Balancing Robot

Izzy Mones and Heidi Dixon

April 23, 2025

Robot Design

Should have list of all the components. Maybe a picture. Do we need to show circuit stuff?

Model

System Description (Lagrange's Method)

Derive the equations here

$$\dot{x} = v \tag{1}$$

$$\dot{v} = \frac{-m^2L^2g\cos(\theta)\sin(\theta) + mL^2(mL\omega^2\sin(\theta) - \delta v) + mL^2u}{mL^2(M + m(1 - \cos(\theta)^2))} \tag{2}$$

$$\dot{\theta} = \omega \tag{3}$$

$$\dot{\omega} = \frac{(m + M)mgL\sin(\theta) - mL\cos(\theta)(mL\omega^2\sin(\theta) - \delta v) - mL\cos(\theta)u}{mL^2(M + m(1 - \cos(\theta)^2))} \tag{4}$$

where x is the cart position, v is the velocity, θ is the pendulum angle, ω is the angular velocity, m is the pendulum mass, M is the cart mass, L is the pendulum arm length, g is the gravitational acceleration, δ is a friction damping on the cart, and u is the control force applied to the cart.

Linearization

To build a control system for our model we will linearize our system of equations around a fixed point x_r where x_r is the position where the robot is vertical, unmoving and positioned at the origin.

The nonlinear system of differential equations

$$\frac{d}{dt}\boldsymbol{x} = f(\boldsymbol{x}). \tag{5}$$

can be represented as a Taylor series expansion around the point $\boldsymbol{x}_r$.

$$f(\boldsymbol{x}) = f(\boldsymbol{x}_r) + \left.\frac{\boldsymbol{df}}{\boldsymbol{dx}}\right|_{x_r}(\boldsymbol{x} - \boldsymbol{x}_r) + \left.\frac{\boldsymbol{d^2f}}{\boldsymbol{dx^2}}\right|_{x_r}(\boldsymbol{x} - \boldsymbol{x}_r)^2 + \cdots \tag{6}$$

Because $\boldsymbol{x}_r$ is a fixed point, we know that $f(\boldsymbol{x}_r) = 0$. Additionally, this approximation is only accurate in a small neighborhood around $\boldsymbol{x}_r$. In this neighborhood, we can assume that the value of $(\boldsymbol{x} - \boldsymbol{x}_r)$ is small, so higher order terms of this series will go to zero. So a fair estimate of our system is

$$\frac{d}{dt}\boldsymbol{x} \simeq \left.\frac{\boldsymbol{df}}{\boldsymbol{dx}}\right|_{x_r}(\boldsymbol{x} - \boldsymbol{x}_r) \tag{7}$$

where $\left.\frac{\boldsymbol{df}}{\boldsymbol{dx}}\right|_{x_r}$ is the Jacobian matrix for our system of equations $f(\boldsymbol{x})$ evaluated at the fixed point $\boldsymbol{x}_r$. The Jacobian matrix for our system of equations evaluated at $\boldsymbol{x}_r = [0\ 0\ \pi\ 0]$ is

$$\frac{\boldsymbol{df}}{\boldsymbol{dx}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{\delta}{M+m(1-\cos(\theta)^2)} & \frac{mg}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{d}{ML} & -\frac{(m+M)g}{ML} & 0 \end{bmatrix} \tag{8}$$

$$A = \left.\frac{\boldsymbol{df}}{\boldsymbol{dx}}\right|_{x_r} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{\delta}{M} & \frac{mg}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{\delta}{ML} & -\frac{(m+M)g}{ML} & 0 \end{bmatrix} \tag{9}$$

$$\frac{d}{dt}\boldsymbol{x} \simeq A(\boldsymbol{x} - \boldsymbol{x}_r)$$

$$B = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{ML} \end{bmatrix} \tag{10}$$

LQR

LQG

- Estimate the full state from sensor readings from the x position, and the angular velocity ω.
- Derive the Kalman filter matrix K_f using the `lqr` function from python control library.

```
# This is our state disturbance matrix
Vd = np.eye(4)
# This is our sensor noise matrix
Vn = np.array([[1, 0], \
               [0, 1]])
Kf = lqr(A.transpose(), C.transpose(), Vd, Vn)[0].transpose()
```

- To build the linear state space for our Kalman filter we build new matrices.
  - $A_{kf} = A - K_f C$
  - $B_{kf} = \begin{bmatrix} B & K_f \end{bmatrix}$
  - $C = I_4$
  - D is a 0 matrix with the same dimensions as K_f

  These form a new linear system

$$\frac{d}{dt}\boldsymbol{x} = A_{kf}\boldsymbol{x} + B_{kf}\boldsymbol{u} \tag{11}$$

$$\boldsymbol{y} = C_{kf}\boldsymbol{x} + D\boldsymbol{u} \tag{12}$$

- Our input vector $\boldsymbol{u} = [u, x_s, \omega_s]$ is our motor torque u and our two sensor readings, position x_s and angular velocity ω_s

Experiments

Conclusions