

TMME50 Assignment V

David Wiman
20000120-8495

December 5, 2023

Instructions 2023 for reporting the computer assignments

The computer assignments are reported in writing, and submitted *printed on paper*. The assignments are performed individually. The use of ChatGPT or any similar system is not allowed. It is permissible to discuss the assignments and to show parts of solutions in that context, but *copying of Matlab code or sections of reports is not allowed*. Further, it is not allowed to possess copies of other students reports or Matlab code, either electronically or on paper, or to supply this to another student; this also means that you hand in and pick up your assignments yourself, not with the help of a friend. The reports shall contain:

- *A copy of this page of instructions.*
- Name and complete (10 digits) civic registration number of the student (sometimes called p-number among exchange students).
- *Which aeroplane and which reference condition* that has been used. Specify the number of the column on the data sheet that has been used.
- Answers to all the questions appearing under the headings "Assignment I:a" etc. and all requested plots.
- A complete set of Matlab files for each computer assignment. Choose the most complete set, such as the one for part I:c in assignment I. In assignment II, also include root loci and a graphic representation of your Simulink model for the final version of your model with all numerical values shown explicitly.
- The ODE system implemented in assignments I, III, IV and V must be given in the report in the order actually implemented and written in a *single* frame containing *all* the equations of the ODE and *nothing* else.

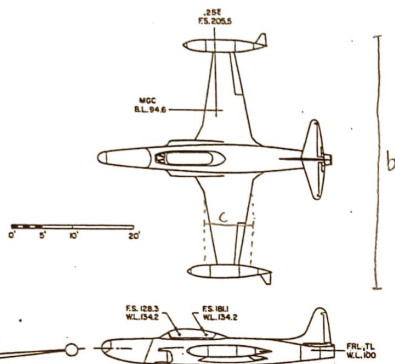
Further, note:

- With the exception of flying qualities tables, illustrations from the lab-PM defining the problem and this page of instructions, no copying of text, figures, equations or code from another document is allowed (unless it is a document you have created yourself).
- It must be clear what data has been used in what way. Data is converted from American to SI units, and this should be done in a way that can be followed in detail either in the text of the report or in the Matlab files, so that mistakes can be found at a glance without making any calculations.
- Nothing written in Matlab syntax or pseudocode is allowed in the main text of the reports: your Matlab code is appended at the end of the report.
- The report must contain sufficient text and illustrations such that it is possible to understand without ever having seen the lab-PM.
- Use the simulation time given in the assignments. For a small number of datasets it is necessary to use a longer simulation time than 100 s in order to see a full phugoid period, but the time should never be shorter than the time given and never longer than 400 s.

T-33 Shooting Star

David Wiman

NT-33A
S = 234.8 ft²
b = 37.54 ft
c = 6.72 ft



surface span chord
S = 234.8 ft², b = 37.54 ft, c = 6.72 ft

	1	2	3	4	5	6	7	8
h ft	5/16	5/16	5/16	5/16	20000	20000	20000	40000
v(uo) ft/s	228	270	447	792	311	570	778	629
m slug	367	426	426	426	426	426	426	426
x slug ft ²	12700	23901	23901	23901	23901	23901	23901	23901
y slug ft ²	20700	21101	21101	21101	21101	21101	21101	21101
z slug ft ²	32601	43802	43802	43802	43802	43802	43802	43802
x2 slug ft ²	480	480	480	480	480	480	480	480
z2 lb/ft ²	61.7	86.7	247	726	61.3	206	383	117

	1	2	3	4	5	6	7	8
$\Sigma u s^{-1}$	-0.0391	-0.00484	-0.0104	-0.0415	0.00477	-0.00735	-0.0511	-0.0355
$\Sigma a ft/s^2$	18.58	35.37	25.12	-16.50	20.43	22.29	7.67	24.59
$\Sigma u s^{-1}$	-0.248	-0.153	-0.128	-0.162	-0.114	-0.107	-0.0703	-0.0266
$\Sigma a ft/s^2$	-21341	-267.57	-773.31	-2807	-14026	-712.5	-1400	-432.9
$M u ft/s^1$	0.000318	0.000603	0.000283	-0.00076	0.000114	-0.000193	-0.00151	-0.000185
$M a s^{-2}$	-1.89	-1.81	-9.21	-33.7	-0.23	-9.95	-18.59	-5.42
$M u s^{-1}$	-0.35	-0.40	-0.63	0	-0.24	-0.31	-0.16	-0.06
$M a s^{-1}$	-0.644	-0.806	-1.37	-2.80	-0.50	-0.981	-1.56	-0.535
$\Sigma a ft/s^2$	0.516	1.47	0.62	-2.65	1.88	0.50	-0.432	0.996
$\Sigma a ft/s^2$	-13.4	-16.2	-44.4	-152	-11.3	-40.9	-82.4	-23.8
$M a s^{-2}$	-4.19	-5.83	-16.0	-52.7	-4.13	-14.2	-28.7	-8.28

	1	2	3	4	5	6	7	8
$V_p ft/s^2$	-28.4	-30.1	-81.0	-26.4	-21.6	-72.2	-14.4	-42.4
$L_p s^{-2}$	-5.49	-4.72	-8.02	-18.0	-4.06	-7.42	-9.89	-5.08
$L_p s^{-1}$	-2.03	-1.32	-2.15	-4.51	-0.82	-1.56	-2.23	-0.877
$L_r s^{-1}$	0.641	0.305	0.320	0.495	0.214	0.256	0.328	0.179
$N_p s^{-2}$	0.667	0.94	2.71	10.6	0.54	2.60	6.24	1.68
$N_p s^{-1}$	-0.116	-0.112	-0.0512	0.0118	-0.103	-0.0393	-0.0411	-0.0428
$N_r s^{-1}$	-0.207	-0.173	-0.291	-0.561	-0.104	-0.204	-0.318	-0.110
$V_{\delta r} ft/s^2$	0.0295	0.0301	0.0503	0.102	0.0185	0.0363	0.0571	0.0195
$L_{\delta r} s^{-2}$	-0.0125	0.443	1.57	5.99	0.287	1.39	3.20	0.908
$N_{\delta r} s^{-2}$	-1.24	-1.25	-3.50	-12.6	-0.883	-3.21	-6.99	-1.92
$L_{\delta a} s^{-2}$	6.01	4.53	12.6	47.0	3.14	11.7	24.0	7.13
$N_{\delta a} s^{-1}$	0.0286	0.134	0.165	0.260	0.164	0.121	0.195	0.118

Background

In this assignment, the full equations of motion for a rigid body moving in three dimensions are to be implemented, as well as the gravitational forces and the linear model for aerodynamic forces.

The force model is extended from assignment I with lateral forces. The reference state is given by all phase variables being equal to zero except $u_0 = V$ and $z_f^0 = -H$. Reference is indicated with a sub- or super-script 0.

The data sheet for the tasks at hand were given in imperial units and must therefore be converted to SI units before use. Two of the values given were:

$$\begin{aligned} V &= 228 \text{ ft/s} \cdot 0.3048 = 69.4944 \text{ m/s}, \\ H &= 100 \text{ m}. \end{aligned} \tag{1}$$

I use data from a T-33 Shooting Star at sea level flight (column one on the data sheet).

The model for the aerodynamical forces is:

$$\begin{aligned} X/m &= g \sin(\theta_0) + X_u(u - u_0) + X_w(w - w_0) \\ Y/m &= -\cos(\theta_0) \sin(\varphi_0) + Y_v(v - v_0) \\ Z/m &= -g \cos(\theta_0) \cos(\varphi_0) + Z_u(u - u_0) + Z_w(w - w_0) \\ L/I_{xx} &= L_v(v - v_0) + L_p(p - p_0) + L_r(r - r_0) \\ M/I_{yy} &= M_w(w - w_0) + M_q(q - q_0) \\ N/I_{zz} &= N_v(v - v_0) + N_p(p - p_0) + N_r(r - r_0) \end{aligned} \tag{2}$$

V:a

We begin by writing down the twelve ordinary differential equations governing the problem.

$$\begin{aligned} \dot{u} &= -qw + rv - g \sin(\theta) + X/m \\ \dot{v} &= -ru + pw + g \sin(\varphi) \cos(\theta) + Y/m \\ \dot{w} &= qu - pv + g \cos(\varphi) \cos(\theta) + Z/m \end{aligned} \tag{3}$$

$$\begin{aligned} \dot{p} &= \frac{1}{1 - \frac{I_{xx}I_{zz}}{I_{xz}^2}} \left(qrI_{zz} \frac{I_{zz} - I_{yy}}{I_{xz}^2} - pq \frac{I_{zz}}{I_{xz}} - L \frac{I_{zz}}{I_{xz}^2} + pq \frac{I_{yy} - I_{xx}}{I_{xz}} + qr - N/I_{xz} \right) \\ \dot{q} &= -rp \frac{I_{xx} - I_{zz}}{I_{yy}} - (p^2 - r^2) \frac{I_{xz}}{I_{yy}} + M/I_{yy} \\ \dot{r} &= \dot{p} \frac{I_{xx}}{I_{xz}} + qr \frac{I_{zz} - I_{yy}}{I_{xz}} - pq - L/I_{xz} \end{aligned} \tag{4}$$

$$\begin{aligned} \dot{\varphi} &= p + q \tan(\theta) \sin(\varphi) + r \tan(\theta) \cos(\varphi) \\ \dot{\theta} &= q \cos(\varphi) - r \sin(\varphi) \\ \dot{\psi} &= q \frac{\sin(\varphi)}{\cos(\theta)} + r \frac{\cos(\varphi)}{\cos(\theta)} \end{aligned} \tag{5}$$

$$\begin{aligned} \dot{x}_f &= u \cos(\theta) \cos(\psi) - v(-\cos(\varphi) \sin(\psi) + \sin(\varphi) \sin(\theta) \cos(\psi)) + w(\sin(\varphi) \sin(\psi) + \cos(\varphi) \sin(\theta) \cos(\psi)) \\ \dot{y}_f &= u \cos(\theta) \sin(\psi) + v(\cos(\varphi) \cos(\psi) + \sin(\varphi) \sin(\theta) \sin(\psi)) + w(-\sin(\varphi) \cos(\psi) + \cos(\varphi) \sin(\theta) \sin(\psi)) \\ \dot{z}_f &= -u \sin(\theta) + v \sin(\varphi) \cos(\theta) + w \cos(\varphi) \cos(\theta) \end{aligned} \tag{6}$$

V:b

When these equations are implemented in Matlab and a simulation is run for 100 s with the initial conditions

$$\begin{aligned}
 u_i &= V \\
 \theta_i &= 0.1 \text{ rad} \\
 z_f^i &= -H \\
 \text{all others} &= 0
 \end{aligned} \tag{7}$$

we get the following graph for all state variables, see Figure 1. We also get a 3D plot of the position of the aeroplane, see Figure 2.

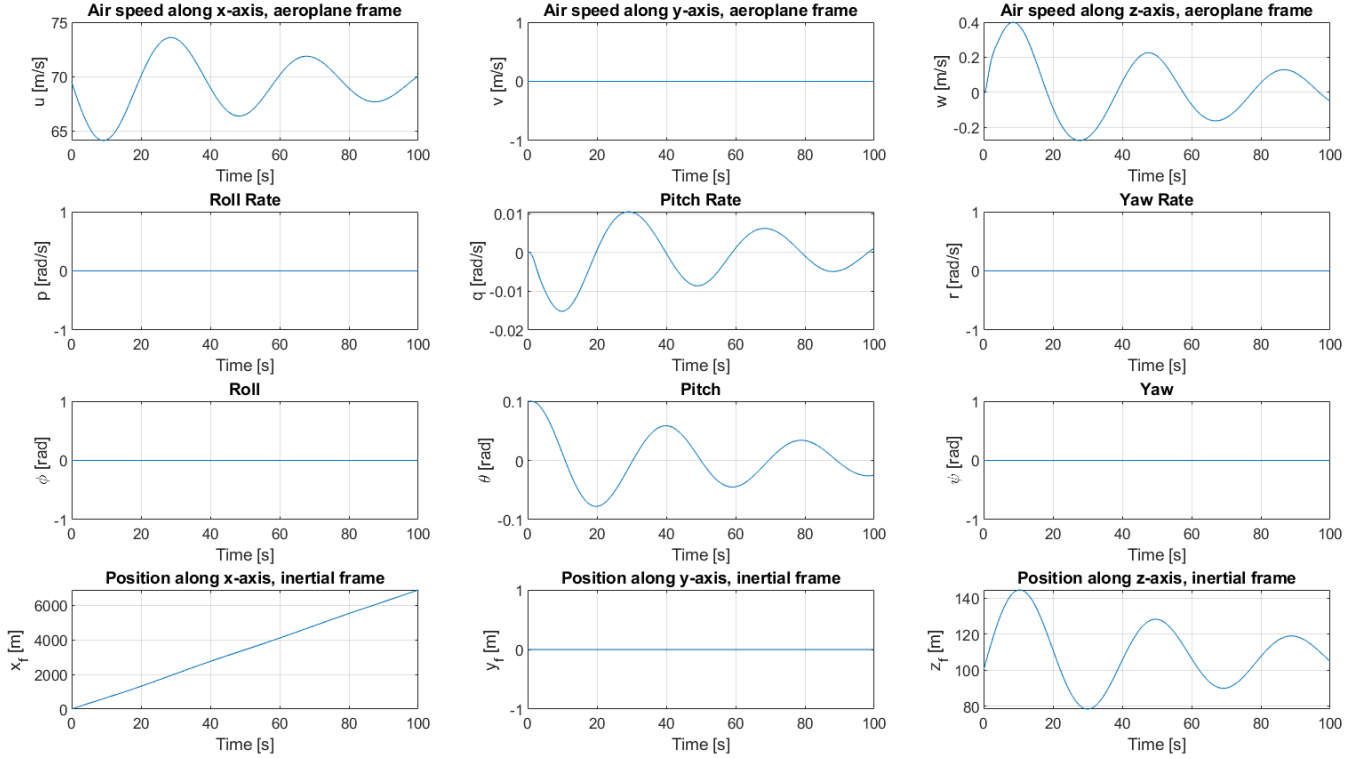


Figure 1: All phase variables.

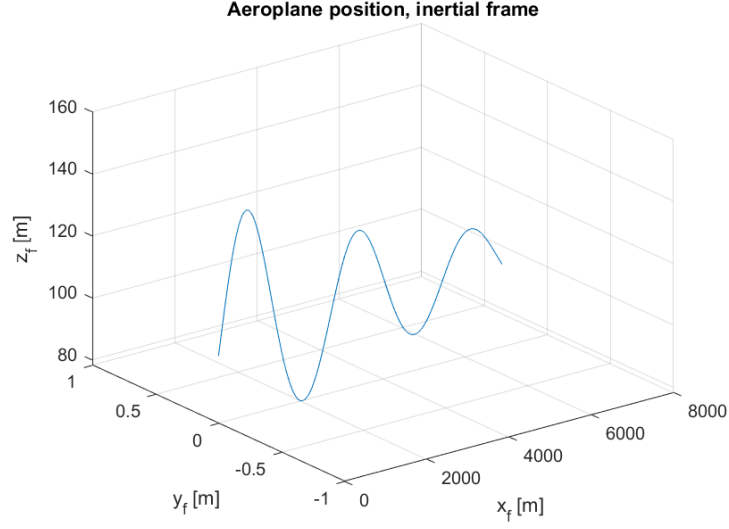


Figure 2: Position of the aeroplane.

If we compare this with the result from assignment I:c, marked with dashed red in Figures 3 and 4, we can see that there is no difference. This is an expected result since the longitudinal quantities do not spill over into lateral motion. Even though we have added lateral motion into our models, nothing we do triggers the aeroplane to leave the xz -plane.

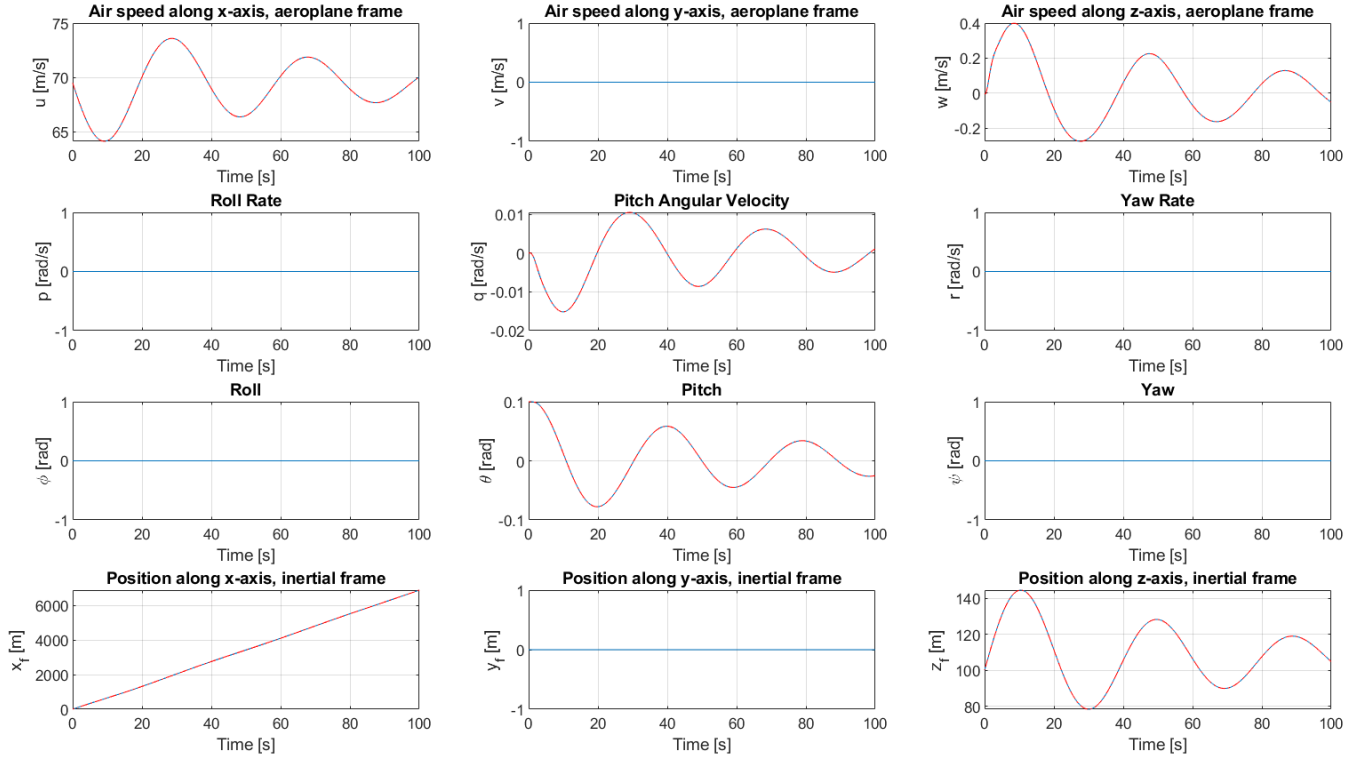


Figure 3: All phase variables with some compared to the result from assignment I:c.

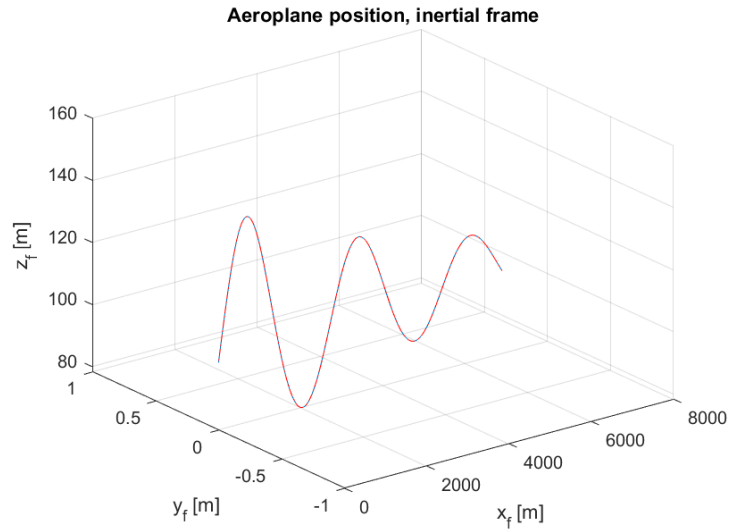


Figure 4: Position of the aeroplane.

V:c

If we let $\theta_i = 0$ but instead set $p_i, q_i, r_i = 0.1$ rad one at the time and simulate for 10 s, we get the results in Figures 5 and 6.

If we compare the results in the position graph, we can see that $q_i = 0.1$ stays in the xz -plane while the others pull to the side.

As for the phase variables, q does not cause a roll or yaw compared to the other two.

We can see that longitudinal motion does not spill over into lateral motion, but lateral motion does cause longitudinal motion. This can also be seen by the mixed longitudinal and lateral quantities in the differential- and force equations.

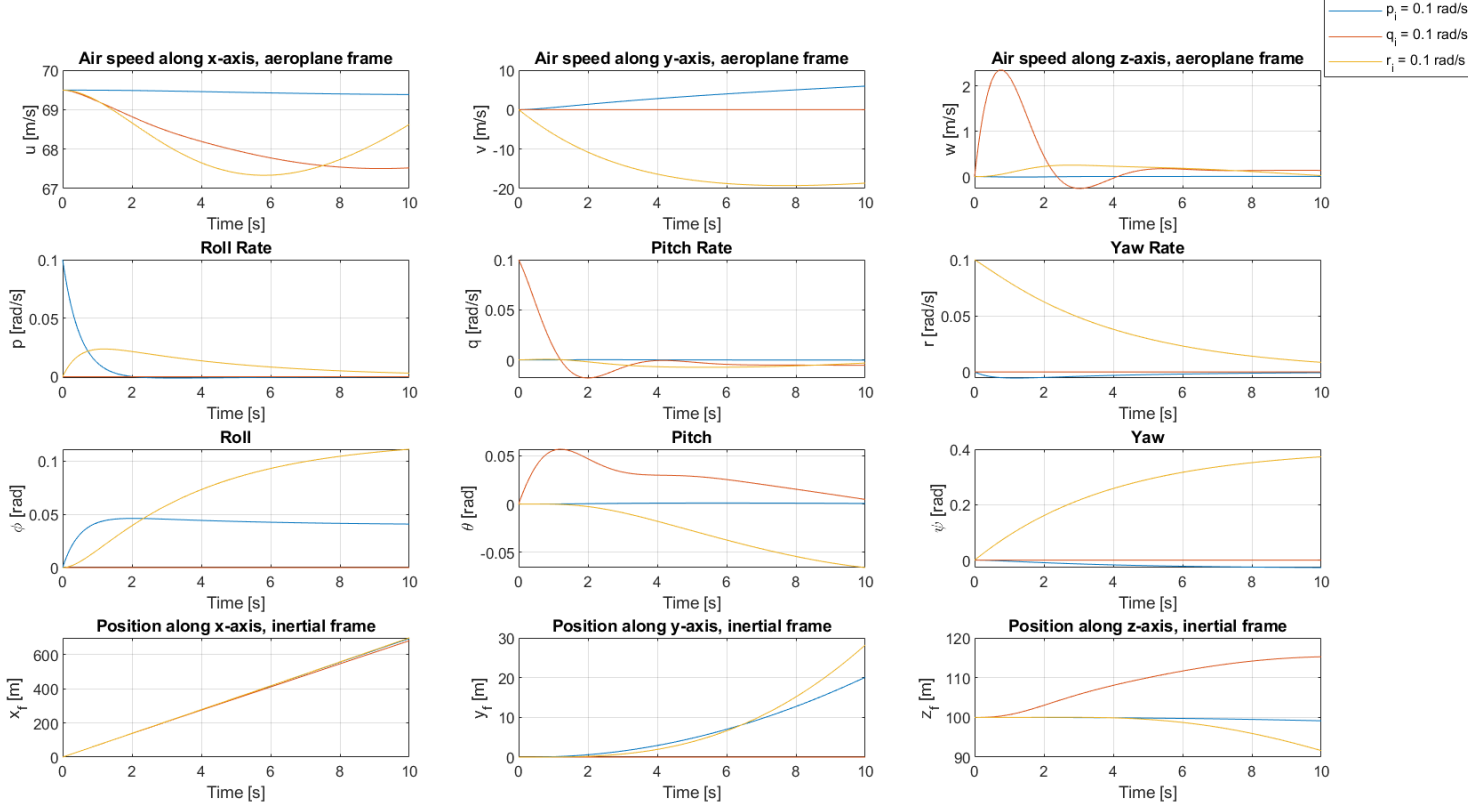


Figure 5: All phase variables compared with different initial conditions.

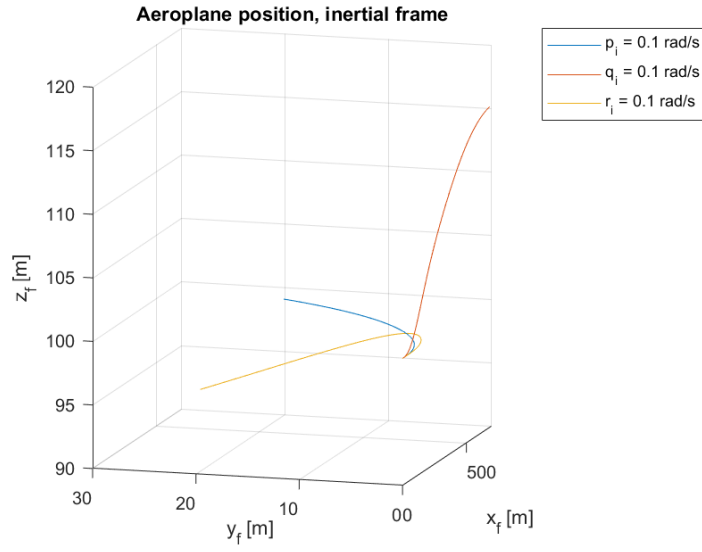


Figure 6: Position of the aeroplane with different initial conditions.

1 Matlab code

1.1 Assignment V:b

```
1 % Set variables
2 run('T33_Shooting_Star_parameters.m');
3
4 % [u, v, w, p, q, r, phi, theta, psi, x-f, y-f, z-f]
5
6 % Reference state around which linearization happens
7 reference_state = [v, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -h];
8
9 % Set ODE options
10 initia_theta = 0.1;
11 initial_conditions = [v, 0, 0, 0, 0, 0, 0, initia_theta, 0, 0, 0, -h];
12 time_interval = [0, 100];
13
14 % Compute ODE
15 [time, state] = ode45(@(t, state) state_propagation(t, state, reference_state)
16     , time_interval, initial_conditions);
17
18 % Extract all states for plotting
19 u = state(:,1);
20 v = state(:,2);
21 w = state(:,3);
22 p = state(:,4);
23 q = state(:,5);
24 r = state(:,6);
25 phi = state(:,7);
26 theta = state(:,8);
27 psi = state(:,9);
28 x_f = state(:,10);
29 y_f = state(:,11);
30 z_f = state(:,12);
31
32 %% Plot
33 figure(1)
34
35 % Air speeds
36 subplot(4,3,1)
37 plot(time, u);
38 title('Air speed along x-axis, aeroplane frame')
39 xlabel('Time [s]')
40 ylabel('u [m/s]')
41 grid
42 hold on
43
44 subplot(4,3,2)
45 plot(time, v);
46 title('Air speed along y-axis, aeroplane frame')
47 xlabel('Time [s]')
48 ylabel('v [m/s]')
49 grid
```

```

50 subplot(4,3,3)
51 plot(time, w);
52 title('Air speed along z-axis, aeroplane frame')
53 xlabel('Time [s]')
54 ylabel('w [m/s]')
55 grid
56 hold on
57
58 % Turn rates
59 subplot(4,3,4)
60 plot(time, p);
61 title('Roll Rate')
62 xlabel('Time [s]')
63 ylabel('p [rad/s]')
64 grid
65
66 subplot(4,3,5)
67 plot(time, q);
68 title('Pitch Rate')
69 xlabel('Time [s]')
70 ylabel('q [rad/s]')
71 grid
72 hold on
73
74 subplot(4,3,6)
75 plot(time, r);
76 title('Yaw Rate')
77 xlabel('Time [s]')
78 ylabel('r [rad/s]')
79 grid
80
81 % Euler angles
82 subplot(4,3,7)
83 plot(time, phi);
84 title('Roll')
85 xlabel('Time [s]')
86 ylabel('\phi [rad]')
87 grid
88
89 subplot(4,3,8)
90 plot(time, theta);
91 title('Pitch')
92 xlabel('Time [s]')
93 ylabel('\theta [rad]')
94 grid
95 hold on
96
97 subplot(4,3,9)
98 plot(time, psi);
99 title('Yaw')
100 xlabel('Time [s]')
101 ylabel('\psi [rad]')
102 grid
103

```

```

104 % Position
105 subplot(4,3,10)
106 plot(time, x_f);
107 title('Position along x-axis, inertial frame')
108 xlabel('Time [s]')
109 ylabel('x_f [m]')
110 grid
111 hold on
112
113 subplot(4,3,11)
114 plot(time, y_f);
115 title('Position along y-axis, inertial frame')
116 xlabel('Time [s]')
117 ylabel('y_f [m]')
118 grid
119
120 subplot(4,3,12)
121 plot(time, -z_f);
122 title('Position along z-axis, inertial frame')
123 xlabel('Time [s]')
124 ylabel('z_f [m]')
125 grid
126 hold on
127
128
129 figure(2)
130 plot3(x_f, y_f, -z_f)
131 title('Aeroplane position, inertial frame')
132 xlabel('x_f [m]')
133 ylabel('y_f [m]')
134 zlabel('z_f [m]')
135 grid
136 hold on

```

1.2 Assignment V:b, state propagation

```

1 % Function to propagate the state
2 function next_state = state_propagation(t, state, reference_state)
3
4 % I don't want to pass too many variables to this function
5 run('T33_Shooting_Star_parameters.m');
6
7 % Rename state components for readability
8 u = state(1);
9 v = state(2);
10 w = state(3);
11
12 p = state(4);
13 q = state(5);
14 r = state(6);
15
16 phi = state(7);
17 theta = state(8);

```

```

18     psi = state(9);
19
20     x_f = state(10);
21     y_f = state(11);
22     z_f = state(12);
23
24     % Extract reference state
25     u0 = reference_state(1);
26     v0 = reference_state(2);
27     w0 = reference_state(3);
28
29     p0 = reference_state(4);
30     q0 = reference_state(5);
31     r0 = reference_state(6);
32
33     phi_0 = reference_state(7);
34     theta_0 = reference_state(8);
35     psi_0 = reference_state(9);
36
37     x_f_0 = reference_state(10);
38     y_f_0 = reference_state(11);
39     z_f_0 = reference_state(12);
40
41     % Forces
42     X = m*g*sin(theta_0) + m*Xu*(u - u0) + m*Xw*(w - w0);
43     Y = -m*g*cos(theta_0)*sin(phi_0) + m*Yv*(v - v0);
44     Z = -m*g*cos(theta_0)*cos(phi_0) + m*Zu*(u - u0) + m*Zw*(w - w0);
45
46     % Moments
47     L = I_xx*Lv*(v - v0) + I_xx*Lp*(p - p0) + I_xx*Lr*(r - r0);
48     M = I_yy*Mw*(w - w0) + I_yy*Mq*(q - q0);
49     N = I_zz*Nv*(v - v0) + I_zz*Np*(p - p0) + I_zz*Nr*(r - r0);
50
51     % State dynamics
52     next_state(1) = -q*w + r*v - g*sin(theta) + X/m; % u
53     next_state(2) = -r*u + p*w + g*sin(phi)*cos(theta) + Y/m; % v
54     next_state(3) = q*u - p*v + g*cos(phi)*cos(theta) + Z/m; % w
55
56     next_state(4) = (1/(1 - I_zz*I_xx/I_xz^2))*( I_zz*q*r*(I_zz - I_yy)/I_xz^2
57         - I_zz*p*q/I_xz - I_zz*L/I_xz^2 + p*q*(I_yy - I_xx)/I_xz + q*r - N/
58         I_xz ); % p
59
60     next_state(5) = -r*p*(I_xx-I_zz)/I_yy - I_xz*(p^2 - r^2)/I_yy + M/I_yy; %
61     q
62
63     next_state(6) = I_xx*next_state(4)/I_xz + q*r*(I_zz - I_yy)/I_xz - p*q - L
64     /I_xz; % r
65
66     next_state(7) = p + q*tan(theta)*sin(phi) + r*tan(theta)*cos(phi); % phi
67     next_state(8) = q*cos(phi) - r*sin(phi); % theta
68     next_state(9) = q*sin(phi)/cos(theta) + r*cos(phi)/cos(theta); % psi
69
70     R = [cos(theta)*cos(psi) , cos(theta)*sin(psi) , -sin(theta);
71         -cos(phi)*sin(psi) + sin(phi)*sin(theta)*cos(psi) , cos(phi)*cos(psi)
72         + sin(phi)*sin(theta)*sin(psi) , sin(phi)*cos(theta);

```

```

66         sin(phi)*sin(psi) + cos(phi)*sin(theta)*cos(psi) , -sin(phi)*cos(psi)
        + cos(phi)*sin(theta)*sin(psi) , cos(phi)*cos(theta)];
67
68     R_inv = inv(R);
69
70     next_state(10) = R_inv(1,1)*u + R_inv(1,2)*v + R_inv(1,3)*w; % x_f
71     next_state(11) = R_inv(2,1)*u + R_inv(2,2)*v + R_inv(2,3)*w; % y_f
72     next_state(12) = R_inv(3,1)*u + R_inv(3,2)*v + R_inv(3,3)*w; % z_f
73
74     % Transpose to get row vector
75     next_state = next_state';
76 end

```

1.3 Assignment I:c, comparison

```

1 % Set variables
2 run('T33-Shooting-Star-parameters.m');
3
4 % Reference state around which linearization happens
5 reference_state = [v, 0, 0, 0, 0, h];
6
7 % Force linearization parameters, [Xu, Xw, Zu, Zw, Mw, Mq]
8 force_lin_param = [Xu, Xw, Zu, Zw, Mw, Mq];
9
10 % Set ODE options
11 initial_theta = 0.1;
12 initial_conditions = [v, 0, 0, initial_theta, 0, h];
13 time_interval = [0, 100];
14
15 % Compute ODE
16 [time, state] = ode45(@(t, state) state_propagation_1_c(t, state, m, I_y,
    reference_state, force_lin_param), time_interval, initial_conditions);
17
18 % Extract all states for plotting
19 u = state(:,1);
20 w = state(:,2);
21 q = state(:,3);
22 theta = state(:,4);
23 x_f = state(:,5);
24 z_f = state(:,6);
25
26 % Plot
27 figure(1)
28 subplot(4,3,1)
29 plot(time, u, '-r');
30 title('Air speed along x-axis, aeroplane frame')
31 xlabel('Time [s]')
32 ylabel('u [m/s]')
33 grid on
34
35 subplot(4,3,3)
36 plot(time, w, '-r');
37 title('Air speed along z-axis, aeroplane frame')

```

```

38 xlabel('Time [s]')
39 ylabel('w [m/s]')
40 grid on
41
42 subplot(4,3,8)
43 plot(time, theta, '-r');
44 title('Pitch')
45 xlabel('Time [s]')
46 ylabel('\theta [rad]')
47 grid on
48
49 subplot(4,3,5)
50 plot(time, q, '-r');
51 title('Pitch Angular Velocity')
52 xlabel('Time [s]')
53 ylabel('q [rad/s]')
54 grid on
55
56 subplot(4,3,10)
57 plot(time, x_f, '-r');
58 title('Position along x-axis, inertial frame')
59 xlabel('Time [s]')
60 ylabel('x_f [m]')
61 grid on
62
63 subplot(4,3,12)
64 plot(time, z_f, '-r');
65 title('Position along z-axis, inertial frame')
66 xlabel('Time [s]')
67 ylabel('z_f [m]')
68 grid on
69
70 figure(2)
71 plot3(x_f, zeros(length(x_f),1), z_f, '-r')
72 title('Aeroplane position, inertial frame')
73 xlabel('x_f [m]')
74 ylabel('y_f [m]')
75 zlabel('z_f [m]')
76 grid on

```

1.4 Assignment V:c

```

1 % Set variables
2 run('T33.Shooting_Star_parameters.m');
3
4 % [u, v, w, p, q, r, phi, theta, psi, x-f, y-f, z-f]
5
6 % Reference state around which linearization happens
7 reference_state = [v, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -h];
8
9 % Set ODE options
10 initial_p = 0.0;
11 initial_q = 0.0;

```

```

12 initial_r = 0.1;
13 initial_conditions = [v, 0, 0, initial_p, initial_q, initial_r, 0, 0, 0, 0, 0,
    -h];
14 time_interval = [0, 10];
15
16 % Compute ODE
17 [time, state] = ode45(@(t, state) state_propagation(t, state, reference_state)
    , time_interval, initial_conditions);
18
19 % Extract all states for plotting
20 u = state(:,1);
21 v = state(:,2);
22 w = state(:,3);
23 p = state(:,4);
24 q = state(:,5);
25 r = state(:,6);
26 phi = state(:,7);
27 theta = state(:,8);
28 psi = state(:,9);
29 x_f = state(:,10);
30 y_f = state(:,11);
31 z_f = state(:,12);
32
33 %% Plot
34 figure(1)
35
36 % Air speeds
37 subplot(4,3,1)
38 plot(time, u);
39 title('Air speed along x-axis, aeroplane frame')
40 xlabel('Time [s]')
41 ylabel('u [m/s]')
42 grid on
43 hold on
44
45 subplot(4,3,2)
46 plot(time, v);
47 title('Air speed along y-axis, aeroplane frame')
48 xlabel('Time [s]')
49 ylabel('v [m/s]')
50 grid on
51 hold on
52
53 subplot(4,3,3)
54 plot(time, w);
55 title('Air speed along z-axis, aeroplane frame')
56 xlabel('Time [s]')
57 ylabel('w [m/s]')
58 grid on
59 hold on
60
61 % Turn rates
62 subplot(4,3,4)
63 plot(time, p);

```



```

64 title('Roll Rate')
65 xlabel('Time [s]')
66 ylabel('p [rad/s]')
67 grid on
68 hold on
69
70 subplot(4,3,5)
71 plot(time, q);
72 title('Pitch Rate')
73 xlabel('Time [s]')
74 ylabel('q [rad/s]')
75 grid on
76 hold on
77
78 subplot(4,3,6)
79 plot(time, r);
80 title('Yaw Rate')
81 xlabel('Time [s]')
82 ylabel('r [rad/s]')
83 grid on
84 hold on
85
86 % Euler angles
87 subplot(4,3,7)
88 plot(time, phi);
89 title('Roll')
90 xlabel('Time [s]')
91 ylabel('\phi [rad]')
92 grid on
93 hold on
94
95 subplot(4,3,8)
96 plot(time, theta);
97 title('Pitch')
98 xlabel('Time [s]')
99 ylabel('\theta [rad]')
100 grid on
101 hold on
102
103 subplot(4,3,9)
104 plot(time, psi);
105 title('Yaw')
106 xlabel('Time [s]')
107 ylabel('\psi [rad]')
108 grid on
109 hold on
110
111 % Position
112 subplot(4,3,10)
113 plot(time, x_f);
114 title('Position along x-axis, inertial frame')
115 xlabel('Time [s]')
116 ylabel('x_f [m]')
117 grid on

```

```

118 hold on
119
120 subplot(4,3,11)
121 plot(time, y_f);
122 title('Position along y-axis, inertial frame')
123 xlabel('Time [s]')
124 ylabel('y_f [m]')
125 grid on
126 hold on
127
128 subplot(4,3,12)
129 plot(time, -z_f);
130 title('Position along z-axis, inertial frame')
131 xlabel('Time [s]')
132 ylabel('z_f [m]')
133 grid on
134 hold on
135
136 figure(2)
137 plot3(x_f, y_f, -z_f)
138 title('Aeroplane position, inertial frame')
139 xlabel('x_f [m]')
140 ylabel('y_f [m]')
141 zlabel('z_f [m]')
142 grid on
143 hold on

```

1.5 Unit conversion=

```

1 % Constants
2 g = 9.81;
3
4 % Unit conversions
5 feet_to_meter = 0.3048;
6 feet2_to_m2 = 0.09290;
7 lb_to_kg = 0.4536;
8 slug_to_kg = 14.59;
9 slug_feet2_to_kg_m2 = 1.356;
10 lb_to_N = 4.448;
11
12 % Parameters in SI units
13 h = 100;
14 v = 228*feet_to_meter;
15 m = 367*slug_to_kg;
16 I_xx = 12700*slug_feet2_to_kg_m2;
17 I_yy = 20700*slug_feet2_to_kg_m2;
18 I_zz = 32001*slug_feet2_to_kg_m2;
19 I_xz = 480*slug_feet2_to_kg_m2;
20 Q = 61.7*lb_to_N/feet2_to_m2;
21
22 Xu = -0.0391;
23 Xalpha = 18.58*feet_to_meter;
24 Xw = Xalpha/v;

```

```

25 Zu = -0.248;
26 Zalpha = -213.41*feet_to_meter;
27 Zw = Zalpha/v;
28 Mu = 0.000318/feet_to_meter;
29 Malpha = -1.89;
30 Mw = Malpha/v;
31 Malpha_dot = -0.35;
32 Mw_dot = Malpha_dot/v;
33 Mq = -0.694;
34 Xdelta_e = 0.516*feet_to_meter;
35 Zdelta_e = -13.4*feet_to_meter;
36 Mdelta_e = -4.19;
37
38 Ybeta = -28.4*feet_to_meter;
39 Lbeta = -5.49;
40 Lp = -2.03;
41 Lr = 0.641;
42 Nbeta = 0.667;
43 Np = -0.116;
44 Nr = -0.207;
45 Ydelta_r = 0.0295*feet_to_meter;
46 Ldelta_r = -0.0125;
47 Ndelta_r = -1.24;
48 Ldelta_a = 6.01;
49 Ndelta_a = 0.0286;
50
51 S = 234.8*feet2_to_m2;
52 b = 37.54*feet_to_meter;
53 c = 6.72*feet_to_meter;
54
55 % Not found in data sheet
56 Yp = 0;
57 Yr = 0;
58 Yv = 0;
59 Lv = 0;
60 Nv = 0;

```