Balancing Robot

Izzy Mones and Heidi Dixon

May 5, 2025

Robot Design

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

Model

We modeled our balancing robot as a pendulum cart system.The wheels and motors are considered to be the cart. All other parts of the robot pivot around the axis created by the wheels and are considered part of the pendulum. A thorough discussion of this model is found in Brunton and Kutx (2019).

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

$$\ddot{x} = \frac{-mg\cos(\theta)\sin(\theta) + mL\dot{\theta}^2\sin(\theta) - \delta\dot{x} + u}{M + m\sin^2\theta} \tag{2}$$

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

$$\ddot{\theta} = \frac{(m+M)g\sin(\theta) - \cos(\theta)(mL\dot{\theta}^2\sin(\theta) - \delta\dot{x}) - \cos(\theta)u}{L(M + m\sin^2\theta)} \tag{4}$$

where x is the cart position, $\dot{x}$ is the cart velocity, θ is the pendulum angle, $\dot{\theta}$ is the angular velocity, m is the pendulum mass, M is the cart mass, L is the distance from the pivot point to the center of mass of the pendulum, g is the gravitational acceleration, δ is a friction damping on the cart, and u is the control force applied to the cart. This forms a set of differential equations

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

where $\boldsymbol{x}$ is the state vector $\boldsymbol{x} = [x, \dot{x}, \theta, \dot{\theta}]$ and $\boldsymbol{u}$ is the input vector.$\boldsymbol{u} = u * [0, 1, 0, 1]$.

Linearization

To build a control system for our model we will linearize the non-linear system of equations (1), (2), (3), and (4) around a fixed point $(\boldsymbol{x}_r, \boldsymbol{u}_r)$ where $\boldsymbol{x}_r$ is the position where the robot is vertical, unmoving and positioned at the origin and $\boldsymbol{u}_r$ is the input with motor torque at zero.

The nonlinear system of differential equations (5) can be represented as a Taylor series expansion around the point $\boldsymbol{x}_r$.

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

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

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

where $\left.\frac{\boldsymbol{df}}{\boldsymbol{dx}}\right|_{x_r}$ is the Jacobian matrix for our system of equations $f(\boldsymbol{x}, \boldsymbol{u})$ with respect to $\boldsymbol{x}$ and $\left.\frac{\boldsymbol{df}}{\boldsymbol{du}}\right|_{u_r}$ is the Jacobian matrix with respect to $\boldsymbol{u}$. Both are then evaluated at the fixed point $(\boldsymbol{x}_r, \boldsymbol{u}_r)$. Our partial differentials with respect to $\boldsymbol{x}$ are

$$\frac{\partial f_1}{\partial x} = \frac{\partial f_1}{\partial \theta} = \frac{\partial f_1}{\partial \dot{\theta}} = \frac{\partial f_3}{\partial x} = \frac{\partial f_3}{\partial \dot{x}} = \frac{\partial f_3}{\partial \theta} = 0$$
$$\frac{\partial f_1}{\partial \dot{x}} = \frac{\partial f_3}{\partial \dot{\theta}} = 1$$

$$\begin{aligned}
\frac{\partial f_2}{\partial x} &= 0 \\
\frac{\partial f_2}{\partial \dot{x}} &= \frac{-\delta}{M+m\sin^2\theta} \\
\frac{\partial f_2}{\partial \theta} &= \frac{(M+m\sin^2\theta)(-mg(\cos^2\theta-\sin^2\theta)+mL\dot{\theta}^2\cos\theta)-(-mg\cos(\theta)\sin(\theta)+mL\dot{\theta}^2\sin(\theta)-\delta\dot{x}+u)(2m\sin\theta\cos\theta)}{(M+m\sin^2\theta)^2} \\
\frac{\partial f_2}{\partial \dot{\theta}} &= \frac{mL\sin\theta}{M+m\sin^2\theta} \cdot 2\dot{\theta}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial f_4}{\partial x} &= 0 \\
\frac{\partial f_4}{\partial \dot{x}} &= \frac{\delta\cos\theta}{L(M+m\sin^2\theta)} \\
\frac{\partial f_4}{\partial \theta} &= \frac{(L(M+m\sin^2\theta))((M+m)g\cos\theta-mL\dot{\theta}^2(\cos^2\theta-\sin^2\theta)-\delta\dot{x}\sin\theta+u\sin\theta)-((m+M)g\sin(\theta)-\cos(\theta)(mL\dot{\theta}^2\sin(\theta)-\delta\dot{x})-\cos(\theta)u)(2mL\sin\theta\cos\theta)}{L^2(M+m\sin^2\theta)^2} \\
\frac{\partial f_4}{\partial \dot{\theta}} &= -\frac{2\dot{\theta}mL\cos\theta\sin\theta}{L(M+m\sin^2\theta)}
\end{aligned}$$

The Jacobian matrix for our system of equations evaluated at $\boldsymbol{x}_r = [0\ 0\ \pi\ 0]$ and $\boldsymbol{u}_r = [0]$ gives the matrix

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{\delta}{M} & -\frac{mg}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{\delta}{ML} & -\frac{(m+M)g}{ML} & 0 \end{bmatrix} \tag{8}$$

Our partial differentials with respect to $\boldsymbol{u}$ are

$$\begin{aligned}
\frac{\partial f_1}{\partial u} &= 0 \\
\frac{\partial f_2}{\partial u} &= \frac{1}{M+m\sin^2\theta} \\
\frac{\partial f_3}{\partial u} &= 0 \\
\frac{\partial f_4}{\partial u} &= -\frac{\cos\theta}{L(M+m\sin^2\theta)}
\end{aligned}$$

Evaluating these equations at $\boldsymbol{x}_r = [0\ 0\ \pi\ 0]$ and $\boldsymbol{u}_r = [0]$ gives the matrix

$$B = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{ML} \end{bmatrix} \tag{9}$$

Two Variable Model

We also built a two variable model for the balancing robot that has a state consisting of only the robot angle and angular velocity. This model works if you don't need to control the position of the robot. Working with this model was useful for testing purposes since it is a simpler system. It has the equations of motion

$$\begin{aligned} \dot{\theta} &= \dot{\theta} \\ \ddot{\theta} &= \frac{(m+M)g\sin(\theta) - \cos(\theta)(mL\dot{\theta}^2\sin(\theta)) - \cos(\theta)u}{L(M + m\sin^2\theta)} \end{aligned}$$

and the linearized matrices

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{(m+M)g}{ML} & 0 \end{bmatrix}$$

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

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

$$\frac{d}{dt}\boldsymbol{x} = A_{kf}\boldsymbol{x} + B_{kf}\boldsymbol{u} \tag{10}$$

$$\boldsymbol{y} = C_{kf}\boldsymbol{x} + D\boldsymbol{u} \tag{11}$$

- 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

References

Steven Brunton and J. Nathan Kutx. *Data Driven Science & Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019.