Tugas Besar Sistem Kendali Multivariabel (SKM)

Analisa Performansi

Inverted Pendulum on a Cart System

SKM – 41 – C

Gede Haris Widiarta (1102174038)

DAFTAR ISI

Ketentuan Tugas Besar SKM

1. Tinjau sebuah sistem fisik (minimal sistem orde 2) lalu modelkan sistem tersebut kedalam bentuk state-space. Jelaskan secara detail sistem tersebut dan parameter-parameter nya. Pastikan dan tunjukkan sistem tersebut *controllable* dan *observable*.
2. Analisis sistem tersebut (kestabilan, performansi -> unit step open-loop system response)
3. Improve performansi sistem dengan menerapkan state-feeback control (pole placement atau lqr). Nyatakan dalam spesifikasi respon sistem.
4. Rancang state-feeback control agar spesifikasi tersebut tercapai dan tunjukkan nilai control gain k.
5. Asumsikan bahwa seluruh state tidak tersedia lalu rancang closed loop estimator untuk sistem tanpa state-feedback control.
6. Asumsikan bahwa seluruh state tidak tersedia lalu rancang closed loop estimator untuk sistem dengan state-feedback control.
7. Analisis hasil rancangan pengontrol yang telah anda desain.

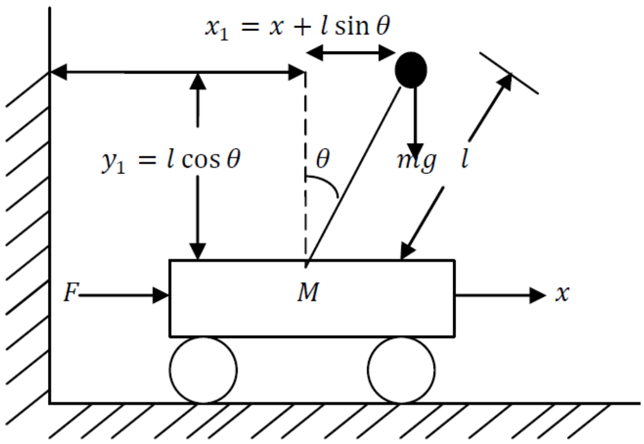
Notes:

- Tugas dikerjakan secara individual. Sistem boleh sama maksimal 3 orang dalam satu kelas namun parameter harus berbeda.
- Setiap tahap dilengkapi dengan penjelasan dan gambar simulasi jika ada
- Dokumen yang perlu dikumpulkan di lms (60%):
    - Laporan
    - Code
    - Referensi.
- Link video penjelasan (40%)

  Presentasi secara online pada minggu uas dengan durasi maksimum 7 menit. (bonus)

A. Pemodelan Matematis Sistem

Sebelum merancang desain kendali suatu sistem, penting untuk mengetahui dinamika sistem agar dapat menentukan parameter – parameter apa saja yang akan diatur. Berikut merupakan gambaran dari sistem *inverted pendulum on a cart*[1].



Gambar 1. Cart Inverted Pendulum System

Keterangan parameter:

Simbol	Keterangan	Nilai
M	massa *cart* (kg)	1.5
m	massa pendulum (kg)	0.3
b	koefisien gesek *cart* massa *cart* (Ns/m)	0.05
l	panjang pendulum dari titik massa massa *cart* (m)	0.4
I	momen inersia pendulum (kg/m^2)	0.099
F	gaya yang diberikan (N)	
X	posisi *cart*	
θ	sudut pendulum	

Berdasarkan gambar tersebut, didapat persamaan model matematika sistem yaitu:

$$x_1 = x + l\ sin\theta$$

$$y_1 = l\ cos\theta$$

$$\dot{x}_1 = \dot{x} + l\dot{\theta}\ cos\theta$$

$$\dot{y}_1 = -l\dot{\theta}\ sin\theta$$

Kemudian, V_1 sebagai resultan kecepatan *cart* dan pendulum, yaitu:

$V_1^2 = \dot{x}_1{}^2 + \dot{y}_1{}^2$

$V_1^2 = (\dot{x} + l\dot{\theta}\ cos\theta)^2 + (-l\dot{\theta}\ sin\theta)^2$

$V_1^2 = \dot{x}^2 + 2\dot{x}l\dot{\theta}\ cos\theta + l^2\dot{\theta}^2$.. (1)

Lalu, langkah selanjutnya yaitu menghitung energi kinetik dan potensial *cart* dan pendulum.

- Energi kinetik pendulum dengan substitusi persamaan (1)

  $K_1 = \frac{1}{2}mV_1^2 + \frac{1}{2}I\omega^2$

  $K_1 = \frac{1}{2}m(\dot{x}^2 + 2\dot{x}l\dot{\theta}\ cos\theta + l^2\dot{\theta}^2) + \frac{1}{2}I\dot{\theta}^2$

  $K_1 = \frac{1}{2}(m\dot{x}^2 + 2m\dot{x}l\dot{\theta}\ cos\theta + ml^2\dot{\theta}^2) + \frac{1}{2}I\dot{\theta}^2$.......................... (2)

- Energi potensial pendulum

  $P_1 = mgh$

  $P_1 = mgy_1$

  $P_1 = mgl\ cos\theta$.. (3)

- Energi kinetik *cart*

  $K_2 = \frac{1}{2}M\dot{x}^2$.. (4)

- Energi potensial *cart*

  karena, *cart* hanya bergerak pada sumbu x (horizontal), maka

  $P_2 = 0$... (5)

Sehingga, dari persamaan (2/3/4/5) didapat persamaan *(langlarian (L))* yaitu:

$L = Energi\ Kinetik - Energi\ Potensial$

$L = (K_1 + K_2) - (P_1 + P_2)$

$L = \left[\frac{1}{2}\left(m\dot{x}^2 + 2m\dot{x}l\dot{\theta}\ cos\theta + ml^2\dot{\theta}^2\right) + \frac{1}{2}I\dot{\theta}^2 + \frac{1}{2}M\dot{x}^2\right] - [mgl\ cos\theta]$...... (6)

Dari persamaan (6) didapat persamaan *Euler-Langrangian*, yaitu:

$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} + d\dot{\theta} = 0$

$(l + ml^2)\ddot{\theta} + ml\ cos\theta\ \ddot{x} - mgl\ sin\theta + d\dot{\theta} = 0$................................. (7)

$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} + d\dot{x} = F$

$(M + m)\ddot{x} + ml\ cos\theta\ \ddot{\theta} - ml\ sin\theta\ \dot{\theta}^2 + b\dot{x} = F$................................ (8)

B. Linearisasi

Untuk linearisasi, kita asumsikan bahwa untuk θ yang bernilai kecil[1][2], maka:

$sin\theta = \theta$; $cos\theta = \theta$; dan $\dot{\theta}^2 = 0$

Sehingga, apabila asumsi tersebut kita substitusikan ke dalam persamaan (7) dan (8), maka persamaan (7) dan (8) menjadi:

$(M+m)\ddot{x} + ml\ddot{\theta} + b\dot{x} = F$ .... (9)

$(l+ml^2)\ddot{\theta} - mgl\theta = -ml\ddot{x}$ .... (10)

Apabila dipindah ruas, maka

$\ddot{x} = \frac{-ml}{M+m}\ddot{\theta} - \frac{b}{M+m}\dot{x} + \frac{1}{M+m}F$ .... (11)

$\ddot{\theta} = \frac{-ml}{l+ml^2}\ddot{x} + \frac{mgl}{l+ml^2}\theta$ .... (12)

Substitusi persamaan (11) ke (10) dan persamaan (12) ke (9), maka

$$\ddot{x} = \frac{-gm^2l^2}{I(M+m)+Mml^2}\theta - \frac{b(I+ml^2)}{I(M+m)+Mml^2}\dot{x} + \frac{I+ml^2}{I(M+m)+Mml^2}F$$

$$\ddot{\theta} = \frac{mgl(M+m)}{I(M+m)+Mml^2}\theta + \frac{mlb}{I(M+m)+Mml^2}\dot{x} - \frac{ml}{I(M+m)+Mml^2}F$$

Untuk mempermudah kita misalkan, $p = I(M+m) + Mml^2$

C. Pemodelan *State Space*

Misalkan,

$x_1 = \theta \rightarrow \dot{x}_1 = x_2$

$x_2 = \dot{\theta} \rightarrow \dot{x}_2 = \ddot{\theta} = \frac{mgl(M+m)}{p}x_1 + \frac{mlb}{p}x_4 - \frac{ml}{p}u$

$x_3 = x \rightarrow \dot{x}_3 = x_4$

$x_4 = \dot{x} \rightarrow \dot{x}_4 = \ddot{x} = \frac{-gm^2l^2}{p}x_1 - \frac{b(I+ml^2)}{p}x_4 + \frac{I+ml^2}{p}u$

Sehingga, bentuk matriks *state space* nya yaitu:

$$\dot{x} = \begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0\\ \frac{mgl(M+m)}{p} & 0 & 0 & \frac{mlb}{p}\\ 0 & 0 & 0 & 1\\ \frac{-gm^2l^2}{p} & 0 & 0 & -\frac{b(I+ml^2)}{p}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0\\ -\frac{ml}{p}\\ 0\\ \frac{I+ml^2}{p}\end{bmatrix}u$$

$$y = \begin{bmatrix}y_1\\ y_2\end{bmatrix} = \begin{bmatrix}0 & 0 & 1 & 0\\ 1 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0\\ 0\end{bmatrix}u$$

Source Code Matlab:

```
% Define State Space Matrix
A = [          0              1      0            0         ;
       (m*g*l*(M+m))/p        0      0        (m*l*b)/p     ;
               0              0      0            1         ;
     -g*(m^2)*(l^2)/p         0      0    -(b*(I+m*l^2))/p;]

B = [0 ; -(m*l)/p ; 0 ; (I+m*l^2)/p];
C = [0 0 1 0 ; 1 0 0 0];
D = [0 ; 0];
system = ss(A, B, C, D)
```

D. Performansi Sistem *Open Loop*

Untuk mengetahui kestabilan sistem *open loop* dapat menggunakan *eigenvalues* dari matriks A.

Source Code Matlab:

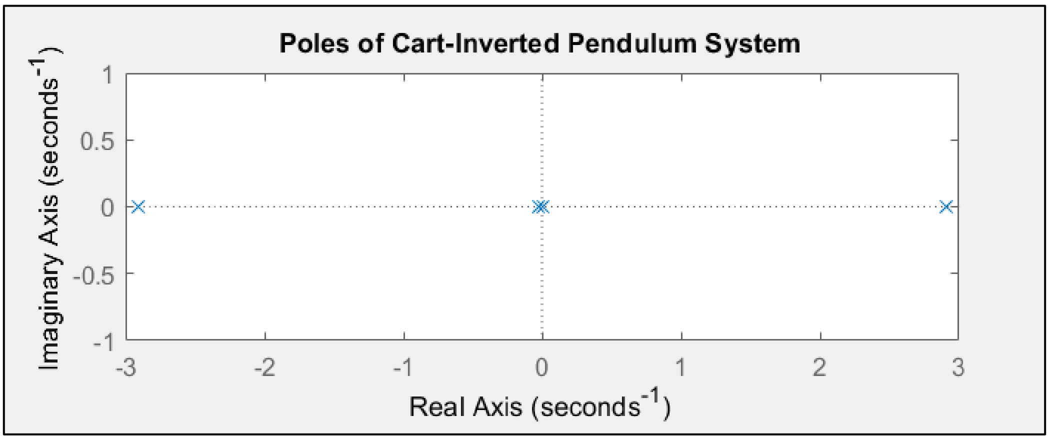
```
% Open-Loop Response
disp('Poles Sistem: ')
poles = eig(A)
subplot(2, 2, 3:4)
pzmap(system)
title('Poles of Cart-Inverted Pendulum System')
```

Hasil:

```
Poles Sistem:

poles =

         0
    2.9079
   -2.9095
   -0.0278
```



Analisa Hasil:

Berdasarkan nilai *poles* sistem, hasil yang didapat menunjukkan sistem *cart-inverted pendulum* tidak stabil, karena salah satu *pole* yaitu 2.9079 berada pada sisi kanan atau *right half plane* (dapat dilihat pada grafik). Sedangkan, syarat suatu sistem *open-loop* stabil yaitu ketika semua *poles* berada pada sisi kiri atau *left half plane*.

E. *Controllable* dan *Observeable*

Dalam suatu sistem, penting untuk mengetahui apakah sistem tersebut *controllable* agar dapat dikendalikan serta penting juga untuk mengetahui apakah sistem tersebut *observeable* agar kita dapat merancang *estimator* sistem.

Source Code Matlab:

```
% Controllability
Matrix_Co = ctrb(A,B)
Rank_Co   = rank(ctrb(A,B))
if Rank_Co > 0;
    fprintf('System is Controllable\n')
else
    fprintf('System is Uncontrollable\n')
end
```

```
% Observeability
Matrix_Ob = obsv(A,C)
Rank_Ob   = rank(obsv(A,C))
if Rank_Ob > 0;
    fprintf('System is Observeable\n')
else
    fprintf('System is Unobserveable\n')
end
```

Hasil:

```
Matrix_Co =

         0   -0.4796    0.0141   -4.0582
   -0.4796    0.0141   -4.0582    0.1257
         0    0.5875   -0.0173    0.2710
    0.5875   -0.0173    0.2710   -0.0159
```

```
Rank_Co =

     4

System is Controllable
```

```
Matrix_Ob =

         0         0    1.0000         0
    1.0000         0         0         0
         0         0         0    1.0000
         0    1.0000         0         0
   -0.5640         0         0   -0.0294
    8.4604         0         0    0.0240
    0.0166   -0.5640         0    0.0009
   -0.0135    8.4604         0   -0.0007
```

```
Rank_Ob =

     4

System is Observeable
```

Analisa Hasil:

Berdasarkan simulasi diatas, terlihat bahwa sistem tersebut *controllable* dan *observeable* yang dilihat dari rank matrik *controllability* dan *observeability*, dimana syaratnya yaitu rank matriks *co* dan *ob* harus *full rank* atau *rank(Mx) = n*, dimana *n* adalah jumlah *state*.

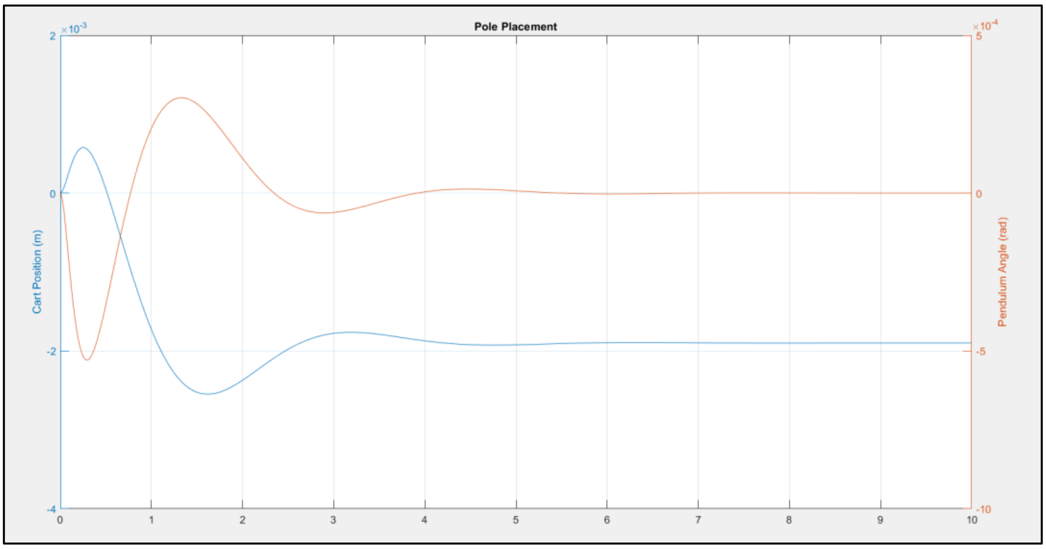
F. *State Feedback Control (Pole Placement)*

Untuk membuat sistem *cart-inverted pendulum* stabil, salah satu metode kendali yang dapat digunakan yaitu *pole placement* dengan mengatur dan menempatkan nilai pole atau eigen sesuai keinginan untuk mencapai kestabilan.

Source Code Matlab:

```
% Pole Placement
pp  =[-11 -9 -1+2j -1-2j]
Kpp = place(A,B,pp)
Acp = A-B*Kpp;
sys_pp = ss(Acp,B,C,D);
% Plot
t = 0:0.01:10;
r =0.2*ones(size(t));
figure(1);clf
[y,t,x]=lsim(sys_pp,r,t);
[AX,H1,H2] = plotyy(t,y(:,1),t,y(:,2),'plot');
set(get(AX(1),'Ylabel'),'String','Cart Position (m)')
set(get(AX(2),'Ylabel'),'String','Pendulum Angle (rad)')
title('Pole Placement')
grid
```

Hasil:



Analisa Hasil:

Berdasarkan simulasi dengan metode *pole placement*, dapat dilihat bahwa sistem mencapai keadaan stabil saat menuju t = 6s. Pemilihan dan penempatan *poles* sangat berpengaruh pada kestabilan sistem dalam segi waktu yang dibutuhkan sistem untuk mencapai kestabilan dan *overshoot* yang dialami sistem. Asumsinya jika *poles* diletakkan semakin ke kiri atau *left half plane* maka sistem akan lebih cepat untuk mencapai titik stabilnya.

G. *State Feedback Control (LQR)*

Selain menggunakan metode *Pole Placement*, metode kendali lainnya yang dapat digunakan yaitu *Linear Quadratic Regulator* (LQR) dengan mengatur nilai parameter Q dan R.

Source Code Matlab:

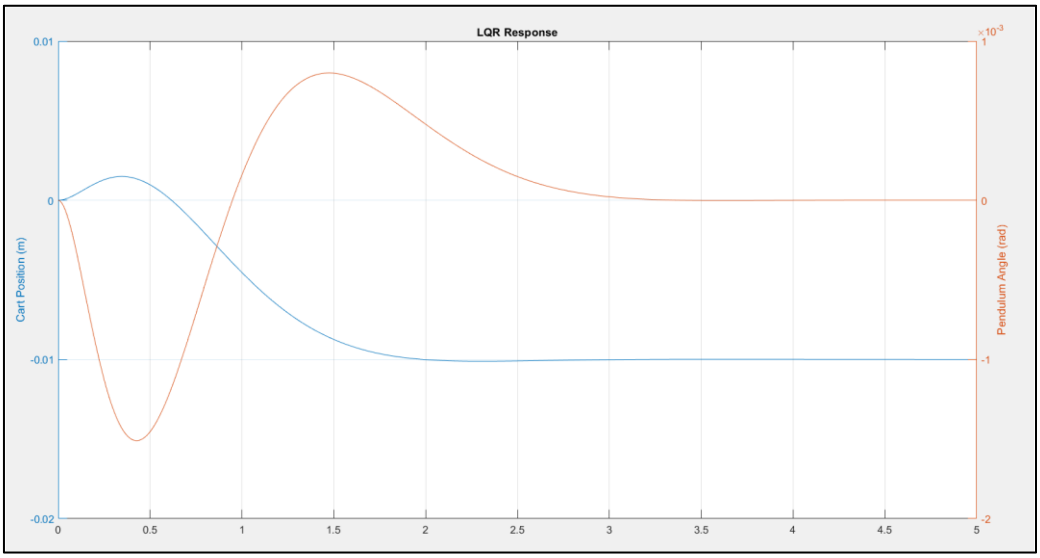
```
% Linear Quadratic Regulator (LQR)
% Define LQR Parameter
Q  = diag([40 40 400 0]);
R  = 1;
```

```
% Calculate LQR Gain
K  = lqr(A,B,Q,R);
Ac = A-B*K; %control matrix
sys_lqr1 = ss(Ac,B,C,D)
% Plot
t = 0:0.01:5;
r =0.2*ones(size(t));
figure(2);clf
[y,t,x]=lsim(sys_lqr1,r,t);
[AX,H1,H2] = plotyy(t,y(:,1),t,y(:,2),'plot');
set(get(AX(1),'Ylabel'),'String','Cart Position (m)')
set(get(AX(2),'Ylabel'),'String','Pendulum Angle (rad)')
title('LQR Response')
grid
```

Hasil:

```
K =

 -144.6983   -51.1613   -20.0000   -22.8230
```



Source Code Matlab LQR dengan *Kr:*

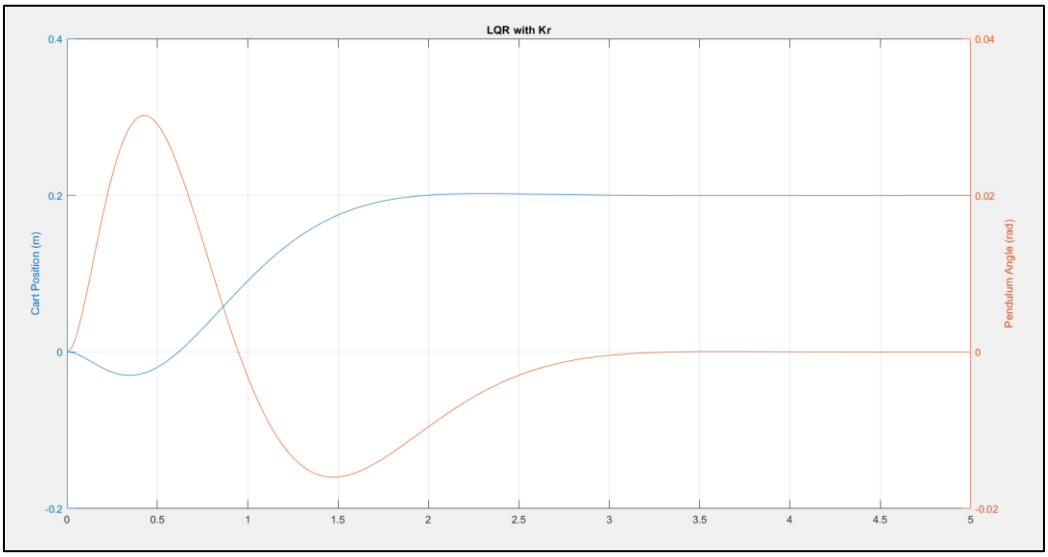
```
% LQR dengan Feedforward Gain (Kr)
Cn = [0 0 1 0];                    %modifikasi matriks C, karena input ref.
Kr = -inv(Cn*(inv(A-B*K))*B);      %menghitung Kr
sys_lqr2 = ss(Ac,B*Kr,C,D)
```

```
%Plot
t = 0:0.01:5;
r =0.2*ones(size(t));
figure(3);clf
[y,t,x]=lsim(sys_lqr2,r,t);
[AX,H1,H2] = plotyy(t,y(:,1),t,y(:,2),'plot');
set(get(AX(1),'Ylabel'),'String','Cart Position (m)')
set(get(AX(2),'Ylabel'),'String','Pendulum Angle (rad)')
title('LQR with Kr')
grid
```

Hasil:

```
Kr =

  -20.0000
```



Analisa Hasil:

Berbeda dengan metode *pole placement* yang mendapatkan respon *closed loop* dari pemilihan letak *K*, LQR memilih K dengan mempertimbangkan *input* untuk memperoleh performansi sistem. Dengan menggunakan LQR penting untuk mengetahui nilai Q dan R yang kita gunakan karena sangat berpengaruh pada performansi sistem. Pada simulasi diatas nilai Q yang digunakan adalah Q = diag[40 40 400 0] dan R = 1 dengan hasil sistem mencapai titik kestabilan saat t = ±3.5s. Kemudian untuk keperluan *tracking* disajikan juga simulasi dengan *gain compensator (Kr)* yang hasilnya dapat dilihat diatas.

H. *State Estimator (Observer)*

Untuk merancang sebuah estimator dalam suatu sistem, diperlukan ada perhitungan *observeability* sistem, dan kemudian penentuan L.

Source Code Matlab:

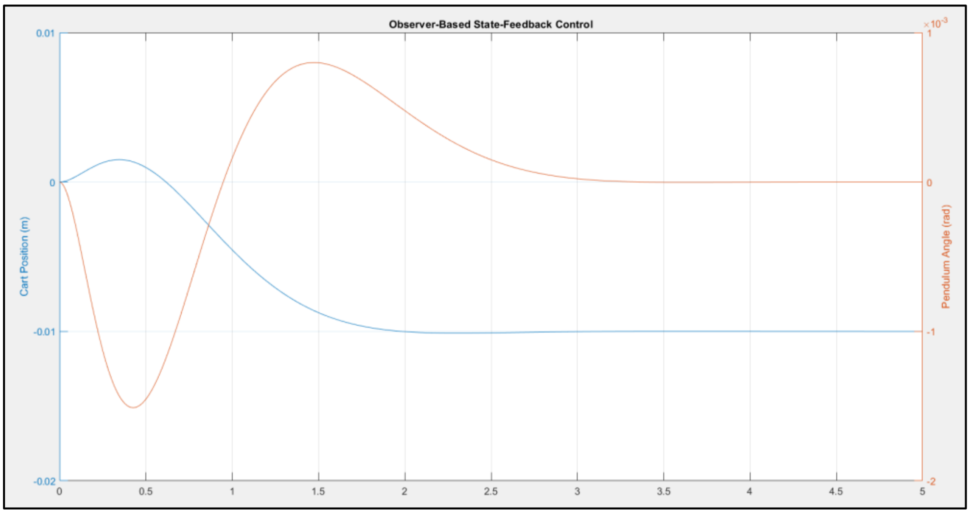
```
% State Estimator (Observer)
ob = obsv(sys_lqr1);
observability = rank(ob)
ctr_poles     = eig(Ac)
obsr_poles    = [-25 -26 -27 -28]
L  = place(A',C',obsr_poles)'
```

a. *Observer – Based State Feedback Control*

Source Code Matlab:

```
% Observer-Based State-Feedback Control
Aco = [(A-B*K) (B*K);zeros(size(A)) (A-L*C)];
Bco = [B;zeros(size(B))];
Cco = [C zeros(size(C))];
Dco = [0;0];
sys_ob = ss(Aco,Bco,Cco,Dco);
%Plot
t = 0:0.01:5;
r = 0.2*ones(size(t));
figure(4);clf
[y,t,x]=lsim(sys_ob,r,t);
[AX,H1,H2] = plotyy(t,y(:,1),t,y(:,2),'plot');
set(get(AX(1),'Ylabel'),'String','Cart Position (m)')
set(get(AX(2),'Ylabel'),'String','Pendulum Angle (rad)')
title('Observer-Based State-Feedback Control')
grid
```

Hasil:

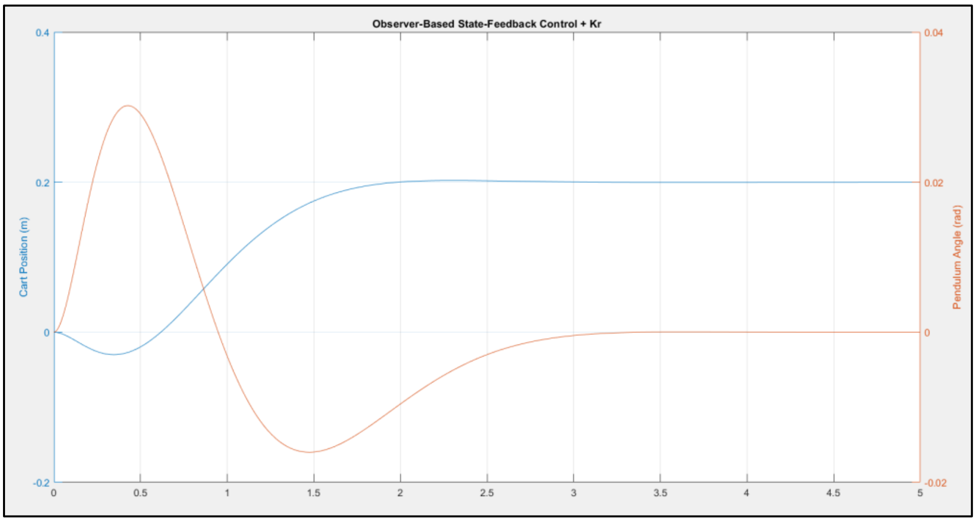


b. *Observer-Based State-Feedback Control + Kr*

Source Code Matlab:

```
% Observer-Based State-Feedback Control + Kr
Acl   = [(A-B*K)  (B*K);zeros(size(A))  (A-L*C)];
Bcl   = [B;zeros(size(B))];
Ccl   = [C zeros(size(C))];
Dcl   = [0;0];
Ccln = [Cn zeros(size(Cn))];
Kr2   = -inv(Ccln*(Acl\Bcl))
Bclt = [B*Kr2;zeros(size(B))];
sys_ob2 = ss(Acl,Bclt,Ccl,Dcl);
% Plot
t = 0:0.01:5;
r = 0.2*ones(size(t));
figure(5);clf
[y,t,x]=lsim(sys_ob2,r,t);
[AX,H1,H2] = plotyy(t,y(:,1),t,y(:,2),'plot');
set(get(AX(1),'Ylabel'),'String','Cart Position (m)')
set(get(AX(2),'Ylabel'),'String','Pendulum Angle (rad)')
title('Observer-Based State-Feedback Control + Kr')
grid
```

Hasil:



Analisa Hasil *State Estimator*:

Sebelum merancang sebuah estimator, kita harus mengetahui terlebih dahulu apakah sistem tersebut *observeable* dengan syarat *full rank matrix ob.* Pemilihan *obsv_poles* pada simulasi diatas, didapat dari hasil *eig(A)* kemudian dikalikan sepuluh kali lebih cepat dari sistem. Berdasarkan dari hasil simulasi yang didapat, respon sistem dengan estimator identik dengan sistem tanpa estimator (semua variabel state diketahui). Hal ini disebabkan karena *observeable poles* jauh lebih cepat dari sistem awal, dan juga rancangan sistem estimator identik atau sama dengan sistem yang sebenarnya, termasuk kondisi awal sehingga tidak banyak memerlukan *effort* untuk menyesuaikan dengan sistem referensi/awal.

I. Link Video

Berikut adalah tautan video penjelasan.

https://drive.google.com/file/d/1rqEdCocBsj4bRhSHYcTapGEW4bW-TvGG/view?usp=sharing atau https://tinyurl.com/tubesharis

J. Referensi

[1] P. Kumar, "Controller Design of Inverted Pendulum Using Pole Placement and Lqr," *Int. J. Res. Eng. Technol.*, vol. 01, no. 04, pp. 532–538, 2012, doi: 10.15623/ijret.2012.0104003.

[2] Michigan University, "Inverted Pendulum: State-Space Methods for Controller Design," 2019. https://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum&section=ControlStateSpace.