Brandon White

MAE 5010 – Autopilot Design and Test

Homework #2 (Due 09/14/2019)

==

2)

BRANDON WHITE

Per BEARD + McLAIN:

$$\begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = \frac{1}{2}\rho V_T^2 S \begin{pmatrix} C_x(\alpha) + C_{x_q}(\alpha)\frac{c}{2V_T} q + C_{x_{\delta_e}}(\alpha)\delta_e \\ \overset{\text{sym.}}{\cancelto{0}{C_{Y_0}}} + \cancelto{\alpha}{C_{Y_\beta}}\beta + C_{Y_p}\frac{b}{2V_T} p + C_{Y_r}\frac{b}{2V_T} r + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r \\ C_Z(\alpha) + C_{Z_q}(\alpha)\frac{c}{2V_T} q + C_{Z_{\delta_e}}(\alpha)\delta_e \end{pmatrix}$$

Where:

$$C_x(\alpha) = -C_D(\alpha)\cos(\alpha) + C_L(\alpha)\sin(\alpha)$$

$$C_{x_q}(\alpha) = -C_{D_q}\cos(\alpha) + C_{L_q}\sin(\alpha)$$

$$C_{x_{\delta_e}}(\alpha) = -C_{D_{\delta_e}}\cos(\alpha) + C_{D_{\delta_e}}\sin(\alpha)$$

$$C_Z(\alpha) = -\left(C_D(\alpha)\sin(\alpha) + C_L(\alpha)\cos(\alpha)\right)$$

$$C_{Z_q}(\alpha) = -\left(C_{D_q}\sin(\alpha) + C_{L_q}\cos(\alpha)\right)$$

$$C_{Z_{\delta_e}}(\alpha) = -\left(C_{D_{\delta_e}}\sin(\alpha) + C_{L_{\delta_e}}\cos(\alpha)\right)$$

and $C_L(\alpha) \triangleq eq(6)$ on HW #2.

and $C_D(\alpha) = C_{D_0} + \dfrac{(C_{L_0} + C_{L_\alpha}\alpha)^2}{\pi(0.8)AR}$

where $e = 0.8$ by assumption.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = R_{BF}(\phi,\theta,\psi)\begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} + \begin{bmatrix} T_{max}\,\delta_T \\ 0 \\ 0 \end{bmatrix}$$

3)

Discussion

To obtain the stability derivatives for the Boeing 737, the moments of inertia were calculated using provided radii of gyration. These and other provided geometric properties were compiled into a .mass file. Utilizing the Boeing 737 .avl file provided for the class and custom .mass file, the stability derivatives were calculated through use of Athena Vortex Lattice (AVL) software provided through the MIT website. There are some areas for concern related to the coefficients (particularly $C_{m\alpha}, C_{mq}, and\ C_{Lq}$), however, I have not had an introduction to AVL sufficient for debugging potential issues.

$$I_{xx} = mK_x^2 = 1767812.6\ kg * m^2$$
$$I_{yy} = mK_y^2 = 6414510.5\ kg * m^2$$
$$I_{zz} = mK_z^2 = 7480654.9\ kg * m^2$$

```
Stability-axis derivatives...

                             alpha                beta
                      ----------------     ----------------
z' force CL |     CLa =   6.247231     CLb =   0.000000
y  force CY |     CYa =  -0.000000     CYb =  -0.609577
x' mom.  Cl'|     Cla =  -0.000000     Clb =  -0.045212
y  mom.  Cm |     Cma = -29.242979     Cmb =   0.000000
z' mom.  Cn'|     Cna =   0.000000     Cnb =   0.420378

                     roll rate  p'      pitch rate  q'        yaw rate  r'
                  ----------------     ----------------     ----------------
z' force CL |     CLp =   0.000000     CLq =  71.850571     CLr =   0.000000
y  force CY |     CYp =  -0.039097     CYq =  -0.000000     CYr =   1.242737
x' mom.  Cl'|     Clp =  -0.487152     Clq =   0.000000     Clr =   0.095460
y  mom.  Cm |     Cmp =   0.000000     Cmq =-379.958527     Cmr =  -0.000000
z' mom.  Cn'|     Cnp =   0.047273     Cnq =   0.000000     Cnr =  -1.020538

Neutral point  Xnp =  19.823826

Clb Cnr / Clr Cnb  =   1.149798    (  > 1 if spirally stable )
```

Fig 1. Results of AVL Analysis of 737-800

4)

Discussion

Integration of the functions for lift, drag, and sideslip were successful and operated like expected with changes to the control inputs as seen in figure 4. Integration of the moments, however, threw the MAV simulation “for a loop” as seen in figure 5. It is unknown at this time, why the moment equations from Beard and McLain are not operating as expected. Review of the derivation of equations has not provided any artifacts of sign convention error, nor any inconsistencies in the code’s equations. Further areas of investigation will include reorganization of MAV class to make code more visually appealing and integration of a Eulerian integration method for debugging.

$$\begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = \frac{1}{2}\rho V_a^2 S \begin{pmatrix} C_X(\alpha) + C_{X_q}(\alpha)\frac{c}{2V_a}q + C_{X_{\delta_e}}(\alpha)\delta_e \\ C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}\frac{b}{2V_a}p + C_{Y_r}\frac{b}{2V_a}r + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r \\ C_Z(\alpha) + C_{Z_q}(\alpha)\frac{c}{2V_a}q + C_{Z_{\delta_e}}(\alpha)\delta_e \end{pmatrix}$$

$$\begin{pmatrix} l \\ m \\ n \end{pmatrix} = \frac{1}{2}\rho V_a^2 S \begin{pmatrix} b\left[C_{l_0} + C_{l_\beta}\beta + C_{l_p}\frac{b}{2V_a}p + C_{l_r}\frac{b}{2V_a}r + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_r}}\delta_r\right] \\ c\left[C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{c}{2V_a}q + C_{m_{\delta_e}}\delta_e\right] \\ b\left[C_{n_0} + C_{n_\beta}\beta + C_{n_p}\frac{b}{2V_a}p + C_{n_r}\frac{b}{2V_a}r + C_{n_{\delta_a}}\delta_a + C_{n_{\delta_r}}\delta_r\right] \end{pmatrix}$$

$$C_X(\alpha) \triangleq -C_D(\alpha)\cos\alpha + C_L(\alpha)\sin\alpha$$
$$C_{X_q}(\alpha) \triangleq -C_{D_q}\cos\alpha + C_{L_q}\sin\alpha$$
$$C_{X_{\delta_e}}(\alpha) \triangleq -C_{D_{\delta_e}}\cos\alpha + C_{L_{\delta_e}}\sin\alpha$$
$$C_Z(\alpha) \triangleq -C_D(\alpha)\sin\alpha - C_L(\alpha)\cos\alpha$$
$$C_{Z_q}(\alpha) \triangleq -C_{D_q}\sin\alpha - C_{L_q}\cos\alpha$$
$$C_{Z_{\delta_e}}(\alpha) \triangleq -C_{D_{\delta_e}}\sin\alpha - C_{L_{\delta_e}}\cos\alpha,$$

Fig 2. Beard and McLain Body-Frame Aerodynamic Equations

```
#Rotated coefficients
Cx = -self.CD(alpha, w_coeff[4], w_coeff[0:2], self.wing[0]/self.wing[1])*cos(alpha) + self.CL_stall(alpha, self.wing[5], w_coeff[0:2])*sin(alpha)
Cxq = -w_coeff[6]*cos(alpha) + w_coeff[2]*sin(alpha)
Cxdele = -h_coeff[7]*cos(alpha) + h_coeff[3]*sin(alpha)
Cz = -self.CD(alpha, w_coeff[4], w_coeff[0:2], self.wing[0]/self.wing[1])*sin(alpha) - self.CL_stall(alpha, self.wing[5], w_coeff[0:2])*cos(alpha)
Czq = -w_coeff[6]*sin(alpha) - w_coeff[2]*cos(alpha)
Czdele = -h_coeff[7]*sin(alpha) - h_coeff[3]*cos(alpha)

#NEEDS REVISION
#FORCES
X = Q*self.wing[0]*self.wing[1]*(Cx + Cxq*self.wing[1]/(2*V_t)*q + Cxdele*self.controls[0])
Y = Q*self.wing[0]*self.wing[1]*(v_coeff[3]*self.controls[3])
Z = Q*self.wing[0]*self.wing[1]*(Cz + Czq*self.wing[1]/(2*V_t)*q + Czdele*self.controls[0])

#MOMENTS
L = Q*self.wing[0]**2*self.wing[1]*(w_coeff[8][0]*self.controls[2] + v_coeff[8][0]*self.controls[3])
M = Q*self.wing[0]*self.wing[1]**2*(w_coeff[8][3] + w_coeff[8][4]*alpha + w_coeff[8][5]*self.wing[1]/(2*V_t)*q + h_coeff[8][1]*self.controls[0])
N = Q*self.wing[0]**2*self.wing[1]*(w_coeff[8][2]*self.controls[2])
```

Fig 3. Code Execution of Beard and McLain Equations

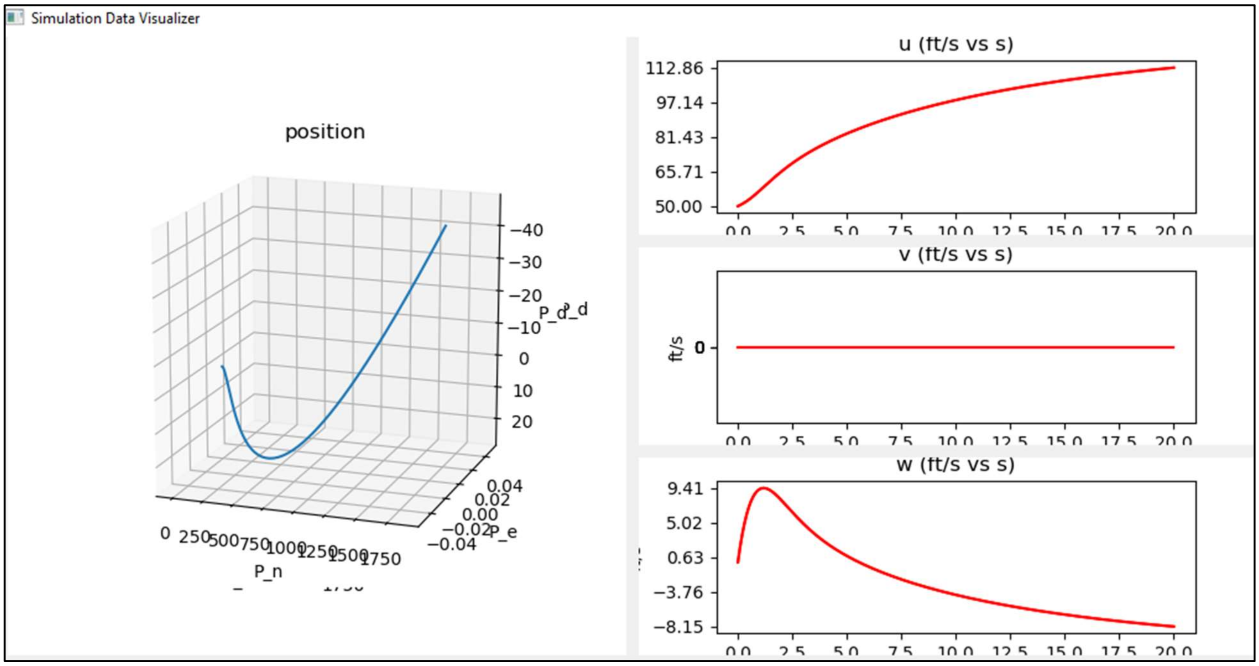

Fig 4. Simulation Results for Model with Only Force Terms

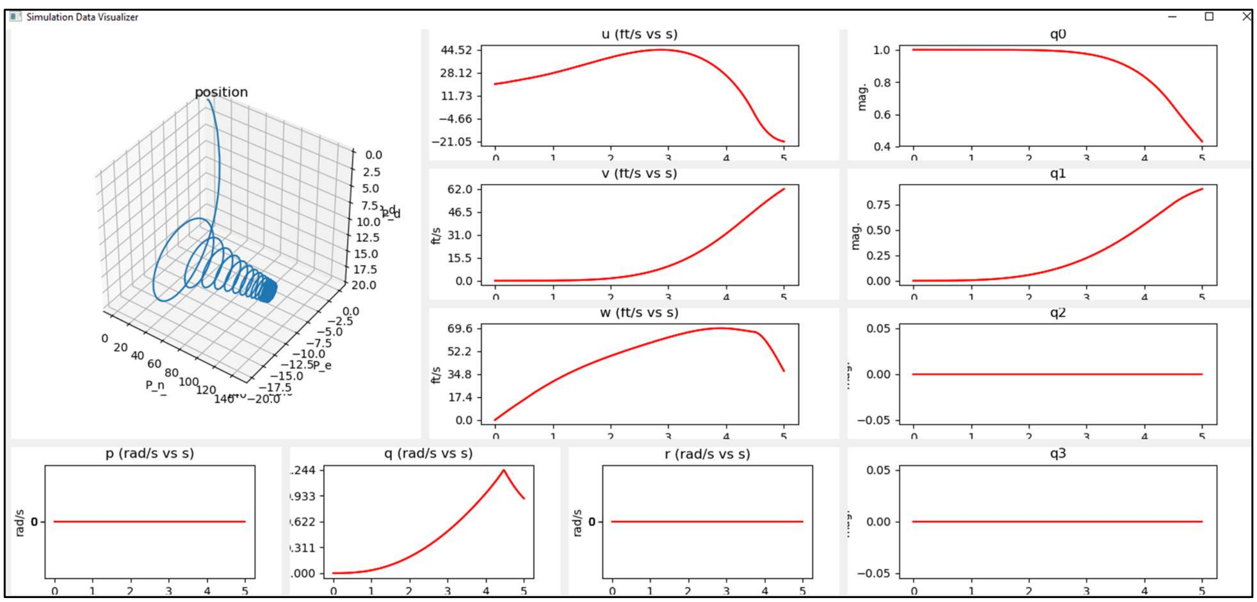

Fig 5. Simulation Results for Model with Force and Moment Terms

5)

Discussion

No amount of control inputs or good intentions were able to stabilize the simulation with moments added. The use of control inputs to alter the "No Moment Simulation" was considered instead, to verify the operation of controls in the sim. A combination of conditions for which the perturbation to altitude was mitigated over a two-minute flight time was found with u = 120ft/s, $\delta = .14\,rad$, and $T = T_{max}\delta_T|_{\delta_T=40\%}$. Note that there is oscillation of the flight path at the beginning as the MAV falls ¾ ft, but over the two minutes only nets a positive ¾ ft. It is well-expected that any gusts would play a role larger than this disturbance on a sub-30lb aircraft, so this was considered successful control configuration (albeit impractical due to the high speed requirement). This need for high speed is likely due to the neglect of moment terms in the simulation.

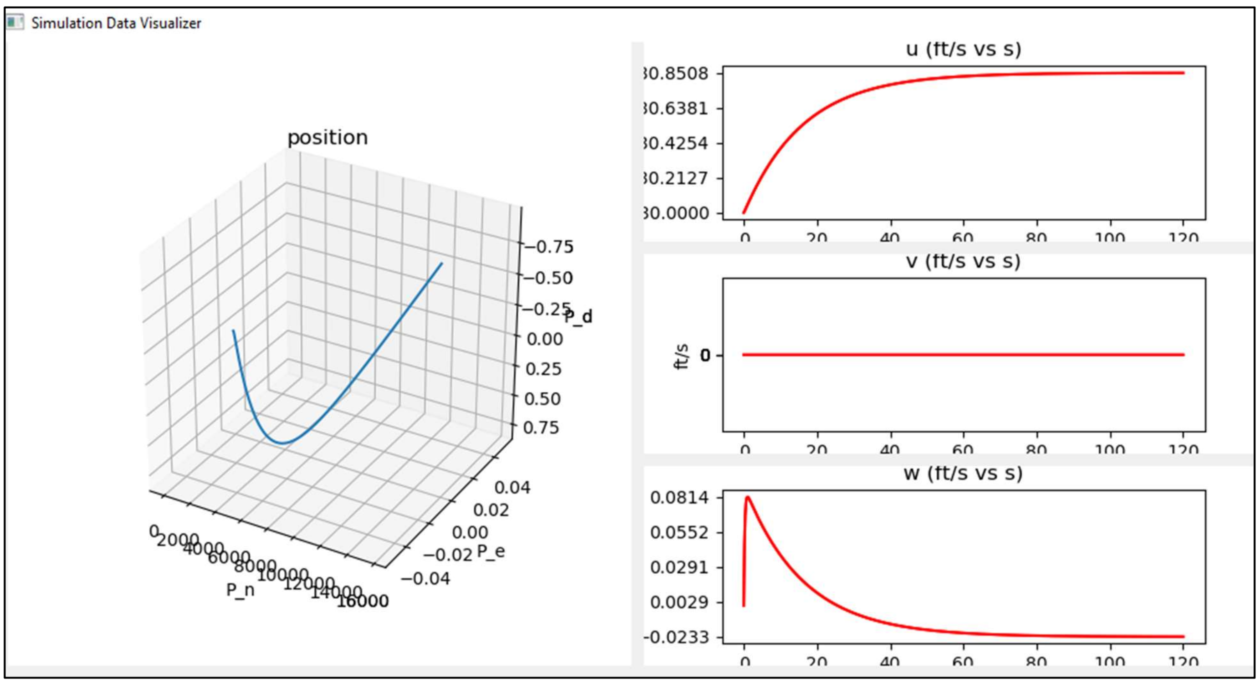

Fig. 6 Flight Path After Stabilization