Aero 2 Half-Quadrotor: Modelling and Parameter Estimation

Concept Review

The free-body diagram of the Aero 2 when in the half-quadrotor configuration is shown in Figure 1.

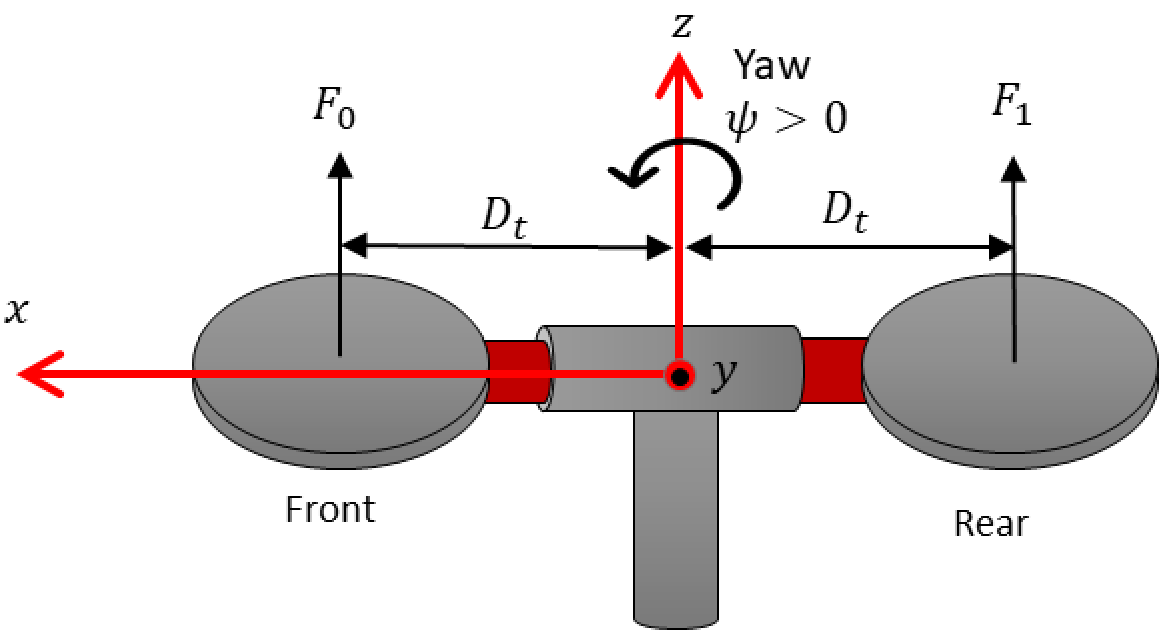


Figure 1: Aero 2 Half-Quadrotor model

The torque produced by the rotors causes the system to rotate about the yaw axis. The following linear equation of motion describes the yaw motion, ψ, relative to torque applied

$$J_y\ddot{\psi} + D_y\dot{\psi} = \tau_y$$

where J_y is the moment of inertia about the yaw axis, D_y is the viscous damping coefficient about the yaw axis, and τ_y is the torque acting about the yaw-axis. Because the pitch-axis is locked and only the yaw motions are considered, the same voltage is applied to both motors. The equation can be redefined in terms of the motor voltage applied to both rotors as

$$J_y\ddot{\psi} + D_y\dot{\psi} = D_t K_f u$$

where K_f is the thrust force gain, D_t is the distance between the pivot and the middle of the propeller, and u is the motor voltage that is applied to motor 0 and motor 1 on the rotors. Applying a positive voltage $u > 0$ results in a positive yaw response, $\dot{\psi} > 0$.

Note: In order to realize this on the actual hardware, the control input is defined as $V_0 = -u$ and $V_1 = u$, where V_0 is the voltage applied to motor 0 (the front motor), and V_1 is the voltage applied to the motor 1 (the rear motor).

Transfer Function Model

From the equation of motion, the transfer function model of the half-quad system is

$$J_y(\Psi(s)s^2 - \psi(0^-)s - \dot{\psi}(0^-)) + D_y(\Psi(s)s - \psi(0^-)) = D_t K_f U(s).$$

Since the system starts from rest, the initial conditions are zero, i.e., $\psi(0^-) = 0$ and $\dot{\psi}(0^-) = 0$. Solving for $\Psi(s)/U(s)$ we can obtain the following transfer function of the plant

$$\frac{\Psi(s)}{U(s)} = \frac{D_t K_f}{J_y s^2 + D_y s}.$$

The transfer function for the yaw rate can be expressed by

$$\frac{\beta(s)}{U(s)} = \frac{D_t K_f}{J_y s + D_y}$$

where $\beta(t) = \dot{\psi}(t)$.

State-Space Model

The linear state-space equations are:

$\dot{x} = Ax + Bu$ and $y = Cx + Du$

Defining the state $x^T = [\psi(t), \dot{\psi}(t)]$, the output vector $y = \psi(t)$, and the control variable u, we can find the following state-space matrices for the equations of motion given above

$A = \begin{bmatrix} 0 & 1 \\ 0 & -D_y/J_y \end{bmatrix}$ and $B = \begin{bmatrix} 0 \\ D_t K_f / J_y \end{bmatrix}$

$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$ and $D = [0]$

Note that only the position measurement of the yaw axis is being measured directly in this model, therefore $y = \psi(t)$.

Lab Procedure

Find the viscous damping and thrust force gain of the Aero 2 Half-Quadrotor model and validate the model by running the transfer function with the identified parameters in parallel with the hardware.

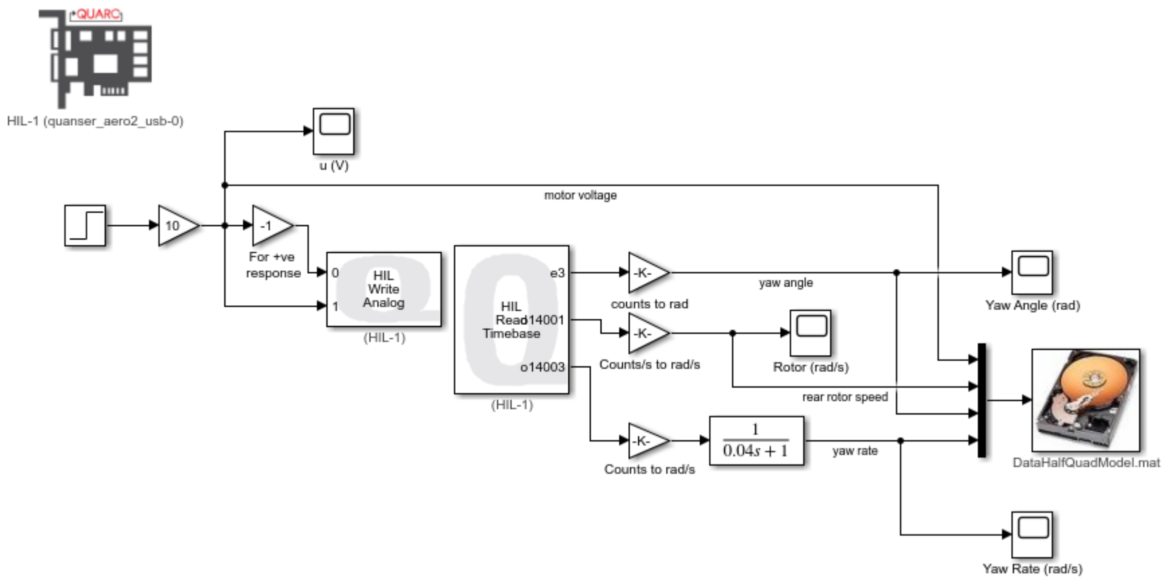
Setup

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

Lab 1: Get Open-Loop Response

The open-loop response can be obtained by running the `q_aero2_half_quad_model` Simulink model shown below in QUARC.



The Simulink model uses the `HIL Write Analog` and `HIL Read Timebase` blocks from the *QUARC Targets* library to apply a voltage to the rotors and measure the corresponding response. The response is saved into a MATLAB *.mat files using the `To Host File` block. This can then be used to plot the results and perform the parameter estimation analysis.

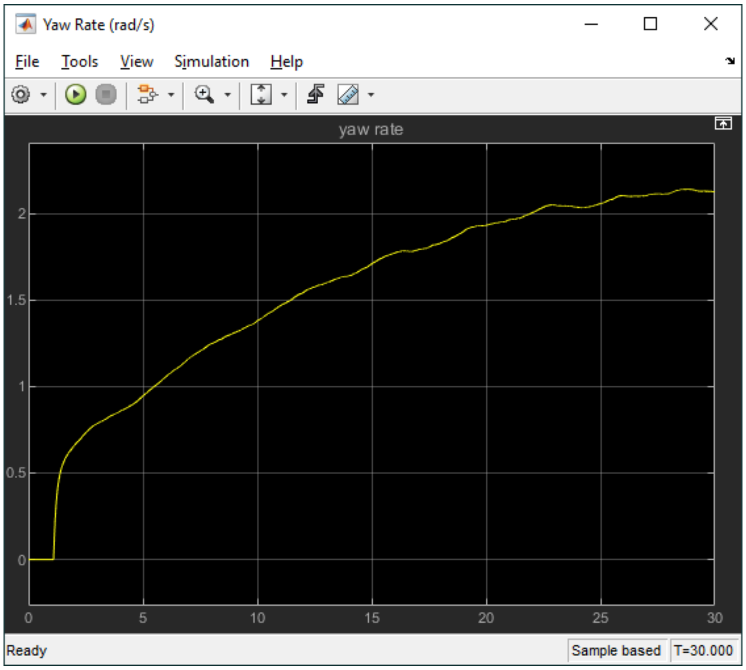
Load the Aero 2 Parameters

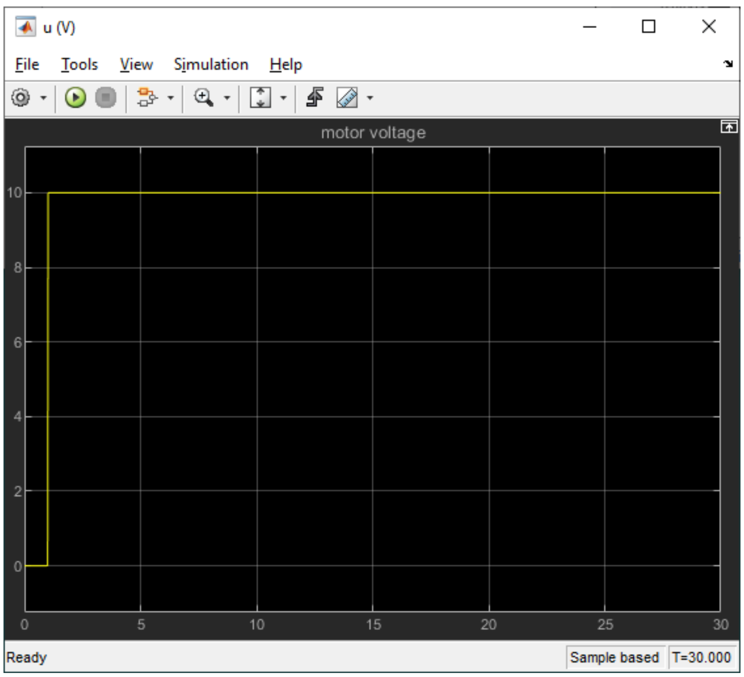
```
aero2_parameters;
```

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

```
% Load Simulink model
open("q_aero2_half_quad_model.slx");
```

Here is a sample response

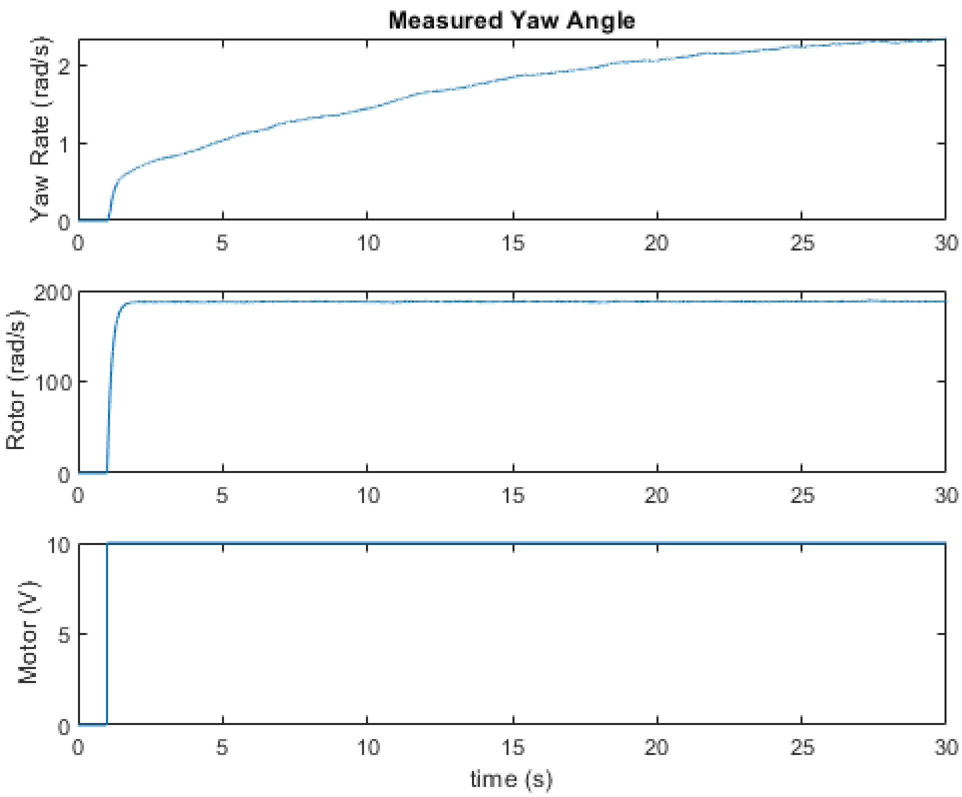




Plot results

```
% Load measured data from past run
load('DataHalfQuadModel.mat');
% store in variables
t = DataHalfQuadModel(1,:); % time (s)
u = DataHalfQuadModel(2,:); % pitch/front motor voltage (V)
wm = DataHalfQuadModel(3,:); % pitch/front rotor speed (rad/s)
psi = DataHalfQuadModel(4,:); % yaw angle (rad)
psi_dot = DataHalfQuadModel(5,:); % yaw angular rate (rad/s)
%
subplot(3,1,1);
```

```
plot(t,psi_dot);
title('Measured Yaw Angle');
ylabel('Yaw Rate (rad/s)');
subplot(3,1,2);
plot(t,wm);
ylabel('Rotor (rad/s)');
subplot(3,1,3);
plot(t,u);
ylabel('Motor (V)');
xlabel('time (s)');
```



Lab 2: Parameter Estimation

The measured yaw angle rate response to a step voltage can be modeled using a first-order transfer function. The actual response is higher-order, but a first-order model can give us an approximation to obtain fairly accurate parameters.

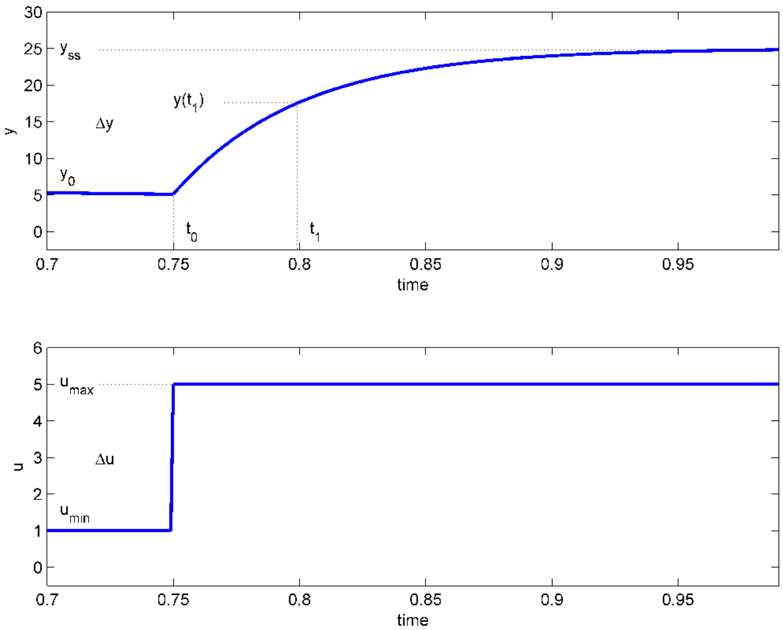
We can find the damping and thrust force gain of the half-quadrotor yaw rate transfer function model by measuring the DC gain and time constant of the response.

First-Order Response

The response of a first-order system can be described by the following transfer function:

$$\frac{Y(s)}{U(s)} = \frac{K}{\tau s + 1}$$

where $K$is the known as the steady-state, or DC gain, and τ is the time constant. The step response shown below is generated using this transfer function with $K = 5$ rad/s/V and $\tau = 0.05$ s.



The step input begins at time t_0. The input signal has a minimum value of u_{min} and a maximum value of u_{max}. The resulting output signal is initially at y_0. Once the step is applied, the output tries to follow it and eventually settles at its steady-state value y_{ss}. The steady-state gain of the system can be found from its response using the input and output signals:

$$K = \frac{\Delta y}{\Delta u}$$

where $\Delta y = y_{ss} - y_0$ and $\Delta u = u_{max} - u_{min}$. The time constant of a system, τ, is defined as the time is takes the output of system to reach $1 - e^{-1} = 63.2$% of its steady-state value from the initial step time, t_0. The point where the output reaches 63.2% of its steady-state value can be defined

$$y(t_1) = 0.632\Delta y + y_0.$$

where

$$t_1 = t_0 + \tau$$

The time t_1 that corresponds to $y(t_1)$ can be found from the response data shown in the step response figure to obtain the model time constant

$$\tau = t_1 - t_0$$

Find DC gain.

Measure the steady-state yaw rate and enter the value in y_ss.

```
% Output at start time (rad/s)
y0 = 0;
% Steady-state yaw angle rate (rad/s)
y_ss = 2.31;
% Change in output (rad/s)
dy = y_ss - y0;
% Change in input signal (V)
du = max(u)-min(u);
% DC gain (rad/s/V)
Kdc = dy/du
```

```
Kdc = 0.2310
```

Find time constant

```
% Step start time (s)
t0 = 1;
% Speed at first decay time (rad/s)
y_t1 = (1-exp(-1))*dy + y0;
% Find index of first decay (s)
i1 = find((y_t1 < psi_dot),1,'first');
% Find time of first decay (s)
t1 = t(i1);
% Experimentally derived time constant (s)
tau = t1 - t0
```

```
tau = 9.2560
```

Calculate the Thrust Force Gain and Viscous Damping

Find the damping and thrust force gain of the half-quadrotor yaw rate transfer function model from the measured DC gain and time constant as well as the Aero 2 model parameters using the equations: $\tau = \frac{J_y}{D_y}$ and $K = \frac{D_t K_f}{D_y}$.

```
% Damping (N/(rad/s))
Dy_id = Jy/tau
```

```
Dy_id = 0.0026
```

```
% Thrust (N/V)
Kf_id = Kdc*Dy_id/Dt
```

```
Kf_id = 0.0035
```

The thrust force gain parameter is $K_f = 0.0147$ N/m and viscous damping is $D_y = 0.0359$ N/m/(rad/s).

Lab 3: Model Validation

The model can be validated by running the hardware in parallel with the identified model using the `q_aero2_half_quad_model_val.slx` Simulink model in QUARC.

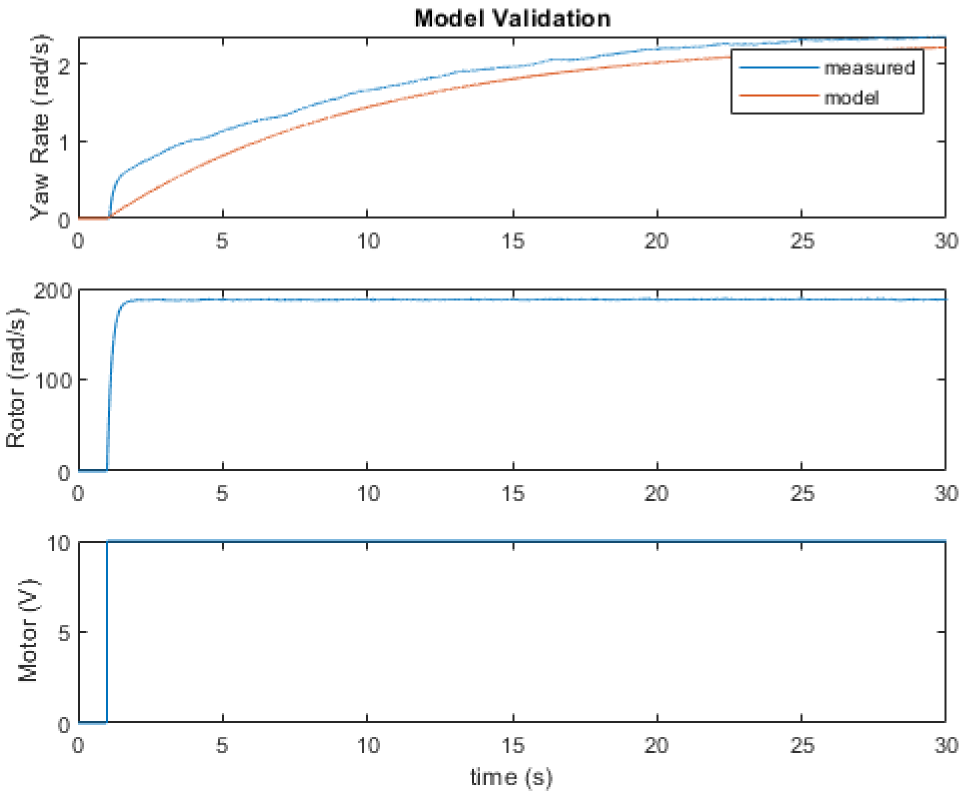
To do this:

1. Make sure with the identified parameters `Kf_id` and `Dy_id` are loaded.
2. Build and run the following Simulink model in QUARC by clicking on the *Monitor & Tune* button.

```
open("q_aero2_half_quad_model_val.slx")
```

Plot results

```
% Load measured data from past run
load('DataHalfQuadModelVal.mat');
% store in variables
t_val = DataHalfQuadModelVal(1,:); % time (s)
u_val = DataHalfQuadModelVal(2,:); % pitch/front motor voltage (V)
wm_val = DataHalfQuadModelVal(3,:); % pitch/front rotor speed (rad/s)
psi_val = DataHalfQuadModelVal(4,:); % yaw angle (rad)
psi_dot_val = DataHalfQuadModelVal(5,:); % yaw angular rate (rad/s)
psi_dot_model = DataHalfQuadModelVal(6,:); % yaw angular rate (rad/s)
%
subplot(3,1,1);
plot(t_val,psi_dot_val,t_val,psi_dot_model);
title('Model Validation');
ylabel('Yaw Rate (rad/s)');
legend('measured','model');
subplot(3,1,2);
plot(t_val,wm_val);
ylabel('Rotor (rad/s)');
subplot(3,1,3);
plot(t_val,u_val);
ylabel('Motor (V)');
xlabel('time (s)');
```



The model does not quite match the hardware response. As mentioned earlier, using a first-order model to represent the motion of the Aero 2 half-quadrotor is an *approximation*. The actual system is higher-order. For the purposes of designing a simple controller, this model is adequate. Of course, having a more representative model could lead to better results but would require a more advanced control design.