Linear Quadratic Control with Reference Input

[latexpage]The last post was our introduction to the Linear Quadratic Regulator (LQR). We saw there that as we started with initial conditions or introduced a disturbance the LQR will drive the states to zero. In the simulations we saw the graphic of the copter converge on the zero state: zero roll, pitch, yaw, and respective rates all to zero.

The, “Control Law” (feedback gain K) was obtained through a solution of the matrix Ricatti equation buried in Matlab. We’ll dig into that math at some other time.

This same solution is relevant for the, “tracking” problem or servo case: when we desire the plant to be controlled to a particular set of non-zero state values.

Basic Idea

We can think of it as the reverse of our last simulated cases where we started with non-zero state initial conditions (some angles for roll, pitch, and yaw) and observed the platform converge on the zero-state attitude.

Think of this new problem as starting with zero initial conditions (a level quadcopter) and desiring the attitude, “step response” to a non-zero input.

This is a bit artificial for the quadcopter because a non-zero attitude will mean thrust in a particular lateral direction which we are ignoring at present. Our controller design will still be relevant, as this is the attitude control loop.

Ultimate Goal

Later we'll introduce an outer position control loop that supplies the reference inputs to the attitude controller: the reference input that is the topic here. In flight it will not be a set value but a continually changing attitude reference input based on the outer position control loop's desire to move the platform by altering the attitude and the resultant thrust vector.

Before we can illustrate how an outer position controller will command attitude to this inner attitude controller we need to understand how to introduce the reference for roll, pitch, and yaw.

The respective rates are states also, but our goal is to regulate these to zero as the attitude tracks the body angle input reference. Our reference vector will supply zero for the rate states.

Theory

Feedback Control of Dynamic Systems by Franklin, Paul, and Emami-Naeini introduces the topic via the material below. The following page copies are from a 1994 edition of their textbook.

7.3.2 Introducing the Reference Input with Full State Feedback

Thus far, the control has been given by Eq. (7.58), or $u = -\mathbf{K}\mathbf{x}$. In order to study the transient response of the pole-placement designs to input commands, it is necessary to introduce the reference input into the system. An obvious way to do this is to change the control to $u = -\mathbf{K}\mathbf{x} + r$. However, the system will

now almost surely have a nonzero steady-state error to a step input. The way to correct this problem is to compute the steady-state values of the state and the control input that will result in zero output error and then force them to take these values. If the desired final values of the state and the control input are $\mathbf{x}_{ss}$ and u_{ss}, then the new control formula should be

$$u = u_{ss} - \mathbf{K}(\mathbf{x} - \mathbf{x}_{ss}), \tag{7.76}$$

so that when $\mathbf{x} = \mathbf{x}_{ss}$ (no error), $u = u_{ss}$. To pick the correct final values, we must solve the equations so that the system will have zero steady-state error to *any* constant input. The system differential equations are the standard ones:

$$\dot{\mathbf{x}} = \mathbf{F}\mathbf{x} + \mathbf{G}u, \tag{7.77a}$$

$$y = \mathbf{H}\mathbf{x} + Ju. \tag{7.77b}$$

In the steady state, Eqs. (7.77a) and (7.77b) reduce to the pair

$$\mathbf{0} = \mathbf{F}\mathbf{x}_{ss} + \mathbf{G}u_{ss}, \tag{7.78a}$$

$$y_{ss} = \mathbf{H}\mathbf{x}_{ss} + Ju_{ss}. \tag{7.78b}$$

We want to solve for the values for which $y_{ss} = r_{ss}$ for any value of r_{ss}. To do this we make $\mathbf{x}_{ss} = \mathbf{N}_x r_{ss}$ and $u_{ss} = N_u r_{ss}$. With these substitutions we can write Eqs. (7.78) as a matrix equation; the common factor of r_{ss} cancels out to give the equation for the gains:

Gain calculation for reference input

$$\begin{bmatrix} \mathbf{F} & \mathbf{G} \\ \mathbf{H} & J \end{bmatrix} \begin{bmatrix} \mathbf{N}_x \\ N_u \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}. \tag{7.79}$$

This equation can be solved for $\mathbf{N}_x$ and N_u to get

$$\begin{bmatrix} \mathbf{N}_x \\ N_u \end{bmatrix} = \begin{bmatrix} \mathbf{F} & \mathbf{G} \\ \mathbf{H} & J \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}.$$

With these values we finally have the basis for introducing the reference input so as to get zero steady-state error to a step input:

Control equation with reference input

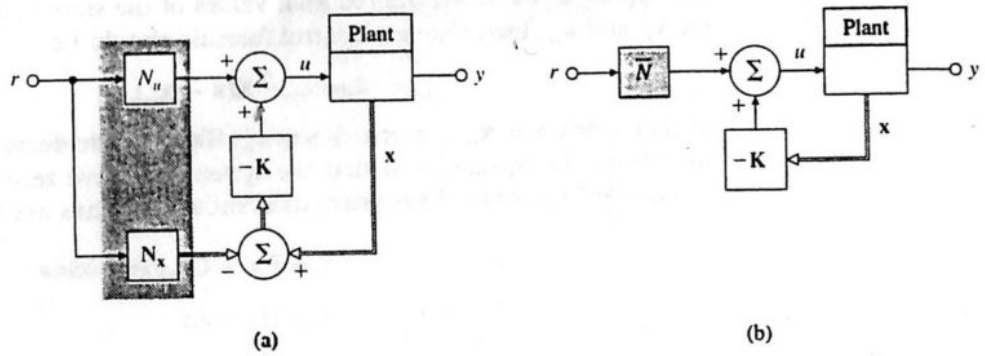
$$\begin{aligned} u &= N_u r - \mathbf{K}(\mathbf{x} - \mathbf{N}_x r) \\ &= -\mathbf{K}\mathbf{x} + (N_u + \mathbf{K}\mathbf{N}_x)r. \end{aligned} \tag{7.80}$$

The coefficient of r in parentheses is a constant that can be computed beforehand. We give it the symbol $\bar{N}$, so

$$u = -\mathbf{K}\mathbf{x} + \bar{N}r. \tag{7.81}$$

The block diagram of the system is shown in Fig. 7.9.

FIGURE 7.9
Block diagram for introducing the reference input with full-state feedback: (a) with state and control gains; (b) with a single composite gain



Quadrotor Implementation

We are dealing with more states and a multi-input, multi-output (MIMO) problem. The equation to solve from the reference above is shown here with **bold** matrix elements representing matrices themselves.

Our '**A**' and '**C**' Matrices are 6×6. The '**B**' and '**D**' Matrices are 6×4. This results in the matrix needing an inverse being 12×10, which is non-invertible. On page 510 Stengel writes we can employ the right pseudoinverse in this case.

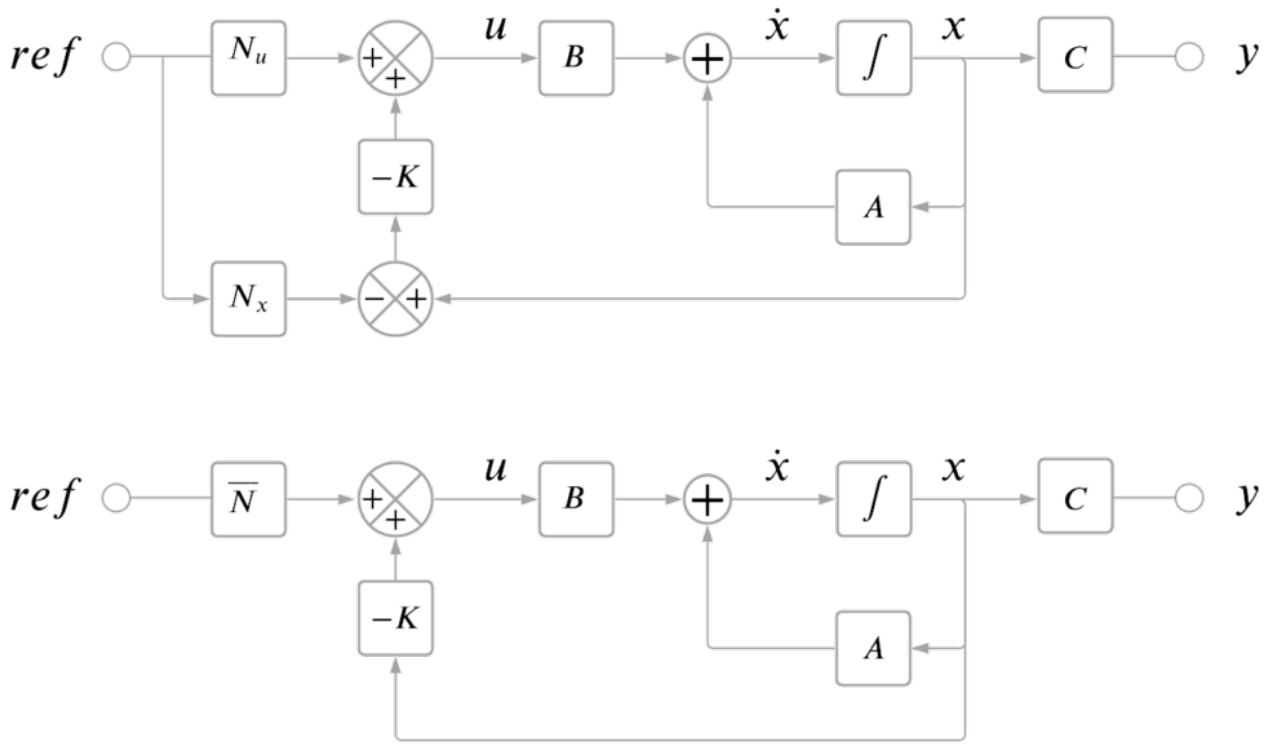
```
\begin{equation}
\begin{bmatrix}
\mathbf{N_{x}} \\
\mathbf{N_{u}} \\
\end{bmatrix}
=
\begin{bmatrix}
\mathbf{A} & \mathbf{B} \\
\mathbf{C} & \mathbf{D} \\
\end{bmatrix}^{-1}
```

\begin{bmatrix}
\mathbf{0} \\
\mathbf{1} \\
\end{bmatrix}
\end{equation}

We can then solve for the reference gains. The resulting $\mathbf{N_x}$ is 6×6. The resulting $\mathbf{N_u}$ is 6×4. We now have the matrices we need to operate on our state reference.

$\mathbf{N_x}$ and $\mathbf{N_u}$ are simplified into a composite gain in the second block diagram according to the reference textbook above:

\begin{equation}
\bar{N}=N_u+KN_x
\end{equation}



We’re now ready to see how the quadrotor LQ controller will track a reference input. The controller gain matrix K is from our LQR solution, so it’s the same controller.

Matlab Simulation

This Matlab script is a generalized version of the script in the last post covering the LQR simulation. In this script you will see the reference gain N is established and applied to a reference input. Yaw-axis sinusoidal reference tracking is illustrated in the following video generated by running the script.

You'll need the same three additional files from the last post (see the matlab script section near the end).

Download the script, find yourself a Matlab seat, and explore!

Quadrotor Reference Input Tracking: a yaw-axis sinusoidal reference

Conclusion

We've covered how to solve for the reference input gain matrix and modified our LQR Matlab simulation script accordingly. The simple simulation shared in the video above indicates we're stable and generally tracking the yaw sinusoid.

There's more to do in this area: examine disturbance rejection, tracking performance, and general analysis of our LQR controller. Through this process we'd apply some performance assessment metrics and iteratively adjust our cost function weights.

We're not yet to a real design and iteration process yet. We're still building the tools and the understanding of how this MIMO Linear Quadratic quadcopter math works.

Happy Learning!