AE353: Design Problem 02

Robotic Controllability

March 8, 2019

1 Goal

The code, `DesignProblem02` simulates a "gravity-assisted underactuated robot arm." This robot arm has two joints: one controlled, one free. Optical encoders measure joint angles (q) and velocities (v). The goal is to make the second joint angle track a piecewise-constant reference trajectory.

1.1 Requirements

A *requirement* is a property that the system must have in order to solve the problem.

The second joint angle, q_2 *shall reach within* $\pm 2^\circ$ *of the desired constant reference value of* $\theta = 60^\circ$ *in under* 5 *seconds and shall hold for a duration of at least* 15 *seconds.*

1.2 Verification

A *verification* is a test that is performed to verify the system meets the requirements set in section (4).

`DesignProblem02('Controller', 'datafile', 'data.mat')` will run $i = \mathbf{5}$ times with randomly generated initial values. The data points from `'data.mat'` will calculate the error between the second joint angle, q_2, and reference value, θ, at each time-step value of $\Delta t = 0.1[s]$ by the following computation:

$$E = \max E_i = |q_2 - \theta|, \text{ s.t. } \theta = 60^\circ \text{ for } i \in [1, 5]$$

If $E \leq 2^\circ$, then the requirement is satisfied.

2 State Space Model

The motion of the robot is governed by ordinary differential equations with the form

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + N(q, \dot{q}) = \tau \text{ where,} \tag{1}$$

$$q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}, \dot{q} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \tau = \begin{bmatrix} \tau_1 \\ 0 \end{bmatrix}$$

q is a matrix of joint angles, $\dot{q}$ is a matrix of joint velocities, τ is a matrix of applied torques, and $M(q), C(q,\dot{q}), N(q,\dot{q})$, are matrix-valued functions of q and/or $\dot{q}$. These functions depend on a number of parameters. Before we can linearize, we need to solve for $\ddot{q}$:

$$\ddot{q} = \frac{\tau - C(q,\dot{q})\dot{q} - N(q,\dot{q})}{M(q)} \tag{2}$$

Now we can use `Jacobian()` of $\ddot{q}$ and $\dot{q}$ for A and $\frac{\tau}{M(q)}$ to find B:

$$\mathbf{A} = [\ddot{q}]_{(\dot{q})} = \begin{bmatrix} 0 & 60 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 5.5465 & -61.2335 & -1.5762 & -0.8327 \\ -1.9024 & -10.7824 & -0.2776 & -0.3690 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 3.1525 \\ 0.5551 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Now that we found A, B, C, D, we can design and implement a controller for the *State Space Model*.

3 Control Design

The goal, as stated in requirements, is to make the arm converge asymptotically, tracing points at $\theta = 60°$ within 5 seconds. In order to implement this, the A, B, C, D values must be structured in the following State Space model:

$$\dot{x} = Ax + Bu, \;\; y = Cx + Du, \;\; u = -Kx + K_{ref}r + K_{int}v(t) + d$$

(WIP)

4 Results (WIP)

As there currently is no controller input designated, the graphs dampen and appear to converge to $v_j = 0$ for $j \in [1, 2]$.

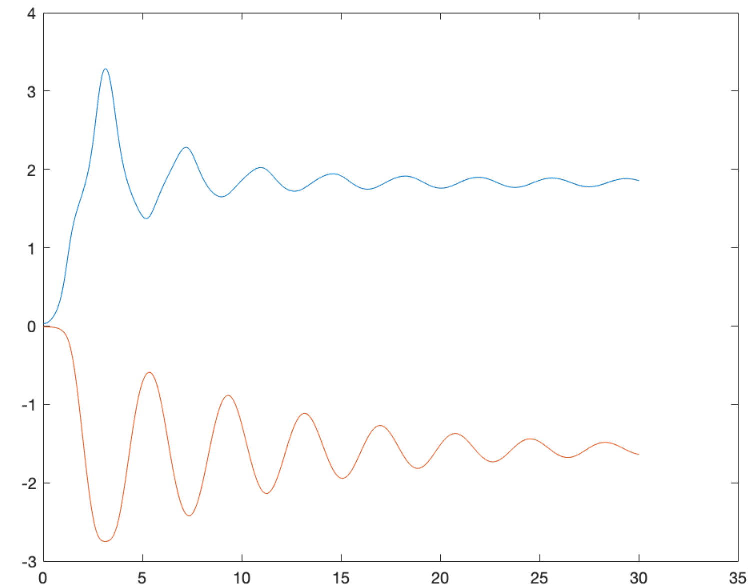

Figure 1: q vs. t

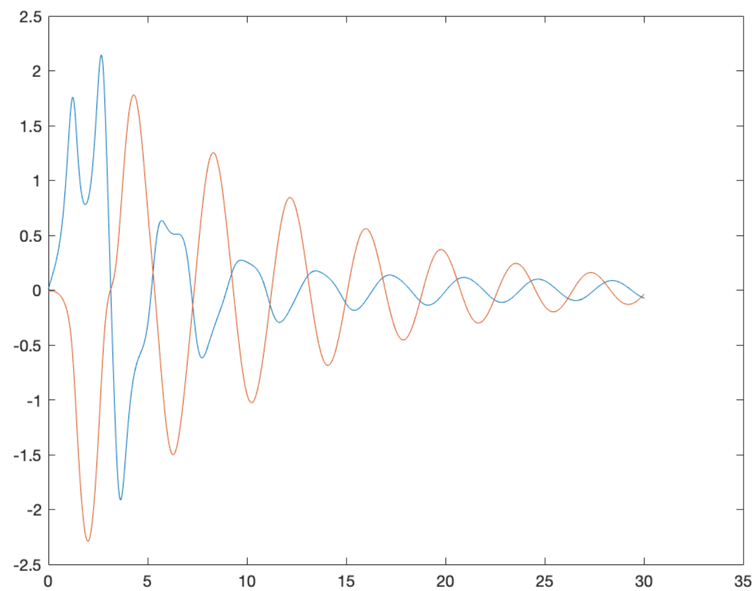

Figure 2: v vs. t