

Homework Total:

89.5 / 102

ASEN 3128 - Assignment 1

Seth Hill
Gregory Kondor
Connor Ott
Group 14 - Station 7b

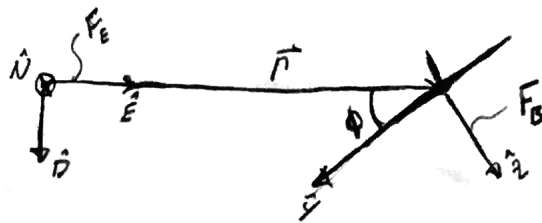
January 29, 2018
University of Colorado - Boulder

5 / 5

Problem 1

Ψ - Azimuth-rotation about \hat{D} , measured from \hat{N}

$$\boxed{\Psi = 180^\circ}$$



Θ - Elevation-rotation about \hat{E} , $\boxed{\Theta = 0^\circ}$

Φ - Bank-rotation about \hat{N} , $\boxed{\Phi = -\phi^\circ}$

Problem 2

$\dot{\Psi}$ - rotation rate about \hat{D}

1 rotation in 1 period $\Rightarrow 360^\circ / \tau$

$$\tau = \frac{2\pi \text{ rad/s}}{0.1 \text{ rad/s}} = 62.83 \text{ s} = 1 \text{ hr}$$

$$\dot{\Psi} = \frac{360^\circ}{62.83} = \boxed{5.73^\circ/\text{s} = 0.1 \text{ rad/s}}$$

$\dot{\Theta}$ - rotation rate about \hat{E}

$$\boxed{\dot{\Theta} = 0 \text{ deg/s}}$$

$\dot{\Phi}$ - rotation rate about \hat{N}

$$\dot{\Phi} = \frac{d}{dt} \Phi = \boxed{0 \text{ deg/s} = \dot{\Phi}}$$

Problem 3

P - roll rate, the rate at which the plane rotates about its body-fixed \hat{x} axis

$$\boxed{P = 0 \text{ deg/s}}$$

q - Pitch rate, the rate at which the plane rotates about its body-fixed \hat{y} axis

$$\boxed{q = 0 \text{ deg/s}}$$

r - Yaw rate, the rate at which the plane rotates about its body-fixed \hat{z} axis $\boxed{r = 0 \text{ deg/s}}$

8 / 12

Problem 4

$$\dot{p} - \hat{x} \text{ component of } \frac{d^B}{dt} \vec{\omega}^{EB} = \frac{d}{dt}(0) = 0 \text{ rad/s}^2$$

$$\dot{q} - \hat{y} \text{ component of } \frac{d^B}{dt} \vec{\omega}^{EB} = \frac{d}{dt}(0.1 \text{ rad/s}) = 0 \text{ rad/s}^2$$

$$\dot{r} - \hat{z} \text{ component of } \frac{d^B}{dt} \vec{\omega}^{EB} = \frac{d}{dt}(0.1 \text{ rad/s}) = 0 \text{ rad/s}^2$$

Problem 5

$\vec{G} = \frac{d^E}{dt} \vec{h}^E$, the net moment about the c.m. of the aircraft expressed in body frame coordinates

$$\vec{G} = \frac{d^E}{dt} \vec{I} \vec{\omega}^{EB} = \vec{I}_B \frac{d}{dt} \vec{\omega}^{EB} + \vec{\omega}^{EB} \times \vec{h}_B$$

~~$\vec{G} = \frac{d^E}{dt} \vec{I} \vec{\omega}^{EB} = \vec{I}_B \frac{d}{dt} \vec{\omega}^{EB} + \vec{\omega}^{EB} \times \vec{h}_B$~~

$$\vec{G} = \begin{bmatrix} (I_y - I_z) \dot{\phi} r \\ (I_z - I_x) \dot{\phi} p \\ (I_x - I_y) \dot{\phi} q \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\phi} \\ \dot{\phi} \end{bmatrix} \vec{0}$$

$p = 0$

$$\vec{G} = -(I_y - I_z) \dot{\phi} r$$

$$\vec{G} = -0.1^2 (I_y - I_z) \cos \phi \sin \phi \hat{x}$$

- \vec{G} is not fixed in the Inertial Frame since \hat{x} is constantly changing direction

- \vec{G} is fixed in the body frame since ϕ isn't changing

Problem 6

\vec{h} is the ^{Inertial} Angular momentum vector of the plate

$$\vec{h} = \vec{I} \vec{\omega}^{EB}$$

$$= \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0.1 \end{bmatrix}$$

$$\vec{h} = 0.1 I_z \hat{D} = 0.1 I_z (\sin\phi \hat{y} + \cos\phi \hat{z})$$

\vec{h} is ^{not} fixed in the inertial frame since $\vec{\omega}^{EB}$ is non-zero

\vec{h} is fixed in the body frame since ϕ is not changing

Problem 7

5/5

Any value dependent on mass or moment of inertia would change

These would be $\neq \vec{f}(\vec{f}_i, \vec{f}_e, \vec{f}_s)$ from assignment 1,
+ problems 1 ($\vec{p}, \vec{q}, \vec{i}$), 5 ($\vec{G}, \vec{G}_e, \vec{G}_s$) + 6 ($\vec{h}, \vec{h}_e, \vec{h}_s$)

Problem 8

Attitude will definitely induce translation on a quadcopter since it will introduce some horizontal force

Translation does not necessarily cause changes in attitude, however

Problem 9

This problem asked for a simulation of a quad copter in steady, hovering flight. Figure 1 indicates that the simulation is working for level, steady hover.

5 / 5

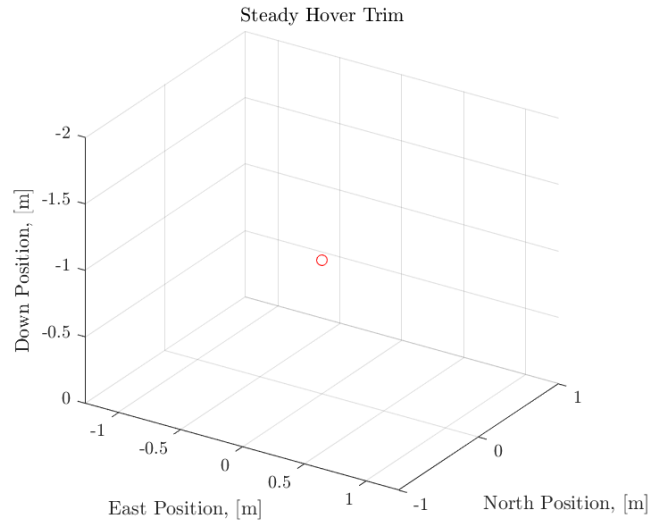


Figure 1: Steady hovering flight for quad copter.

Problem 10

This problem asked to determine trim forces and attitudes that would allow for steady flight travelling 5 m/s East. This was determined using a free body diagram and solving in MATLAB. The math is show in Figure 2. Figure 3 is a plot of the path of the quad copter.

well commented code 5/5
hover trim state verification 0/5
azimuth 0 9/10
azimuth 90 9/10

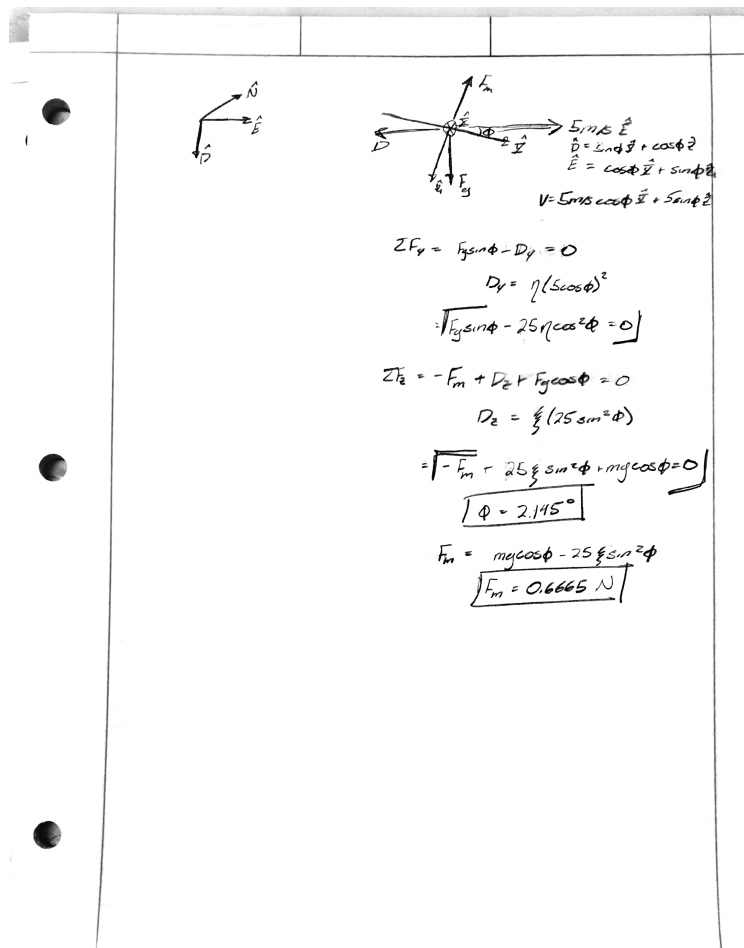


Figure 2: Math determining steady East trim forces and attitudes.

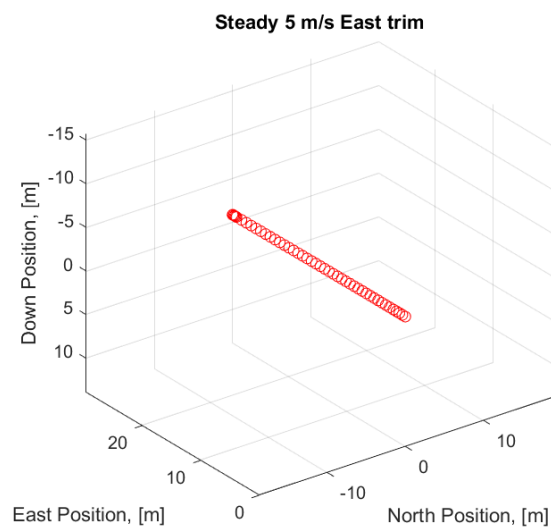


Figure 3: Steady East trim state.

For a 90 degree azimuth, the free body diagram and math remain mostly unchanged, the only difference being that the bank becomes a negative elevation angle. This case is shown in Figure 4.

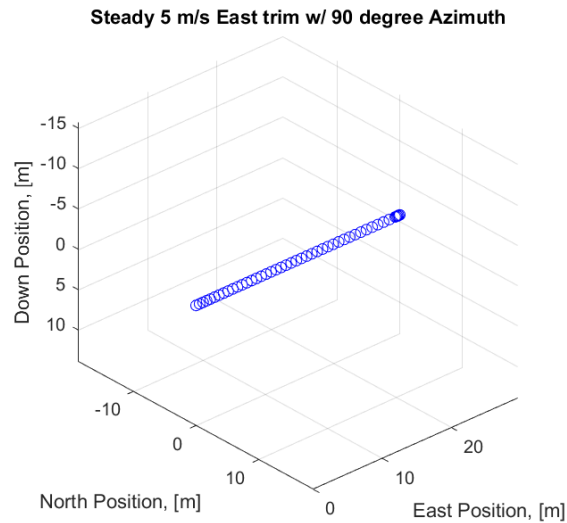


Figure 4: Steady East trim state w/ 90 degree azimuth.

Problem 11

5 / 5

This problem asked to determine if steady hovering flight is a stable state for the quad copter. It was seen to be unstable in both experiment and in simulation. In the simulation, a random moment was applied for 0.2 seconds on the steady hovering state, and the trajectory was plotted. This is shown in Figure 5. In experiment, the quad copter was set in a hover state and its feedback control was turned off. The experimental results are shown in Figure 6 the trajectory indicate the instability of the system without feedback control.

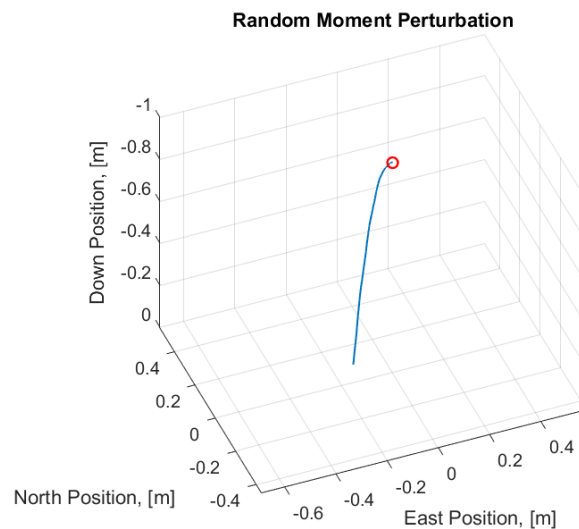


Figure 5: Simulation with random moment applied to it for 0.2 seconds.

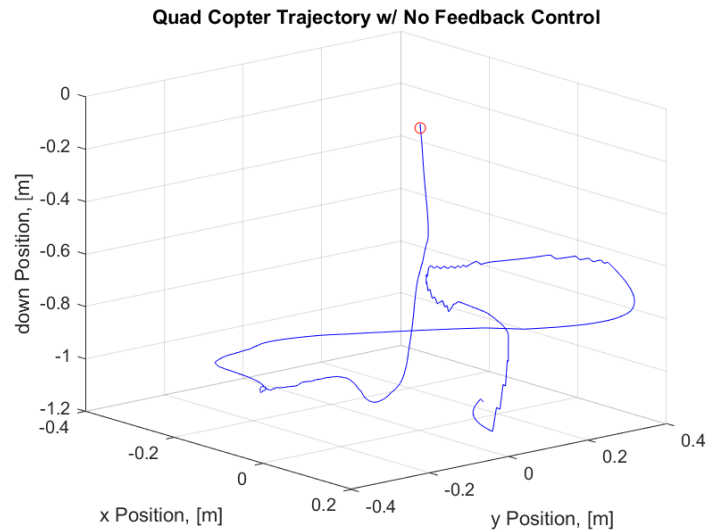


Figure 6: Experimental quad copter data.

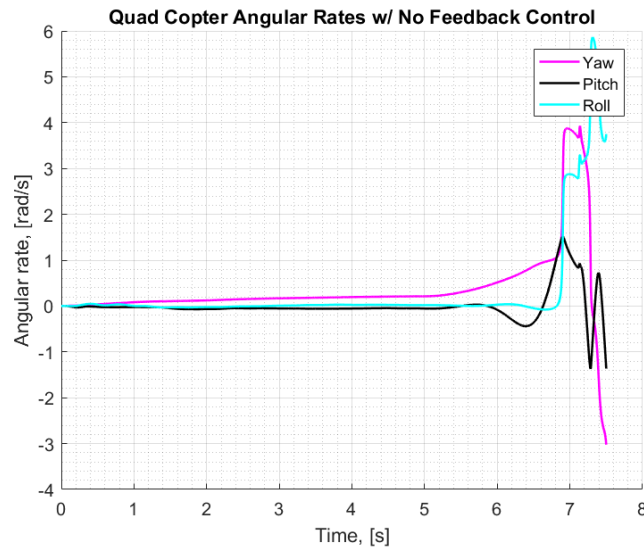


Figure 7: Experimental quad copter data.

MATLAB Code

```

1 %
2 % quadCopterSim simulates the dynamics of a small quad copter using
3 % numerical integration of Euler's Moment Equations
4 %
5 % Created: 1/30/18 – Connor Ott
6 % Last Modified: 1/30/18 – Connor Ott
7 %
8
9 clc; clear; close all;
10
11 %% Hover Trim

```



```

12 ti = 0;
13 tf = 5; % s
14 momPert = zeros(1, 3); % No perturbation
15 analyticalTrim = ones(1, 4) * 0.068 * 9.81 / 4;
16 initConds = zeros(1, 12);
17 initConds(12) = -1;
18
19 options = odeset('Events', @termEvents, 'RelTol', 1e-8);
20 [t, F] = ode45(@(t, F)quadCopterODE(t, F, analyticalTrim, momPert), ...
21               [ti, tf], initConds, options);
22
23 figure
24 hold on; grid on; axis equal;
25 plot3(F(:, 10), F(:, 11), F(:, 12), 'ro')
26 set(gca, 'zdir', 'reverse')
27 zlabel('Down Position, [m]')
28 xlabel('East Position, [m]')
29 ylabel('North Position, [m]')
30 title('Steady Hover Trim')
31 set(gca, 'ticklabelinterpreter', 'latex');
32
33 %% 5 m/s East trim
34 ti = 0;
35 tf = 5; % [s]
36
37 momPert = zeros(1, 3); % No perturbation
38 V = 5; % [m/s]
39 heta = 1e-3; % [N/(m/s)^2]
40 xsi = 3e-3; % [N/(m/s)^2]
41 m = 0.068; % [kg]
42 g = 9.81; % [m/s^2]
43 solveF = @(x) m*g*sin(x) - 25*heta*cos(x).^2;
44 phi = fzero(solveF, 0);
45 f_Mag = m*g*cos(phi) + 25*xsi*sin(phi).^2;
46
47 steadyTrim_East = ones(1, 4) * f_Mag / 4;
48 initConds = zeros(1, 12);
49 initConds(12) = -1;
50 initConds(2) = V*cos(phi);
51 initConds(3) = -V*sin(phi);
52 initConds(7) = phi;
53
54 options = odeset('Events', @termEvents, 'RelTol', 1e-8);
55 [t, F] = ode45(@(t, F)quadCopterODE(t, F, steadyTrim_East, momPert), ...
56               [ti, tf], initConds, options);
57
58 figure
59 hold on; grid on; axis equal;
60 plot3(F(:, 10), F(:, 11), F(:, 12), 'bo')
61 set(gca, 'zdir', 'reverse')
62 zlabel('Down Position, [m]')
63 ylabel('East Position, [m]')
64 xlabel('North Position, [m]')
65 title('Steady 5 m/s East trim w/ 90 degree Azimuth')
66

```

```

67 %% Velocity Plots
68 figure
69 subplot(3, 1, 1)
70 hold on; grid on; grid minor;
71 plot(t, F(:, 1), 'b-', 'linewidth', 1.2)
72 xlabel('Time, [s]')
73 ylabel('North Velocity, [m/s]')
74
75 subplot(3, 1, 2)
76 hold on; grid on; grid minor;
77 plot(t, F(:, 2), 'b-', 'linewidth', 1.2)
78 xlabel('Time, [s]')
79 ylabel('East Velocity, [m/s]')
80
81 subplot(3, 1, 3)
82 hold on; grid on; grid minor;
83 plot(t, F(:, 3), 'b-', 'linewidth', 1.2)
84 xlabel('Time, [s]')
85 ylabel('Down Velocity, [m/s]')
86
87 %% Position Plots
88 figure
89 subplot(3, 1, 1)
90 hold on; grid on; grid minor;
91 plot(t, F(:, 10), 'm-', 'linewidth', 1.2)
92 xlabel('Time, [s]')
93 ylabel('North Position, [m/s]')
94
95 subplot(3, 1, 2)
96 hold on; grid on; grid minor;
97 plot(t, F(:, 11), 'm-', 'linewidth', 1.2)
98 xlabel('Time, [s]')
99 ylabel('East Position, [m/s]')
100
101 subplot(3, 1, 3)
102 hold on; grid on; grid minor;
103 plot(t, F(:, 12), 'm-', 'linewidth', 1.2)
104 xlabel('Time, [s]')
105 ylabel('Down Position, [m/s]')
106
107 %% Repeat of above with azimuth at 90 deg
108 % It travels north now, since the body frame has been rotated and the bank
109 % angle no longer causes motion in East direction.
110
111 ti = 0;
112 tf = 5; % [s]
113
114 momPert = zeros(1, 3); % No perturbation
115 u = 5; % [m/s]
116 heta = 1e-3; % [N/(rad/s)^2]
117 m = 0.068; % [kg]
118 g = 9.81; % [m/s^2]
119 theta = atan((heta^2 * u^2)/(m*g)); % [rad] - Bank
120 psi = pi/2; % [rad] - Azimuth
121 f.Mag = sqrt((m*g)^2 + (heta^2*u^2)^2); %[N]

```

```

122
123 steadyTrim_East = ones(1, 4) * f_Mag / 4;
124 initConds = zeros(1, 12);
125 initConds(12) = -1;
126 initConds(1) = u;
127 initConds(8) = theta;
128 initConds(9) = psi;
129
130 options = odeset('Events', @termEvents, 'RelTol', 1e-8);
131 [t, F] = ode45(@(t, F)quadCopterODE(t, F, steadyTrim_East, momPert), ...
132               [ti, tf], initConds, options);
133
134
135 figure
136 hold on; grid on; axis equal;
137 plot3(F(:, 10), F(:, 11), F(:, 12), 'ro')
138 set(gca, 'zdir', 'reverse')
139 zlabel('Down Position, [m]')
140 xlabel('East Position, [m]')
141 ylabel('North Position, [m]')
142 title('Steady 5 m/s East trim - 90 deg azimuth')
143
144 %% Position Plots
145 figure
146 subplot(3, 1, 1)
147 hold on; grid on; grid minor;
148 plot(t, F(:, 10), 'm-', 'linewidth', 1.2)
149 xlabel('Time, [s]')
150 ylabel('East Position, [m/s]')
151
152 subplot(3, 1, 2)
153 hold on; grid on; grid minor;
154 plot(t, F(:, 11), 'm-', 'linewidth', 1.2)
155 xlabel('Time, [s]')
156 ylabel('North Position, [m/s]')
157 axis([0, 5, -0.5, 0.5])
158
159 subplot(3, 1, 3)
160 hold on; grid on; grid minor;
161 plot(t, F(:, 12), 'm-', 'linewidth', 1.2)
162 xlabel('Time, [s]')
163 ylabel('Down Position, [m/s]')
164 axis([0, 5, -1.5, -0.5])
165
166 %% Simulate a perturbation in steady hover to see how quad copter reacts
167
168 ti = 0;
169 tf = 5; % s
170 analyticalTrim = ones(1, 4) * 0.068 * 9.81 / 4;
171 initConds = zeros(1, 12);
172 initConds(12) = -1;
173
174 figure
175 hold on; grid on; axis equal;
176 set(gca, 'zdir', 'reverse')

```

```

177 zlabel('Down Position, [m]')
178 xlabel('East Position, [m]')
179 ylabel('North Position, [m]')
180 title('Random Moment Perturbation')
181 plot3(0, 0, -1, 'ro', 'linewidth', 1.2)
182 momPert = randn(3, 1)./10;
183 options = odeset('Events', @termEvents, 'RelTol', 1e-8);
184 [t, F] = ode45(@(t, F)quadCopterODE(t, F, analyticalTrim, momPert), ...
185               [ti, tf], initConds, options);
186
187 plot3(F(:, 10), F(:, 11), F(:, 12), '-', 'linewidth', 1.1)
188
189 %% Plotting experimental quad copter data
190 load('RSdata_Drone01_1330.mat');
191 times = rt_estim.time(:);
192 xdata = rt_estim.signals.values(:, 1);
193 ydata = rt_estim.signals.values(:, 2);
194 zdata = rt_estim.signals.values(:, 3);
195
196 figure
197 hold on; grid on;
198 plot3(xdata(1), ydata(1), zdata(1), 'ro')
199 plot3(xdata, ydata, zdata, 'b-')
200 title('Quad Copter Trajectory w/ No Feedback Control')
201 xlabel('x Position, [m]')
202 ylabel('y Position, [m]')
203 zlabel('down Position, [m]')

1 function dfdt = quadCopterODE(t, F, trim, momPert)
2 % Defines dynamics of quad copter
3
4
5 %% Physical properties and constants
6 alpha = 2e-6; % [N/(m/s)^2]
7 beta = 1e-6; % [N/(m/s)^2]
8 heta = 1e-3; % [N/(rad/s)^2]
9 xsi = 3e-3; % [N/(rad/s)^2]
10
11 I_x = 6.8e-5; % [kg m^2]
12 I_y = 9.2e-5; % ['']
13 I_z = 1.35e-4; % ['']
14
15 m = 0.068; % [kg]
16 d = 0.06; % [m]
17 k = 0.0024; % [~]
18 rad = d/sqrt(2); % [m]
19 g = 9.81; % [m/s^2]
20
21 %% Pulling from input, F and trim
22 % Inertial velocity in body coords
23 u = F(1);
24 v = F(2);
25 w = F(3);
26
27 % Inertial angular velocity in body coords
28 p = F(4);

```

```

29 q = F(5);
30 r = F(6);
31
32 % Euler angles
33 phi = F(7);
34 theta = F(8);
35 psi = F(9);
36
37 % Position vector from origin to COM in inertial coords
38 x_E = F(10);
39 y_E = F(11);
40 z_e = F(12);
41
42 f1 = trim(1); % motor forces
43 f2 = trim(2);
44 f3 = trim(3);
45 f4 = trim(4);
46
47 %% Aerodynamic and Control Moments and Forces
48
49 % Moments
50 L_a = - alpha^2 * p^2 * sign(p);
51 M_a = - alpha^2 * q^2 * sign(q);
52 N_a = - beta^2 * r^2 * sign(r);
53
54 L_c = ((f2 + f3) - (f1 + f4)) * rad;
55 M_c = ((f3 + f4) - (f1 + f2)) * rad;
56 N_c = (f2 + f4 - (f1 + f2)) * k;
57
58 L = L_a + L_c;
59 M = M_a + M_c;
60 N = N_a + N_c;
61
62 if t > 1 && t < 1.2 % Throw a moment perturbation in for 0.2 seconds
63     L = L + momPert(1);
64     M = M + momPert(2);
65     N = N + momPert(3);
66 end
67
68 % Forces
69 X_a = - heta^2 * u^2 * sign(u);
70 Y_a = - heta^2 * v^2 * sign(v);
71 Z_a = - xsi^2 * w^2 * sign(w);
72
73 X_c = 0;
74 Y_c = 0;
75 Z_c = -sum(trim);
76
77 X = X_a + X_c;
78 Y = Y_a + Y_c;
79 Z = Z_a + Z_c;
80
81 % Determining Roll, Pitch, and Yaw rates of change
82 p_dot = (I_y - I_z)/I_x * q * r + 1/I_x * L;
83 q_dot = (I_z - I_x)/I_y * p * r + 1/I_y * M;

```

```

84 r_dot = (I_x - I_y)/I_z * p * q + 1/I_z * N;
85 dOmega_bdt = [p_dot, q_dot, r_dot]';
86
87 % Determining translation rates
88 u_dot = r*v - q*w - g*sin(theta) + 1/m * X;
89 v_dot = p*w - r*u + g*sin(phi)*cos(theta) + 1/m * Y;
90 w_dot = q*u - p*v + g*cos(phi)*cos(theta) + 1/m * Z;
91 dV_bdt = [u_dot, v_dot, w_dot]';
92
93 % Tranlating body to inertial coordinates
94 x_dot = u * cos(theta)*cos(psi) + ...
95         v * (sin(phi)*sin(theta)*cos(psi) - cos(phi)*sin(psi)) + ...
96         w * (cos(phi)*sin(theta)*cos(psi) + sin(phi)*sin(psi));
97 y_dot = u * cos(theta)*sin(psi) + ...
98         v * (sin(phi)*sin(theta)*sin(psi) + cos(phi)*cos(psi)) + ...
99         w * (cos(phi)*sin(theta)*sin(psi) - sin(phi)*cos(psi));
100 z_dot = -u * sin(theta) + ...
101         v * sin(phi)*cos(theta) + ...
102         w * cos(phi)*cos(theta);
103 dV_Edt = [x_dot, y_dot, z_dot]';
104
105 % Translating Angular Momentum to Euler angles
106 phi_dot = p + (q*sin(phi)+ r*cos(phi))*tan(theta);
107 theta_dot = q*cos(phi) - r*sin(phi);
108 psi_dot = q*sin(phi)*sec(theta) + r*cos(phi)*sec(theta);
109 dEuldt = [phi_dot, theta_dot, psi_dot]';
110
111 % Concatenating for output
112 dfdt = [dV_bdt; dOmega_bdt; dEuldt; dV_Edt];
113
114
115
116 end

1 function [value, isTerm, direction] = termEvents(t, F)
2 % Define events for quad copter ODE
3 value = F(12);
4 isTerm = 1;
5 direction = [];
6 end

```