CONTROL LAB EXERCISE

The basic lab exercise assumes that you have some experience with Matlab and computer programming. More proficient programmers may wish to do the additional exercise in section 3.2.

1. Objective

The objective of the lab is to study the effect of feedback gains on autopilot performance.

2. Theory

A simple pitch plane autopilot with two feedback loops is represented by the block diagram below:

For a flight speed $V = 380$ m/s, the missile perturbations about the trim condition is modelled by the short period approximation:

\[
\begin{align*}
\alpha' &= -3.3 \alpha + q - 0.89 \delta e \\
q' &= -248 \alpha - 662 \delta e
\end{align*}
\] (1)

The servo dynamics is represented by a first order system with a time constant $T = 0.001$

\[
\delta e' = \left( \delta e_{\text{cmd}} - \delta e \right) / T
\] (3)

The feedback consists of the pitch rate $q$ and the integrated acceleration error $e$:

\[
e' = a_{\text{cmd}} - a_z
\] (4)

Hence the control law takes the form:

\[
\delta e_{\text{cmd}} = K_a e + K_q q
\] (5)
We substitute the control law (5) into the servo model (3) and noting that the acceleration is given by 
\[ a_z = V(\alpha' - q) = V(-3.3 \alpha - 0.89 \delta e), \]
the closed loop system take the form:

\[
\begin{align*}
\alpha' &= -3.3 \alpha + q - 0.89 \delta e \\
q' &= -248 \alpha - 662 \delta e \\
\delta e' &= (K_a q - \delta e + K_q e) / T \\
e' &= -V(-3.3 \alpha - 0.89 \delta e) + a_{cmd} \tag{6}
\end{align*}
\]

For this lab exercise we will select the gains \(K_a\) and \(K_q\) so that the autopilot will follow a unit step acceleration command. ie \(a_{cmd} = 1\) for \(t > 0\).

### 3. Procedure

#### 3.1 Basic Lab exercise

1. Create a directory on your PC and place the following Matlab files (downloaded from the course website) in it
   i) ASM.m
   ii) fm.m
   iii) missile.m
   iv) perf.m

2. Start Matlab on your PC and change Matlab's path to your directory by typing at the prompt

   \[ cd \ c:\your\_directory\_name \]

3. Run the Matlab programme ASM.m by typing \texttt{asm} at the prompt. Enter the recommended gains \(K_a = 0.008\) and \(K_q = 2.0\) and observe the resulting step response. Does the response follow the commanded acceleration closely?

4. Now vary the gains to see if you can improve the response time of the autopilot without incurring any overshoot. By varying one gain at a time, can you deduce the effect of each gain on the autopilot performance?

5. In the file \texttt{fm.m} we have constructed a simple performance index by measuring the total deviation and overshoot of the acceleration for the first 4s. We can plot this performance index as a function of both gains using a contour plot, type

   \[ perf \]

   The range of gains used for the contour plot are \(K_a = [0.001, 0.003]\) and \(K_q = [0.01, 0.03]\). Is there an optimal choice of gains? Can you determine it’s approximate value?
6. You can let Matlab determine the “optimal” gains by typing at the prompt

\texttt{fminserach( 'fm', [ka, kq] )}

\textit{ka} and \textit{kq} are your guess for the optimal values of \( K_a \) and \( K_q \). This may take a few minutes. Check the step response for the computed optimal gains using \textit{asm}. Is there any improvement in response time? If you didn’t have the benefit of a contour plot can you still locate the optimal gains? Try an initial guess like \([0.008, 2.0]\) and find out..

3.2 Additional Lab Exercise

If you are proficient in Matlab programming, try modifying the performance index used in \textit{fm.m} and try computing the corresponding optimal gains. What is a “good” performance index? Can we use the autopilot bandwidth as a performance index?

4. Matlab code

4.1 ASM.m

```matlab
% stability derivatives at 380 m/s
V = 380;
Za = -3.3; Zq = 0; Ze = -0.89;
Ma = -248; Mq = 0; Me = -662;

% servo time constant
T = 0.001;

% accelerometer feedback gain (0.0021)
Ka = input('enter integrated acceleration error gain (0.008) : ');

% pitch rate gyro feedback gain (0.0262)
Kq = input('enter pitch rate gain (2.0) : ');

% closed loop system (\( a \ q \ e \ i \))
A = [ Za 1 Ze 0;
    Ma Mq Me 0;
    0 Kq/T -1/T Ka/T;
    -V*Za 0 -V*Ze 0];

B = [ 0; 0; 0; 1];
C = [V*Za 0 V*Ze 0];
D = [0];

sys = ss(A,B,C,D);
time = 0:0.01:4;
step(sys, time)
```
4.2 fm.m
function err = fm(K)
% err = fm(K)
% returns performance index = total deviation + overshoot penalty
% K = [acceleration gain, pitch rate gain]

% stability derivatives at V = 380 m/s
V = 380;
Za = -3.3; Zq = 0; Ze = -0.89;
Ma = -248; Mq = 0; Me = -662;

% servo time constant
T = 0.001;

% closed loop system - step input
% x = [aoa, pitch rate, elevator, integrated az error]

% monitor step response for 4 sec
Tf = 4; dt = 0.01;
xo = [0 0 0 0];
[t, y] = ode23('missile', 0:dt:Tf, xo, [], K);

% step acceleration cmd
ac = ones(length(t) , 1);
az = V*(Za*y(:,1) + Zq*y(:,2) + Ze*y(:,3));

% check for overshoot
maz = max(az);
e0 = 0;
if (maz > 1.0)
    e0 = (maz-1);
end

% composite error based on overshoot and deviation
err = sum(abs( ac - az ))*dt + e0;

4.3 missile.m
function f = missile(t, x, flag, K)
% missile closed loop equations for a step input
Ka = K(1); Kq = K(2);

% stability derivatives at V = 380 m/s
V = 380;
Za = -3.3; Zq = 0; Ze = -0.89;
Ma = -248; Mq = 0; Me = -662;

% servo time constant
T = 0.001;

% closed loop system - step input
% x = [aoa, pitch rate, elevator, integrated az error]
ac = 1;
\[ azt = (Z_a x(1) + Z_q x(2) + Z_e x(3)); \]

\[ f = \text{zeros}(1,4); \]
\[ f = [ azt + x(2); \]
\[ \quad M_a x(1) + M_q x(2) + M_e x(3); \]
\[ \quad (K_q x(2) - x(3) + K_a x(4))/T; \]
\[ \quad a_c - V \times azt]; \]