A Design Study Approach to Classical Control

Randal W. Beard Timothy W. McLain
Brigham Young University

Updated: May 17, 2016

Homework C.11

The objective of this problem is to implement state feedback controller using the full state. Start with the simulation files developed in Homework **??.??**.

(a) Using the values for ω_{n_ϕ}, ζ_ϕ, ω_{n_θ}, and ζ_θ selected in Homework **??.??**, find the desired closed loop poles.

(b) Add the state space matrices A, B, C, D derived in Homework **??.??** to your param file.

(c) Verify that the state space system is controllable by checking that $\text{rank}(\mathcal{C}_{A,B}) = n$.

(d) Find the feedback gain K so that the eigenvalues of $(A - BK)$ are equal to desired closed loop poles. Find the reference gain k_r so that the DC-gain from ϕ_r to ϕ is equal to one.

(e) Implement the state feedback scheme and tune the closed loop poles to get good response. You should be able to get much faster response using state space methods.

Solution

From HW **??.??**, the state space equations for the satellite are given by

$$
\begin{aligned}
\dot{x} &= \begin{pmatrix} 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 1.0000 \\ -0.0300 & 0.0300 & -0.0100 & 0.0100 \\ 0.1500 & -0.1500 & 0.0500 & -0.0500 \end{pmatrix} x + \begin{pmatrix} 0 \\ 0 \\ 0.2 \\ 0 \end{pmatrix} u \\
y &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \end{pmatrix} x \qquad (1)
\end{aligned}
$$

where Equation (1) represents the measured outputs. The reference output is the angle ϕ, which implies that

$$y_r = \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix} x.$$

Step 1. The controllability matrix is

$$\mathcal{C}_{A,B} = [B, AB, A^2B, A^3B] = \begin{pmatrix} 0 & 0.2000 & -0.0020 & -0.0059 \\ 0 & 0 & 0.0100 & 0.0294 \\ 0.2000 & -0.0020 & -0.0059 & 0.0007 \\ 0 & 0.0100 & 0.0294 & -0.0036 \end{pmatrix}.$$

The determinant is $det(\mathcal{C}_{A,B}) = -36,000 \neq 0$, therefore the system is controllable.

Step 2. The open loop characteristic polynomial is

$$\Delta_{ol}(s) = \det(sI - A) = s^4 + 0.0600s^3 + 0.18s^2$$

which implies that

$$
\begin{aligned}
\mathbf{a}_A &= (0.06, 0.18, 0, 0) \\
\mathcal{A}_A &= \begin{pmatrix} 1 & 0.06 & 0.18 & 0 \\ 0 & 1 & 0.06 & 0.18 \\ 0 & 0 & 1 & 0.06 \\ 0 & 0 & 0 & 1 \end{pmatrix}.
\end{aligned}
$$

Step 3. The desired closed loop polynomial is

$$
\begin{aligned}
\Delta_{cl}^d(s) &= (s^2 + 2\zeta_\theta \omega_{n_\theta} s + \omega_{n_\theta}^2)(s^2 + 2\zeta_\phi \omega_{n_\phi} s + \omega_{n_\phi}^2) \\
&= s^4 + 4.9275s^3 + 12.1420s^2 + 14.6702s + 8.8637,
\end{aligned}
$$

when $\omega_\theta = 1.9848$, $\zeta_\theta = 0.707$, $\omega_\phi = 1.5$, $\zeta_\phi = 0.707$, which implies that

$$\boldsymbol{\alpha} = (4.9275, 12.1420, 14.6702, 8.8637).$$

Step 4. The gains are therefore given by

$$\begin{aligned} K &= (\boldsymbol{\alpha} - \mathbf{a}_A)\mathcal{A}_A^{-1}\mathcal{C}_{A,B}^{-1} \\ &= (40.2842, 255.1732, 24.3375, 366.1825) \end{aligned}$$

The feedforward reference gain $k_r = -1/C_r(A - BK)^{-1}B$ is computed using $C_r = (0, 1, 0, 0)$, which gives

$$\begin{aligned} k_r &= \frac{-1}{C_r(A - BK)^{-1}B} \\ &= 295.4573. \end{aligned}$$

Alternatively, we could have used the following Matlab script

```
% state space design
A = [...
    0, 0, 1, 0;...
    0, 0, 0, 1;...
    -P.k/P.Js, P.k/P.Js, -P.b/P.Js, P.b/P.Js;...
    P.k/P.Jp, -P.k/P.Jp, P.b/P.Jp, -P.b/P.Jp;...

];
B = [0; 0; 1/P.Js; 0];
Cr = [0, 1, 0, 0];


% gains for pole locations
wn_th    = 2*0.9924;
zeta_th = 0.707;
wn_phi    = 1*1.5;
zeta_phi  = 0.707;
ol_char_poly = charpoly(A);
des_char_poly = conv([1,2*zeta_th*wn_th,wn_th^2],...
                [1,2*zeta_phi*wn_phi,wn_phi^2]);
des_poles = roots(des_char_poly);

% is the system controllable?
if rank(ctrb(A,B))≠4, disp('System Not Controllable'); end
P.K = place(A,B,des_poles);
P.kr = -1/(Cr*inv(A-B*P.K)*B);
```

The Matlab code for the controller is given by

```
function tau=satellite_ctrl(in,P)
    phi_d   = in(1);
    phi     = in(2);
    theta   = in(3);
    t       = in(4);

    % use a digital differentiator to find phidot and thetadot
    persistent phidot
    persistent phi_d1
    persistent thetadot
    persistent theta_d1
    % reset persistent variables at start of simulation
    if t<P.Ts,
        phidot        = 0;
        phi_d1        = 0;
        thetadot    = 0;
        theta_d1    = 0;
    end
    phidot = (2*P.tau-P.Ts)/(2*P.tau+P.Ts)*phidot...
        + 2/(2*P.tau+P.Ts)*(phi-phi_d1);
    thetadot = (2*P.tau-P.Ts)/(2*P.tau+P.Ts)*thetadot...
        + 2/(2*P.tau+P.Ts)*(theta-theta_d1);
    phi_d1 = phi;
    theta_d1 = theta;

    % construct the state
    x = [phi; theta; phidot; thetadot];
    % compute the state feedback controller
    tau = sat( -P.K*x + P.kr*phi_d, P.taumax);
end
```