Linear Systems: LQR Case Studies

Daniel M Bodily

Brigham Young University, Provo UT 84602, USA

1 Introduction

The problem of using a Linear-Quadratic Regulator (LQR) to control a dynamic system is fundamental within the field of optimal control theory. This work presents a brief derivation of the LQR controller following the derivation presented by Hespanha [1]. It also gives several examples that demonstrate how LQR controllers may be used to control nonlinear systems. In this work two systems are investigated in depth; an inverted pendulum, and a quadcopter. For both systems, equations of motion are derived and linearized. An LQR controller is applied using Python's control toolbox. Results are presented in graphical form and links to visualizations are provided that demonstrate controller performance.

2 LQR Formulation

Given a continuous-time linear system,

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{1}$$

the optimal control problem may be formulated as,

$$\min_{u(t)} J = \int_0^\infty y^T Q y + u^T R u \; dt \tag{2}$$

where Q and R are defined as a positive definite cost matrices of dimension n x n and k x k respectively. We desire to drive the output state y to zero as fast as possible using minimal control effort. We define the feedback invariant function $H(x, u)$,

$$\begin{aligned} H(x,u) &= -\int_0^\infty \dot{x}^T P x + x^T P \dot{x} \; dt \\ &= -\int_0^\infty \frac{d}{dt}(x^T P x) \; dt \\ &= x_o^T P x_o - \lim_{x\to\infty} x(t)^T P x(t) \quad = x_o^T P x_o \end{aligned} \tag{3}$$

where we have assumed that the limit of $x(t)$ at infinity is zero. We add and subtract this term within our optimal control formulation, and expand using our dynamics equations given in Eq. 1 to get,

$$\begin{aligned}\min_{u(t)} J &= H + \int_0^\infty y^T Q y + u^T R u \, dt - H \\ &= H + \int_0^\infty x^T C^T Q C X + u^T R u \, dt + \int_0^\infty \dot{x}^T P x + x^T P \dot{x} \, dt \\ &= H + \int_0^\infty x^T (C^T Q C + A^T P + P A) x + u^T R u + 2 u^T B^T P x \, dt\end{aligned} \tag{4}$$

where we have assumed D to be identically zero and have utilized the fact that $x^T P B u = u^T B^T P x$. By completing the square and assuming R and P to be symmetric,

$$\begin{aligned}\min_{u(t)} J = H + \int_0^\infty & x^T (C^T Q C + A^T P + P A - P B R^{-1} B^T P) x \\ & + (u + R^{-1} B^T P x)^T R (u + R^{-1} B^T P x) \, dt\end{aligned} \tag{5}$$

This function is minimized when the algebraic Ricotti Equation is solved for P,

$$A^T P + P A + C^T Q C - P B R^{-1} B^T P = 0 \tag{6}$$

and the following feedback control law is adopted,

$$u = -R^{-1} B^T P x(t) = -K x(t) \tag{7}$$

The closed-loop system with LQR feedback control then becomes,

$$\dot{x} = (A - B R^{-1} B^T P) x \tag{8}$$

This represents the optimal feedback controller with respect to the user-defined weighting matrices Q and R. In practice these weightings must be tuned to achieve desired performance objectives.

The LQR controller can be applied to nonlinear systems by first linearizing the system about a desired trajectory (or point), and then applying feedback control to drive the differences between state variables and their desired values to zero. We present two examples of performing LQR on nonlinear systems here.

3 Inverted Pendulum

Vertically balancing an inverted pendulum is a classic nonlinear control problem. In this example we assume a mass M is attached to a fixed pin joint by a massless rod of length L. We assume we can apply a torque about the joint in a counter-clockwise fashion. We also include friction applied about the pin joint that resists motion.

The equation of motion is found by summing moments about the pin joint using Newton-Euler methods,

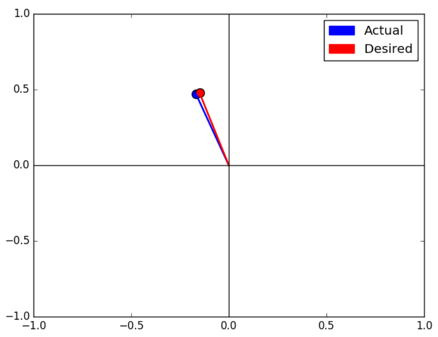

Fig. 1: An inverted pendulum is controlled using LQR. A visualization can be found at https://youtu.be/JGDmS5Go2-I.

$$I\ddot{\theta} = MgL\sin\theta - b\dot{\theta} - \tau \tag{9}$$

We can linearize this system about any desired point $(\theta_d, \dot{\theta}_d)$. The state space representation of the linearized system becomes,

$$\begin{bmatrix} \dot{\tilde{\theta}} \\ \ddot{\tilde{\theta}} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{mgL\cos\theta_d}{I} & -\frac{b}{I} \end{bmatrix} \begin{bmatrix} \tilde{\theta} \\ \dot{\tilde{\theta}} \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{1}{I} \end{bmatrix} \tilde{\tau} \tag{10}$$

where the tilde above each variable represents a deviation of the variable from a desired value (e.g. $\tilde{x} = x - x_{desired}$). The desired τ_d is approximated as that which maintains static equilibrium, $\tau_d = mgL\sin\theta_d$. Using the LQR formulation, this system is controllable in θ and can be driven to follow desired trajectories. We define initial conditions, Q, and R weighting matrices and a desired trajectory,

$$\theta_o = 200^\circ \qquad \dot{\theta}_o = 0 \qquad Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} .001 \end{bmatrix} \qquad \theta_d = \frac{\pi}{6}\sin(t)$$

The optimal feedback controller is obtained by solving the LQR problem as previously discussed,

$$\tau = -K\hat{x} + \tau_d \tag{11}$$

This system was simulated and controlled using Python's control toolbox. Results are shown in Fig. 2. As can be seen, the pendulum is struck at the beginning of its trajectory as to push it towards the desired trajectory. Then only small, sinusoidal inputs are required to keep the pendulum on target. Additional functionality was included within the implementation of this model to account for wrapping that occurs in the state variable θ at multiples of 2π. This caused inputs to drive the pendulum towards an equivalent $\theta_d = \theta_d + 2\pi$ that was closer to the pendulum's initial starting point. A visualization of the system's simulated dynamics with LQR control can be found at https://youtu.be/JGDmS5Go2-I.

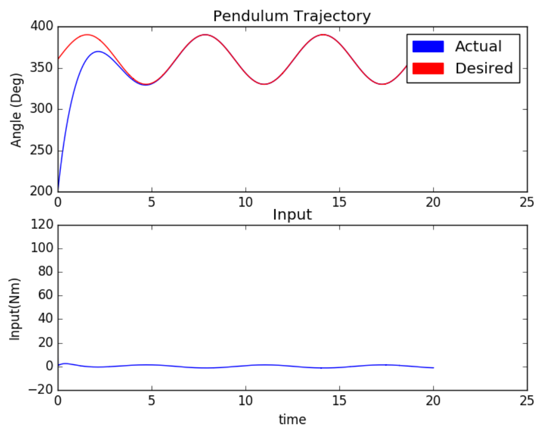

Fig. 2: A pendulum is controlled to track a desired trajectory using LQR control.

4 Quadcopter

We apply LQR to a quadcopter model and demonstrate LQR performance in tracking a desired 40cm-diameter circular trajectory, 50cm above the ground. This quadcopter has four propellers, each at length L from its center and each exerting a force in the negative z direction with respect to a body fixed coordinate frame (see Fig. 3).

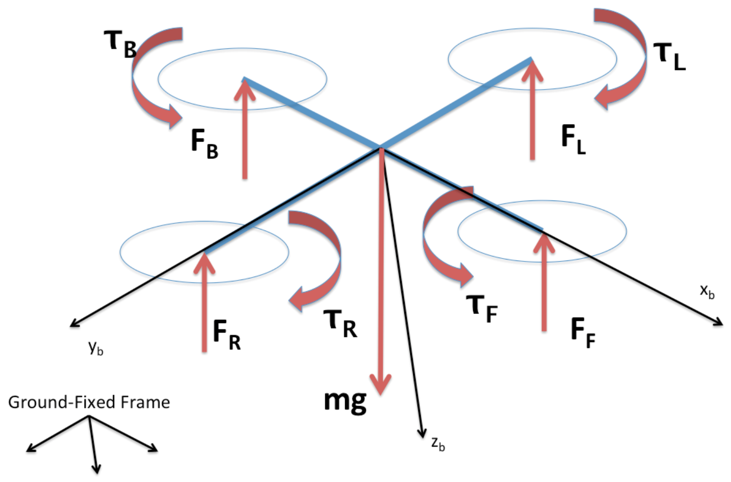

Fig. 3: A body fixed coordinate frame is attached with z point downwards as depicted.

Each propeller also induces a moment that we will assume to be proportional to the force applied ($\tau_i = k_i F_i$). We first identify the equations of motion for rotational velocity ω which we define to occur about the body-fixed axis.

$$\omega = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{12}$$

We define p, q, and r to represent components of ω directed about the $+x_b$, $+y_b$ and $+z_b$ axes respectively. We can sum moments about the center point and use Newton-Euler methods to derive the state equations,

$$\begin{aligned} \dot{p} &= \frac{(I_y - I_z)qr + (F_L - F_R)L}{I_x} \\ \dot{q} &= \frac{(I_z - I_x)pr + (F_F - F_B)L}{I_y} \\ \dot{r} &= \frac{(I_x - Iy)pq + (\tau_R + \tau_L - \tau_B - \tau_F)L}{I_z} \end{aligned} \tag{13}$$

Here we have assumed the body-fixed frame is centered about the principle axes of inertia, thus $I_{xy} = I_{yz} = I_{zx} = 0$.

Next, we use an Euler angle representation to relate the orientation of the body frame to a fixed inertial frame at the ground. We define the following Euler sequence,

1. Rotate ψ about the +z axis of the inertial frame, R_z
2. Rotate θ about the +y axis of the resulting frame, $R_{y'}$
3. Rotate ϕ about the +x axis of the resulting frame, $R_{x''}$

We can relate angular velocities as represented in the inertial frame $\dot{\psi}$, $\dot{\theta}$, and $\dot{\phi}$ to angular velocities in the body frame p, q and r by rotating each of these components into the body frame,

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = R_{x''}R_{y'}R_z \begin{bmatrix} \dot{\psi} \\ 0 \\ 0 \end{bmatrix} + R_{x''}R_{y'} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R_{x''} \begin{bmatrix} 0 \\ 0 \\ \dot{\phi} \end{bmatrix} = B \begin{bmatrix} \dot{\psi} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} \tag{14}$$

where R_z, $R_{y'}$ and $R_{x''}$ are 3x3 rotation matrices which are functions of ψ, θ, and ϕ respectively. We can invert the B matrix to find three more equations of motion,

$$\begin{bmatrix} \dot{\psi} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & \frac{s\phi}{c\theta} & \frac{c\phi}{c\theta} \\ 0 & c\phi & -s\phi \\ 1 & \frac{s\theta s\phi}{c\theta} & \frac{s\theta c\phi}{c\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{15}$$

We now derive state equations to describe translational motion. We describe the velocity of the quadcopter with respect to its body frame,

$$v_b = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{16}$$

where u, v and w represent velocities along the $+x_b$, $+y_b$ and $+z_b$ axes respectively. We differentiate to find the acceleration of the body as measured in the moving body frame,

$$\alpha_b = \frac{dv_b}{dt} + \omega \times v_b = \begin{bmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{bmatrix} \tag{17}$$

and sum forces in the body frame,

$$F_b = -(F_R + F_L + F_B + F_F) + R_{x''} Ry' Rz \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \tag{18}$$

Using Newton's second law $F_b = m\alpha_b$, and solving for $\dot{u}$,$\dot{v}$, and $\dot{w}$ we can achieve three more state equations,

$$\begin{aligned} \dot{u} &= rv - qw - g\sin\theta \\ \dot{v} &= pw - ru + g\cos\theta\sin\phi \\ \dot{w} &= qu - pv + g\cos\theta\cos\phi - \frac{F_L + F_B + F_L + F_R}{m} \end{aligned} \tag{19}$$

We can directly relate velocities expressed in the body frame to velocities expressed in the inertial frame p_n, p_e and p_d to find the final three state equations,

$$\begin{bmatrix} \dot{p_n} \\ \dot{p_e} \\ \dot{p_d} \end{bmatrix} = (Rx'' Ry' Rz)^T \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{20}$$

Defining our state vector and control vector to be,

$$x = [\psi, \theta, \phi, p, q, r, p_n, p_e, p_d, u, v, w]^T \quad u = [F_L, F_R, F_F, F_B]^T \tag{21}$$

and linearizing about a simple, planar trajectory, our state space representation becomes,

$$\dot{\tilde{x}} = A\tilde{x} + Bu \tag{22}$$

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -g & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & g & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ L/I_x & -L/I_x & 0 & 0 \\ 0 & 0 & L/I_y & -L/I_y \\ k/I_z & k/I_z & -k/I_z & -k/I_z \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1/m & -1/m & -1/m & -1/m \end{bmatrix}$$

Solving the LQR problem with these equations gives a set of gains K that may be applied in the following control law,

$$\begin{bmatrix} F_L \\ F_R \\ F_F \\ F_B \end{bmatrix} = -uK\tilde{x} + \begin{bmatrix} mg/4 \\ mg/4 \\ mg/4 \\ mg/4 \end{bmatrix} \tag{23}$$

We apply this formulation to the following desired trajectory.

$$p_{n_d} = .2\cos t \qquad p_{e_d} = .2\sin t \qquad p_{d_d} = .5 \tag{24}$$

Desired velocities may also be defined in the body frame by differentiating this trajectory and using the relationship given in Eq. 20. The result of applying the formulated controller to a quadcopter that initially lies at rest at a point on the ground 20cm west and 20cm south of the inertial frame yielded the trajectory shown in Fig. 4.

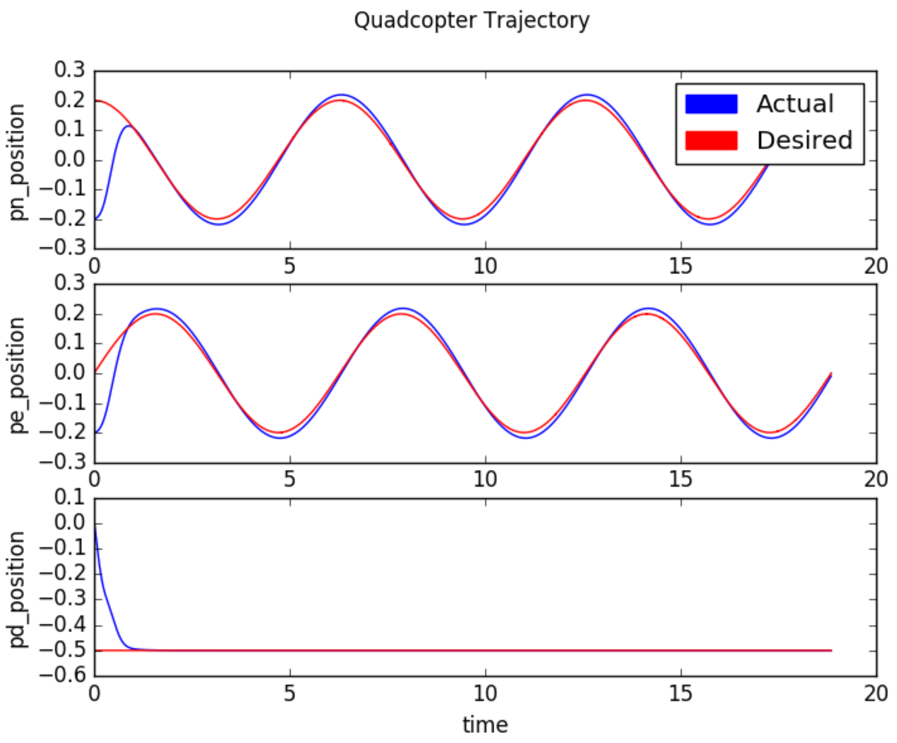

Fig. 4: An LQR controller allows a quadcopter to follow a circular trajectory

A simple visualization was implemented in Python for this example. It clearly shows the quadcopter converging on the desired trajectory marked by a moving red dot. This visualization can be found at https://youtu.be/-S5H1mTbF7I.

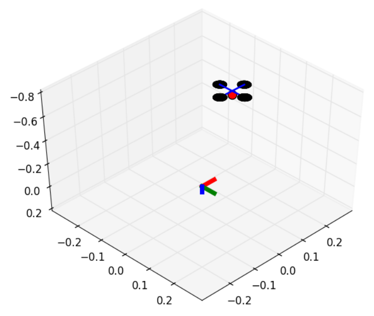

Fig. 5: A quadcopter is controlled to track a desired trajectory using LQR. A visualization of the controlled system can be found at https://youtu.be/-S5H1mTbF7I.

5 Conclusion

This work has demonstrated the efficacy of applying LQR to control both simple and complex nonlinear systems. The LQR formulation has been clearly derived and applied to two nonlinear systems. In both cases LQR allows the system to smoothly track a desired trajectory.

While LQR is an optimal formulation for linear systems, it suffers from several weaknesses. In some cases it has been reported that tuning the weight matrices Q and R to achieve desired performance is nontrivial. In the two cases presented here, we found tuning Q and R to be relatively simple however. LQR also may fail to give desired control when applied to highly nonlinear systems in which a locally linearized model may not be appropriate. In this case using nonlinear control methods may provide better control performance.

References

1. Hespanha, J.P.: Linear systems theory. Princeton university press (2009)