ESTIMASI PADA SISTEM ROTATIONAL/TRANSLATIONAL ACTUATOR MENGGUNAKAN NON-RECURSIVE LEAST SQUARE METHOD

LAPORAN TUGAS AKHIR
IDENTIFIKASI SISTEM
(TKEE163232)

Disusun Oleh:

Resha Dwika Hefni Al-Fahsi
16/394959/TK/44251

PROGRAM STUDI TEKNIK ELEKTRO
DEPARTEMEN TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI
FAKULTAS TEKNIK UNIVERSITAS GADJAH MADA
2019

Abstrak

Laporan tugas akhir ini membahas tentang estimasi pada sistem *rotational/translational actuator*. Kinematis dari sistem tersebut dirumuskan dari model non-linearnya. Kemudian untuk mempermudah dalam analisis kendali digital dilakukan linearisasi dan diskretisasi sistem sehingga dapat dicari *state space*-nya. Dilakukan juga analisis *controllable* dan *observable* pada sistem tersebut. Menggunakan *Non-Recursive Least Square Method* untuk mencari estimasi sistem yang tepat dengan mengamati respon sistem terhadap AWGN (*Additive White Gaussian Noise*). Semua simulasi perhitungan pada sistem dilakukan menggunakan Matlab® [1] R2016a.

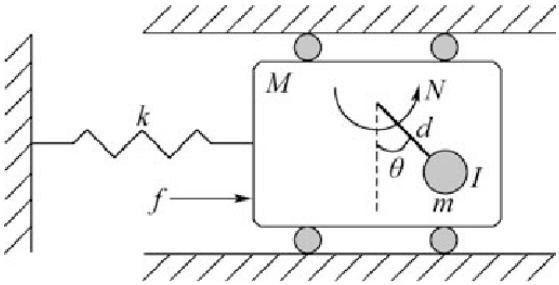
[1]Matlab and Simulink are trademarks of MathWorks Inc. USA.

Daftar Isi

Bab 1

Pendahuluan

Tujuan dari pembuatan laporan ini untuk memberikan penjelasan tentang perancangan *robust H_∞ control* pada sistem *rotational/translational actuator*. Sistem *rotational/translational actuator* atau biasa disingkat RTAC merupakan sebuah sistem berupa gabungan dua aktuator yang salah satunya bergerak secara rotasi dan aktuator lainnya bergerak secara translasi[2]. Dalam praktiknya aktuator tersebut berupa sistem osilasi yaitu pendulum yang berperan dalam gerakan rotasi dan pegas yang berperan dalam gerakan translasi. Sistem RTAC sendiri sering digambarkan dengan sebuah gerobak yang terpasang sistem pegas yang dipasang secara tegak lurus pada dinding dan di dalam gerobak tersebut terdapat pendulum yang berosilasi (bergerak secara rotasi) sehingga akan menyebabkan gerakan translasi pada gerobak[3].



Gambar 1. Pemodelan Sistem RTAC

Pada bab 2, kita mendefinisikan sistem RTAC dalam *state-space*. Sistem RTAC sendiri pada dasarnya merupakan sistem non-linear sehingga diperlukan proses linearisasi untuk mendapatkan bentuk *state-space*-nya.
Pada bab 3 dilakukan proses diskretisasi pada sistem sehingga dapat dilakukan analisis kendali digital untuk mendapatkan *state-space* dalam ranah diskret. Dengan mengetahui bentuk *state-space* tersebut dapat diperiksa apakah sistem tersebut *observable* dan *controllable.*
Pada bab 4 dilakukan estimasi sistem tersebut menggunakan *Non-Recursive Least Square Method* dengan melihat respon sistem terhadap AWGN (*Additive White Gaussian Noise).*

Bab 2

Sistem *Rotational/Translational Actuator*

Seperti yang dijelaskan sebelumnya sistem RTAC merupakan sistem non-linear. Sistem non-linear sendiri memiliki *state* yang *dependent* dengan *state* lainnya. Pada sistem yang akan dibahas, terdapat 4 *state* yang didefiniskan pada *state-space*. Untuk mendapatkan *state-space* sistem tersebut dilakukan linearisasi dengan metode linearisasi Jacobian.

2.1 Pemodelan

Pemodelan sistem RTAC didefinisikan dengan *state-space* non-linearnya[4]:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\end{bmatrix} = \begin{bmatrix} x_2 \\ \dfrac{(I+mr^2)(mrx_4^2\sin x_3 - kx_1) - (mr\cos x_3)N + (I+mr^2)F}{(M+m)(I+mr^2) - m^2r^2\cos^2 x_3} \\ x_4 \\ \dfrac{mr\cos x_3(kx_1 - mrx_4^2\sin x_3 - F) + (M+m)N}{(M+m)(I+mr^2) - m^2r^2\cos^2 x_3} \end{bmatrix}$$

Di mana:

x_1= perpindahan gerobak dari titik setimbang, m
x_2= kecepatan linear gerobak, m/s
x_3= posisi sudut pendulum, rad
x_4= kecepatan sudut pendulum, rad/s
M = massa gerobak, 1.3608 kg
m = massa pendulum, 0.096 kg
r = panjang tali pendulum, 0.0592 m
k = konstanta/kekakuan pegas, 186.3 N/m
F = gaya translasi, gaya gangguan, 5 N
N = torsi pada pendulum, kontrol pada sistem, 0.41861 Nm
I = inersia pendulum, 0.0002175 kg/m^2

2.2. Linearisasi

Linearisasi Jacobian sendiri menggunakan turunan parsial setiap *state* terhadap suatu fungsi yang mendefinisikan salah satu turunan *state* pada sistem. Dengan linearisasi Jacobian menggunakan *operating point*, $[x_1 \quad x_2 \quad x_3 \quad x_4] = \left[0 \quad 0 \quad \frac{\pi}{4} \quad 0.785\right]$ dan $[F \quad N] = [0 \quad 0]$, didapatkan matriks-matriks *state-space*-nya:

Untuk matriks A-nya:

$$A = \begin{bmatrix} x_2 \\ \dfrac{(I+mr^2)(mrx_4^2\sin x_3 - kx_1) - (mr\cos x_3)N + (I+mr^2)F}{(M+m)(I+mr^2) - m^2r^2\cos^2 x_3} \\ x_4 \\ \dfrac{mr\cos x_3(kx_1 - mrx_4^2\sin x_3 - F) + (M+m)N}{(M+m)(I+mr^2) - m^2r^2\cos^2 x_3} \end{bmatrix} \begin{bmatrix}\partial x_1\\ \partial x_2\\ \partial x_3\\ \partial x_4\end{bmatrix}^T$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\dfrac{k(I+mr^2)}{(M+m)(I+(0.5mr^2))} & 0 & -\dfrac{m^2r^3(0.555((M+m)I+(Mmr^2))+m(0.616I+0.555mr^2))}{((M+m)(I+(0.5mr^2)))^2} & \dfrac{1.57\,Imr+1.11\,m^2r^3}{(M+m)(I+(0.5mr^2))} \\ 0 & 0 & 0 & 1 \\ \dfrac{0.707\,mrk}{(M+m)(I+(0.5mr^2))} & 0 & 0 & -\dfrac{0.785\,m^2r^2}{(M+m)(I+(0.5mr^2))} \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -183.66 & 0 & -0.00279 & 0.00723 \\ 0 & 0 & 0 & 1 \\ 1332.1 & 0 & 0 & -0.04512 \end{bmatrix}$$

Sedangkan untuk matriks B-nya:

$$B = \begin{bmatrix} x_2 \\ \dfrac{(I+mr^2)(mrx_4^2\sin x_3 - kx_1) - (mr\cos x_3)N + (I+mr^2)F}{(M+m)(I+mr^2) - m^2r^2\cos^2 x_3} \\ x_4 \\ \dfrac{mr\cos x_3(kx_1 - mrx_4^2\sin x_3 - F) + (M+m)N}{(M+m)(I+mr^2) - m^2r^2\cos^2 x_3} \end{bmatrix} \begin{bmatrix} \partial F \\ \partial N \end{bmatrix}^T$$

$$B = \begin{bmatrix} 0 & 0 \\ \dfrac{(I+mr^2)}{(M+m)(I+(0.5mr^2))} & -\dfrac{0.707\,mr}{(M+m)(I+(0.5mr^2))} \\ 0 & 0 \\ -\dfrac{0.707\,mr}{(M+m)(I+(0.5mr^2))} & \dfrac{M+m}{(M+m)(I+(0.5mr^2))} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0.98581 & -7.1505 \\ 0 & 0 \\ -7.1505 & 2592.5 \end{bmatrix}$$

Dengan matriks $C = \mathbf{I_{4\times4}}$ dan $D = \mathbf{0_{4\times2}}$. Sehingga bentuk *state-space*, $\dot{x} = Ax + Bu$ dan $\dot{x} = Cx + Du$, dari sistem tersebut telah terbentuk.

Bab 3

Sistem Digital

Pada praktiknya, sistem yang ada di dunia ini dikendalikan menggunakan komputer. Komputer sendiri merupakan suatu sistem yang bekerja pada ranah diskret. Sedangkan sistem yang ada di alam berbentuk kontinyu. Sehingga perlu dilakukan proses diskretisasi yaitu mengubah sistem pada ranah analog menjadi ranah digital.

3.1. Diskretisasi

Pemodelan sistem pada bab sebelumnya dilakukan pada ranah kontinyu waktu. Sehingga sistem tersebut perlu diubah ke bentuk sistem dalam waktu diskret. Berikut persamaan *state-space* waktu diskret[1].

$$\mathbf{A_d} = \boldsymbol{\Phi} = \mathcal{L}^{-1}[(s\boldsymbol{I} - \boldsymbol{A})^{-1}]$$

$$\boldsymbol{B_d} = \int_0^T \boldsymbol{\Phi}(\lambda)\, d\lambda\, \boldsymbol{B}$$

$$\boldsymbol{C_d} = \boldsymbol{C}$$

$$\boldsymbol{D_d} = \boldsymbol{D}$$

Persamaan di atas akan mengubah model *state-space* dalam ranah kontinyu waktu menjadi diskret.

3.2. *Contrabillity* dan *Observability*

Dengan mengetahui bentuk lengkap dari *state-space* dapat ditentukan *observability* dan *controllability* dari sistem tersebut. Untuk mencari *observability* dari suatu sistem digunakan matriks $O = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$ dan mencari *observability* dari suatu sistem digunakan matriks $C = [B \quad AB \quad \cdots \quad A^{m-1}B]$. Syarat sistem *observable* ketika matriks O merupakan matriks *full-column rank* dan sistem *controllable* ketika matriks C merupakan matriks *full-row rank.*

3.3. Program Matlab®

Dengan menggunakan Matlab® dapat dicari:

```
%Diskretisasi Sistem
fs = 250; %Hz
Ts = 1/fs; % periode sampling (sekon)
sys_ss = ss(A,B,C,D); % representasi state space
sys_ss_d = c2d(sys_ss, Ts)
[Ad, Bd, Cd, Dd] = ssdata(sys_ss_d)

Ob=obsv(Ad,Cd);
[n,m]=size(Ob);
unob = m-rank(Ob); %observable
if(unob==0)
   disp('Given System is Observable.');
else
   disp('Given System is Unobservable');
end

Co = ctrb(Ad,Bd);
[n,m]=size(Co);
unco=n-rank(Co); %controllable
if(unco==0)
   disp('Given System is Controllable.');
else
   disp('Given System is Uncontrollable');
end
```

Dari hasil simulasi didapatkan *state-space* waktu diskret dengan frekuensi cuplik 250 Hz didapatkan.

$$A_d = \begin{bmatrix} 0.9985 & 0.004 & 0 & 0 \\ -0.7342 & 0 & 0 & 0 \\ 0.0107 & 0 & 1 & 0.004 \\ 5.3253 & 0.0107 & 0 & 0.9998 \end{bmatrix}$$

$$B_d = \begin{bmatrix} 0 & -0.0001 \\ 0.0039 & -0.0284 \\ -0.0001 & 0.0207 \\ -0.0286 & 10.369 \end{bmatrix}$$

Dengan matriks $C_d = \mathbf{I}_{4\times4}$ dan $D_d = \mathbf{0}_{4\times2}$. *State-space* waktu diskret tersebut merupakan sistem yang *controllable* dan *observable*.

Bab 4

Estimasi Sistem

Sistem yang ada di dunia ini memiliki sifat yang unik, beragam dan kadang sulit untuk diketahui modelnya. Apabila sistem ingin diketahui modelnya dapat dicari persamaan matematisnya. Terkadang sistem yang begitu kompleks persamaan matematis bukanlah solusi yang optimal. Sistem dapat diketahui sifatnya salah satunya dengan menganalisis respon sistem tersebut terhadap tanggapan acak yang diberikan pada sistem. Sehingga dapat dianalisis bentuk tanggapannya dan diestimasi dengan metode-metode estimasi sistem salah satunya adalah *Non-Recursive Least Square Method.*

4.1. *Non-Recursive Least Square Method*

Parameter *state-space* dapat diestimasi dari data runtun input dan output. Misal runtun output selama N cuplikan adalah:

$$y[k-N+1] = CAx[k-N] + CBu[k-N]$$

Agar dapat dilakukan estimasi, diasumsikan seluruh state sistem terukur untuk seluruh cuplikan. Karena itu, matriks C yang menunjukkan hubungan antara state dengan output yang dapat diukur adalah sebuah matriks identitas berukuran n×n, di mana n adalah jumlah state. Persamaan di atas dapat ditulis ulang menjadi:

$$Y[k] = \psi\hat{\theta}[k] + E[k]$$

di mana

$$Y[k] = \begin{bmatrix} y^T[k-N+1] \\ \vdots \\ y^T[k] \end{bmatrix}$$

berisi nilai variable output yang terukur selama N cuplikan.

$$\psi^T[k] = [\psi[k-N+1] \quad \cdots \quad \psi[k]] \\ \begin{bmatrix} x[k-N] & \cdots & x[k-1] \\ u[k-N] & \cdots & u[k-1] \end{bmatrix}$$

adalah matriks data identifikasi yang berisi input dan state sistem, sedangkan

$$\hat{\theta}[k] = \begin{bmatrix} \hat{A}^T C^T \\ \hat{B}^T C^T \end{bmatrix}$$

berisi nilai parameter yang diestimasi hingga cuplikan terakhir. $E[k]$ adalah *error* estimasi hingga cuplikan terakhir.

Non-Recursive Least Square Method didefinisikan,

$$\hat{\theta}[k] = \left[\psi^T[k]\psi[k]\right]^{-1}\psi^T[k]Y[k]$$

dengan meminimumkan jumlah *error*

$$V_{Ls} = \sum_{i=k-N+1}^{k} e^T[i]e[i]$$

Non-Recursive Least Square Method diterapkan pada sistem yang telah didiskretkan. Sistem diberikan input AWGN (Additive White Gaussian Noise) agar seluruh komponen frekuensi dapat tereksitasi dan keluar pada output, kemudian runtun input dan output direkam selama 10 detik.

4.2. Program Matlab®

Dengan menggunakan Matlab® dapat dicari:

```
Tf = 10;
t = 0:Ts:Tf;
u = awgn(heaviside(t), 0.1); %Input Additive White Gaussian Noise (Derau
Putih)
x0 = [0; 0; 0; 0]; %Nilai awal state
[y,t,x]=lsim(sys_ss_d,[u;u],t,x0);

%Non-Recursive Least Square
sy=size(y); %ukuran runtun output
sx=size(x); %ukuran runtun state
uT = transpose([u; u]); %ditranspose agar menjadi vektor kolom
su=size(uT); %ukuran runtun input,

%Bacaan Output y[k+1]
for i=1:(sy(1)-1)
    for j=1:sy(2)
     Y(i,j)=y(i+1,j);
    end
end

%Bacaan State x[k]
for i=1:(sx(1)-1)
    for j=1:sx(2)
        psiX(i,j) = x(i,j);
    end
end

%Bacaan Input u[k]
for i=1:(su(1)-1)
    for j=1:su(2)
        psiU(i,j) = uT(i,j);
    end
end
```

```
%Identification Data Matrix Psi dengan menggabungkan State x[k] dan u[k]
psi = horzcat(psiX, psiU);
%Gabungan transpose matriks A dan B hasil estimasi ThetaHat
thetaHat = inv(transpose(psi)*psi)*transpose(psi)*Y

%Mendapatkan matriks A transpose
sth = size(thetaHat);
for(i=1:(sth(1)-su(2)))
    for(j=1:sth(2))
       At(i,j)=thetaHat(i,j);
    end
end
%Mendapatkan matriks B transpose
for(i=(sth(1)-su(2)+1):sth(1))
    for(j=1:sth(2))
       Bt(i-(sth(1)-su(2)),j)=thetaHat(i,j);
    end
end

%Hasil Estimasi State Space
Aest = transpose(At)
Best = transpose(Bt)
Cest = eye(sy(2),sx(2))
Dest = zeros(sy(2),su(2))
sysEst = ss(Aest, Best, Cest, Dest)
```

Dari hasil simulasi didapatkan *state-space* waktu diskret dengan frekuensi cuplik 250 Hz didapatkan.

$$A_{est} = \begin{bmatrix} 0.9985 & 0.003998 & -2.232 \times 10^{-8} & 5.78 \times 10^{-8} \\ -0.7342 & 0.9985 & -1.115 \times 10^{-5} & 2.888 \times 10^{-5} \\ 0.01065 & 1.421 \times 10^{-5} & 1 & 0.004 \\ 5.325 & 0.01065 & -3.964 \times 10^{-8} & 0.9998 \end{bmatrix}$$

$$B_{est} = \begin{bmatrix} 0.08916 & 0.05273 \\ 0.01542 & 0.007552 \\ -1.251 \times 10^{4} & -8865 \\ -2683 & -2195 \end{bmatrix}$$

Dengan matriks $C_d = \mathbf{I_{4\times4}}$ dan $D_d = \mathbf{0_{4\times2}}$. Didapatkan estimasi sistem yang hampir sama dengan sistem aslinya, namun untuk matriks B_{est}, matriks estimasi dari matriks B, didapatkan nilai yang berbeda dengan aslinya hanya salah satu input saja yang terestimasi dan nilai dari estimasi tidak sama bahkan jauh berbeda. Hal tersebut disebabkan karena proses estimasi yang acak tergantung pada AWGN.

Daftar Pustaka

[1] Ogata, K. *Discrete Time Control Systems*. Upper Saddle River, NJ: Prentice Hall, 1995.

[2] Tavakoli, Mahdi, Hamid D. Taghirad, and Mehdi Abrishamchian. "Identification and robust H∞ control of the rotational/translational actuator system." *International Journal of Control, Automation, and Systems* 3.3 (2005): 387-396.

[3] Adlgostar, R., H. Azimian, and H. D. Taghirad. "Robust H∞, H 2/H∞ controller for rotational/translational actuator (RTAC)." *Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE*. IEEE, 2006.

[4] Rosales, Andres, et al. "Controller designed by means of numeric methods for a benchmark problem: RTAC (Rotational Translational Actuator)." *Electronics, Robotics and Automotive Mechanics Conference, 2006*. Vol. 1. IEEE, 2006.

Appendix A

Lampiran Program

Berikut lampiran program yang digunakan untuk simulasi sistem RTAC ini. Program dibuat menggunakan Matlab® dan tersedia secara online di https://github.com/reshalfahsi/rtacsystem.

```
clear all;
close all;

M = 1.3608;
m = 0.096;
R = 0.0592;
I = 2.175*10^-4;
k = 186.3;

%pembagi di persamaan
h = (M+m)*(I+0.5*m*R^2);

%nilai di matriks A
a11 = 0;
a12 = 1;
a13 = 0;
a14 = 0;
a21 = (-k*(I+m*R^2))/h;
a22 = 0;
a23 = -
(((m^2)*(R^3)*(0.555*((M+m)*I+(M*m*R^2))+m*(0.616*I+0.555*m*R^2)))/(h^2));
a24 = (1.57*I*m*R+1.11*(m^2)*(R^3))/h;

a31 = 0;
a32 = 0;
a33 = 0;
a34 = 1;

a41 = (0.707*m*R*k)/h;
a42 = 0;
a43 = 0;
a44 = -((0.785*(m^2)*(R^2))/h);

%nilai di matriks B
b11 = 0;
b12 = 0;
b21 = (I+m*R^2)/h;
b22 = (-0.707*m*R/h);
b31 = 0;
b32 = 0;
b41 = (-0.707*m*R/h);
b42 = (M+m)/h;
```

```
%matriks A,B,C, dan D
A = [a11 a12 a13 a14; a21 a22 a23 a24; a31 a32 a33 a34; a41 a42 a43 a44];
B = [0 0; b21 b22; 0 0; b41 b42];
C = eye(4);
D = zeros(4,2);

sys_ss = ss(A,B,C,D); % representasi state space

%Diskretisasi Sistem
fs = 250; %Hz
Ts = 1/fs; % periode sampling (sekon)

sys_ss_d = c2d(sys_ss, Ts)
[Ad, Bd, Cd, Dd] = ssdata(sys_ss_d)
Ob=obsv(Ad,Cd);
[n,m]=size(Ob);
unob = m-rank(Ob); %observable
if(unob==0)
   disp('Given System is Observable.');
else
   disp('Given System is Unobservable');
end

Co = ctrb(Ad,Bd);
[n,m]=size(Co);
unco=n-rank(Co); %controllable
if(unco==0)
   disp('Given System is Controllable.');
else
   disp('Given System is Uncontrollable');
end

Tf = 10;
t = 0:Ts:Tf;
u = awgn(heaviside(t), 0.1); %Input Additive White Gaussian Noise (Derau
Putih)
x0 = [0; 0; 0; 0]; %Nilai awal state
[y,t,x]=lsim(sys_ss_d,[u;u],t,x0);

%Non-Recursive Least Square
sy=size(y); %ukuran runtun output
sx=size(x); %ukuran runtun state
uT = transpose([u; u]); %ditranspose agar menjadi vektor kolom
su=size(uT); %ukuran runtun input,

%Bacaan Output y[k+1]
for i=1:(sy(1)-1)
    for j=1:sy(2)
     Y(i,j)=y(i+1,j);
    end
end
```

```
%Bacaan State x[k]
for i=1:(sx(1)-1)
    for j=1:sx(2)
        psiX(i,j) = x(i,j);
    end
end

%Bacaan Input u[k]
for i=1:(su(1)-1)
    for j=1:su(2)
        psiU(i,j) = uT(i,j);
    end
end

%Identification Data Matrix Psi dengan menggabungkan State x[k] dan u[k]
psi = horzcat(psiX, psiU);
%Gabungan transpose matriks A dan B hasil estimasi ThetaHat
thetaHat = inv(transpose(psi)*psi)*transpose(psi)*Y

%Mendapatkan matriks A transpose
sth = size(thetaHat);
for(i=1:(sth(1)-su(2)))
    for(j=1:sth(2))
       At(i,j)=thetaHat(i,j);
    end
end
%Mendapatkan matriks B transpose
for(i=(sth(1)-su(2)+1):sth(1))
    for(j=1:sth(2))
       Bt(i-(sth(1)-su(2)),j)=thetaHat(i,j);
    end
end

%Hasil Estimasi State Space
Aest = transpose(At)
Best = transpose(Bt)
Cest = eye(sy(2),sx(2))
Dest = zeros(sy(2),su(2))
sysEst = ss(Aest, Best, Cest, Dest)
```