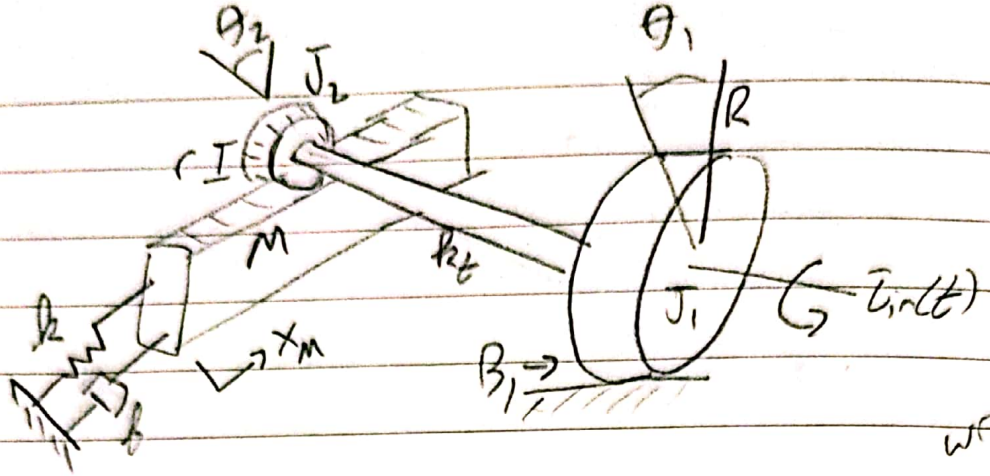
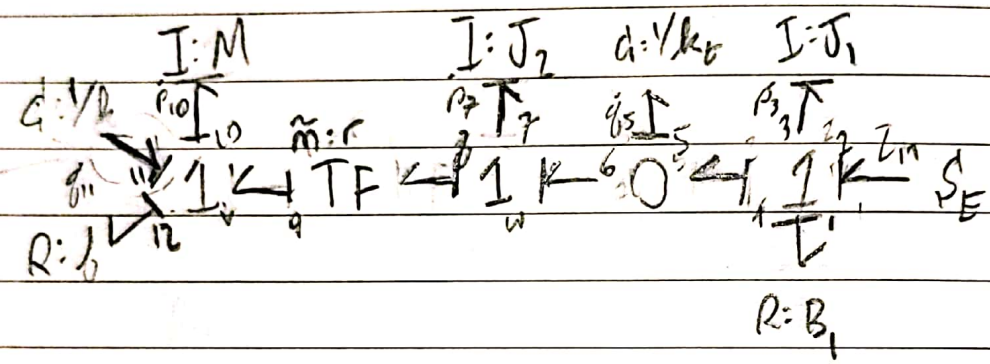


WF=V

1)



2)

Independent State Variables

q_{11}, p_7, q_5, p_3 (p_{10} dependent)

↳ derivative causality

→ 7 independent → 4th order

3) $\dot{q}_{11} = f_{11}$ $\quad b_{12} f_{11} = e_{12}$ $\quad f_{11} = \tilde{m} f_8$

$\downarrow$ $\quad k_{11} \cdot q_{11} = e_{11}$ $\quad f_8 = p_7/J_2$

$$\boxed{\dot{q}_{11} = \tilde{m} \frac{p_7}{J_2}}$$

e_{10} $\quad e_9$ $\quad e_4$

$e_6 = e_5$

$e_9 \tilde{m} = e_8$

$\dot{p}_7 = e_7 = e_6 - e_8 = q_5/k_t - \tilde{m} e_9$

$\dot{p}_7 = q_5/k_t - \tilde{m}(e_{10} + e_{11} + e_{12})$

$\dot{p}_7 = q_5/k_t - \tilde{m}(e_7/\tilde{m} + q_{11}/k_{11} + b_{12} f_{11})$

$e_9 = e_{10} + e_{11} + e_{12}$

$e_5 = q_5 \cdot k_t$

$e_4 = S_E' - e_3 - e_1$

$e_1 = f_1 B_1$

$f_3 = p_3/J_1$

$f_{11} = p_{10}/M$

$$\dot{p}_3 = e_3 = \dot{S}_E - e_1 - e_4 = T_{in} - f_1 B_1 - q_5/k_t$$

$$\boxed{\dot{p}_3 = T_{in} - f_1 B_1 - q_5/k_t} \quad \boxed{\dot{p}_3 = T_{in} - \frac{p_3 B_1}{J_1} - q_5 \cdot k_T}$$

$$\dot{q}_5 = f_5 = f_4 - f_6 = \boxed{p_3/J_1 - p_7/J_2 = q_5}$$

$$\dot{p}_7 = e_7 = e_6 - e_8 = (q_5 \cdot k_t) - \tilde{m}(e_{10} + e_{11} + e_{12})$$

$$= q_5 k_t - \tilde{m}(\dot{p}_{10} + k_{11} q_{11} + b_{12} p_{10}/M)$$

$$p_{10} = M f_{11} = M \tilde{m} f_7 = M \tilde{m} p_7/J_2$$

$$\dot{p}_{10} = \frac{M \tilde{m} \dot{p}_7}{J_2}$$

$$\dot{p}_7 = q_5 k_t - \tilde{m}\left(\frac{M \tilde{m} \dot{p}_7}{J_2} + k_{11} q_{11} + b_{12} p_{10}/M\right)$$

$$\dot{p}_7\left(1 + \frac{M \tilde{m}^2}{J_2}\right) = q_5 k_t - \tilde{m}(k_{11} q_{11} + b_{12} p_{10}/M)$$

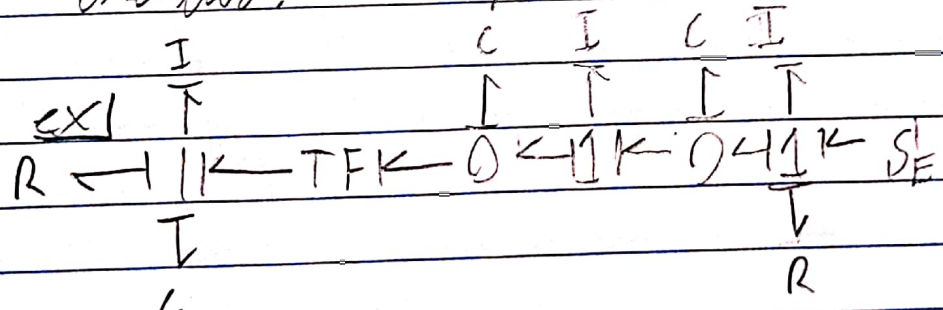
$$\dot{p}_7 = \frac{q_5 k_t - \tilde{m}(k_{11} q_{11} + b_{12} p_{10}/M)}{\left(1 + \frac{M \tilde{m}^2}{J_2}\right)} \qquad \frac{M \tilde{m} p_7}{J_2}$$

$$\begin{bmatrix} \dot{p}_3 \\ \dot{p}_7 \\ \dot{q}_5 \\ \dot{q}_{11} \end{bmatrix} = \begin{bmatrix} -B_1/J_1 & 0 & -k_T & 0 \\ 0 & -\frac{\tilde{m}^2 b_{12}}{J_2}\Big/\left(1+\frac{M\tilde{m}^2}{J_2}\right) & \frac{k_T}{\left(1+\frac{M\tilde{m}^2}{J_2}\right)} & \frac{-\tilde{m}k_{11}}{\left(1+\frac{M\tilde{m}^2}{J_2}\right)} \\ 1/J_1 & -1/J_2 & 0 & 0 \\ 0 & \frac{\tilde{m}}{J_2} & 0 & 0 \end{bmatrix} \begin{bmatrix} p_3 \\ p_7 \\ q_5 \\ q_{11} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \cdot U_{in}$$

5) Conceptual Question

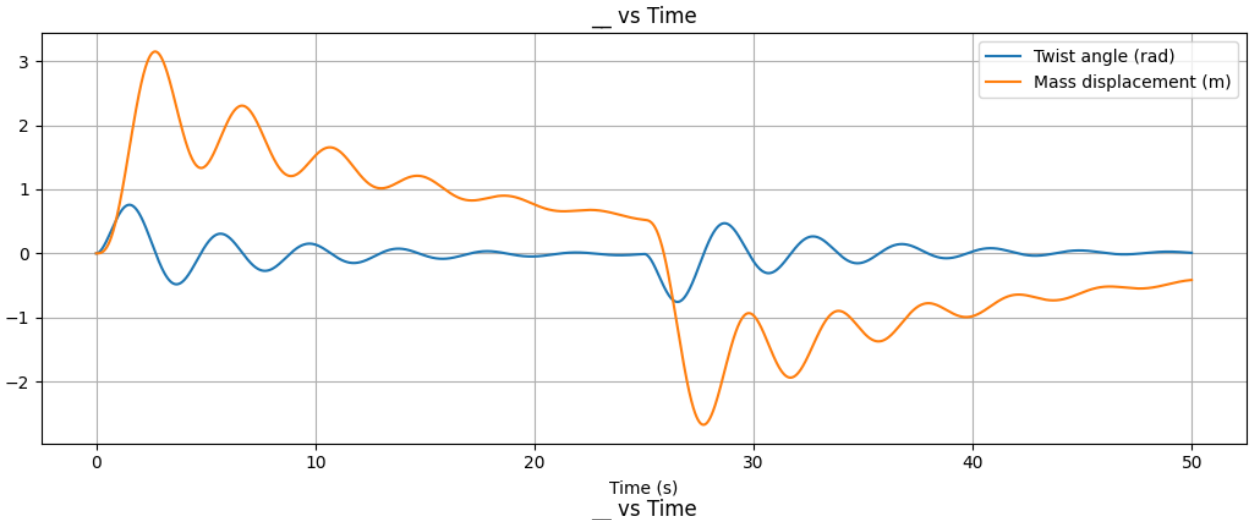
- Add slop to the system. Imperfect gear meshing will make it so p_7 & p_{10} are not linked by derivative causality anymore.

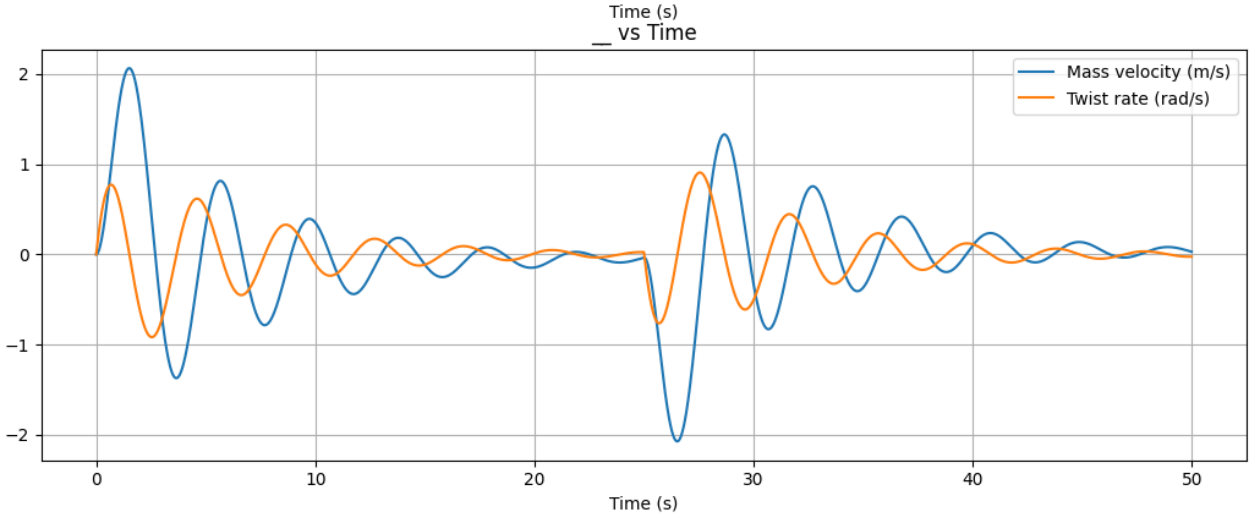
This would make it a 6th order system as the energy stored is no longer linked between the two.



```
import numpy as np
import matplotlib.pyplot as plt

def main():
    J1 = 2.2
    J2 = 1
    B1 = 3
    Kt = 2
    M = 16
    b = 2
    k = 20
    R = 0.5
    r = 0.1
    m = r

    # define the system ODEs
    def equations(x, t):
        line_1 = np.array([-B1/J1, 0, -Kt, 0])
        convt = (1 + M*m**2/J2)
        line_2 = np.array([0, -m**2*b/J2/convt, Kt/convt, -m*k/convt])
        line_3 = np.array([1/J1, -1/J2, 0, 0])
        line_4 = np.array([0, m/J2, 0, 0])

        A = np.array([line_1, line_2, line_3, line_4])
        B = np.array([1, 0, 0, 0])
        t_in = 5 if t < 25 else 0
        return np.dot(A, x) + B * t_in
    x0 = np.array([0, 0, 0, 0])

    t = np.array([x*0.01 for x in range(5000)])

    sol = rk4fixed(equations, x0, t)
    fig, (ax1, ax2) = plt.subplots(2)
    p3 = sol[:,0]
    p7 = sol[:,1]
    p10 = M*m*p7/J2
    q5 = sol[:,2]
    q11 = sol[:,3]
    twist = q5-q11

    dots = np.array([equations(sol[i],t[i]) for i in range(len(sol))])
    p7_dot = dots[:,1]
    p10_dot = M*m*p7_dot/J2

    ax1.plot(t, twist, label = "Twist angle (rad)")
```

```
    ax1.plot(t, p10, label = "Mass displacement (m)")
    ax1.legend()
    ax1.set_title("__ vs Time")
    ax1.set_xlabel("Time (s)")
    ax1.grid()

    ax2.plot(t,p10_dot, label = "Mass velocity (m/s)")
    ax2.plot(t,dots[:,2] - dots[:,3], label = "Twist rate (rad/s)")
    ax2.grid()
    ax2.set_title("__ vs Time")
    ax2.set_xlabel("Time (s)")
    ax2.legend()
    plt.show()


def rk4fixed(f, x0, t):
    import numpy
    n = len(t)
    x = numpy.zeros((n, len(x0)))
    x[0] = x0
    for i in range(n - 1):
        h = t[i+1] - t[i]
        k1 = f(x[i], t[i])
        k2 = f(x[i] + k1 * h / 2., t[i] + h / 2.)
        k3 = f(x[i] + k2 * h / 2., t[i] + h / 2.)
        k4 = f(x[i] + k3 * h, t[i] + h)
        x[i+1] = x[i] + (h / 6.) * (k1 + 2*k2 + 2*k3 + k4)
    return x

if __name__ == "__main__":
    main()
```


__ vs Time
Twist angle (rad)
Mass displacement (m)
3
2
1
0
−1
−2
0
10
20
30
40
50
Time (s)


__ vs Time
Mass velocity (m/s)
Twist rate (rad/s)
2
1
0
−1
−2
0
10
20
30
40
50
Time (s)