Aero 2 Half-Quadrotor LQR Control Design

Concept Review

State-Feedback Control

This section shows how to design a state-feedback controller to control the position of the yaw of the Aero 2 half-quadrotor system. By using the state-space model derived in the *Parameter Estimation* lab, we can find a control gain K. The control gains are computed using the Linear-Quadratic Regulator (LQR) algorithm.

The general state-feedback control is illustrated in Figure 1.

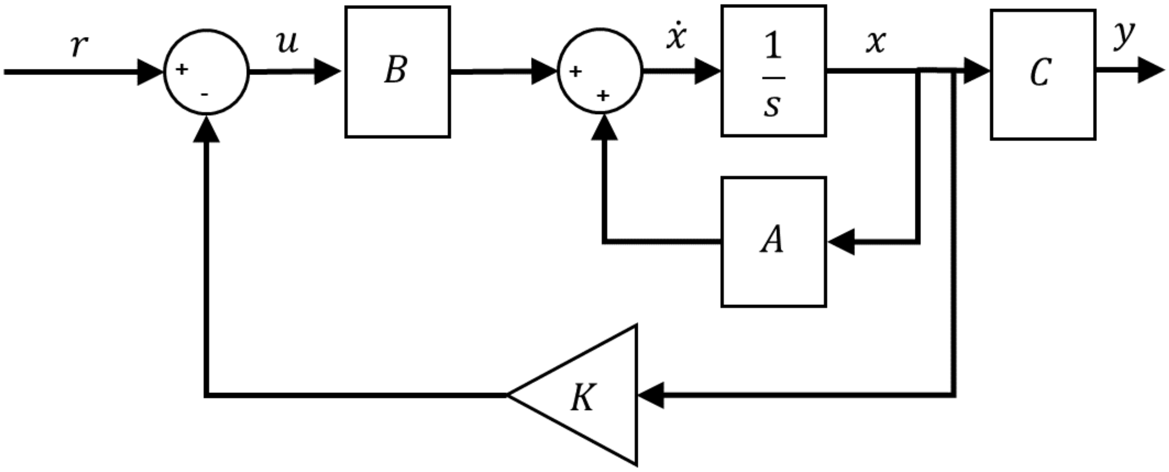


Figure 1: State-feedback control

The state-feedback controller is defined

$$u = K(x_d - x)$$

where $x^{\top} = [\psi(t), \dot{\psi}(t)]$ is the state, $x_d^{\top} = [\psi_d(t), 0]$ is the reference state with the desired setpoint yaw angle ψ_d, and u is the control input, which is the voltage applied to both rotors. Based on this, we can define the following state-feedback control

$$u = K(x_d - x) = k_p(\psi_d - \psi) - k_d\dot{\psi}$$

This shows that, when expanded, the state-feedback control has a same structure as the PD control that was used in the *PD Control* lab. However, the gains are derived using the LQR optimization technique, i.e., not using the time-domain based second-order design.

LQR Control Design

Our desired closed-loop response should match the following specifications

1. Peak time: $t_p \leq 4$ s.
2. Percent Overshoot: $PO \leq 5\%$.

MATLAB will be used to run the LQR optimization routine and generate K.

Load the Aero 2 Parameters

```
aero2_parameters;
```

Load damping and thrust parameters values .

```
aero2_half_quad_id_param;
```

Load the state-space model in MATLAB

```
% State-space model
A = [0 1;
    0 -Dy/Jy];
B = [0;
    Dt*Kf/Jy];
C = [1 0]; % assuming only yaw position measurement
D = 0;
%
ss_halfquad = ss(A,B,C,D)
```

```
ss_halfquad =

  A = 
            x1       x2
   x1        0        1
   x2        0  -0.1079

  B = 
            u1
   x1        0
   x2  0.01793

  C = 
       x1  x2
   y1   1   0

  D = 
       u1
   y1   0

Continuous-time state-space model.
```

LQR Control Simulation

The closed-loop response is the simulated in Simulink using the `s_aero2_half_quad_lqr` model shown in Figure 2. The *Aero 2 Half-Quadrotor Model* subsystem includes a block diagram model of the system.

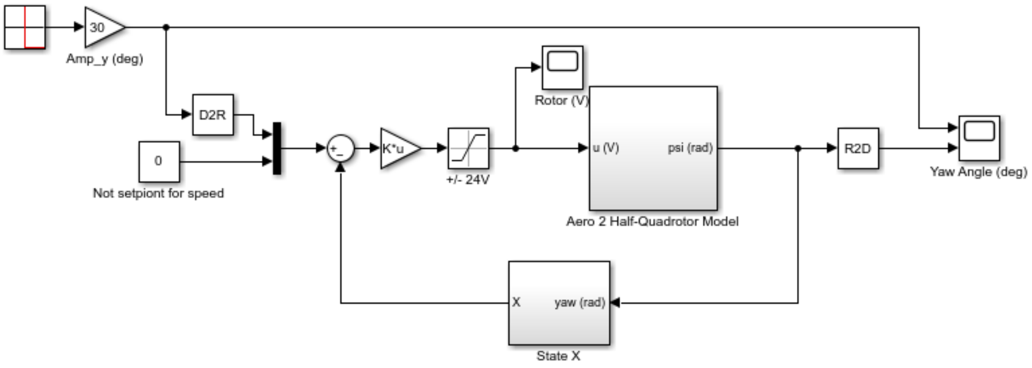


Figure 2: The `s_aero2_half_quad_lqr` model is used to simulate the Aero 2 LQR response.

LQR Control Design

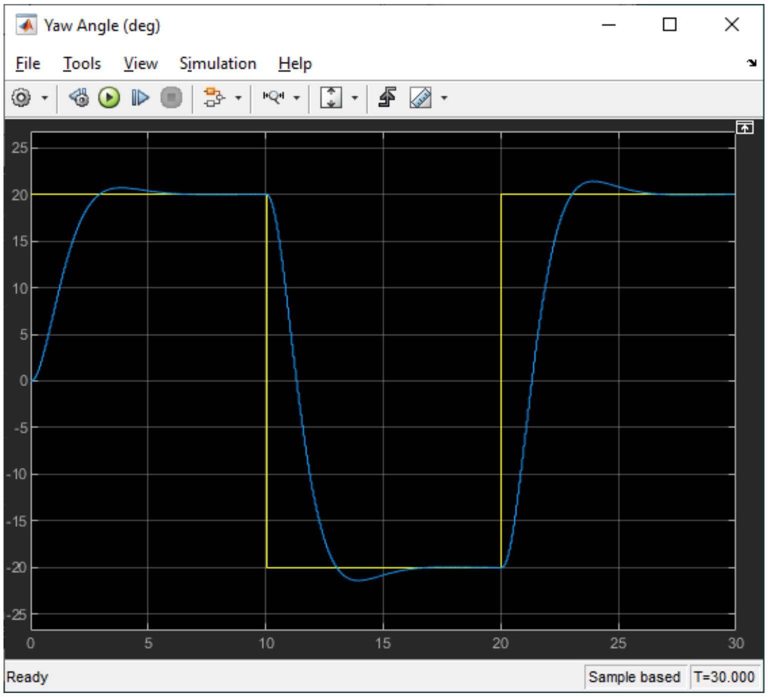
```
Q = diag([1000 0]);
R = 0.2;
K = lqr(A,B,Q,R)
```

```
K = 1×2
   70.7107   82.9927
```

Open the Simulink and run the simulation.

```
open("s_aero2_half_quad_lqr.slx")
sim("s_aero2_half_quad_lqr.slx")
```

See the sample response shown in Figure 3.



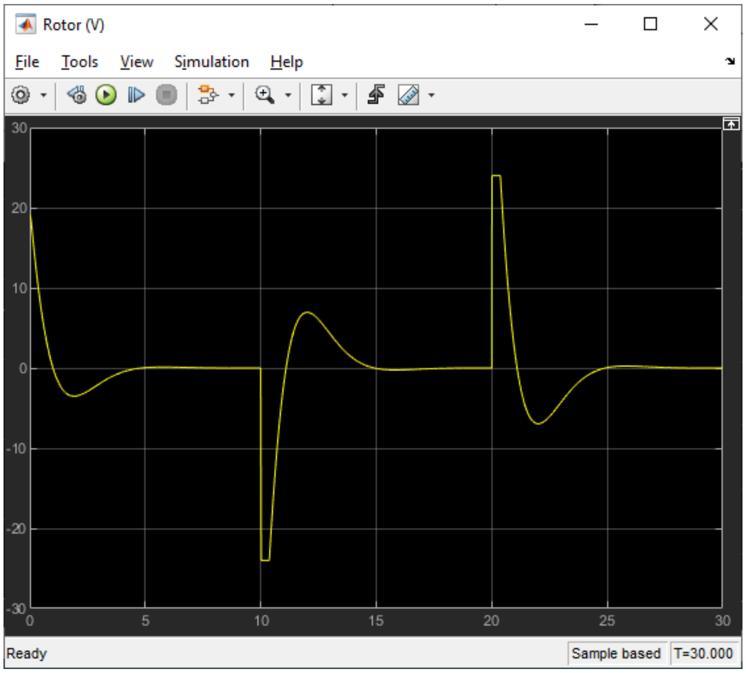
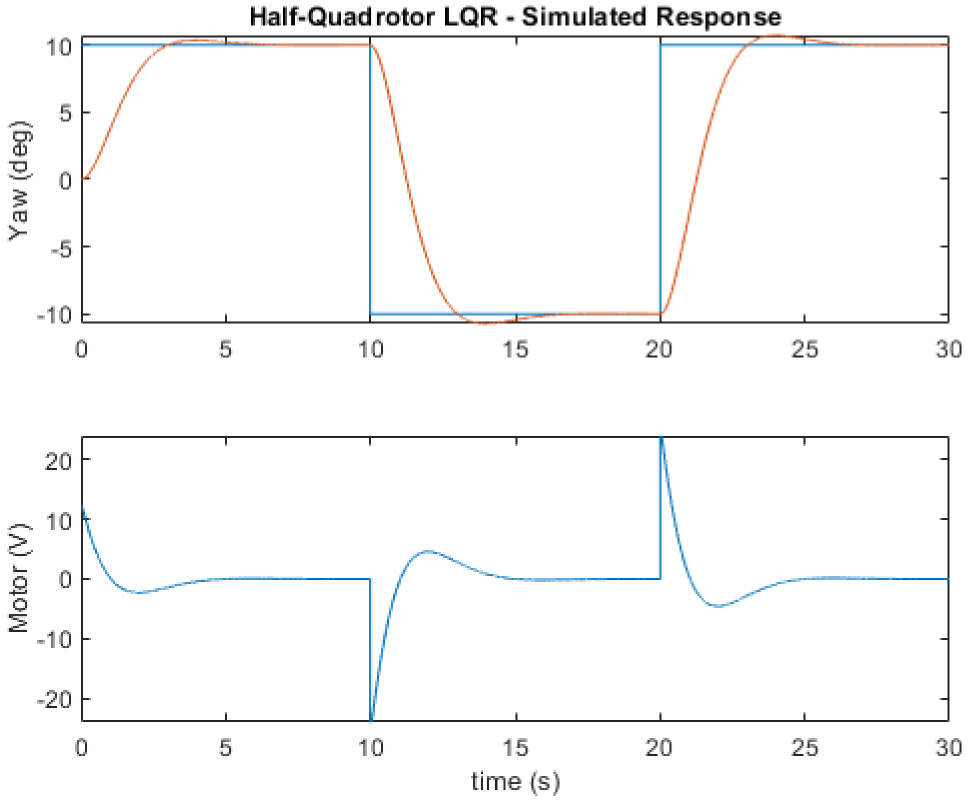


Figure 3: Aero 2 LQR control closed-loop response simulation.

Plot and analyze results.

```
% Load measured data from past run
% store in variables
t_sim = DataHalfQuadLQRSimVm.time; % time (s)
u_sim = DataHalfQuadLQRSimVm.signals.values; % pitch/front motor voltage (V)
psi_d_sim = DataHalfQuadLQRSimYaw.signals(1).values; % desired yaw angle (rad)
psi_sim = DataHalfQuadLQRSimYaw.signals(2).values; % yaw angle (rad)
%
subplot(2,1,1);
plot(t_sim,psi_d_sim,t_sim,psi_sim);
title('Half-Quadrotor LQR - Simulated Response');
ylabel('Yaw (deg)');
subplot(2,1,2);
plot(t_sim,u_sim);
ylabel('Motor (V)');
xlabel('time (s)');
```



```
% stepinfo(psi_sim(19/0.002:end),t_sim(19/0.002:end))
```

```
ans = struct with fields:
        RiseTime: 1.9107
    SettlingTime: 25.0952
     SettlingMin: 8.0057
     SettlingMax: 10.7076
       Overshoot: 7.1212
      Undershoot: 100.0003
            Peak: 10.7076
        PeakTime: 23.9620
```

The peak time and percent overshoot of the system are:

$t_p = 23.97 - 23 = 3.97$ s

$$PO = 100 \times \frac{10.71 - 10}{20} = 3.55\%$$

The percent overshoot and peak time control design specifications are satisfied in simulation. Note that if a larger setpoint is used, e.g., 20 deg, then the actuator will saturate and cause a longer peak time that will not satisfy the requirement, i.e., > 4 s.

LQR Control Hardware Implementation

The LQR control will be applied to the Aero 2 half-quadrotor system by running the `q_aero2_half_quad_lqr.slx` Simulink model shown in Figure 4 in QUARC.

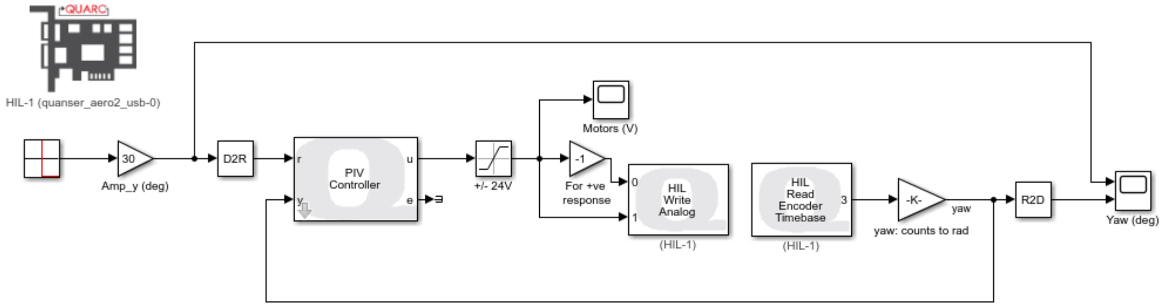


Figure 4: The `q_aero2_half_quad_lqr.slx` Simulink model uses QUARC to run the LQR control on the Aero 2

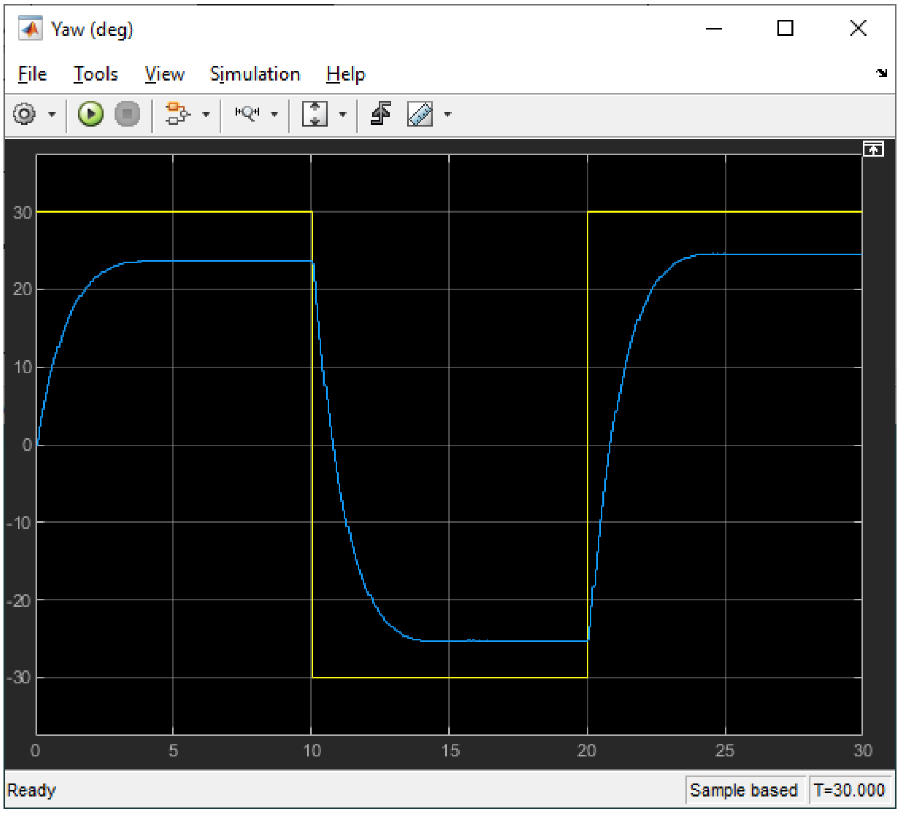
Aero 2 Setup

1. Make sure the Aero 2 has been tested as instructed in the Quick Start Guide.
2. Launch MATLAB and browse to the working directory that includes the Simulink models for this lab.
3. Configure the Aero 2 in the Half-Quadrotor configuration:
4. **Lock** the pitch axis and **unlock** the yaw axis.
5. Both rotors are **horizontal** (i.e., rotor shields are parallel with the ground).
6. Mount weight on each rotor.
7. Connect the USB cable to your PC/laptop.
8. Connect the power and turn the power switch ON. The Aero base LED should be red

Build and run the following Simulink model in QUARC by clicking on the *Monitor & Tune* button.

```
open("q_aero2_half_quad_lqr.slx")
```

See the sample response in Figure 5.



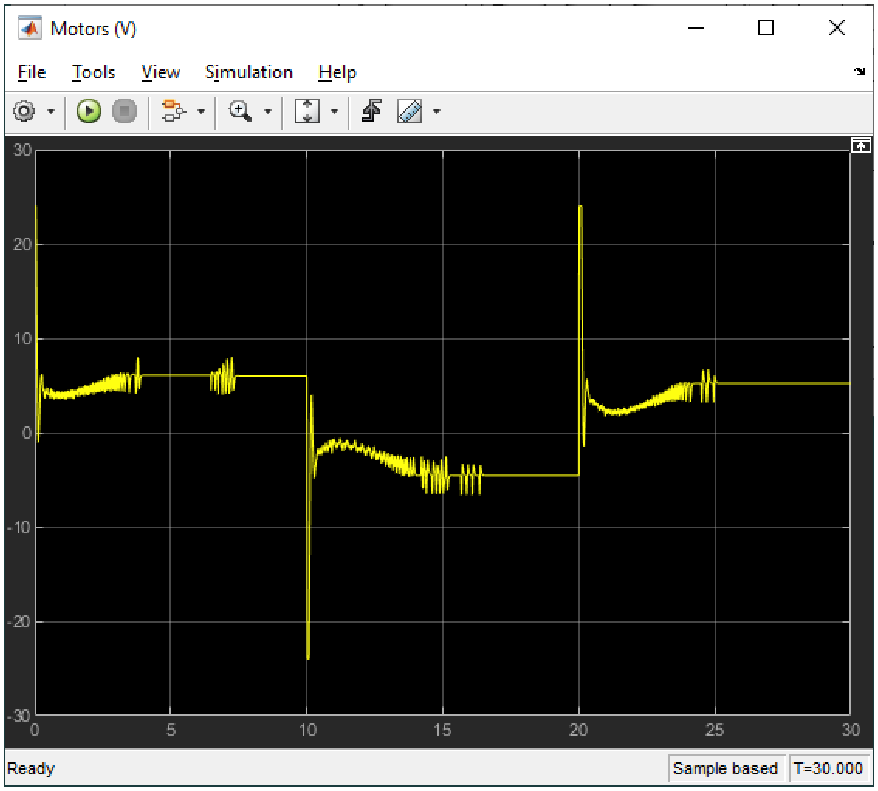
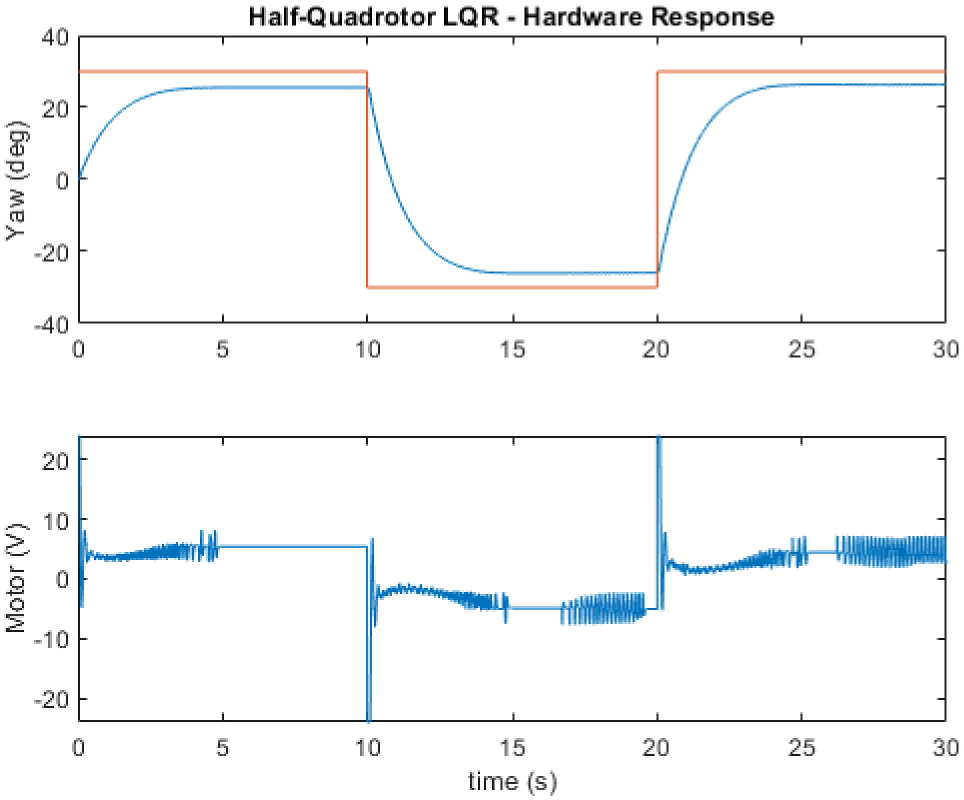


Figure 5: Sample LQR control response on Aero 2.s

Plot results

```
% Load measured data from past run
```

```
load('DataHalfQuadLQR.mat');
% store in variables
t = DataHalfQuadLQR(1,:); % time (s)
u = DataHalfQuadLQR(2,:); % pitch/front motor voltage (V)
psi_d = 180/pi*DataHalfQuadLQR(3,:); % desired yaw angle (deg)
psi = 180/pi*DataHalfQuadLQR(4,:); % yaw angle (deg)
%
subplot(2,1,1);
plot(t,psi_d,t,psi);
title('Half-Quadrotor LQR - Hardware Response');
ylabel('Yaw (deg)');
subplot(2,1,2);
plot(t,u);
ylabel('Motor (V)');
xlabel('time (s)');
```



The peak time and percent overshoot of the system are:

$t_p = 24.83 - 20 = 4.83$ s

$PO = 0\%$

The percent overshoot control design specification is satisfied on the hardware but the peak time goes above the requirement. Increasing the control gain would increase the response time for this to be met.

Similarly as with the PD control, the hardware response has a steady-state error and less overshoot than in the simulation. As mentioned in the *PD Control* lab, this is most likely due to umodelled dynamics such as the Coulomb friction.