Simulation of Engineering System 3 (ENG3036)

Assignment Report Template Part 2

Student Name: Andy Mannikum

Student Number: 2811078M
Assignment Topic: Instrumental landing system lateral beam guidance system

2.0 Introduction for Part 2

It was previously observed that the simulated ILS system did not align the aircraft with the center line before it reached the airfield. This presents an issue, as the aircraft must be properly aligned during its interaction with the ILS system to ensure a safe landing. Although the ILS helped to reduce and stabilise the aircraft's oscillating heading, it was not fast enough within the available time. A potential improvement has been identified by modifying the coupler in two stages. First, the optimal gain value for the coupler can be determined, followed by adding an integral term to improve accuracy. Additionally, the distance between the aircraft and the airfield has so far been treated as constant, but in reality, this distance decreases over time as the plane approaches the landing site. Accounting for this change will improve the accuracy of the simulation. However, the distance is only measured at six specific points in time, so an interpolating polynomial can be computed and integrated into the MATLAB simulation to address this limitation.

Control System Design & Implementation

1. G_c Control Gain Variation Results & Analysis

By experimenting with different values of the gain Gc, the overall trend of its effect was determined through trial and error.

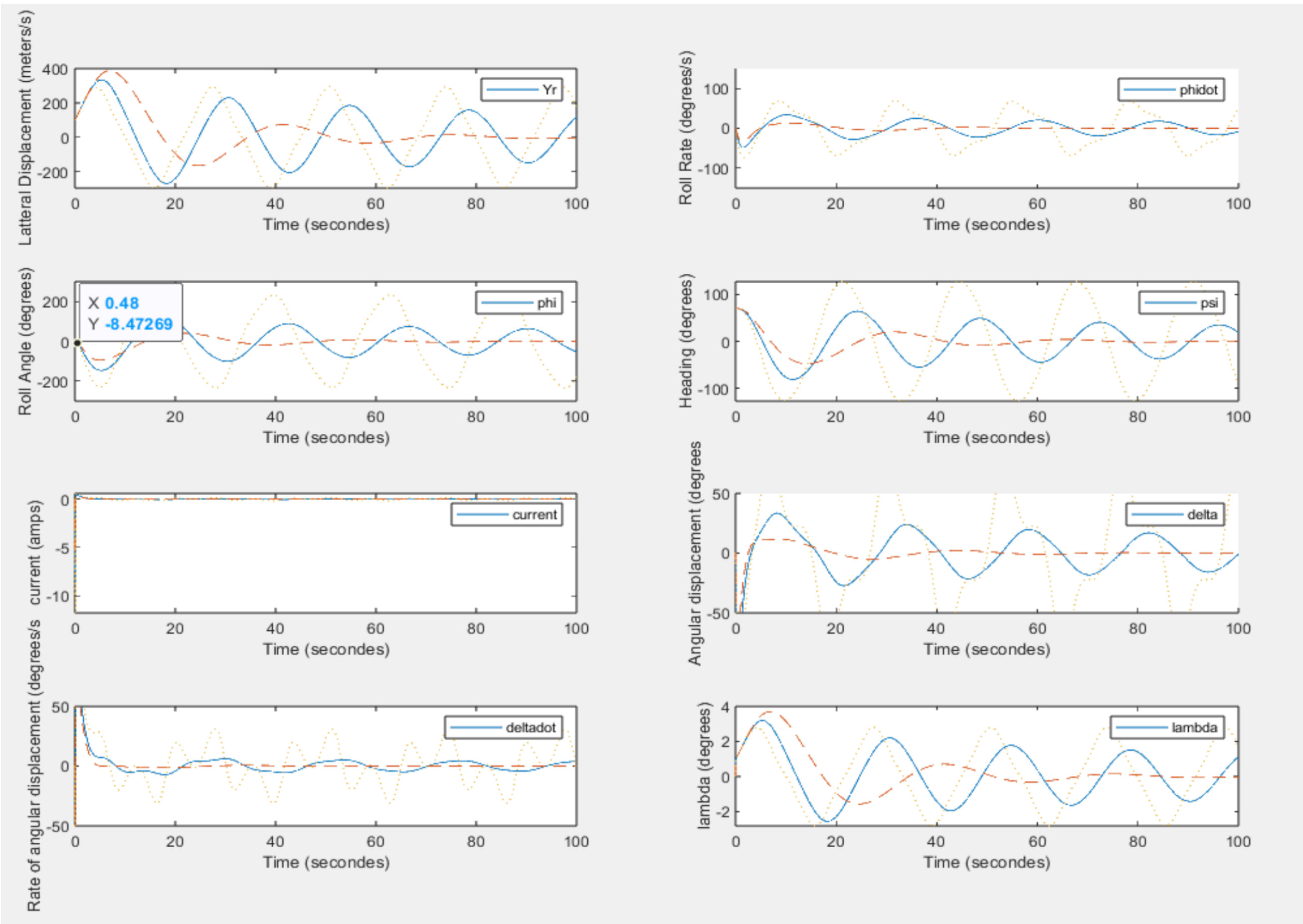

Figure 1 All parameters with varying Value of Gain Gc: Blue Solid line Gc = 45.5, Red Dash Line Gc = 20, Yellow Dotted Line Gc=91

Figure 1 illustrates the effect of changing the gain Gc. When Gc is doubled to 91, all the measured variables in the simulation become unstable, with significantly larger oscillations. Reducing Gc to 20 results in a noticeable reduction in the measured variables compared to Gc = 45.5, but they still do not stabilise at the centre line within the time limit. Therefore, further adjustments are needed to find the ideal value of Gc that will critically dampen the system.

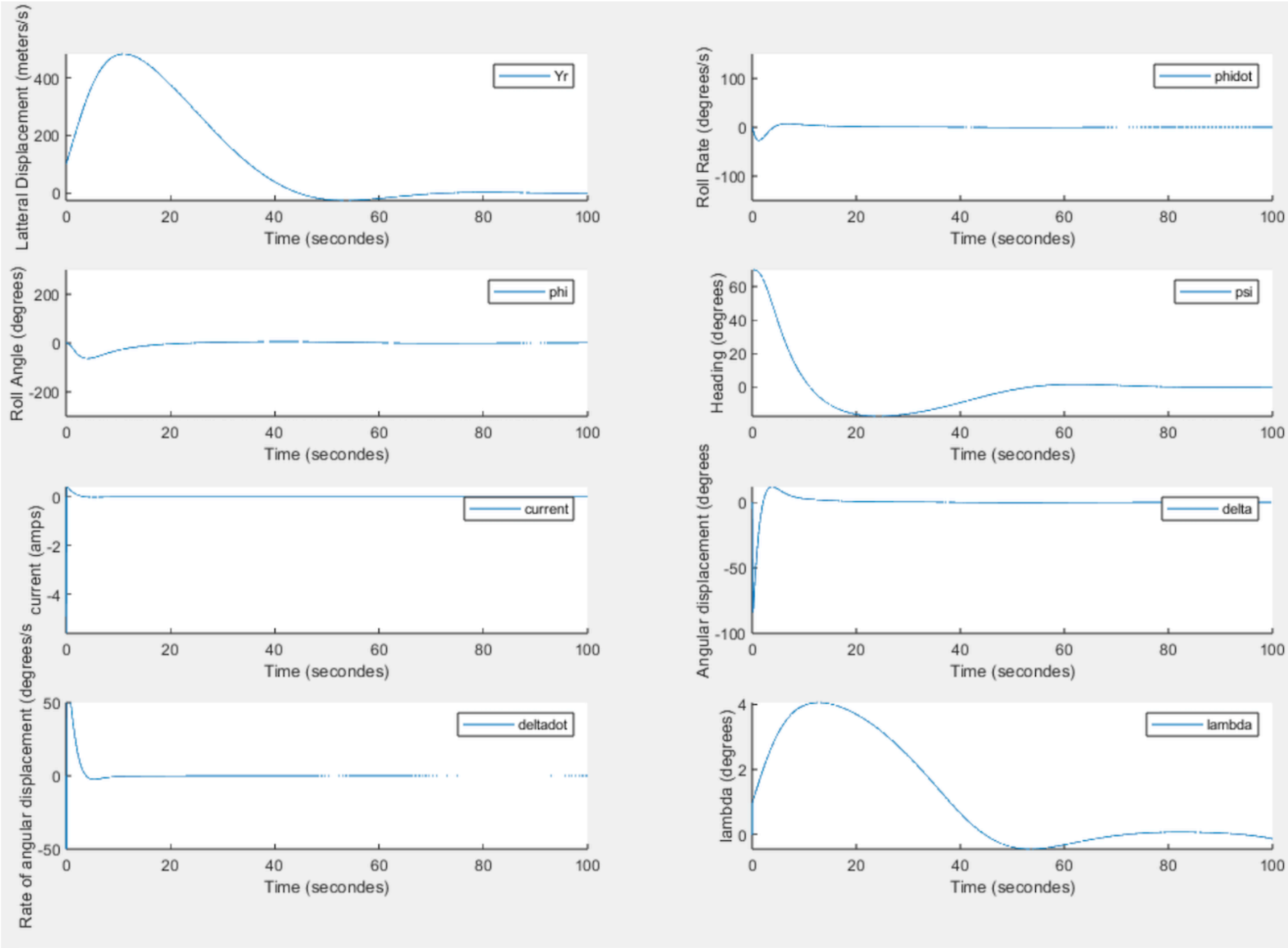

Figure 2 Graphs of Parameters with the Gain Gc = 5

After testing different values for the gain, Gc = 5 was chosen as the ideal value due to at this value, the system appears to be critically damped and stabilises within the 100 second time frame.

2. *Integration Implementation in Control System in Matlab Code K_I Control Gain Variation Results & Analysis*

The implementation of the integral term into the control system allows for another method to enhance the performance of the simulation.

We can then take the integral of delta lambda (λref –λ) and add that in the MatLab code as seen if figure 3, ensuring to also add it to the equation for V_a.

Euler's method of integration was chosen as integration method due to its simplicity and rapidity.

```
lambda = asin(x(7)/R);
delta_lambda = lamda_ref - lambda;              %change in lambda
integrator = integrator + stepsize*delta_lambda;
Psic = GC*delta_lambda + GC*Ki*integrator;      %Psi of C

xdot = ILSmodel(x,Psic);
```

Figure 3 Added Integral Term for Coupler

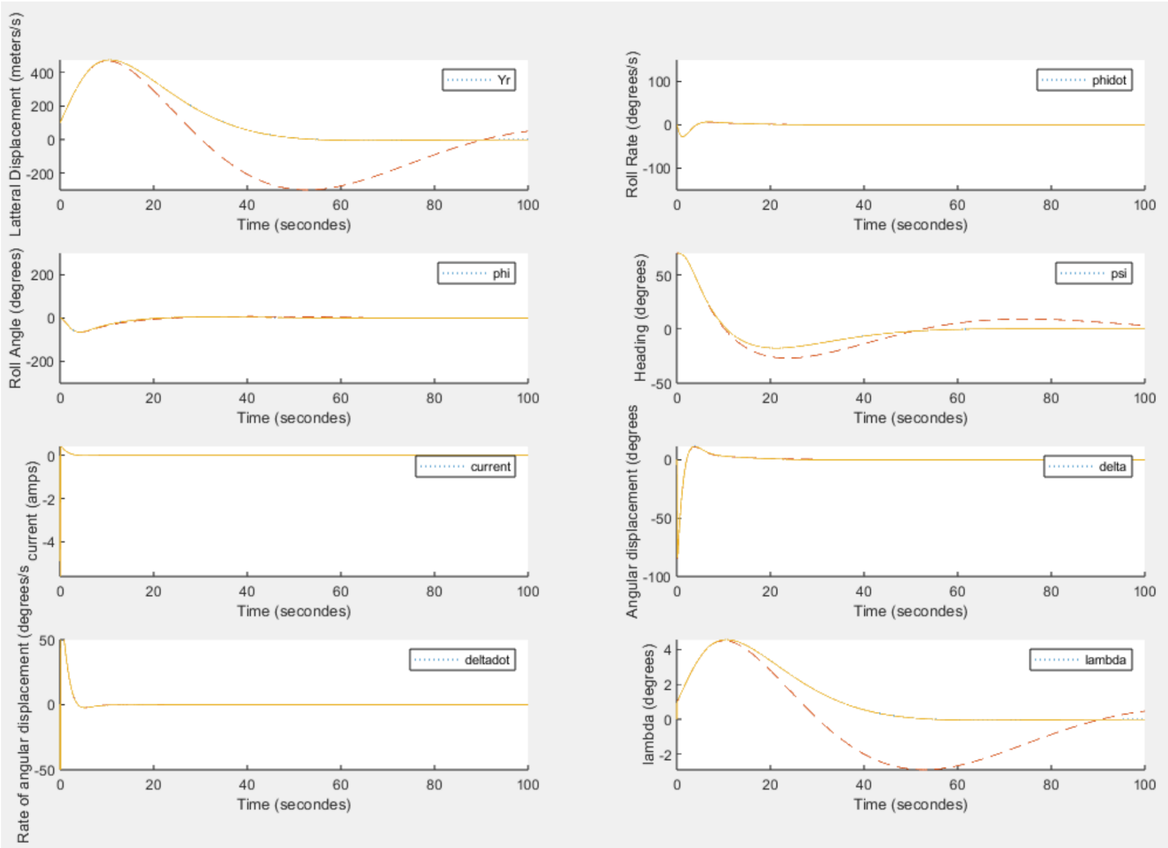

Figure 4 Experimentation with Different Values for Ki(Red Dashed line Ki=1, Yellow Solid line Ki=0.001)

Figure 4 illustrates the impact of changing the integral gain Ki. The red dashed line represents Ki = 1, while the yellow solid line shows Ki = 0.001. It is evident that Ki affects the heading (psi), lateral displacement (Yr), and error lambda. A larger Ki causes these values to become unstable, while a smaller value does not have the same effect. Ki seems to destabilise the plane's heading, so a smaller value of Ki = 0.001 is selected as the optimal choice.

Interpolation

3. Interpolation Algorithm Calculation

One of the issues with the original simulation program is that the distance between the plane and the runway remained constant. In reality, as the plane approaches the runway, this distance decreases at a significant rate. To address this, a table with six data points is provided in the brief, showing the distance between the plane and the runway at six different times. To accurately model the distance as a function of time, interpolation can be applied.

Time(s)	0	24	30	56	88	100
Range(m)	6500	5200	4000	3100	1900	430

Table 1 Distance(meters) of the Plane in Function of Time(seconds)

The interpolation polynomial will have a form seen in equation 1.

$$P_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_k(x - x_0) \ldots (x - x_{n-1}) \qquad \text{Eq 1}$$

with n and k being integers and a representing coefficients that can be found by using equation 2.

$$a_k = f[x_o, x_1, \ldots, x_k] \qquad \text{Eq 2}$$

For k = 0,1,2, . . . n and with equation 3;

$$f[x_i, x_{i+1}, \ldots, x_{i+k}] = \frac{f[x_{i+1}, x_{i+2}, \ldots, x_{i+k}] - f[x_i, x_{i+1}, \ldots, x_{i+k-1}]}{x_{i+k} - x_i} \qquad \text{Eq 3}$$

We can now find the coefficients a_0 through a_5 using a table 2 to simplify and render to operation more intuitive:

	x	F[]	f[,]	f[,,]	f[,,,]	f[,,,,]	f[,,,,,]
x0	0	6500					
			-54.1667				
x1	24	5200		-4.8611			
			-200		0.1791		
x2	30	4000		5.1683		-0.00296	
			-34.6154		-0.08153		0.00037
x3	56	3100		-0.0497		0.000719	
			-37.5		-0.02689		
x4	88	1900		-1.9318			
			-122.5				
x5	100	430					

Table 2 Calculation of coefficients a_n

Now with the coefficients calculated they can be inserted into equation 1 to achieve the polynomial as seen in equation 5

$$P_n = a_0 + a_1(x-x_0) + a_2(x-x_0)(x-x_1) + a_3(x-x_0)(x-x_1)(x-x_2) + a_4(x-x_0)(x-x_1)(x-x_2)(x-x_3) + a_5(x-x_0)(x-x_1)(x-x_2)(x-x_3)(x-x_4) \qquad \text{Eq 4}$$

$$P_4 = 6500 - 54.2(x-0) - 4.9(x)(x-24) + 0.2(x)(x-24)(x-30) - 0.003(x)(x-24)(x-30)(x-56) + 0.0004(x)(x-24)(x-30)(x-56)(x-88) \qquad \text{Eq 5}$$

4. ***Interpolation Matlab Code, Results & Analysis***

The function derived above and then converted into nested form was then implemented into the MATLAB script as shown below in Figure 5:

```
if (time >=0) && (time <=100)
     R = 6500 + time*((-54.1667)+(time-24)*(((-4.8611)+(time-30)*(0.1791+(time-56)*((-0.00296)+(time-88)*(0.00037)))));
     R_out(i) = R;
```

Figure 5 MATLAB Integration of Newton’s Divided Difference Interpolation Method

The interpolation polynomial, presented in equation 5, is implemented in the MATLAB code, specifically in the dynamic section. Ensuring high precision in the constant coefficients of the polynomial is vital for obtaining an accurate result.

Figure 6 illustrates the plane's position over time according to the interpolation polynomial. The polynomial is verified, as the x and y values match those in Table 1.

Finally, Figure 7 presents the lateral displacement, roll angle, roll rate, heading, angular displacement, rate of angular displacement, error lambda, and current, all plotted as functions of time. Overall, the results show much greater stability compared to those from the initial coupler, with all parameters converging to 0 within the 100-second period.

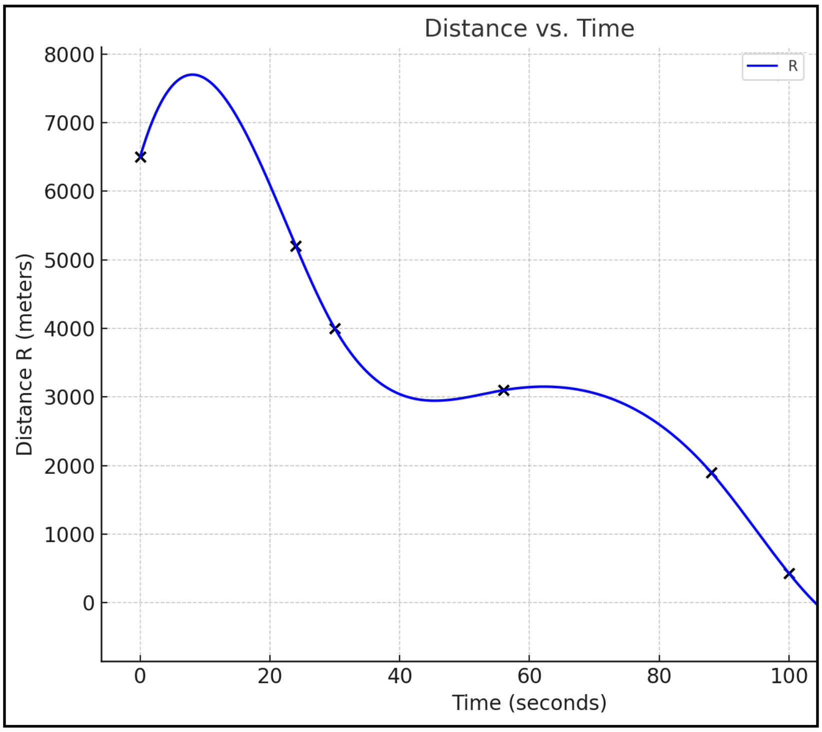

Figure 6 Plane Distance to Runway(meters) in Function of Time(seconds)

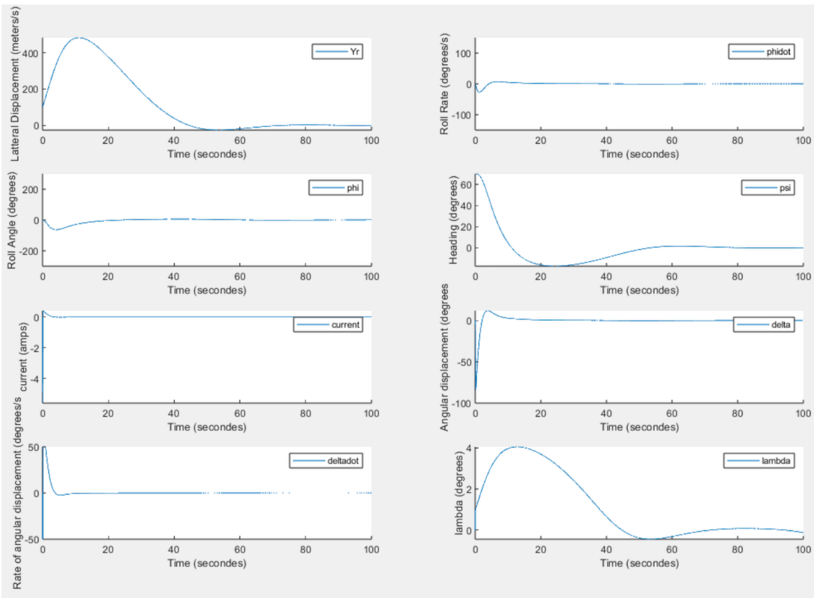

Figure 7 All parameters in Function of Time

5. ***Conclusions***

By adjusting the gain Gc, incorporating the integral into the coupler, and accounting for changes in distance, a more accurate and coherent simulation can be achieved. Selecting the appropriate values for Gc and Ki is crucial to maintain system stability and ensure critical damping. It is essential for any simulation to consider all relevant variables; if the simulation cannot replicate the expected outcome, it becomes ineffective. Compared to the initial simulation conditions, this setup allows the plane to stabilise along the centre line and safely land. To further enhance realism, real-world factors such as wind, rain, and air density must be considered, as they can influence the plane's flight path as it approaches the airfield. Factoring in these variables will ensure a safe landing under various conditions.