

ASEN 3128 - Lab 4

Cole MacPherson, Joshua Schmitz, Nick Herrington, Panitnan Yuvanondha

March 21st, 2020

Problem 1

Below are the calculation by hands for finding the gain values for lateral and longitudinal motion.

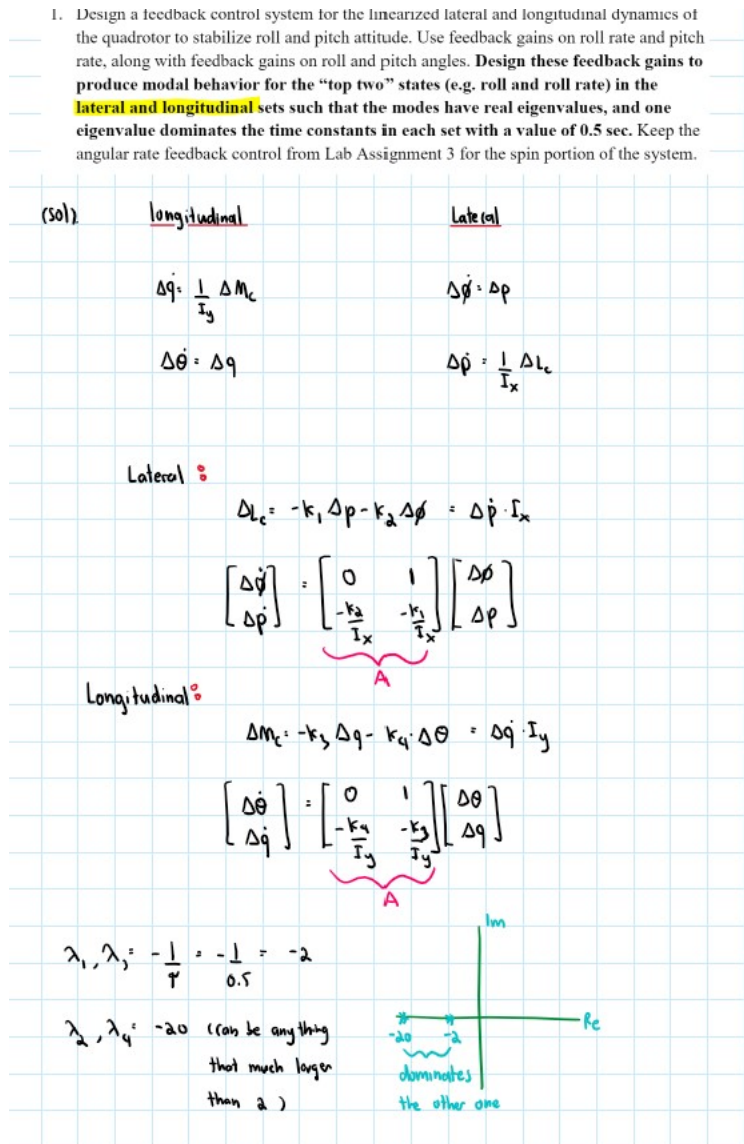


Fig. 1 Question 1 Calculation

Lateral :

$$\Delta L_c = -k_1 \Delta p - k_2 \Delta \phi = \Delta \dot{p} \cdot I_x$$

$$I_x \ddot{\phi} = -k_1 \Delta p - k_2 \Delta \phi$$

$$\ddot{\phi} + \frac{k_1}{I_x} \dot{\phi} + \frac{k_2}{I_x} \phi = 0$$

$$\mathcal{L} \left[\ddot{\phi} + \frac{k_1}{I_x} \dot{\phi} + \frac{k_2}{I_x} \phi = 0 \right]$$

$$s^2 + \frac{k_1}{I_x} s + \frac{k_2}{I_x} = 0$$

$$\frac{C}{s^2 + \frac{k_1}{I_x} s + \frac{k_2}{I_x}} = \frac{C}{(s - \lambda_1)(s - \lambda_2)}$$

$$= \frac{C}{s^2 - (\lambda_1 - \lambda_2)s + \lambda_1 \lambda_2}$$

$$\frac{k_1}{I_x} = -(\lambda_1 - \lambda_2)$$

$$k_1 = -I_x(\lambda_1 - \lambda_2) = -5.8 \cdot 10^{-5}(-20 - 2)$$

$$k_1 = 0.001276$$

$$\frac{k_2}{I_x} = \lambda_1 \lambda_2$$

$$k_2 = I_x \lambda_1 \lambda_2 = 5.8 \cdot 10^{-5}(20 \cdot 2)$$

$$k_2 = 0.00232$$

Fig. 2 Question 1 Calculation

Longitudinal: (same steps except change I_x to I_y)

$$\Delta M_c = -k_3 \Delta q - k_4 \Delta \theta = I_y \Delta \ddot{q}$$

$$I_y \ddot{\theta} + k_3 \dot{\theta} + k_4 \theta = 0$$

$$\mathcal{L}[I_y \ddot{\theta} + k_3 \dot{\theta} + k_4 \theta = 0]$$

$$s^2 + \frac{k_3}{I_y} s + \frac{k_4}{I_y} = 0$$

$$\frac{C}{s^2 + \frac{k_3}{I_y} s + \frac{k_4}{I_y}} = \frac{C}{(s - \lambda_3)(s - \lambda_4)}$$

$$= \frac{C}{s^2 - (\lambda_3 - \lambda_4)s + \lambda_3 \lambda_4}$$

$$k_3 = -I_y(\lambda_3 - \lambda_4) = -7.2 \cdot 10^{-5}(-22)$$

$$k_3 = 0.001584$$

$$k_4 = I_y(\lambda_3 \lambda_4) = 7.2 \cdot 10^{-5} \cdot 40$$

$$k_4 = 0.00288$$

Fig. 3 Question 1 Calculation

Problem 2

Deviation by $+5^\circ$ in Roll [2.a]

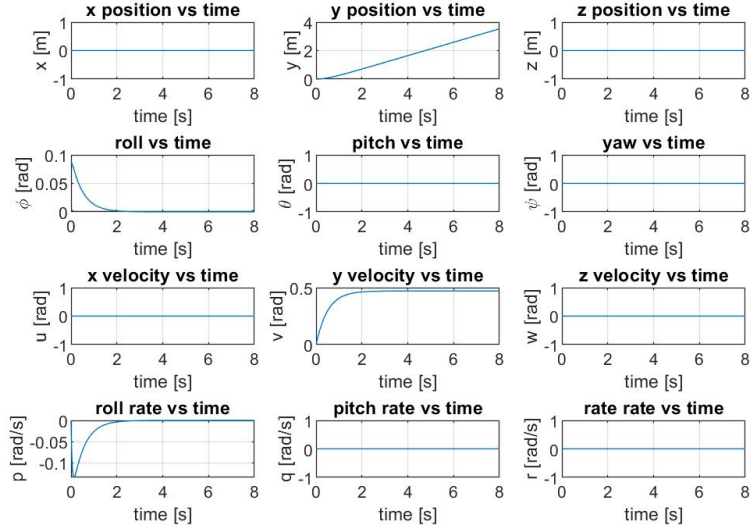


Fig. 4 Linearized Roll

Deviation by $+5^\circ$ in Pitch [2.b]

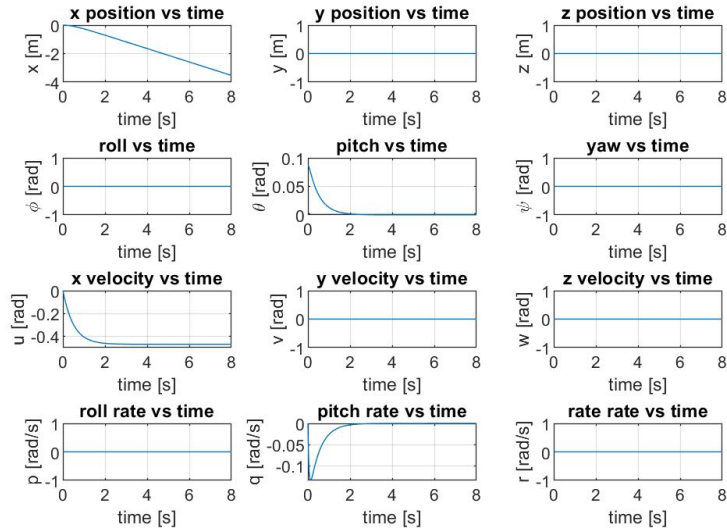


Fig. 5 Linearized Pitch

Deviation by +0.1 rad/s in Roll Rate [2.c]

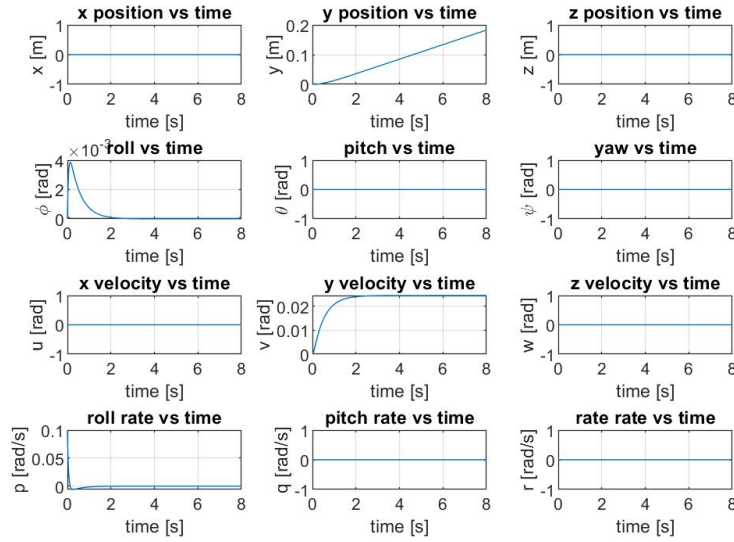


Fig. 6 Linearized Roll Rate

Deviation by +0.1 rad/s in Pitch Rate [2.d]

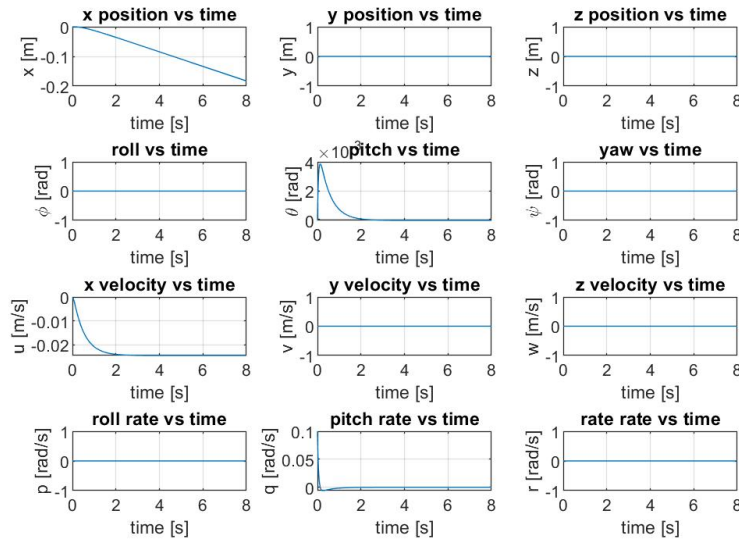


Fig. 7 Linearized Pitch Rate

The results for the linearized model make sense because when we add a perturbation to the quadcopter, the altitude doesn't change. This can be seen in the above graphs, the z-components do not change and remain zero for the duration of the ODE simulation. This is a result from the linearization of the equations of motion. Since we are looking at infinitesimal time steps, it appears that the quadcopter's altitude is not changing with time.

Problem 3

Deviation by $+5^\circ$ in Roll [3.a]

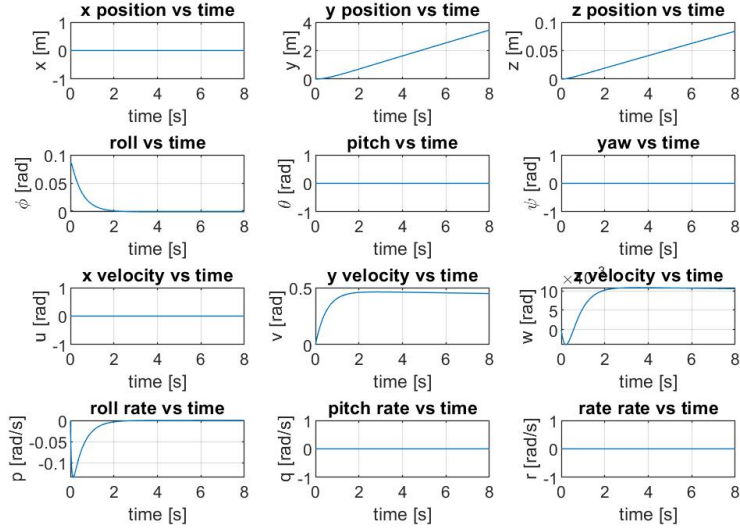


Fig. 8 Non-Linearized Roll

Deviation by $+5^\circ$ in Pitch [3.b]

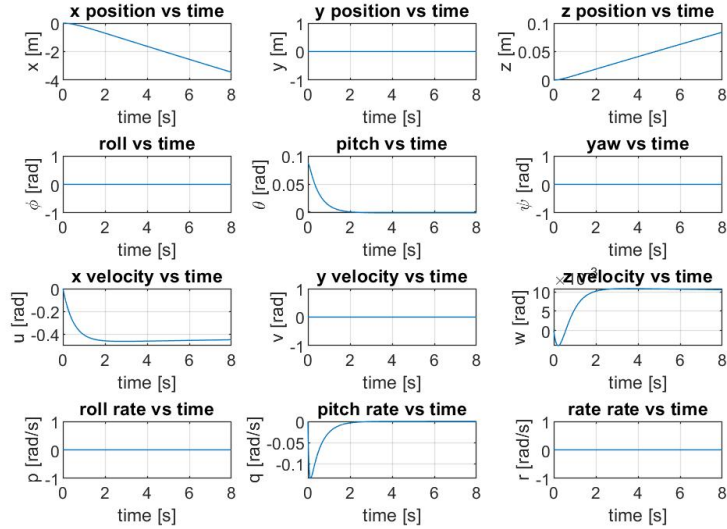


Fig. 9 Non-Linearized Pitch

Deviation by +0.1 rad/s in Roll Rate [3.c]

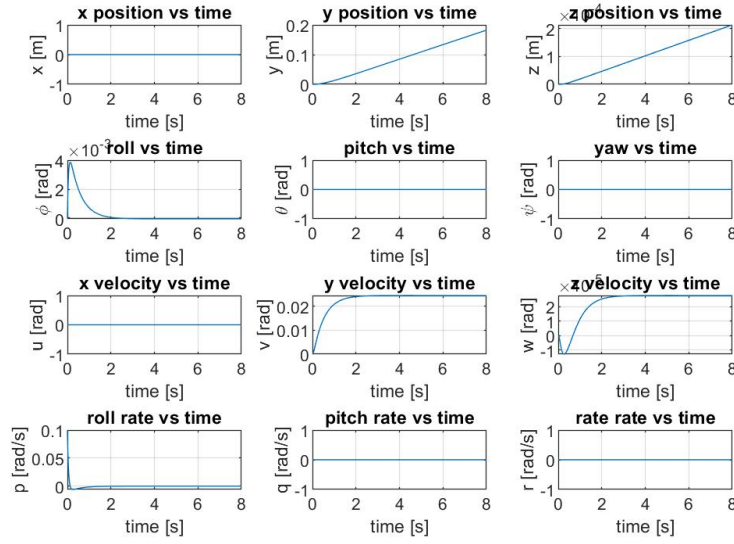


Fig. 10 Non-Linearized Roll Rate

Deviation by +0.1 rad/s in Pitch Rate [3.d]

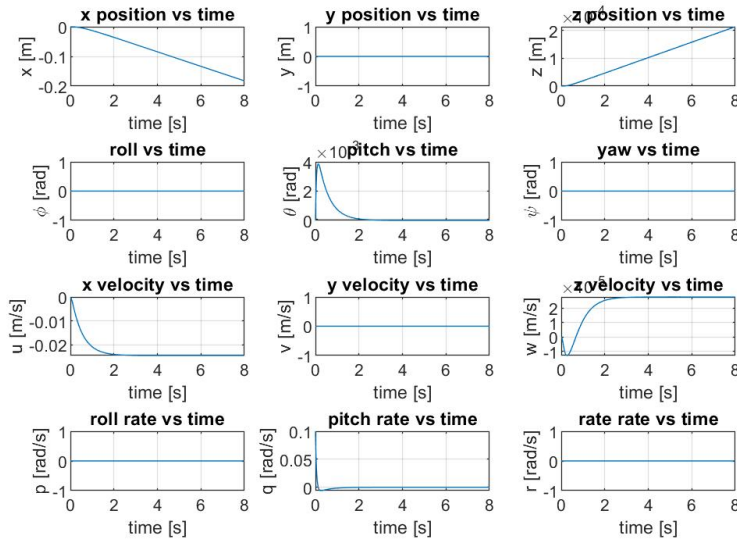


Fig. 11 Non-Linearized Pitch Rate

When modeling the motion of the quadcopter with the non-linearized EOM's, we get similar results as to the linearized model except when get changes in the z-components which is expected. This model verifies that we correctly performed the calculations for the linearized model since the x-components and y-components followed similar changes in the graphs.

Problem 4

The following plots are the eigenvalues obtained from the MATLAB built-in function eig over the range of k_3 values for both lateral and longitudinal motion in the complex plane. In order to meet the design objective, the real eigenvalue must be around 0.8 since the time constant given is 1.25 sec ($\tau = \frac{1}{1.25}$), therefore the k_3 value is picked at the location where the real eigenvalue is or close to 0.8. From doing this, the k_3 value is 0.009 for lateral and 0.0076 for longitudinal.

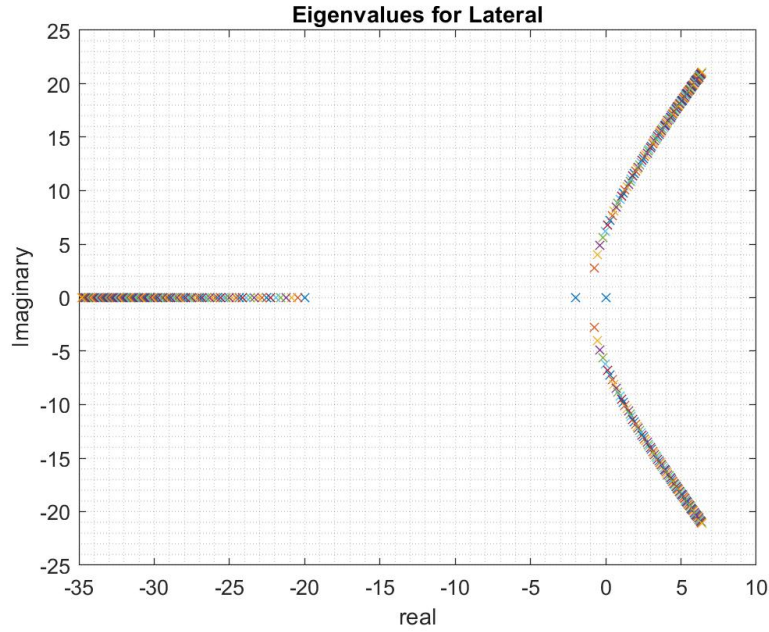


Fig. 12 Eigenvalues for Lateral

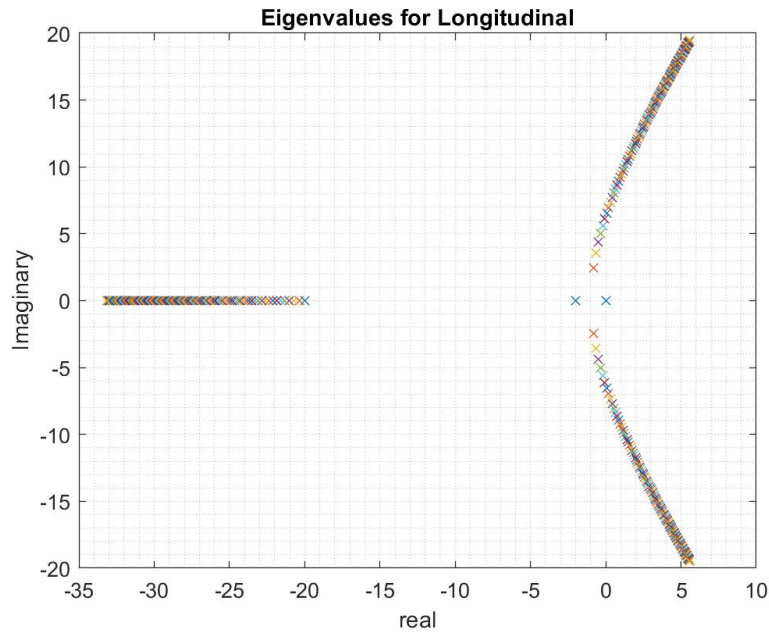


Fig. 13 Eigenvalues for Longitudinal

Problem 5

Problem 5 Model (Lateral)

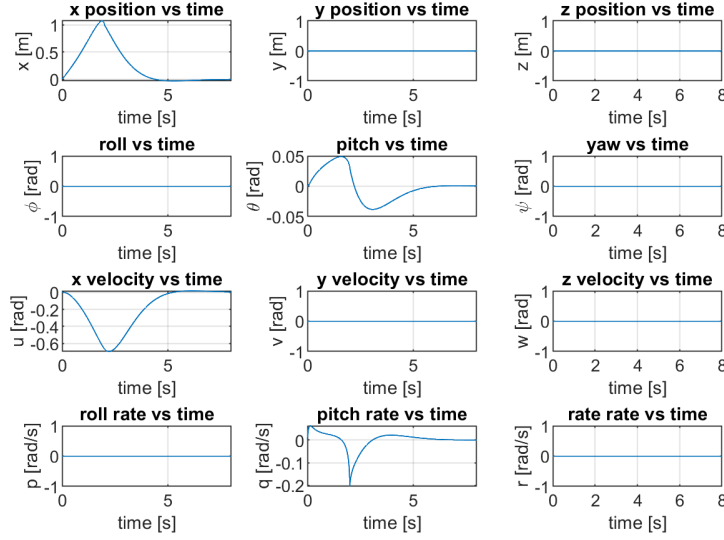


Fig. 14 Question 5 lateral Simulation

Problem 5 Model (Longitudinal)

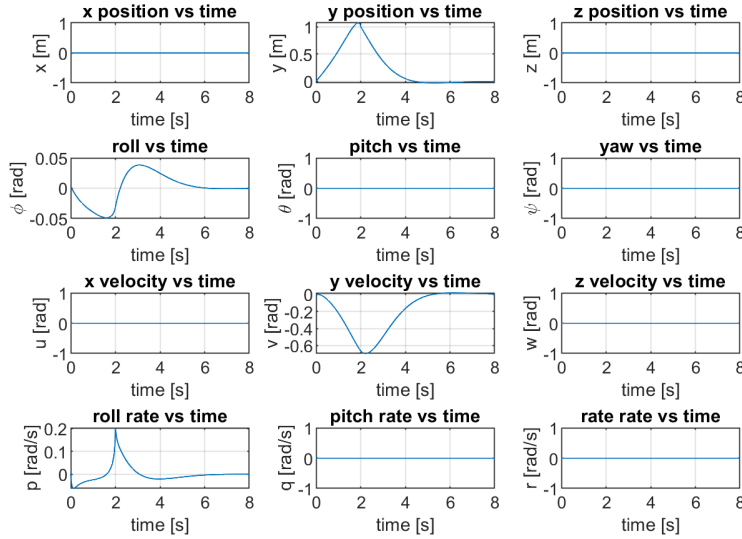


Fig. 15 Question 5 longitudinal Simulation

As seen in figures 14 and 15, the question 5 simulation takes in a steady state hover state vector in and moves the drone to a position of 1 meter in both the y and x directions for separate simulations. After this the drone then steadies itself out to go back to a steady hover state. This simulation can be confirmed as correct as the values presented in figures 14 and 15 act as expected through the simulation and the beginning and end states are steady hover states.

Appendix

A. Team Participation Table

Name	Plan	Model	Experiment	Results	Report	Code	Ack
Cole MacPherson	1	X	X	2	1	2	CM
Joshua Schmitz	1	X	X	1	1	2	JS
Nick Herrington	1	X	X	1	2	1	NH
Panitan Yuvanondha	1	X	X	1	2	1	PY

B. Code

This script is used for problems 2 through 5 and calls the required functions shown after this script:

```
1 %% ASEN 3128 - Lab 4 - Main
2 % Script to compare the lineared and non-linearized control responses of a
3 % quadrotor with a feedback loop incorporated into the equations of motion
4 %
5 %
6 % Author: Cole MacPherson
7 % Collaborators: N. Herrington, J. Schmitz, P. Yuvanondha
8 % Date: 5th Mar 2021
9
10 %% Housekeeping
11 clc;
12 clear;
13 close all;
14 tic
15
16 %% declare constants
17 m = 0.068; % mass of the quadrotor [kg]
18 R = 0.06; % radial distance from CG to propeller [m]
19 k_m = 0.0024; % control moment coefficient [N*m/N]
20 I_x = 5.8e-5; % x-axis moment of inertia [kg*m^2]
21 I_y = 7.2e-5; % y-axis moment of inertia [kg*m^2]
22 I_z = 1.0e-4; % z-axis moment of inertia [kg*m^2]
23 nu = 1e-3; % aerodynamic force coefficient [N/(m/s)^2]
24 mu = 2e-6; % aerodynamic moment coefficient [N*m/(rad/s)^2]
25 g = 9.81; % gravitational constant [m/s^2]
26 lambda_1 = -2; % 1st eigenvalue
27 lambda_2 = -20; % 2nd eigenvalue
28
29 %% Problem 1
30 % Solve for k_1 and k_2 (Lateral)
31 syms k_1 k_2 % declare k1 and k2 as symbolic variables
32 eqn1 = lambda_1^2 + lambda_1*(k_1/I_x) + (k_2/I_x) == 0; % 1st eigenvalue solution
33 eqn2 = lambda_2^2 + lambda_2*(k_1/I_x) + (k_2/I_x) == 0; % 2nd eigenvalue solution
34
35 [A,b] = equationsToMatrix([eqn1, eqn2],[k_1, k_2]); % convert equations to a matrix
36
37 x_lat = double(A\b); % solve the system of equations
38 k_1 = x_lat(1); % define k_1 value from system solutions
39 k_2 = x_lat(2); % define k_2 value from system solutions
40 k_3 = 0;
41
42 % Solve for k_3 and k_4 (Longitudinal)
43 syms k_4 k_5 % declare k1 and k2 as symbolic variables
44 eqn1 = lambda_1^2 + lambda_1*(k_4/I_y) + (k_5/I_y) == 0; % 1st eigenvalue solution
45 eqn2 = lambda_2^2 + lambda_2*(k_4/I_y) + (k_5/I_y) == 0; % 2nd eigenvalue solution
46
47 [C,d] = equationsToMatrix([eqn1, eqn2],[k_4, k_5]); % convert equations to a matrix
```

```

48 x_long = double(C\d); % solve the system of equations
49 k_4 = x_long(1); % define k_3 value from system solutions
50 k_5 = x_long(2); % define k_4 value from system solutions
51 k_6 = 0;
52
53
54 %% Problem 2 (Closed-Loop Linearized)
55 % define known vector to pass into function
56 knowns = [mu nu I_x I_y I_z m g k_1 k_2 k_3 k_4 k_5 k_6 R k_m 0];
57 % define time span for ode45
58 t_span = [0 8];
59 % define motor trim forces
60 forces = (1/4)*m*g.*ones(1,4);
61 % define perturbations
62 perturbations = zeros(1,3);
63
64 % a (+5 deg in roll)
65 state_vec = zeros(1,12);
66 state_vec(4) = deg2rad(5);
67
68 [t_2a, state_2a] = ode45(@(t_2a,state_2a) linearizedEOM(t_2a,state_2a,forces,perturbations,knowns
    ),t_span,state_vec);
69
70 % b (+5 deg in pitch)
71 state_vec = zeros(1,12);
72 state_vec(5) = deg2rad(5);
73
74 [t_2b, state_2b] = ode45(@(t_2b,state_2b) linearizedEOM(t_2b,state_2b,forces,perturbations,knowns
    ),t_span,state_vec);
75
76 % c (+0.1 rad/sec in roll rate)
77 [p,q,r] = rollrate2pqr(0,0,0,0.1,0,0);
78 state_vec = zeros(1,12);
79 state_vec(10:12) = [p,q,r];
80
81 [t_2c, state_2c] = ode45(@(t_2c,state_2c) linearizedEOM(t_2c,state_2c,forces,perturbations,knowns
    ),t_span,state_vec);
82
83 % d (+0.1 rad/sec in pitch rate)
84 [p,q,r] = rollrate2pqr(0,0,0,0,0.1,0);
85 state_vec = zeros(1,12);
86 state_vec(10:12) = [p,q,r];
87
88 [t_2d, state_2d] = ode45(@(t_2d,state_2d) linearizedEOM(t_2d,state_2d,forces,perturbations,knowns
    ),t_span,state_vec);
89
90 %% Problem 2 Plots
91 plotP2N3(t_2a,state_2a,t_2b,state_2b,t_2c,state_2c,t_2d,state_2d,'2');
92
93 %% Problem 3 (Non-Linearized)
94 % a (+5 deg in roll)
95 state_vec = zeros(1,12);
96 state_vec(4) = deg2rad(5);
97 perturbations(2) = 0.5;
98
99 [t_3a, state_3a] = ode45(@(t_3a,state_3a) quadrotorODE(t_3a,state_3a,forces,knowns),t_span,
    state_vec);
100
101 % b (+5 deg in pitch)
102 state_vec = zeros(1,12);
103 state_vec(5) = deg2rad(5);
104
105 [t_3b, state_3b] = ode45(@(t_3b,state_3b) quadrotorODE(t_3b,state_3b,forces,knowns),t_span,
    state_vec);
106
107 % c (+0.1 rad/sec in roll rate)
108 [p,q,r] = rollrate2pqr(0,0,0,0.1,0,0);
109 state_vec = zeros(1,12);

```

```

110 state_vec(10:12) = [p,q,r];
111
112 [t_3c, state_3c] = ode45(@(t_3c,state_3c) quadrotorODE(t_3c,state_3c,forces,knowns),t_span,
    state_vec);
113
114 % d (+0.1 rad/sec in pitch rate)
115 [p,q,r] = rollrate2pqr(0,0,0,0,0.1,0);
116 state_vec = zeros(1,12);
117 state_vec(10:12) = [p,q,r];
118
119 [t_3d, state_3d] = ode45(@(t_3d,state_3d) quadrotorODE(t_3d,state_3d,forces,knowns),t_span,
    state_vec);
120
121 %% Problem 3 Plots
122 plotP2N3(t_3a,state_3a,t_3b,state_3b,t_3c,state_3c,t_3d,state_3d,'3');
123
124 %% Problem 4
125 k_3 = -18.475e-5; % found through guess and check
126 k_6 = 22.95e-5; % found through guess and check
127
128 % define known vector to pass into function
129 knowns = [mu nu I_x I_y I_z m g k_1 k_2 k_3 k_4 k_5 k_6 R k_m 0];
130
131 % a (+5 deg in roll)
132 state_vec = zeros(1,12);
133 state_vec(4) = deg2rad(5);
134
135 [t_4a, state_4a] = ode45(@(t_4a,state_4a) linearizedEOM(t_4a,state_4a,forces,perturbations,knowns
    ),t_span,state_vec);
136
137 % b (+5 deg in pitch)
138 state_vec = zeros(1,12);
139 state_vec(5) = deg2rad(5);
140
141 [t_4b, state_4b] = ode45(@(t_4b,state_4b) linearizedEOM(t_4b,state_4b,forces,perturbations,knowns
    ),t_span,state_vec);
142
143 % c (+0.1 rad/sec in roll rate)
144 [p,q,r] = rollrate2pqr(0,0,0,0.1,0,0);
145 state_vec = zeros(1,12);
146 state_vec(10:12) = [p,q,r];
147
148 [t_4c, state_4c] = ode45(@(t_4c,state_4c) linearizedEOM(t_4c,state_4c,forces,perturbations,knowns
    ),t_span,state_vec);
149
150 % d (+0.1 rad/sec in pitch rate)
151 [p,q,r] = rollrate2pqr(0,0,0,0,0.1,0);
152 state_vec = zeros(1,12);
153 state_vec(10:12) = [p,q,r];
154
155 [t_4d, state_4d] = ode45(@(t_4d,state_4d) linearizedEOM(t_4d,state_4d,forces,perturbations,knowns
    ),t_span,state_vec);
156
157 %% Problem 4 Plots
158 plotP2N3(t_4a,state_4a,t_4b,state_4b,t_4c,state_4c,t_4d,state_4d,'4');
159
160 %% Problem 5
161
162 % define known vector to pass into function
163 knowns = [mu nu I_x I_y I_z m g k_1 k_2 k_3 k_4 k_5 k_6 R k_m 1 1 0 2];
164
165 % a lateral
166 state_vec = zeros(1,12);
167
168 [t_5a, state_5a] = ode45(@(t_5a,state_5a) linearizedEOM(t_5a,state_5a,forces,perturbations,knowns
    ),t_span,state_vec);
169
170 % define known vector to pass into function

```

```

171 knowns = [mu nu I_x I_y I_z m g k_1 k_2 k_3 k_4 k_5 k_6 R k_m 1 0 1 2];
172
173 % b longitudinal
174 state_vec = zeros(1,12);
175
176 [t_5b, state_5b] = ode45(@(t_5b,state_5b) linearizedEOM(t_5b,state_5b,forces,perturbations,knowns
    ),t_span,state_vec);
177
178 %% Problem 5 Plots
179 figure
180 sgtitle(['Problem 5 Model (Lateral)']);
181 % x position vs time
182 subplot(4,3,1);
183 plot(t_5a, state_5a(:,1));
184 grid on;
185 xlim([0 t_5a(end)]);
186 xlabel('time [s]');
187 ylabel('x [m]');
188 title('x position vs time');
189 % y position vs time
190 subplot(4,3,2);
191 plot(t_5a, state_5a(:,2));
192 grid on;
193 xlim([0 t_5a(end)]);
194 xlabel('time [s]');
195 ylabel('y [m]');
196 title('y position vs time');
197 % z position vs time
198 subplot(4,3,3);
199 plot(t_5a, state_5a(:,3));
200 grid on;
201 xlim([0 t_5a(end)]);
202 xlabel('time [s]');
203 ylabel('z [m]');
204 title('z position vs time');
205 % roll vs time
206 subplot(4,3,4);
207 plot(t_5a, state_5a(:,4));
208 grid on;
209 xlim([0 t_5a(end)]);
210 xlabel('time [s]');
211 ylabel('\phi [rad]');
212 title('roll vs time');
213 % pitch vs time
214 subplot(4,3,5);
215 plot(t_5a, state_5a(:,5));
216 grid on;
217 xlim([0 t_5a(end)]);
218 xlabel('time [s]');
219 ylabel('\theta [rad]');
220 title('pitch vs time');
221 % yaw vs time
222 subplot(4,3,6);
223 plot(t_5a, state_5a(:,6));
224 grid on;
225 xlim([0 t_5a(end)]);
226 xlabel('time [s]');
227 ylabel('\psi [rad]');
228 title('yaw vs time');
229 % x velocity vs time
230 subplot(4,3,7);
231 plot(t_5a, state_5a(:,7));
232 grid on;
233 xlim([0 t_5a(end)]);
234 xlabel('time [s]');
235 ylabel('u [rad]');
236 title('x velocity vs time');
237 % y velocity vs time

```

```

238 subplot(4,3,8);
239 plot(t_5a, state_5a(:,8));
240 grid on;
241 xlim([0 t_5a(end)]);
242 xlabel('time [s]');
243 ylabel('v [rad]');
244 title('y velocity vs time');
245 % z velocity vs time
246 subplot(4,3,9);
247 plot(t_5a, state_5a(:,9));
248 grid on;
249 xlim([0 t_5a(end)]);
250 xlabel('time [s]');
251 ylabel('w [rad]');
252 title('z velocity vs time');
253 % roll rate vs time
254 subplot(4,3,10);
255 plot(t_5a, state_5a(:,10));
256 grid on;
257 xlim([0 t_5a(end)]);
258 xlabel('time [s]');
259 ylabel('p [rad/s]');
260 title('roll rate vs time');
261 % pitch rate vs time
262 subplot(4,3,11);
263 plot(t_5a, state_5a(:,11));
264 grid on;
265 xlim([0 t_5a(end)]);
266 xlabel('time [s]');
267 ylabel('q [rad/s]');
268 title('pitch rate vs time');
269 % yaw rate vs time
270 subplot(4,3,12);
271 plot(t_5a, state_5a(:,12));
272 grid on;
273 xlim([0 t_5a(end)]);
274 xlabel('time [s]');
275 ylabel('r [rad/s]');
276 title('rate rate vs time');
277
278 figure
279 sgtitle(['Problem 5 Model (Longitudnal)']);
280 % x position vs time
281 subplot(4,3,1);
282 plot(t_5b, state_5b(:,1));
283 grid on;
284 xlim([0 t_5b(end)]);
285 xlabel('time [s]');
286 ylabel('x [m]');
287 title('x position vs time');
288 % y position vs time
289 subplot(4,3,2);
290 plot(t_5b, state_5b(:,2));
291 grid on;
292 xlim([0 t_5b(end)]);
293 xlabel('time [s]');
294 ylabel('y [m]');
295 title('y position vs time');
296 % z position vs time
297 subplot(4,3,3);
298 plot(t_5b, state_5b(:,3));
299 grid on;
300 xlim([0 t_5b(end)]);
301 xlabel('time [s]');
302 ylabel('z [m]');
303 title('z position vs time');
304 % roll vs time
305 subplot(4,3,4);

```



```

306 plot(t_5b, state_5b(:,4));
307 grid on;
308 xlim([0 t_5b(end)]);
309 xlabel('time [s]');
310 ylabel('\phi [rad]');
311 title('roll vs time');
312 % pitch vs time
313 subplot(4,3,5);
314 plot(t_5b, state_5b(:,5));
315 grid on;
316 xlim([0 t_5b(end)]);
317 xlabel('time [s]');
318 ylabel('\theta [rad]');
319 title('pitch vs time');
320 % yaw vs time
321 subplot(4,3,6);
322 plot(t_5b, state_5b(:,6));
323 grid on;
324 xlim([0 t_5b(end)]);
325 xlabel('time [s]');
326 ylabel('\psi [rad]');
327 title('yaw vs time');
328 % x velocity vs time
329 subplot(4,3,7);
330 plot(t_5b, state_5b(:,7));
331 grid on;
332 xlim([0 t_5b(end)]);
333 xlabel('time [s]');
334 ylabel('u [rad]');
335 title('x velocity vs time');
336 % y velocity vs time
337 subplot(4,3,8);
338 plot(t_5b, state_5b(:,8));
339 grid on;
340 xlim([0 t_5b(end)]);
341 xlabel('time [s]');
342 ylabel('v [rad]');
343 title('y velocity vs time');
344 % z velocity vs time
345 subplot(4,3,9);
346 plot(t_5b, state_5b(:,9));
347 grid on;
348 xlim([0 t_5b(end)]);
349 xlabel('time [s]');
350 ylabel('w [rad]');
351 title('z velocity vs time');
352 % roll rate vs time
353 subplot(4,3,10);
354 plot(t_5b, state_5b(:,10));
355 grid on;
356 xlim([0 t_5b(end)]);
357 xlabel('time [s]');
358 ylabel('p [rad/s]');
359 title('roll rate vs time');
360 % pitch rate vs time
361 subplot(4,3,11);
362 plot(t_5b, state_5b(:,11));
363 grid on;
364 xlim([0 t_5b(end)]);
365 xlabel('time [s]');
366 ylabel('q [rad/s]');
367 title('pitch rate vs time');
368 % yaw rate vs time
369 subplot(4,3,12);
370 plot(t_5b, state_5b(:,12));
371 grid on;
372 xlim([0 t_5b(end)]);
373 xlabel('time [s]');

```

```

374 ylabel('r [rad/s]');
375 title('rate rate vs time');
376
377
378 %% End Housekeeping
379 toc

```

This is the non-linearized function used in problem 3:

```

1 function state_dot = quadrotorODE(t,state_vec,forces,knowns)
2
3 % Function to calculate the response of a non-linearized quadrotor model
4 % with feedback loops intergrated
5 %
6 % Inputs:
7 %     t           -> time
8 %     state_vec   -> state vector
9 %     forces      -> motor forces
10 %     knowns      -> vector of known values
11 % Outputs:
12 %     state_dot   -> derivative of the state vector
13
14 %% Define known variables
15 mu = knowns(1); % [N/(m/s)^2]
16 nu = knowns(2); % [N/(rad/s)^2]
17 I_x = knowns(3); % x moment of inertia [kg*m^2]
18 I_y = knowns(4); % y moment of inertia [kg*m^2]
19 I_z = knowns(5); % z moment of inertia [kg*m^2]
20 m = knowns(6); % mass [kg]
21 g = knowns(7); % gravity [m/s^2]
22 k_1 = knowns(8); % lateral k_1
23 k_2 = knowns(9); % lateral k_2
24 k_3 = knowns(10); % lateral k_3
25 k_4 = knowns(11); % longitudinal k_1
26 k_5 = knowns(12); % longitudinal k_2
27 k_6 = knowns(13); % longitudinal k_3
28 k_m = knowns(15); % moment coefficient
29
30 %% Define variables
31 phi = state_vec(4); % roll angle
32 theta = state_vec(5); % pitch angle
33 psi = state_vec(6); % yaw angle
34 u = state_vec(7); % x velocity
35 v = state_vec(8); % y velocity
36 w = state_vec(9); % z velocity
37 p = state_vec(10); % roll rate
38 q = state_vec(11); % pitch rate
39 r = state_vec(12); % yaw rate
40 f_1 = forces(1); % motor force 1
41 f_2 = forces(2); % motor force 2
42 f_3 = forces(3); % motor force 3
43 f_4 = forces(4); % motor force 4
44
45 %% Define Aerodynamic/Control Forces/Moments
46 M_aero = -mu*norm([p;q;r])*[p;q;r]; % aerodynamic moment
47 F_aero = -nu*norm([u;v;w])*[u;v;w]; % aerodynamic force
48 M_cntl = [-k_1*p-k_2*phi;...
49           -k_4*q-k_5*theta;...
50           k_m*(f_1-f_2+f_3-f_4)]; % control moment
51 F_cntl = [0; 0; -sum(forces)]; % control force
52
53 %% Non-Linear Equations of Motion
54 % velocity
55 xyz_dot = [cos(theta)*cos(psi) sin(phi)*sin(theta)*cos(psi)-cos(phi)*sin(psi) cos(phi)*sin(
56           theta)*cos(psi)+sin(phi)*sin(psi);...
57           cos(theta)*sin(psi) sin(phi)*sin(theta)*sin(psi)+cos(phi)*cos(psi) cos(phi)*sin(theta)*
58           sin(psi)-sin(phi)*cos(psi);...
59           -sin(theta) sin(phi)*cos(theta) cos(phi)*cos(theta)]...

```

```

58     * [u;v;w];
59     % rate of change of the euler angles
60     euler_dot = [1 sin(phi)*tan(theta) cos(phi)*tan(theta);...
61                 0 cos(phi) -sin(phi);...
62                 0 sin(phi)*(1/cos(theta)) cos(phi)*(1/cos(theta))];...
63     * [p;q;r];
64     % acceleration
65     v_dot = [r*v-q*w; p*w-r*u; q*u-p*v]...
66             + g*[-sin(theta);cos(theta)*sin(phi);cos(theta)*cos(phi)]...
67             + (1/m)*F_aero + (1/m)*F_cntl;
68     % angular acceleration
69     v_angular_dot = [((I_y-I_z)/I_x)*q*r; ((I_z-I_x)/I_y)*q*r; ((I_x-I_y)/I_z)*q*r]...
70                     + [(1/I_x)*M_aero(1); (1/I_y)*M_aero(2); (1/I_z)*M_aero(3)]...
71                     + [(1/I_x)*M_cntl(1); (1/I_y)*M_cntl(2); (1/I_z)*M_cntl(3)];
72
73     state_dot = [xyz_dot; euler_dot; v_dot; v_angular_dot]; % vector output for the integrals
74
75 end

```

This is the linearized function used in problems 2, 4, and 5:

```

1 function state_dot = linearizedEOM(t, state_vec, forces, perturbations, knowns)
2
3 % Function to calculate the respose of a linearized quadrotor model with
4 % feedback loops intergrated
5 %
6 % Inputs:
7 %     t           -> time
8 %     state_vec   -> state vector
9 %     forces      -> motor forces
10 %    perturbations -> vector to describe any perturbations
11 %    knowns       -> vector of known values
12 % Outputs:
13 %    state_dot    -> derivative of the state vector
14
15 %% Define known variables
16 mu = knowns(1); % [N/(m/s)^2]
17 I_x = knowns(3); % x moment of inertia [kg*m^2]
18 I_y = knowns(4); % y moment of inertia [kg*m^2]
19 I_z = knowns(5); % z moment of inertia [kg*m^2]
20 g = knowns(7); % gravity [m/s^2]
21 k_1 = knowns(8); % lateral k_1
22 k_2 = knowns(9); % lateral k_2
23 k_3 = knowns(10); % lateral k_3
24 k_4 = knowns(11); % longitudinal k_1
25 k_5 = knowns(12); % longitudinal k_2
26 k_6 = knowns(13); % longitudinal k_3
27 R = knowns(14); % radius from CG to motor force [m]
28 k_m = knowns(15); % control moment coefficient
29
30 %% Define variables
31 x = state_vec(1); % x position
32 y = state_vec(2); % y position
33 z = state_vec(3); % z position
34 phi = state_vec(4); % roll angle
35 theta = state_vec(5); % pitch angle
36 u = state_vec(7); % x velocity
37 v = state_vec(8); % y velocity
38 w = state_vec(9); % z velocity
39 p = state_vec(10); % roll rate
40 q = state_vec(11); % pitch rate
41 r = state_vec(12); % yaw rate
42 f_1 = forces(1); % motor force 1
43 f_2 = forces(2); % motor force 2
44 f_3 = forces(3); % motor force 3
45 f_4 = forces(4); % motor force 4
46
47 %% Define Aerodynamic/Control Forces/Moments

```

```

48 M_aero = -mu*norm([p;q;r])*[p;q;r]; % aerodynamic moment
49 M_cntl = [(R/sqrt(2))*(-f_1-f_2+f_3+f_4);...
50 (R/sqrt(2))*(f_1-f_2-f_3+f_4);...
51 k_m*(f_1-f_2+f_3-f_4)]; % control moment
52
53 %% Linear Equations of Motion
54 % rate of change of euler angles
55 euler_dot = [p; q; r];
56 % acceleration
57 v_dot = g*[-theta; phi; 0];
58
59 if knowns(16) == 1 && (t >= 0 && t <= 2)
60 % time and distance required for required travel
61 t_r = knowns(19);
62 x_r = knowns(17);
63 y_r = knowns(18);
64 % velocity to meet requirements
65 u_r = (x_r-x)/(t_r-t);
66 v_r = (y_r-y)/(t_r-t);
67 % angular acceleration
68 v_angular_dot = [-(k_1*p)/I_x) - ((k_2*phi)/I_x) + (k_3*(v_r-v))/I_x;...
69 -((k_4*q)/I_y) - ((k_5*theta)/I_y) + (k_6*(u_r-u))/I_y;...
70 (M_aero(3) + M_cntl(3) + perturbations(3))/I_z];
71 % velocity
72 xyz_dot = [u_r-u; v_r-v; w];
73 else
74 % angular acceleration
75 v_angular_dot = [-(k_1*p)/I_x) - ((k_2*phi)/I_x) + (k_3*v)/I_x;...
76 -((k_4*q)/I_y) - ((k_5*theta)/I_y) + (k_6*u)/I_y;...
77 (M_aero(3) + M_cntl(3) + perturbations(3))/I_z];
78 % velocity
79 xyz_dot = [u; v; w];
80 end
81
82 state_dot = [xyz_dot; euler_dot; v_dot; v_angular_dot]; % vector output for the integrals
83
84 end

```

This is the function used to plot problems 2, 3, and 4:

```

1 function plotP2N3(t_2a,state_2a,t_2b,state_2b,t_2c,state_2c,t_2d,state_2d,N)
2
3 %% Problem 2 Plots
4 % a (+5 deg in roll)
5 figure
6 sgtitle(['Deviation by +5^{o} in Roll [' N '.a']']);
7 % x position vs time
8 subplot(4,3,1);
9 plot(t_2a, state_2a(:,1));
10 grid on;
11 xlim([0 t_2a(end)]);
12 xlabel('time [s]');
13 ylabel('x [m]');
14 title('x position vs time');
15 % y position vs time
16 subplot(4,3,2);
17 plot(t_2a, state_2a(:,2));
18 grid on;
19 xlim([0 t_2a(end)]);
20 xlabel('time [s]');
21 ylabel('y [m]');
22 title('y position vs time');
23 % z position vs time
24 subplot(4,3,3);
25 plot(t_2a, state_2a(:,3));
26 grid on;
27 xlim([0 t_2a(end)]);
28 xlabel('time [s]');

```

```

29     ylabel('z [m]');
30     title('z position vs time');
31     % roll vs time
32     subplot(4,3,4);
33     plot(t_2a, state_2a(:,4));
34     grid on;
35     xlim([0 t_2a(end)]);
36     xlabel('time [s]');
37     ylabel('\phi [rad]');
38     title('roll vs time');
39     % pitch vs time
40     subplot(4,3,5);
41     plot(t_2a, state_2a(:,5));
42     grid on;
43     xlim([0 t_2a(end)]);
44     xlabel('time [s]');
45     ylabel('\theta [rad]');
46     title('pitch vs time');
47     % yaw vs time
48     subplot(4,3,6);
49     plot(t_2a, state_2a(:,6));
50     grid on;
51     xlim([0 t_2a(end)]);
52     xlabel('time [s]');
53     ylabel('\psi [rad]');
54     title('yaw vs time');
55     % x velocity vs time
56     subplot(4,3,7);
57     plot(t_2a, state_2a(:,7));
58     grid on;
59     xlim([0 t_2a(end)]);
60     xlabel('time [s]');
61     ylabel('u [rad]');
62     title('x velocity vs time');
63     % y velocity vs time
64     subplot(4,3,8);
65     plot(t_2a, state_2a(:,8));
66     grid on;
67     xlim([0 t_2a(end)]);
68     xlabel('time [s]');
69     ylabel('v [rad]');
70     title('y velocity vs time');
71     % z velocity vs time
72     subplot(4,3,9);
73     plot(t_2a, state_2a(:,9));
74     grid on;
75     xlim([0 t_2a(end)]);
76     xlabel('time [s]');
77     ylabel('w [rad]');
78     title('z velocity vs time');
79     % roll rate vs time
80     subplot(4,3,10);
81     plot(t_2a, state_2a(:,10));
82     grid on;
83     xlim([0 t_2a(end)]);
84     xlabel('time [s]');
85     ylabel('p [rad/s]');
86     title('roll rate vs time');
87     % pitch rate vs time
88     subplot(4,3,11);
89     plot(t_2a, state_2a(:,11));
90     grid on;
91     xlim([0 t_2a(end)]);
92     xlabel('time [s]');
93     ylabel('q [rad/s]');
94     title('pitch rate vs time');
95     % yaw rate vs time
96     subplot(4,3,12);

```

```

97     plot(t_2a, state_2a(:,12));
98     grid on;
99     xlim([0 t_2a(end)]);
100    xlabel('time [s]');
101    ylabel('r [rad/s]');
102    title('rate rate vs time');
103
104    % b (+5 deg in pitch)
105    figure
106    sgtitle(['Deviation by +5^{o} in Pitch [' N '.b]']);
107    % x position vs time
108    subplot(4,3,1);
109    plot(t_2b, state_2b(:,1));
110    grid on;
111    xlim([0 t_2b(end)]);
112    xlabel('time [s]');
113    ylabel('x [m]');
114    title('x position vs time');
115    % y position vs time
116    subplot(4,3,2);
117    plot(t_2b, state_2b(:,2));
118    grid on;
119    xlim([0 t_2b(end)]);
120    xlabel('time [s]');
121    ylabel('y [m]');
122    title('y position vs time');
123    % z position vs time
124    subplot(4,3,3);
125    plot(t_2b, state_2b(:,3));
126    grid on;
127    xlim([0 t_2b(end)]);
128    xlabel('time [s]');
129    ylabel('z [m]');
130    title('z position vs time');
131    % roll vs time
132    subplot(4,3,4);
133    plot(t_2b, state_2b(:,4));
134    grid on;
135    xlim([0 t_2b(end)]);
136    xlabel('time [s]');
137    ylabel('\phi [rad]');
138    title('roll vs time');
139    % pitch vs time
140    subplot(4,3,5);
141    plot(t_2b, state_2b(:,5));
142    grid on;
143    xlim([0 t_2b(end)]);
144    xlabel('time [s]');
145    ylabel('\theta [rad]');
146    title('pitch vs time');
147    % yaw vs time
148    subplot(4,3,6);
149    plot(t_2b, state_2b(:,6));
150    grid on;
151    xlim([0 t_2b(end)]);
152    xlabel('time [s]');
153    ylabel('\psi [rad]');
154    title('yaw vs time');
155    % x velocity vs time
156    subplot(4,3,7);
157    plot(t_2b, state_2b(:,7));
158    grid on;
159    xlim([0 t_2b(end)]);
160    xlabel('time [s]');
161    ylabel('u [rad]');
162    title('x velocity vs time');
163    % y velocity vs time
164    subplot(4,3,8);

```



```

165     plot(t_2b, state_2b(:,8));
166     grid on;
167     xlim([0 t_2b(end)]);
168     xlabel('time [s]');
169     ylabel('v [rad]');
170     title('y velocity vs time');
171     % z velocity vs time
172     subplot(4,3,9);
173     plot(t_2b, state_2b(:,9));
174     grid on;
175     xlim([0 t_2b(end)]);
176     xlabel('time [s]');
177     ylabel('w [rad]');
178     title('z velocity vs time');
179     % roll rate vs time
180     subplot(4,3,10);
181     plot(t_2b, state_2b(:,10));
182     grid on;
183     xlim([0 t_2b(end)]);
184     xlabel('time [s]');
185     ylabel('p [rad/s]');
186     title('roll rate vs time');
187     % pitch rate vs time
188     subplot(4,3,11);
189     plot(t_2b, state_2b(:,11));
190     grid on;
191     xlim([0 t_2b(end)]);
192     xlabel('time [s]');
193     ylabel('q [rad/s]');
194     title('pitch rate vs time');
195     % yaw rate vs time
196     subplot(4,3,12);
197     plot(t_2b, state_2b(:,12));
198     grid on;
199     xlim([0 t_2b(end)]);
200     xlabel('time [s]');
201     ylabel('r [rad/s]');
202     title('rate rate vs time');
203
204     % c (+0.1 rad/s in roll rate)
205     figure
206     sgtitle(['Deviation by +0.1 rad/s in Roll Rate [' N '.c']]);
207     % x position vs time
208     subplot(4,3,1);
209     plot(t_2c, state_2c(:,1));
210     grid on;
211     xlim([0 t_2c(end)]);
212     xlabel('time [s]');
213     ylabel('x [m]');
214     title('x position vs time');
215     % y position vs time
216     subplot(4,3,2);
217     plot(t_2c, state_2c(:,2));
218     grid on;
219     xlim([0 t_2c(end)]);
220     xlabel('time [s]');
221     ylabel('y [m]');
222     title('y position vs time');
223     % z position vs time
224     subplot(4,3,3);
225     plot(t_2c, state_2c(:,3));
226     grid on;
227     xlim([0 t_2c(end)]);
228     xlabel('time [s]');
229     ylabel('z [m]');
230     title('z position vs time');
231     % roll vs time
232     subplot(4,3,4);

```

```

233 plot(t_2c, state_2c(:,4));
234 grid on;
235 xlim([0 t_2c(end)]);
236 xlabel('time [s]');
237 ylabel('\phi [rad]');
238 title('roll vs time');
239 % pitch vs time
240 subplot(4,3,5);
241 plot(t_2c, state_2c(:,5));
242 grid on;
243 xlim([0 t_2c(end)]);
244 xlabel('time [s]');
245 ylabel('\theta [rad]');
246 title('pitch vs time');
247 % yaw vs time
248 subplot(4,3,6);
249 plot(t_2c, state_2c(:,6));
250 grid on;
251 xlim([0 t_2c(end)]);
252 xlabel('time [s]');
253 ylabel('\psi [rad]');
254 title('yaw vs time');
255 % x velocity vs time
256 subplot(4,3,7);
257 plot(t_2c, state_2c(:,7));
258 grid on;
259 xlim([0 t_2c(end)]);
260 xlabel('time [s]');
261 ylabel('u [rad]');
262 title('x velocity vs time');
263 % y velocity vs time
264 subplot(4,3,8);
265 plot(t_2c, state_2c(:,8));
266 grid on;
267 xlim([0 t_2c(end)]);
268 xlabel('time [s]');
269 ylabel('v [rad]');
270 title('y velocity vs time');
271 % z velocity vs time
272 subplot(4,3,9);
273 plot(t_2c, state_2c(:,9));
274 grid on;
275 xlim([0 t_2c(end)]);
276 xlabel('time [s]');
277 ylabel('w [rad]');
278 title('z velocity vs time');
279 % roll rate vs time
280 subplot(4,3,10);
281 plot(t_2c, state_2c(:,10));
282 grid on;
283 xlim([0 t_2c(end)]);
284 xlabel('time [s]');
285 ylabel('p [rad/s]');
286 title('roll rate vs time');
287 % pitch rate vs time
288 subplot(4,3,11);
289 plot(t_2c, state_2c(:,11));
290 grid on;
291 xlim([0 t_2c(end)]);
292 xlabel('time [s]');
293 ylabel('q [rad/s]');
294 title('pitch rate vs time');
295 % yaw rate vs time
296 subplot(4,3,12);
297 plot(t_2c, state_2c(:,12));
298 grid on;
299 xlim([0 t_2c(end)]);
300 xlabel('time [s]');

```

```

301     ylabel('r [rad/s]');
302     title('rate rate vs time');
303
304     % d (+0.1 rad/s in pitch rate)
305     figure
306     sgtitle(['Deviation by +0.1 rad/s in Pitch Rate [' N '.d']']);
307     % x position vs time
308     subplot(4,3,1);
309     plot(t_2d, state_2d(:,1));
310     grid on;
311     xlim([0 t_2d(end)]);
312     xlabel('time [s]');
313     ylabel('x [m]');
314     title('x position vs time');
315     % y position vs time
316     subplot(4,3,2);
317     plot(t_2d, state_2d(:,2));
318     grid on;
319     xlim([0 t_2d(end)]);
320     xlabel('time [s]');
321     ylabel('y [m]');
322     title('y position vs time');
323     % z position vs time
324     subplot(4,3,3);
325     plot(t_2d, state_2d(:,3));
326     grid on;
327     xlim([0 t_2d(end)]);
328     xlabel('time [s]');
329     ylabel('z [m]');
330     title('z position vs time');
331     % roll vs time
332     subplot(4,3,4);
333     plot(t_2d, state_2d(:,4));
334     grid on;
335     xlim([0 t_2d(end)]);
336     xlabel('time [s]');
337     ylabel('\phi [rad]');
338     title('roll vs time');
339     % pitch vs time
340     subplot(4,3,5);
341     plot(t_2d, state_2d(:,5));
342     grid on;
343     xlim([0 t_2d(end)]);
344     xlabel('time [s]');
345     ylabel('\theta [rad]');
346     title('pitch vs time');
347     % yaw vs time
348     subplot(4,3,6);
349     plot(t_2d, state_2d(:,6));
350     grid on;
351     xlim([0 t_2d(end)]);
352     xlabel('time [s]');
353     ylabel('\psi [rad]');
354     title('yaw vs time');
355     % x velocity vs time
356     subplot(4,3,7);
357     plot(t_2d, state_2d(:,7));
358     grid on;
359     xlim([0 t_2d(end)]);
360     xlabel('time [s]');
361     ylabel('u [m/s]');
362     title('x velocity vs time');
363     % y velocity vs time
364     subplot(4,3,8);
365     plot(t_2d, state_2d(:,8));
366     grid on;
367     xlim([0 t_2d(end)]);
368     xlabel('time [s]');

```

```

369     ylabel('v [m/s]');
370     title('y velocity vs time');
371     % z velocity vs time
372     subplot(4,3,9);
373     plot(t_2d, state_2d(:,9));
374     grid on;
375     xlim([0 t_2d(end)]);
376     xlabel('time [s]');
377     ylabel('w [m/s]');
378     title('z velocity vs time');
379     % roll rate vs time
380     subplot(4,3,10);
381     plot(t_2d, state_2d(:,10));
382     grid on;
383     xlim([0 t_2d(end)]);
384     xlabel('time [s]');
385     ylabel('p [rad/s]');
386     title('roll rate vs time');
387     % pitch rate vs time
388     subplot(4,3,11);
389     plot(t_2d, state_2d(:,11));
390     grid on;
391     xlim([0 t_2d(end)]);
392     xlabel('time [s]');
393     ylabel('q [rad/s]');
394     title('pitch rate vs time');
395     % yaw rate vs time
396     subplot(4,3,12);
397     plot(t_2d, state_2d(:,12));
398     grid on;
399     xlim([0 t_2d(end)]);
400     xlabel('time [s]');
401     ylabel('r [rad/s]');
402     title('rate rate vs time');
403
404 end

```

This is the function used to convert the roll rate to p, q, and r values

```

1 function [p,q,r] = rollrate2pqr(phi,theta,psi,phi_dot,theta_dot,psi_dot)
2
3 % Function to calculate the roll, pitch, and yaw from a given position with
4 % a known roll rate, pitch rate, and yaw rate
5 %
6 % Inputs:
7 %     phi      -> yaw
8 %     theta    -> pitch
9 %     psi      -> yaw
10 %     phi_dot  -> roll rate
11 %     theta_dot -> pitch rate
12 %     psi_dot  -> yaw rate
13 % Outputs:
14 %     p        -> roll velocity
15 %     q        -> pitch velocity
16 %     r        -> yaw velocity
17
18
19 %% Solve for pqr vector
20 pqr = [phi_dot,theta_dot,psi_dot]...
21     *inv([1, sin(phi)*tan(theta), cos(phi)*tan(theta);...
22     0, cos(phi), sin(phi);...
23     0, sin(phi)*sec(theta), cos(phi)*sec(theta)]);
24
25 %% Define p, q, and r based on previous calculations
26 p = pqr(1);
27 q = pqr(2);
28 r = pqr(3);
29

```

```
30 end
```

This is the eigenvalues plot :

```
1 % Lab 4
2 % Nick Herrington, Cole MacPherson, Joshua Schmitz, Panitnan Yuvanondha
3
4 clc;clear;close all
5
6 %% declare constants
7 m = 0.068; % mass of the quadrotor [kg]
8 R = 0.06; % radial distance from CG to propeller [m]
9 k_m = 0.0024; % control moment coefficient [N*m/N]
10 I_x = 5.8e-5; % x-axis moment of inertia [kg*m^2]
11 I_y = 7.2e-5; % y-axis moment of inertia [kg*m^2]
12 I_z = 1.0e-4; % z-axis moment of inertia [kg*m^2]
13 nu = 1e-3; % aerodynamic force coefficient [N/(m/s)^2]
14 mu = 2e-6; % aerodynamic moment coefficient [N*m/(rad/s)^2]
15 g = 9.81; % graviational constant [m/s^2]
16 k1=0.001276;
17 k2=0.00232;
18 k3=0.001584;
19 k4=0.00288;
20
21
22
23 %% Problem 4
24 % Lateral
25 k3Lat = 0:0.001:0.1;
26 for i = 1:length(k3Lat)
27 A_lat = [0 g 0;0 0 1; -(k3Lat(i)/I_x) -(k2/I_x) -(k1/I_x)];
28 lam_lat(:,i) = eig(A_lat);
29 figure(1)
30 plot(real(lam_lat(:,i)),imag(lam_lat(:,i)),'x')
31 hold on
32 grid minor
33 title('Eigenvalues for Lateral')
34 xlabel('real')
35 ylabel('Imaginary')
36 end
37 % for lateral at index = 10 , the real eigenvalue is about 0.8 (tau=1/1.25)
38 k3lat=k3Lat(10)
39
40
41
42 % Longitudinal
43 k3lon= 0:0.001:0.1;
44 for i = 1:length(k3Lat)
45 A_lon = [0 -g 0;0 0 1; (k3lon(i)/I_y) -(k4/I_y) -(k3/I_y)];
46 lam_lon(:,i) = eig(A_lon);
47 figure(2)
48 plot(real(lam_lon(:,i)),imag(lam_lon(:,i)),'x')
49 hold on
50 grid minor
51 title('Eigenvalues for Longitudinal')
52 xlabel('real')
53 ylabel('Imaginary')
54 end
55
56 % for longitudinal at index = 77 , the real eigenvalue is about 0.8 (tau=1/1.25)
57 k3Lon=k3lon(77)
```

References

- [1] Smead Department of Aerospace Engineering. ASEN 3128 Aircraft Dynamics - Spring 2021 LABORATORY 4 University of Colorado at Boulder. Lab Document and Supplemental Resources.