + \frac{1}{2} g L_2 c_{12} \\ 0 & \left(\frac{1}{2} L_1 L_2 s_2\right) \dot{\theta}_1^2 + \frac{1}{2} g L_2 c_{12} \end{bmatrix}$$

here,

$$\ddot{e}_d + K_v \dot{e} + K_p e = \tilde{J}^{-1} \cdot W(\theta, \dot{\theta}, \ddot{\theta}) \cdot \tilde{\varphi} \longrightarrow \tilde{J}^{-1} \cdot W(\theta, \dot{\theta}, \ddot{\theta}) \cdot \tilde{\varphi} \quad \text{is the reason of steady state error.}$$

So, key problem is how to remove the $W(\cdot) \varphi$ term. We can use information of error in order to remove $W(\cdot) \varphi$.

If we estimate values of mass well, the error should be removed.

6. Adaptive Control(5)

■ Estimation of mass

Information of mass can be updated and estimated by error information every loop.
That is adaptive control's concept.

from the error dynamics

$$\ddot{e}_d + K_v' \dot{e} + K_p' e = \tilde{J}^{-1} \cdot W(\theta, \dot{\theta}, \ddot{\theta}) \cdot \tilde{\varphi} \quad \text{where,} \quad e = \begin{bmatrix} e \\ \dot{e} \end{bmatrix} \quad A = \begin{bmatrix} O_2 & I_2 \\ -K_p' & -K_v' \end{bmatrix}$$

rewrite above eqn. in the state-space form.


$$\dot{e} = Ae + B\tilde{J}^{-1} \cdot W \cdot \tilde{\varphi} \quad B = \begin{bmatrix} O_2 \\ I_2 \end{bmatrix}$$

here, we introduce Lyapunov-like function in order to find appropriate update rule for mass estimation.

$$V = e^T P e + \tilde{\varphi}^T \Gamma^{-1} \tilde{\varphi} \quad \text{where,} \quad \Gamma = \text{diag}(\gamma_1, \gamma_2)$$

$$\left[\dot{\tilde{\varphi}}^T \Gamma^{-1} \tilde{\varphi} \right]^T = \tilde{\varphi}^T \Gamma^{-1} \dot{\tilde{\varphi}} \quad \text{and} \quad \Gamma^T = \Gamma$$

differentiation

$$\dot{V} = \dot{e}^T P e + e^T P \dot{e} + \dot{\tilde{\varphi}}^T \Gamma^{-1} \tilde{\varphi} + \tilde{\varphi}^T \Gamma^{-1} \dot{\tilde{\varphi}} = \dot{e}^T P e + e^T P \dot{e} + 2\tilde{\varphi}^T \Gamma^{-1} \dot{\tilde{\varphi}}$$


6. Adaptive Control(6)

substitute $\dot{e} = A\dot{e} + B\tilde{J}^{-1} \cdot W \cdot \tilde{\phi}$

$$\begin{aligned}
 \dot{V} &= \dot{e}^T P e + e^T P \dot{e} + 2\tilde{\phi}^T \Gamma^{-1} \dot{\tilde{\phi}} \\
 &= \left(A\dot{e} + B\tilde{J}^{-1} \cdot W \cdot \tilde{\phi} \right)^T P e + e^T P \left(A\dot{e} + B\tilde{J}^{-1} \cdot W \cdot \tilde{\phi} \right) + 2\tilde{\phi}^T \Gamma^{-1} \dot{\tilde{\phi}} \\
 &= -e^T (A^T P + P A) e + 2\tilde{\phi}^T \left(\Gamma^{-1} \dot{\tilde{\phi}} + W^T \tilde{J}^{-1} B^T P e \right) \\
 &= -e^T Q e + \underbrace{2\tilde{\phi}^T \left(\Gamma^{-1} \dot{\tilde{\phi}} + W^T \tilde{J}^{-1} B^T P e \right)}
 \end{aligned}$$

where, $-Q = A^T P + P A$

Q should be positive definite, symmetric matrix to satisfy the Lyapunov equation.

For stability, it is always desirable to have \dot{V} at least negative semidefinite.

Therefore, the choice of adaptation update rule becomes obvious.

$$\dot{\tilde{\phi}} = -\Gamma W^T \tilde{J}^{-1} B^T P e \quad \text{then, we can cancel the last term of above equation, } \dot{V}$$

$$\Gamma^{-1} \dot{\tilde{\phi}} + W^T \tilde{J}^{-1} B^T P e = 0 \quad \longrightarrow \quad \dot{V} = -e^T Q e \leq 0 \quad \because \text{matrix Q is positive definite.}$$

6. Adaptive Control(7)

▣ Adaptive Update Rule

$$\dot{\tilde{\phi}} = -\Gamma W^T \tilde{J}^{-1} B^T P e$$

In order to find update rule, we have to calculate every matrices as follow

$$K_v' = \begin{bmatrix} k_v' & 0 \\ 0 & k_v' \end{bmatrix}$$

$$K_p' = \begin{bmatrix} k_p' & 0 \\ 0 & k_p' \end{bmatrix}$$

here, P is one of candidates

$$P = \begin{bmatrix} P_1 I_2 & P_2 I_2 \\ P_2 I_2 & P_3 I_2 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} (k_p' + 1/2 k_v') I_2 & 1/2 I_2 \\ 1/2 I_2 & I_2 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} k_p' + 1/2 k_v' & 0 & 1/2 & 0 \\ 0 & k_p' + 1/2 k_v' & 0 & 1/2 \\ 1/2 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1 \end{bmatrix}$$

$$Q = -(A^T P + P A) = \begin{bmatrix} 1/2 k_p' I_2 & 0_2 \\ 0_2 & (k_v' - 1/2) I_2 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1/2 k_p' & 0 & 0 & 0 \\ 0 & 1/2 k_p' & 0 & 0 \\ 0 & 0 & k_v' - 1/2 & 0 \\ 0 & 0 & 0 & k_v' - 1/2 \end{bmatrix}$$

$$\Gamma = \begin{bmatrix} \gamma_1 & 0 \\ 0 & \gamma_2 \end{bmatrix}$$

$$B = \begin{bmatrix} 0_2 \\ I_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

6. Adaptive Control(8)

$$\dot{\tilde{\phi}} = -\Gamma W^T \tilde{J}^{-1} B^T P e \quad \longrightarrow \quad \tilde{\phi}_{new} = \tilde{\phi}_{old} + \dot{\tilde{\phi}} \cdot \Delta t$$

$$\begin{bmatrix} m_1 - \tilde{m}_1 \\ m_2 - \tilde{m}_2 \end{bmatrix}_{new} = \begin{bmatrix} m_1 - \tilde{m}_1 \\ m_2 - \tilde{m}_2 \end{bmatrix}_{old} + \begin{bmatrix} \dot{\tilde{\phi}}_1 \\ \dot{\tilde{\phi}}_2 \end{bmatrix} \Delta t \quad \text{because} \quad \begin{bmatrix} m_1 \\ m_2 \end{bmatrix}_{new} = \begin{bmatrix} m_1 \\ m_2 \end{bmatrix}_{old}$$

then,

$$\begin{bmatrix} \tilde{m}_1 \\ \tilde{m}_2 \end{bmatrix}_{new} = \begin{bmatrix} \tilde{m}_1 \\ \tilde{m}_2 \end{bmatrix}_{old} - \begin{bmatrix} \dot{\tilde{\phi}}_1 \\ \dot{\tilde{\phi}}_2 \end{bmatrix} \Delta t$$

6. Adaptive Control(9)

■ Consideration of uncertainty

To verify performance of Adaptive Control, we consider uncertainty of mass terms and check the system's response.

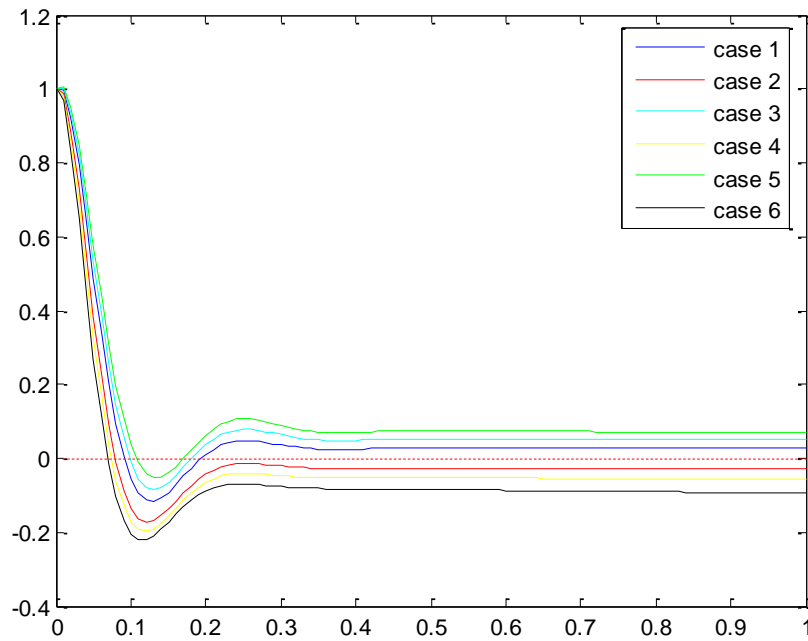
6-cases can be assumed for uncertainty

case ①	$m = m + 0.1m$
case ②	$m = m - 0.1m$
case ③	$m = m + 0.2m$
case ④	$m = m - 0.2m$
case ⑤	$m = m + 0.3m$
case ⑥	$m = m - 0.3m$

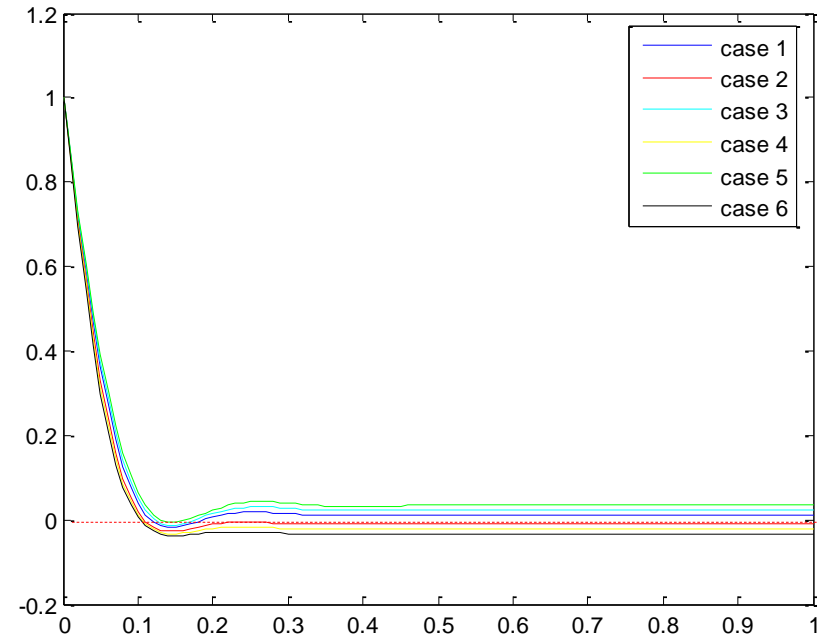
for the all 6 cases, the gains of P.D. didn't changed.

6. Adaptive Control(10)

■ Results of simulation for Adaptive Control with Uncertainty



< the result of error for θ_1 (deg) >



< the result of error for θ_2 (deg) >

✂ We can find that the errors come close to zero.

6. Adaptive Control(11)

■ matlab codes – Adpative_Control_Uncertainty.m

```
function dx = Adpative_Control_Uncertainty(t, x)
%% x - x(1):error1, x(2):error2, x(3):dot_error1, x(4):dot_error2
global m1 m2 L1 L2 g
global m1_U m2_U L1_U L2_U
global dot_PHI Kv Kp

m1 = m1 - dot_PHI(1)*0.001;
m2 = m2 - dot_PHI(2)*0.001;

theta = zeros(2,1);
theta(1) = x(1); theta(2) = x(3);
c1 = cos(theta(1)); s1 = sin(theta(1));
c2 = cos(theta(2)); s2 = sin(theta(2));
c12 = cos(theta(1)+theta(2)); s12 = sin(theta(1)+theta(2));

%% Calculated Parameters %%
J_11 = (m1*L1^2)/3 + m2*L1^2 + (m2*L2^2)/3 + m2*L1*L2;
J_12 = (m2*L2^2)/3 + (m2*L1*L2)/2;
J_21 = (m2*L2^2)/3 + (m2*L1*L2)/2;
J_22 = (m2*L2^2)/3;

D_11 = 0; D_12 = -(m2*L1*L2*s2)/2;
D_21 = (m2*L1*L2*s2)/2; D_22 = 0;
```

```
C_11 = -(m2*L1*L2*s2)/2; C_12 = -(m2*L1*L2*s2)/2;
C_21 = 0; C_22 = 0;
```

```
G_1 = (m1/2 + m2)*g*L1*c1 + (m2*g*L2*c12)/2;
G_2 = (m2*g*L2*c12)/2;
```

```
%% Uncertainty included Parameters %%
```

```
J_11_U = (m1_U*L1_U^2)/3 + m2_U*L1_U^2 + (m2_U*L2_U^2)/3 + m2_U*L1_U*L2_U;
J_12_U = (m2_U*L2_U^2)/3 + (m2_U*L1_U*L2_U)/2;
J_21_U = J_12_U;
J_22_U = (m2_U*L2_U^2)/3;
```

```
D_11_U = 0; D_12_U = -(m2_U*L1_U*L2_U*s2)/2;
D_21_U = (m2_U*L1_U*L2_U*s2)/2; D_22_U = 0;
```

```
C_11_U = -(m2_U*L1_U*L2_U*s2)/2; C_12_U = -(m2_U*L1_U*L2_U*s2)/2;
C_21_U = 0; C_22_U = 0;
```

```
G_1_U = (m1_U/2 + m2_U)*g*L1_U*c1 + (m2_U*g*L2_U*c12)/2;
G_2_U = (m2_U*g*L2_U*c12)/2;
```

6. Adaptive Control(12)

■ matlab codes – Adpative_Control_Uncertainty.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
J = [ J_11, J_12;
      J_21, J_22];
D = [ D_11, D_12;
      D_21, D_22]*[x(2)^2; x(4)^2];
C = [ C_11, C_12;
      C_21, C_22]*[x(1)*x(3); x(3)*x(1)];
G = [ G_1; G_2];

J_U = [ J_11_U, J_12_U;
        J_21_U, J_22_U];
D_U = [ D_11_U, D_12_U;
        D_21_U, D_22_U]*[x(2)^2; x(4)^2];
C_U = [ C_11_U, C_12_U;
        C_21_U, C_22_U]*[x(1)*x(3); x(3)*x(1)];
G_U = [ G_1_U; G_2_U];

Gamma = [ 40; 0;
          0; 40];
B = [ zeros(2,2); eye(2)];

P = 0.5*[ (Kp(1,1)+0.5*Kv(1,1))*eye(2), 0.5*eye(2);
          0.5*eye(2), eye(2) ];
```

```
error    = [x(1); x(2)];
dot_error = [ x(3); x(4) ];

e = [ error;
      dot_error];

W = (D_U-D) + (C_U-C) + (G_U-G);

dot_PHI = -Gamma*(W')*inv(J)*(B')*P*e;

X = -J_UWKv*dot_error - J_UWKp*error - J_UW(D-D_U)...
    - J_UW(C-C_U) - J_UW(G-G_U);

dx = zeros(4,1); % a column vector

dx(1) = x(3);
dx(2) = x(4);
dx(3) = X(1);
dx(4) = X(2);
```

6. Adaptive Control(13)

■ matlab codes – Response_Adaptive_Control_Uncertainty.m

```
clear all;
clc

global m1 m2 L1 L2 g
global m1_U m2_U L1_U L2_U
global dot_PHI Kv Kp

% Calculated Parameter
m1 = 1; m2 = 1; L1 = 1; L2 = 1; g = 9.81;

L1_U = L1; L2_U = L2;

dot_PHI = zeros(2,1);

%% x - x(1):error1, x(2):error2, x(3):dot_error1, x(4):dot_error2

% Initial Conditions
x_0 = [ deg2rad(1); deg2rad(1); 0; 0];
time = [0:0.01:1];

Kv = [ 90, 0;
       0, 80];
Kp = [ 2700, 0;
       0, 2500];
```

```
% for case1의 경우
m1_U = 1 + 0.1; m2_U = 1 + 0.1;
[t1, x_case1] = ode45(@Adpative_Control_Uncertainty, time, x_0);

% for case2의 경우
m1_U = 1 - 0.1; m2_U = 1 - 0.1;
[t2, x_case2] = ode45(@Adpative_Control_Uncertainty, time, x_0);

% for case3의 경우
m1_U = 1 + 0.2; m2_U = 1 + 0.2;
[t3, x_case3] = ode45(@Adpative_Control_Uncertainty, time, x_0);

% for case4의 경우
m1_U = 1 - 0.2; m2_U = 1 - 0.2;
g = 9.81;
[t4, x_case4] = ode45(@Adpative_Control_Uncertainty, time, x_0);

% for case5의 경우
m1_U = 1 + 0.3; m2_U = 1 + 0.3;
[t5, x_case5] = ode45(@Adpative_Control_Uncertainty, time, x_0);

% for case6의 경우
m1_U = 1 - 0.3; m2_U = 1 - 0.3;
[t6, x_case6] = ode45(@Adpative_Control_Uncertainty, time, x_0);
```

6. Adaptive Control(14)

■ matlab codes – Response_Adaptive_Control_Uncertainty.m

```
figure(1)
plot(t1,rad2deg(x_case1(:,1)), 'b', t2,rad2deg(x_case2(:,1)), 'r'...
    , t3,rad2deg(x_case3(:,1)), 'c', t4,rad2deg(x_case4(:,1)), 'y'...
    , t5,rad2deg(x_case5(:,1)), 'g', t6,rad2deg(x_case6(:,1)), 'k')
legend('case 1', 'case 2', 'case 3', 'case 4', 'case 5', 'case 6');

figure(2)
plot(t1,rad2deg(x_case1(:,2)), 'b', t2,rad2deg(x_case2(:,2)), 'r'...
    , t3,rad2deg(x_case3(:,2)), 'c', t4,rad2deg(x_case4(:,2)), 'y'...
    , t5,rad2deg(x_case5(:,2)), 'g', t6,rad2deg(x_case6(:,2)), 'k')
legend('case 1', 'case 2', 'case 3', 'case 4', 'case 5', 'case 6');
```


7. Comparison of Control Algorithms(1)

■ Consideration of uncertainty

To compare performance of PD, PD+Computed Torque and Adaptive Control, we consider uncertainty of mass terms and check the system's response.

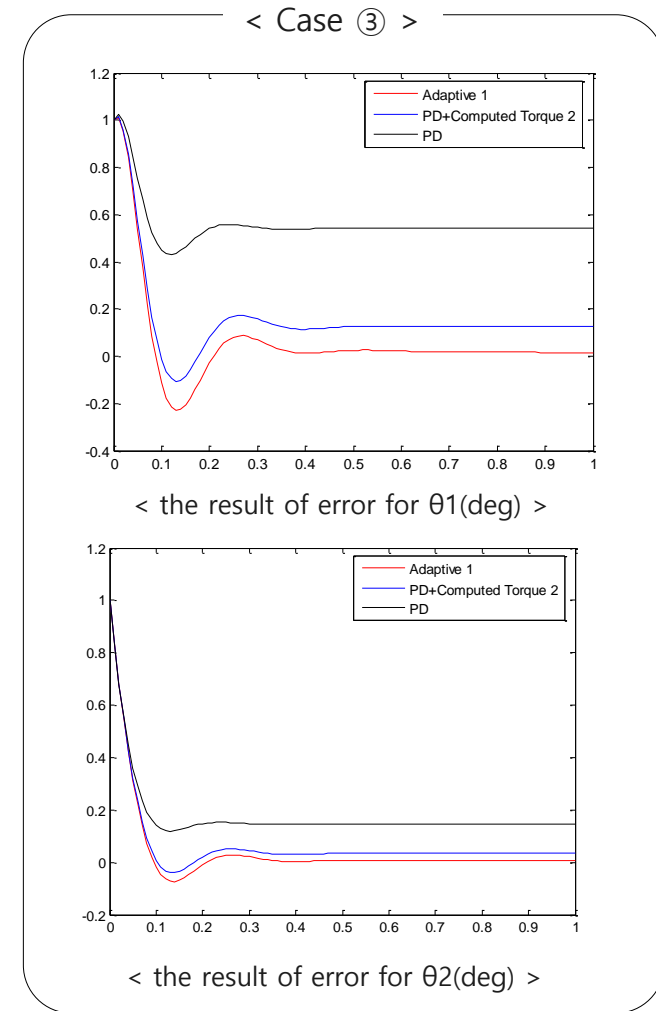
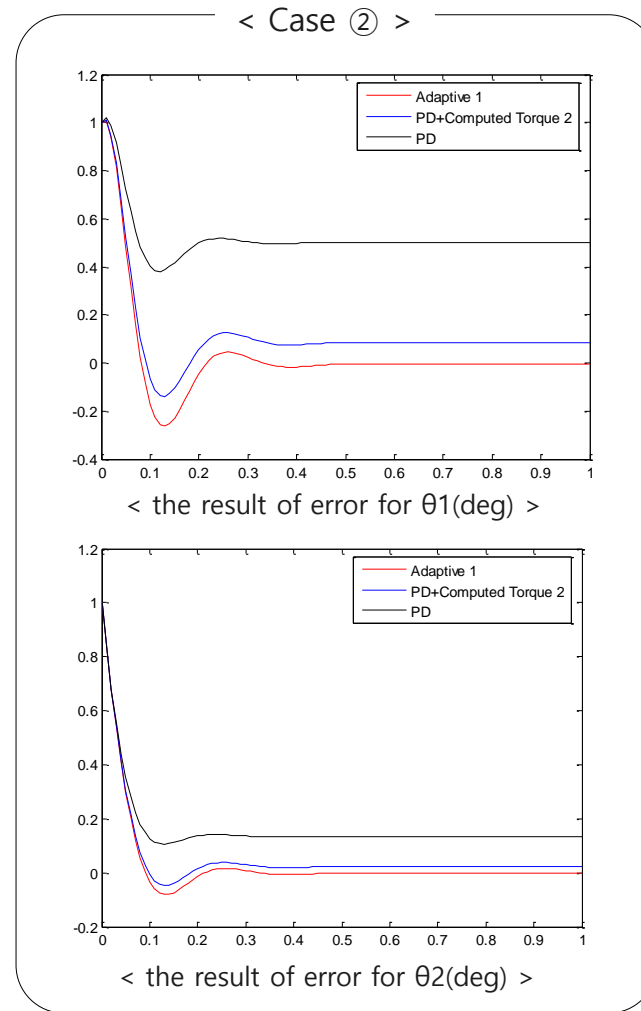
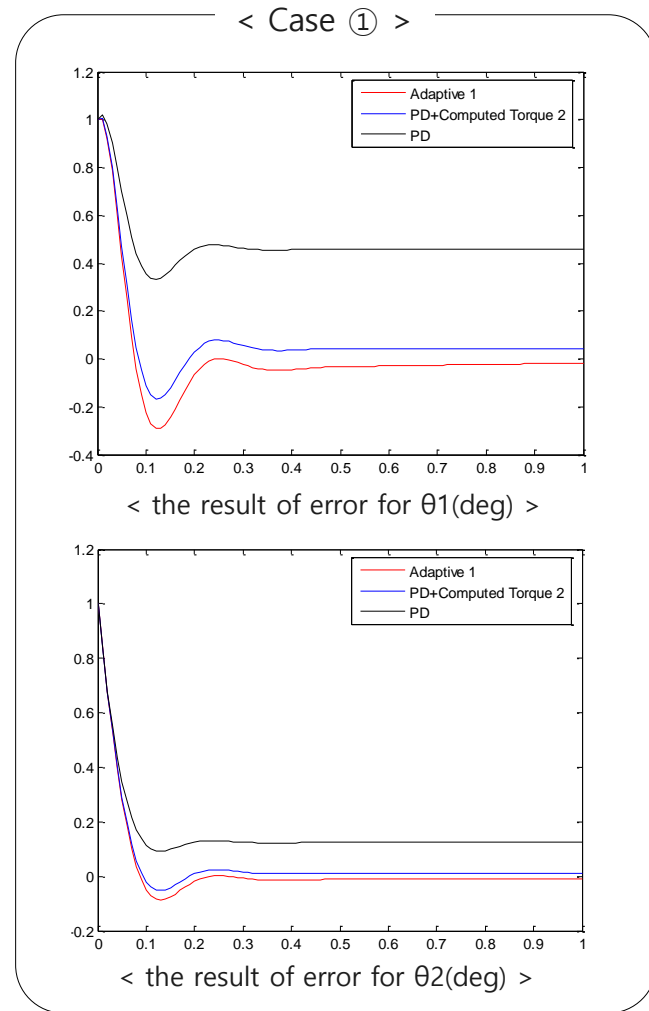
3-cases can be assumed for uncertainty

case ①	$m = m + 0.1m$
case ②	$m = m + 0.2m$
case ③	$m = m + 0.3m$

for the all 6 cases, the gains of P.D. didn't changed.

7. Comparison of Control Algorithms(2)

■ Comparison of simulation result



7. Comparison of Control Algorithms(3)

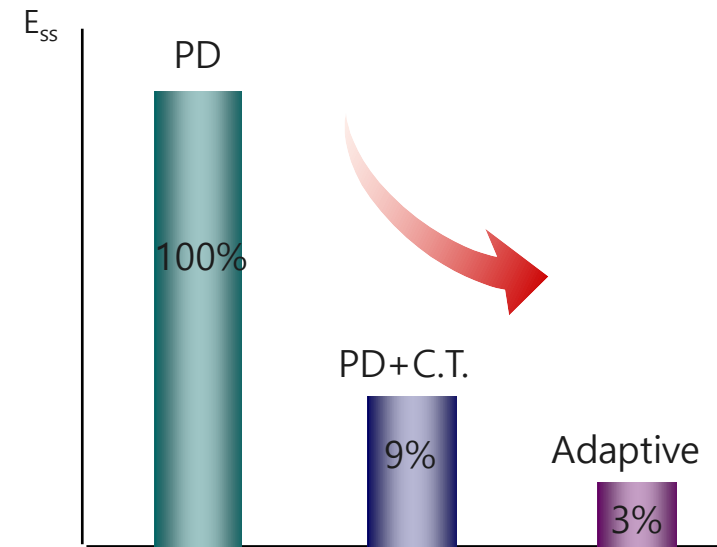
■ Comparison of Steady State Error

< Steady State Error of $\theta_1(\text{deg})$ >

	PD Only	PD + Computed Torque	PD + Computed Torque + Adaptive
case ①	0.4580	0.0416	-0.0178
case ②	0.4996	0.0833	-0.0036
case ③	0.5412	0.1249	0.0148

< Steady State Error of $\theta_2(\text{deg})$ >

	PD Only	PD + Computed Torque	PD + Computed Torque + Adaptive
case ①	0.1236	0.0112	-0.0085
case ②	0.1349	0.0225	-0.0018
case ③	0.1461	0.0337	0.0057



8. Conclusion

- Through kinematics & Motion Generation, we can make our own desired motion or trajectories needed for a robot.
- We can also derive dynamics of a robot using Lagrange Eqns. And understand the meaning of each terms for the dynamic eqns.
- PD control is a good for a linear system and easy to apply. But, the error cannot be compensated for nonlinear systems.
- PD control + Computed Torque method shows better performance when dynamics of systems are well known.
- Adaptive method can be an alternative way to compensate the error when parameter uncertainty exists.

9. Summary

