NTNU

TTK4190

Assignment 2

Author:
Shiv Jeet Rai

MTTK

October 15, 2015

1: Open-loop analysis

- **Dutch roll natural frequency**:

$$\omega_n = \sqrt{\alpha^2 + \beta^2} \quad \text{where} \quad \alpha = Re\{\lambda\} \quad \text{and} \quad \beta = Im\{\lambda\}$$

  Thus:

$$\det(\lambda I - A) \Rightarrow \lambda = \begin{bmatrix} -0.4224 + 3.0633i \\ -0.4224 - 3.0633i \\ -3.6152 \\ -0.0167 \\ -20.2 \\ -20.2 \end{bmatrix}$$

  we look at the complex conjugated pair and get the dutch roll natural frequency as:

$$\omega_n = \sqrt{\alpha^2 + \beta^2} == \sqrt{0.4224^2 + 3.0633^2} \approx \underline{\underline{3.0923}}$$

  Dutch roll relaticve damping ratio:

$$\zeta = \sqrt{1 - \frac{\beta}{\omega_n}^2} = \sqrt{1 - \frac{3.0633}{3.0923}^2} \approx \underline{\underline{0.1366}}$$

- **Explain what dutch roll mode is:**
  The dutch roll mode is a lateral oscillatory motion of oscillatory roll and yaw motion. Yaw is a complex interaction of the lateral DOF in yaw motion and have oscillations. This causes the air on the starboad wing and port wing (right and left) to vary, causing drag perturbations and oscillatory lift. The roll mode is a less intense non-oscillatory lateral mode that has no direct aerodynamic moment created to restore the wing level. [1][2].

- **Comute the spiral-divergence and roll modes**
  By using the formulas (5.43-5.44) in the book (Beard)

$$\lambda_{\text{SPIRAL}} \approx \underline{\underline{-3.615}}$$

  and

$$\lambda_{\text{ROLL}} \approx \underline{\underline{-0.0167}}$$

2: Autopilot for course hold using succesive loop closure

Task 2.1

From the block diagram given in the assignment the transfer function $H_{\frac{p}{\delta_a}}(s)$ was:

$$H_{\frac{p}{\delta_a}}(s) = \frac{p}{\delta_a}(s) = \frac{a_{\phi 2}}{s + a_{\phi 1}} \tag{1}$$

Rewriting eq.(1) to time domain gave:

$$\dot{p} = -a_{\phi 1} p + a_{\phi 2}\delta_a$$

When compared to the expression for $\dot{p}$ in $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$ it can be seen:

$$\begin{matrix} a_{\phi 1} \\ a_{\phi 2} \end{matrix} = \begin{matrix} -a_{33} \\ a_{35} \end{matrix} = \begin{matrix} \underline{\underline{3.6784}} \\ \underline{\underline{-0.7333}} \end{matrix}$$

Task 2.2

Given/Known:

$$\begin{aligned} a_{\phi 1} &= 3.6784 & \delta_a^{\max} &= 45\text{deg} & \zeta_\phi^{\max} &= 0.707 \\ a_{\phi 2} &= -0.7333 & e_\phi^{\max} &= 15\text{deg} \end{aligned}$$

Control gains:

- $K_{p_\phi} = \frac{\delta_a^{\max}}{e_\phi^{\max}} \text{sign}(a_{\phi 2)} = \frac{45}{3}(-1) = \underline{\underline{-3}}$

- $K_{d_\phi} = \frac{2\zeta_\phi \omega_{n_\phi} - a_{\phi 1}}{a_{\phi 2}}$
  - $\omega_{n_\phi} = \sqrt{|a_{\phi 2}| \frac{\delta_a^{\max}}{e_\phi^{\max}}} = \sqrt{0.7333 \cdot 3} \approx 1.4832$
  - ζ_ϕ, a design parameter, is chosen as $\zeta_\phi = \zeta_\phi^{\max} = 0.707$.

  $\Rightarrow K_{d_\phi} \approx \underline{\underline{2.1562}}$

- $K_{i_\phi} = \underline{\underline{0}}$ since there is no noise in the system. If we were to compensate for noise, K_{i_ϕ} could be found by using root locus

- $K_{p_\chi} = 2\zeta_\chi \omega_{n_\chi} \frac{V_g}{g}$
  - $\omega_{n_\chi} = \frac{1}{8}\omega_{n_\phi} = \frac{1.4832}{8} \approx \underline{\underline{0.1854}}$. The denominator is chosen between 5 to 10 and 8 was used here.
  - ζ_χ is a design parameter chosen as 1, because $\zeta = 1$ is critically damped.
  - $V_g = V_a = 552/3.6[m/s]$. Given in the assignment description.
  - $g = 9.81[m/s^2]$

  $\Rightarrow K_{p_\chi} = 2 \cdot 1 \cdot 0.1854 \cdot \frac{552}{3.6 \cdot 9.81} \approx \underline{\underline{5.7957}}$

- $K_{i_\chi} = \omega_{n_\chi}^2 \frac{V_g}{g} = 0.1854^2 \cdot \frac{552}{3.6 \cdot 9.81} \approx \underline{\underline{0.5373}}$

Open loop and closed loop from χ^c to χ

Open loop: $H(s) = \frac{K_{p_\chi} g/V_g s + K_{i_\chi} g/V_g s}{s^2}$
Closed loop: $H(s) = \frac{K_{p_\chi} g/V_g s + K_{i_\chi} g/V_g s}{s^2 + K_{p_\chi} g/V_g s + K_{i_\chi} g/V_g s}$

The plots can be seen when running the Matlab file.

Task 2.3

The simulation results of course hold and maneuvering can be seen when running the Matlab file. In addition there are plots of how the reference, for both course hold and maneuvering, is with and without a 1st order filter. The filter is added because a step response (or responses for maneuvering) happen instantaneously, is in practical impossible and in many cases undesired. We wish to have a satisfactory slope (which varies on how big the difference, e, is) from current position to the reference. The constraints for maximum values, namely $\delta_a^{\max} = 45$deg and $e_\phi^{\max} = 15$deg, were implemented by saturation blocks. The course holding and maneuvering worked quite well, as seen in the simulations.

The steps, value and step time, were chosen as Figure 1 shows. The values were chosen arbitrarily and both negative and positive maneuvering was tried. Note that the maneuvering step values are added, this be seen in the simulink figure (Figure 2). The step times were chosen big such that the autopilot would have enough time to reach the reference before new reference was given.

Note: T = 20 could perhaps been smaller such that a more step response characteristic reference would have been obtained.

The state space model was implemented with only one input, namely δ_a^c, this can be seen in the simulink if it is of interest. In other words the rudder input, δ_r^c was assumed 0.

3: MIMO linear autopilot for course hold

Task 3.1

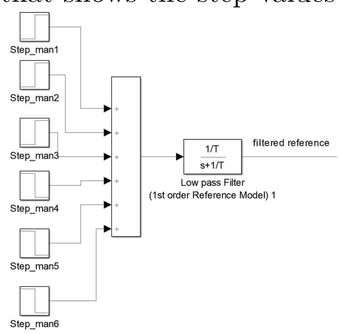
The Figure 3 and Figure 4 shows how the root locus function was found and how the root locus plot was used to find the gain Kr. Note: Ki in Figure 4 is Kr, I just had a spelling mistake there.

Note: Kr = -0.807, but that might not be the optimal. When reading a root locus diagram one wished to reduce the oscillatory property by having the poles closer the real axis while preventing the system to be too slow.

```
%% Course hold
T = 20; % Time constant for 1st order low pass filter as reference model
step_course_hold = 25;
step_course_hold_start_time = 200;
%% Maneuvers (These are added!!)
step_manuever1 = 20;
step_manuever2 = 60;
step_manuever3 = 70;
step_manuever4 = -50;
step_manuever5 = -80;
step_manuever6 = -60;

step_manuever1_start_time = 200;
step_manuever2_start_time = 400;
step_manuever3_start_time = 600;
step_manuever4_start_time = 800;
step_manuever5_start_time = 1000;
step_manuever6_start_time = 1200;
```

Figure 1: Matlab script that shows the step values and step times used



,

Figure 2: Simulink-part that shows the step values were added for maneuvering

In other words my Kr is sufficient (as seen in the maneuvering plot in task 3.2), but perhaps not optimal.

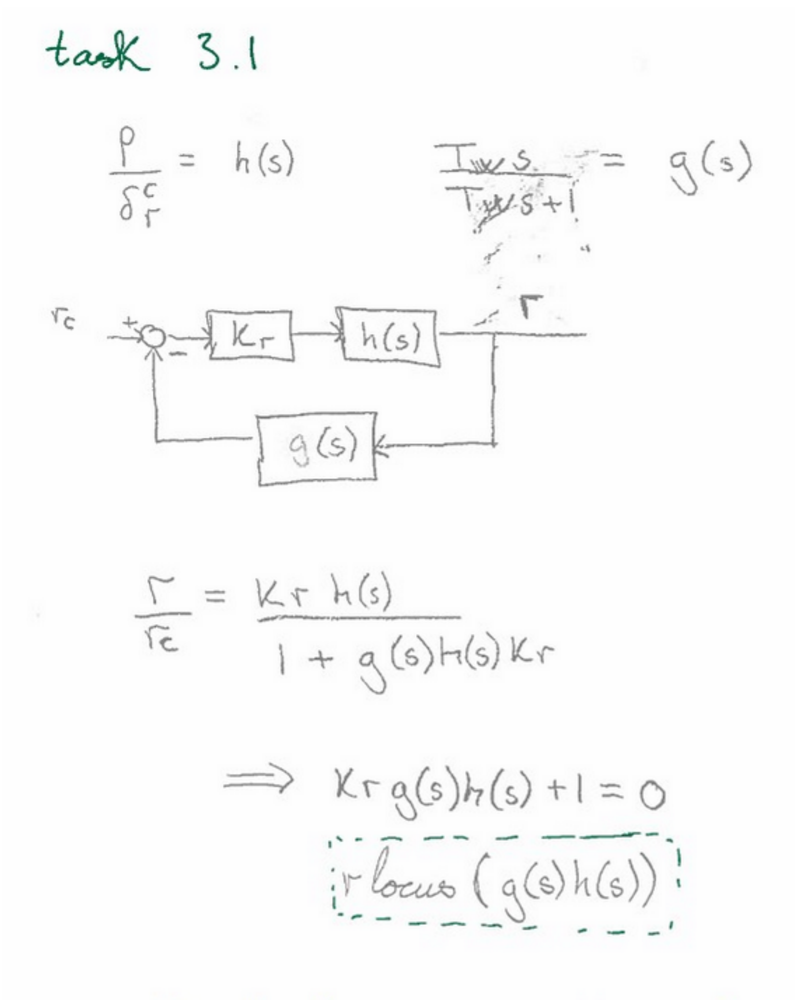


Figure 3: Rewriting the closed loop to Evans form to find root locus function

Comment on the plot showing r_c and r when running this task in Matlab: We clearly see that the actual do not reach the reference. This could have been improved if I had chosen a more optimal Kr gain, however the closed loop also need an integral term to remove the stationary deviations. This can be seen by applying "sluttverditeoremet" with the input as a step, as given in this task.

$$\Rightarrow \lim_{t\to\infty} r(t) = \lim_{s\to 0} s \cdot r(s) = s \cdot H_{\frac{r}{r_c}}(s) \cdot r_c(s) = \text{stationary difference}$$

$$\text{with:} \quad H_{\frac{r}{r_c}}(s) = \frac{57.91s^4 + 286.3s^3 + 300.4s^2 + 118.4s + 46.46}{s^6 + 25.68s^5 + 185.7s^4 + 622.7s^3 + 1066s^2 + 761.3s + 11.69}$$

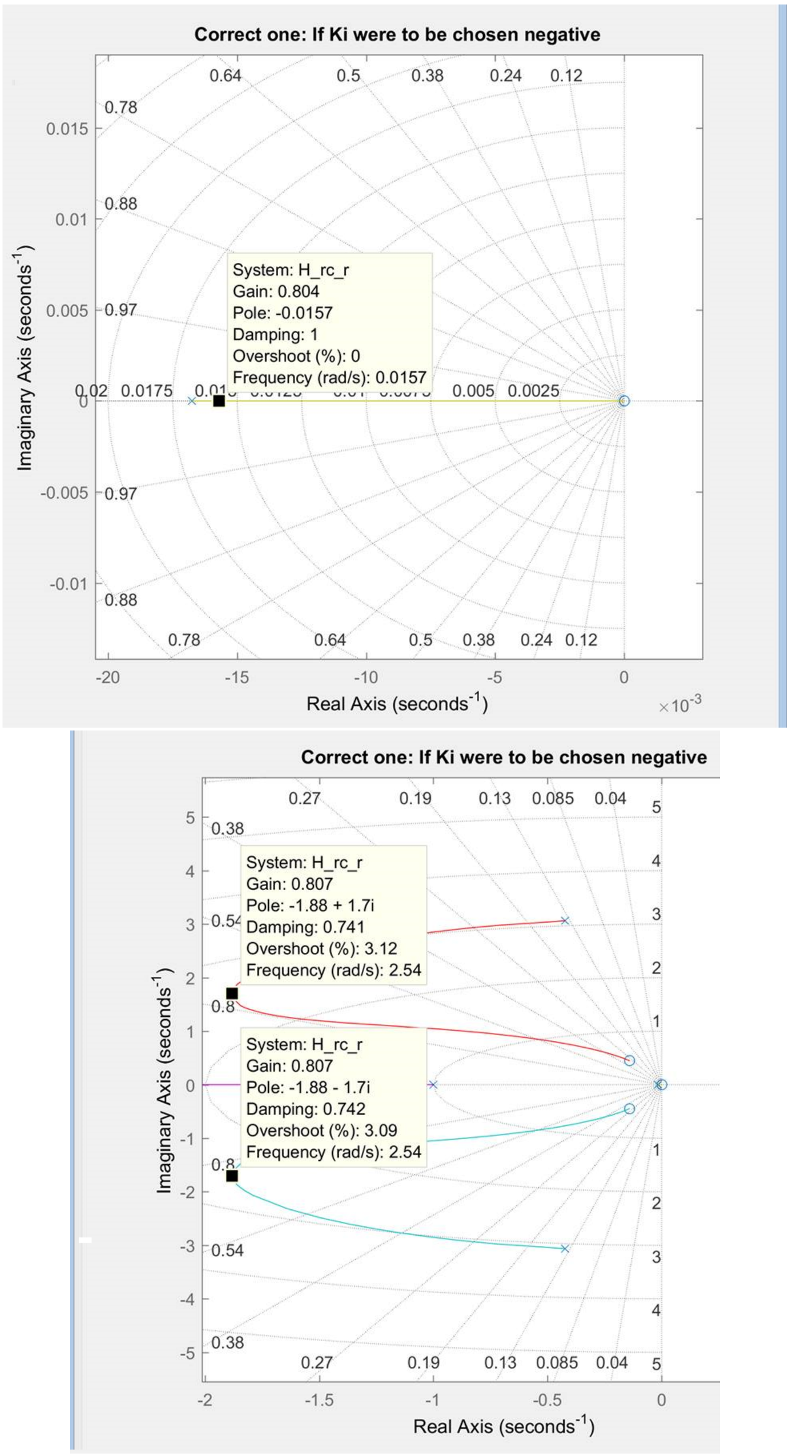


Figure 4: Kr chosen as -0.807 from root locus

The Kr=-0.807 in that transfer function.

The aim with the plot was probably to see that root locus would give a Kr such that the actual might reach reference, or better than mine. I still think that an integral term would be good have.

Task 3.2

Redrawing the original system as in Figure 5 and comparing it with the autopilot in task 2, we see that:

$$K_\phi = K_{p_\phi}$$
$$K_p = K_{d_\phi}$$

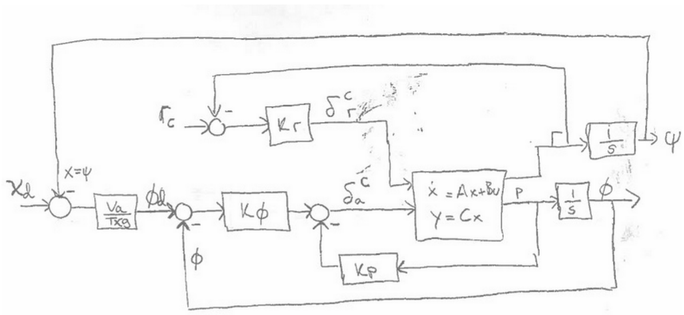


Figure 5: Redrawn block diagram

Regarding $\phi_d = \frac{V_a}{T_\chi g}$, it can be shown by putting:

$$\dot{\chi} = \frac{1}{T_\chi}(\chi_d - \chi)$$

into:

$$\dot{\chi} = \frac{g}{V_a} tan(\phi)$$

Rewriting the expression and using $tan(phi) \approx \phi$ for small ϕ values, gives:

$$\phi = \frac{V_a}{T_\chi g}(\chi_d - \chi)$$

since it is desirable that $e_\phi = (\phi_d - \phi) \rightarrow 0 \quad \text{as} \quad t \rightarrow \infty$, we can say $\phi_d = \phi$. Thus:

$$\phi_d = \frac{V_a}{T_\chi g}(\chi_d - \chi) \qquad \text{Q.E.D}$$

Comments on the simulations in this task

Using the same steps (value and time) as task 2.3, the response is okay, it manages to follow the reference, see the plots. However the (δ_a VS δ_a^c) and (δ_r VS δ_r^c) made little sense. I thought we were to see some symmetry due that the relationship between δ_a and δ_a^c was given by $H_a(s) = \frac{20.2}{s+20.2}$, thus would $\delta_a < \delta_a^c$ and have tops and buttoms and same time. This can not be seen. I think this can either be because of:

- Wrong implementation of the outputs in simulink, but I couldn't seem to find out where.

Task 3.3

The simulations shows the reference along with the two autopilots and it can clearly be seen that the autopilot from task 2.3 was better. However that might be because the controller gains were given by constants given in the description and using formulas. We only used ζ_χ and ω_{n_χ} as design parameters and I chose $\zeta_c hi = 1$ (critcally damped) and $\omega_{n_\chi} = \frac{1}{8}\omega_{n_\phi}$, if chosen differently we could have that the autopilot in task 2 became slower.

In the autopilot in task 3 I found $K_r = -0.807$, but that might differ from the optimal one, since removing oscillatory properties (less imaginary value on the poles in Figure 4) might be more desirable than that the poles are a bit more left in the plane. In addition I used $K_\phi = K_{p_\phi}$ and $K_p = K_{d_\phi}$, which is not completely true. With that I mean that $K_\phi = K_{p_\phi}$ and $K_p = K_{d_\phi}$ are good as starting values, but they should probably have been tuned a bit different (a little different in values) due to the autopilot having somewhat other dynamics (properties) than the autopilot in task 2. Thus I think that the autopilot in task 3 should have been better due to being a MIMO lateral pilot, but that might not be true as well, since having more inputs and outputs might not necessary be better. Thus if I had tuned the autopilot in task 3 better, we might had seen another result.

Given the simulations I presented I would say that the autopilot from task 2 is better.

Bibliography

[1] $https://en.wikipedia.org/wiki/Aircraft\_dynamic\_modes$

[2] $https://en.wikipedia.org/wiki/Dutch\_roll$