

Kendali Kecepatan dan Posisi Motor DC

Priyova M. Rafief¹, Karunia Dini F.¹, Octsana Dhiyaa W.¹, Bodhi Setiawan¹

Universitas Gadjah Mada¹

Praktikum Sistem Kendali Lanjut



UNIVERSITAS
GADJAH MADA

Anggota Kelompok 1

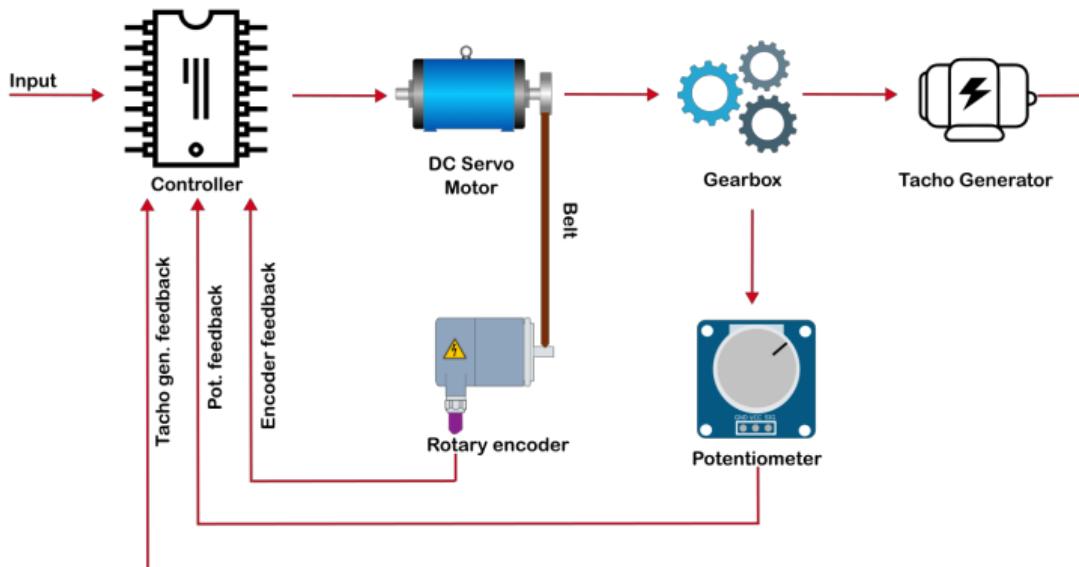
- ① Priyoya M. Rafief (20/457197/SV/17644)
- ② Karunia Dini F. (20/464248/SV/18567)
- ③ Octsana Dhiyaa W. (20/464253/SV/18572)
- ④ Bodhi Setiawan (20/464239/SV/18558)

Isi Pembahasan

- 1 Gambaran Umum Proyek Sistem Kendali Motor DC
- 2 Pemodelan Sistem Motor DC
- 3 Perancangan Kendali PID Motor DC
- 4 Simulasi Kendali PID
- 5 Simulasi LQR
- 6 Sistem Mekanikal dan Elektrikal
- 7 Desain GUI

Gambaran Umum Proyek Sistem Kendali Motor DC

Diagram Sistem



Pemodelan Sistem Motor DC

Pengertian Motor DC

Motor DC atau dalam bahasa Indonesia disebut motor arus searah adalah jenis motor listrik yang mengubah energi listrik arus searah menjadi energi mekanis. Bentuk energi yang dihasilkan berupa putaran. Prinsip kerja motor arus searah berdasarkan pada interaksi antara dua fluks magnetik yang disebut dengan kumparan medan dan kumparan jangkar. Kumparan medan menghasilkan fluks magnet dengan arah dari kutub utara ke kutub selatan, sedangkan kumparan jangkar menghasilkan fluks magnetik yang melingkar.

Sumber: Wikipedia

Ilustrasi Motor DC:

Struktur Fisik

Parameter:

- J : Momen inersia rotor ($Kg.m^2$)
- b : Koefisien gaya gesek viskos ($N.m.s$)
- Ke : Koefisien gaya elektromotif ($V/rad/sec$)
- Kt : Koefisien torsi motor ($N.m/Amp$)
- R : Resistansi kumparan (Ohm)
- L : Induktansi kumparan (H)

Struktur:

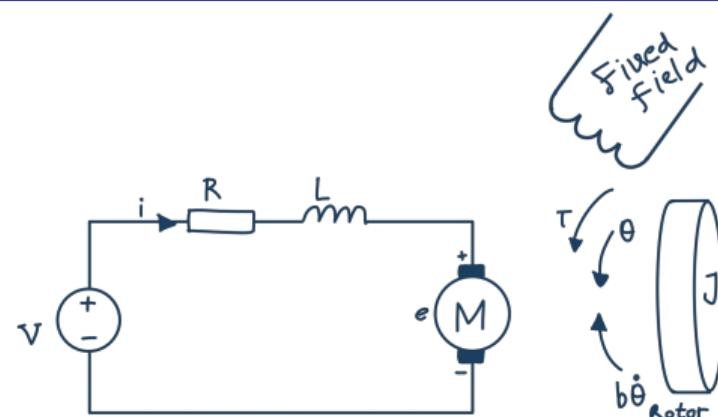
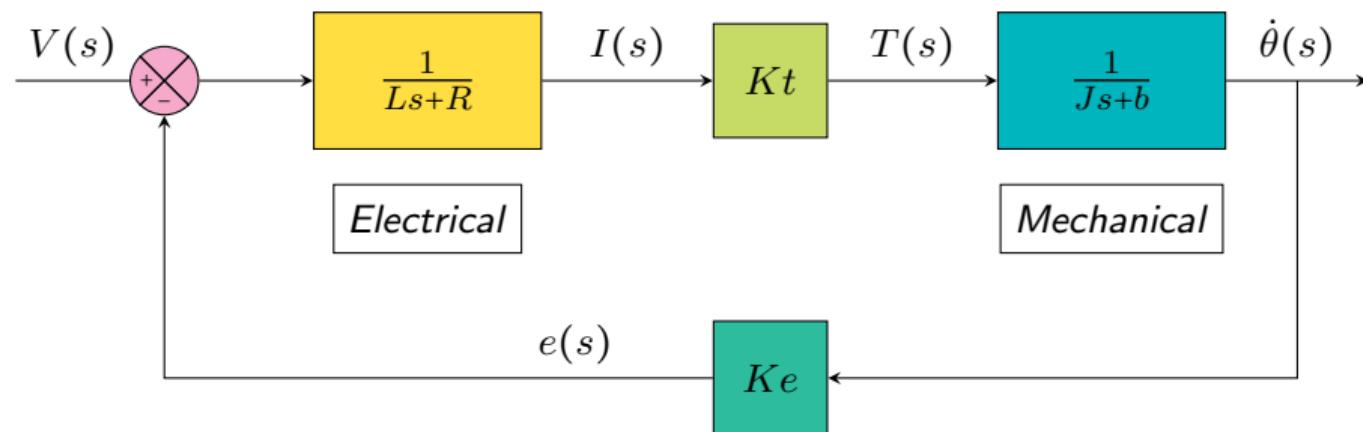


Diagram Blok Plant Motor DC

Struktur motor DC dengan parameter-parameter sebelumnya memiliki diagram blok sebagai berikut:



Fungsi Alih

Diagram blok *plant* motor DC menghasilkan persamaan fungsi alih berikut:

$$\frac{\dot{\theta}(s)}{V(s)} = \frac{Kt}{(Js + b)(Ls + R) + KtKe} \quad \left[\frac{\text{rad/sec}}{V} \right] \quad (1)$$

Persamaan di atas merupakan fungsi alih kecepatan motor DC. Dengan mengintegralkan fungsi alih tersebut, maka diperoleh fungsi alih untuk posisi motor DC:

$$\frac{\theta(s)}{V(s)} = \frac{Kt}{s((Js + b)(Ls + R) + KtKe)} \quad \left[\frac{\text{rad}}{V} \right] \quad (2)$$

State Space

Kecepatan:

$$\frac{\delta}{\delta t} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} -\frac{b}{J} & \frac{Kt}{J} \\ -\frac{Ke}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} V$$

$$y = [1 \quad 0] \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} \quad (3)$$

Struktur:

$$\frac{\delta}{\delta t} \begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{Kt}{J} \\ 0 & -\frac{Ke}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} V$$

$$y = [1 \quad 0 \quad 0] \begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix} \quad (4)$$

Perancangan Kendali PID Motor DC

Apa Itu Kendali PID?

- **PID=Proportional-Integral-Derivative**
- Kendali mekanisme umpan balik yang biasanya dipakai pada sistem kontrol industri
- Secara kontinu menghitung nilai kesalahan sebagai beda antara setpoint yang diinginkan dan variabel proses terukur. Persamaan:

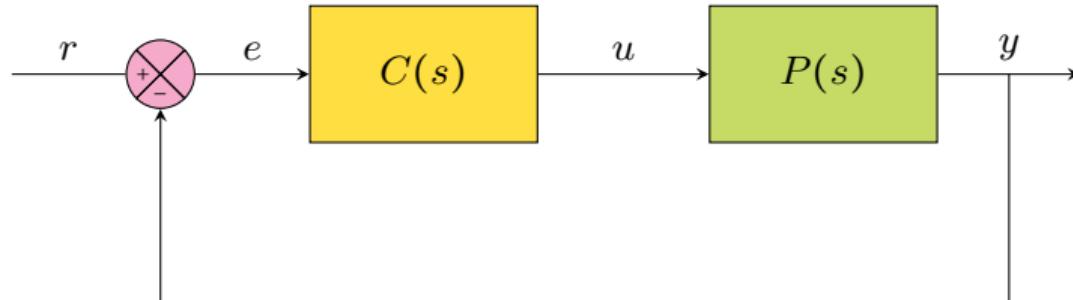
$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \quad (5)$$

Mengapa Kendali PID?

Kendali PID berfungsi untuk meminimalkan nilai kesalahan (*error*) setiap waktu dengan penyetelan variabel kontrol, seperti posisi, kecepatan, damper, daya, dan lain sebagainya.

Contoh perbandingan sistem dengan dan tanpa PID:

Diagram Blok Kendali



Keterangan:

$C(s)$: Controller

$P(s)$: Plant

$r(s)$: Output yang diinginkan

$e(s)$: Nilai error

$u(s)$: Sinyal kendali

$y(s)$: Output sesungguhnya

Diagram Blok Kendali PID Motor DC: Kecepatan

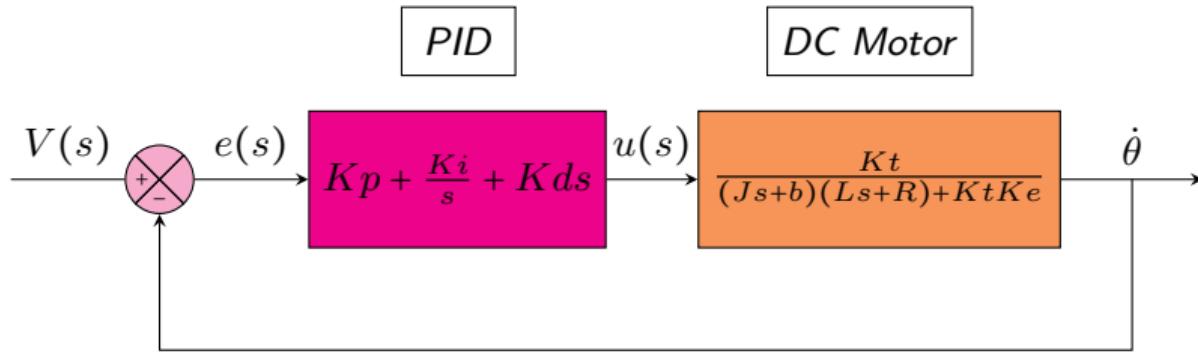
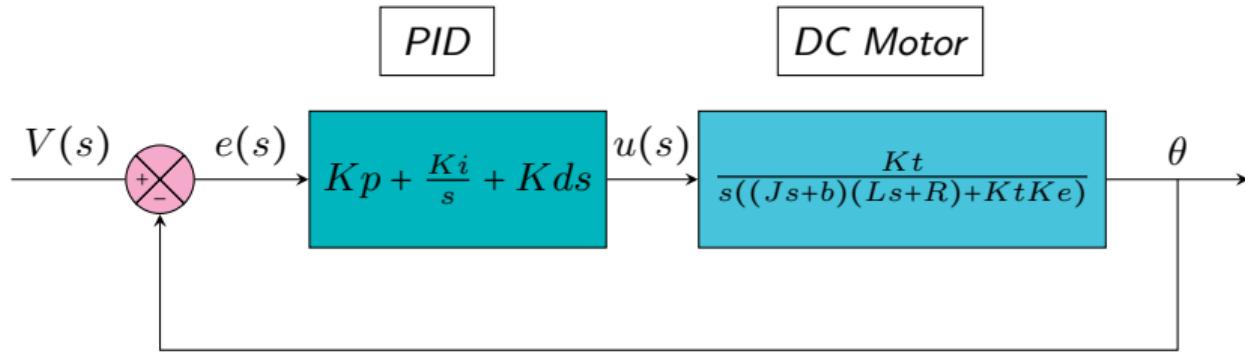


Diagram Blok Kendali PID Motor DC: Posisi



Simulasi Kendali PID

Uji Perbandingan Sistem *Open-loop* dengan *Closed-loop* Motor DC

Program *Open-loop*:

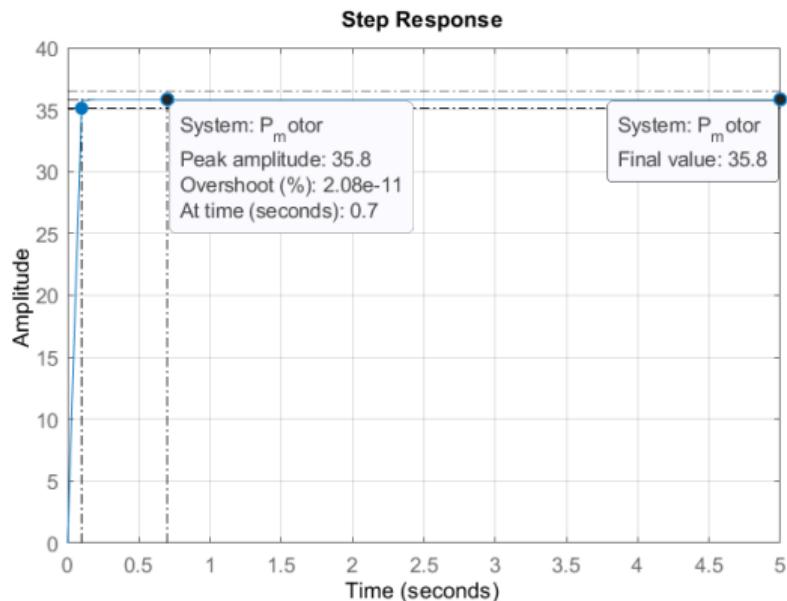
```
J = 3.2284E-6;
b = 3.5077E-6;
Kt = 0.0274;
Ke = 0.0274;
R = 4;
L = 2.75E-6;;
s = tf('s');
P_motor = Kt/((J*s+b)*(L*s+R)+Kt*Ke);
rP_motor = 0.1/(0.5*s+1)
ltvview('step', P_motor, 0:0.1:5);
```

Program *Closed-loop*:

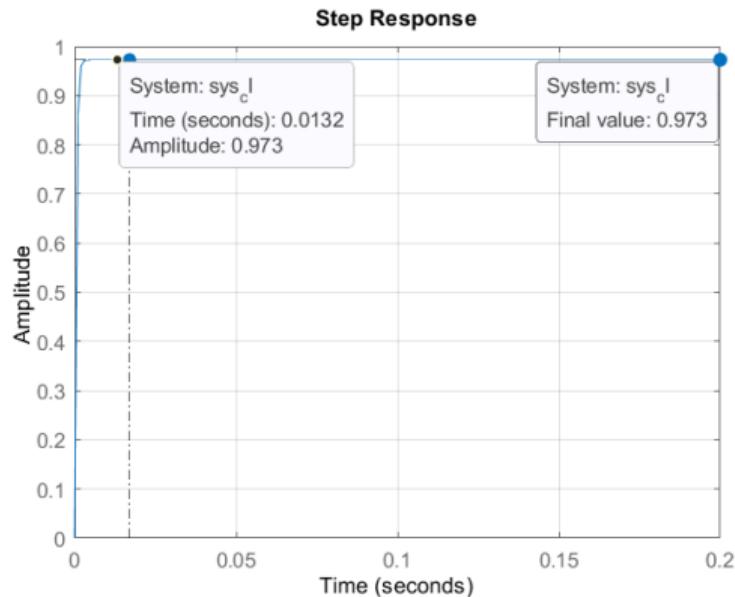
```
J = 3.2284E-6;
b = 3.5077E-6;
Kt = 0.0274;
Ke = 0.0274;
R = 4;
L = 2.75E-6;
s = tf('s');
P_motor = Kt/((J*s+b)*(L*s+R)+Kt*Ke);
t = 0:0.001:0.2;
sys_cl = feedback(P_motor,1)
step(sys_cl,t)
```

Uji Perbandingan Sistem *Open-loop* dengan *Closed-loop* Motor DC

Hasil *Open-loop*:



Hasil *Closed-loop*:



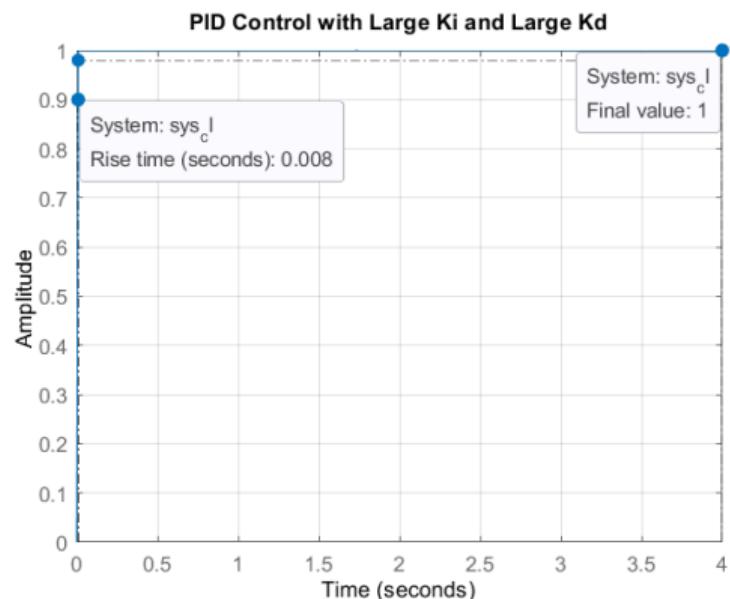
Kendali PID: Kecepatan

Program:

```
J = 3.2284E-6;
b = 3.5077E-6;
Kt = 0.0274;
Ke = 0.0274;
R = 4;
L = 2.75E-6;
s = tf('s');
P_motor = Kt/((J*s+b)*(L*s+R)+Kt*Ke);

Kp = 100;
Ki = 200;
Kd = 10;
C = pid(Kp,Ki,Kd);
sys_cl = feedback(C*P_motor,1);
step(sys_cl, 0:0.01:4)
grid
title('PID Control with Large Ki and Large Kd')
```

Hasil:



Kendali PID: Posisi

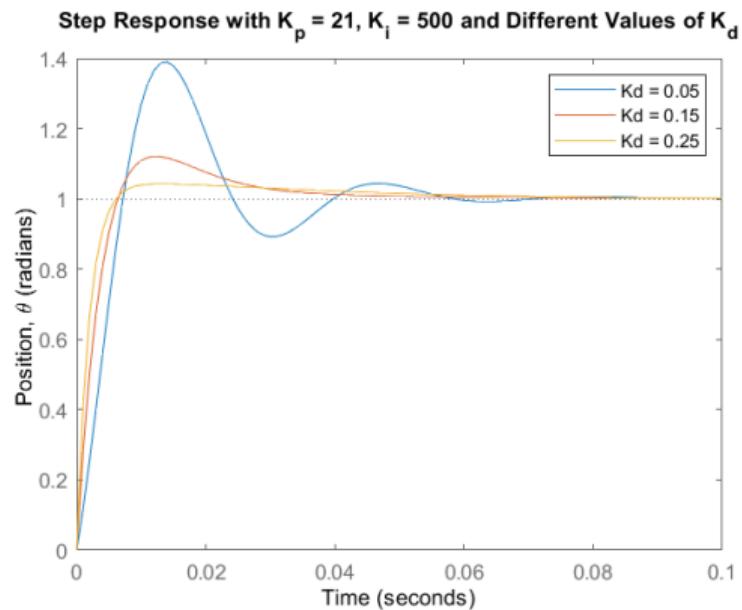
Program:

```
J = 3.2284E-6;
b = 3.5077E-6;
K = 0.0274;
R = 4;
L = 2.75E-6;
s = tf('s');
P_motor = K/(s*((J*s+b)*(L*s+R)+K^2));
Kp = 21;
Ki = 500;
Kd = 0.05;

for i = 1:3
    C(:,:,i) = pid(Kp,Ki,Kd);
    Kd = Kd + 0.1;
end

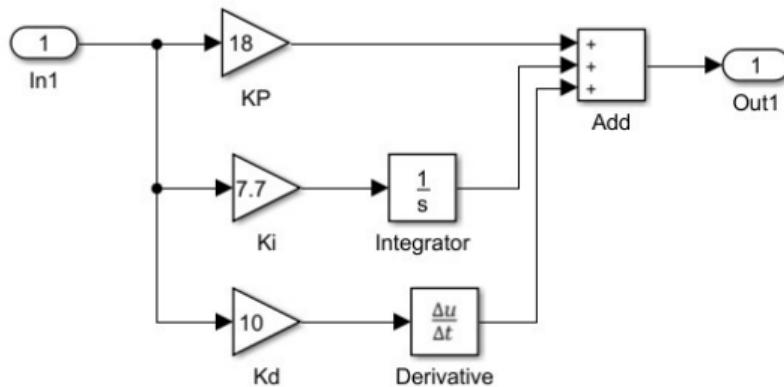
sys_cl = feedback(C*P_motor,1);
t = 0:0.001:0.1;
step(sys_cl(:,:,1), sys_cl(:,:,2), sys_cl(:,:,3), t)
ylabel('Position, \theta (radians)')
title('Step Response with K_p = 21, K_i = 500 and
        Different Values of K_d')
legend('Kd = 0.05', 'Kd = 0.15', 'Kd = 0.25')
```

Hasil:

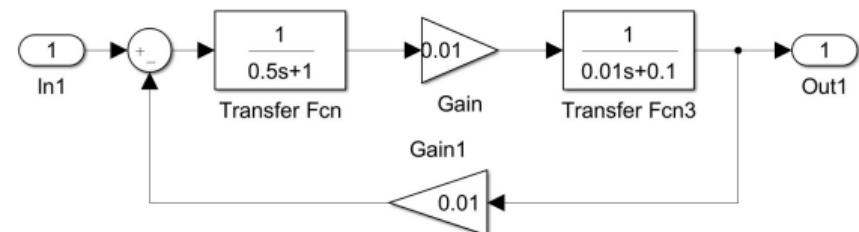


Simulink

Blok PID:

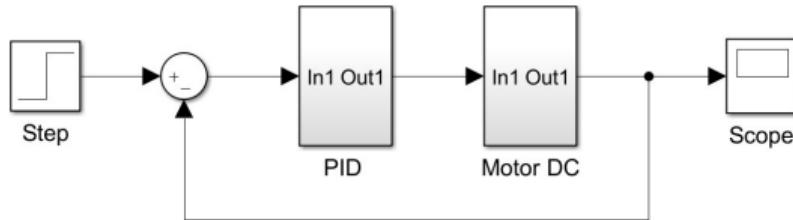


Blok Plant Motor DC

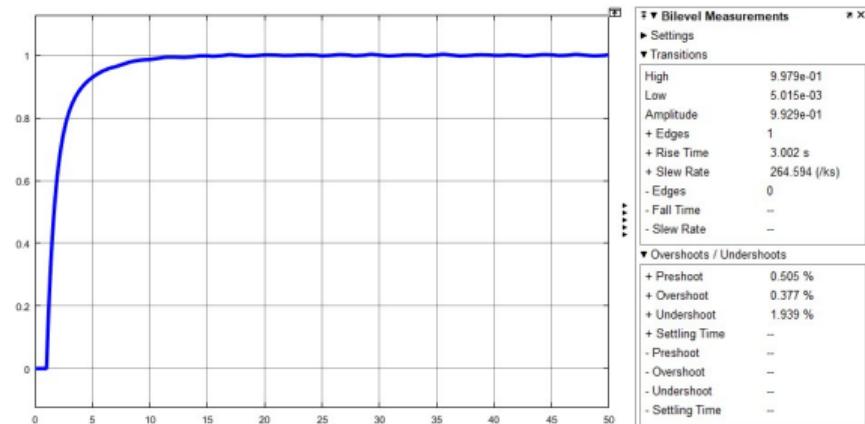


Simulink

Blok Simulink



Hasil



Simulasi LQR

Simulasi LQR: Kecepatan

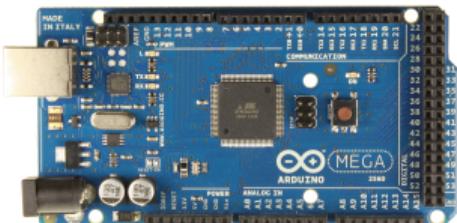
Klik untuk menjalankan animasi scroll:

Simulasi LQR: Posisi

Klik untuk menjalankan animasi scroll:

Sistem Mekanikal dan Elektrikal

Komponen-Komponen Utama



Arduino Mega



Motor DC



Tachogenerator



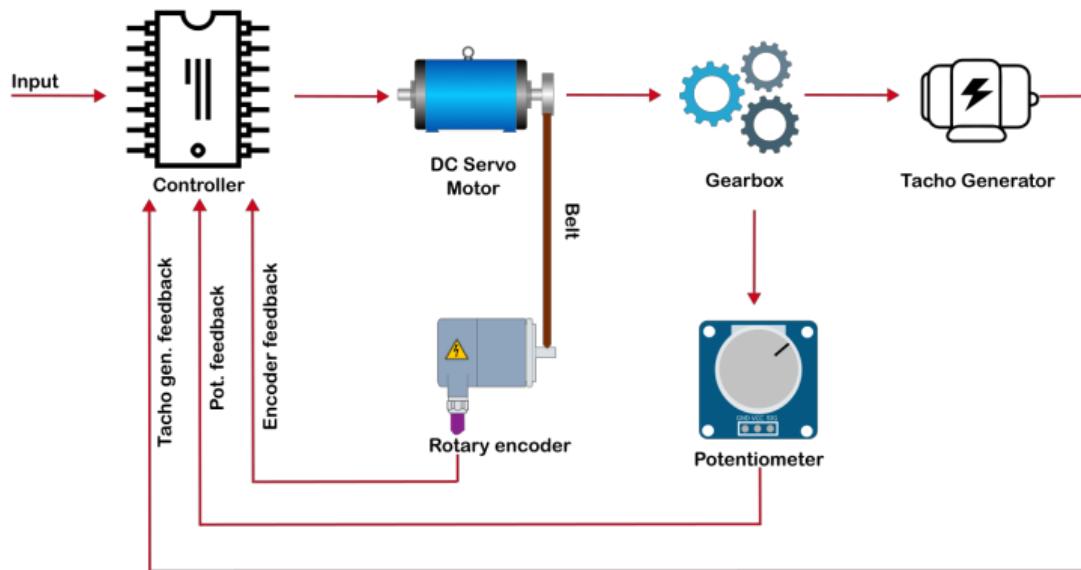
Potensiometer 360°



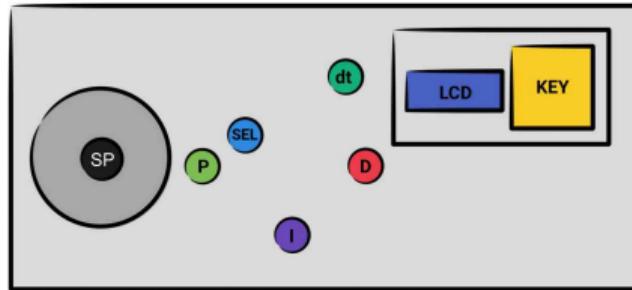
Rotary Encoder

Diagram Sistem

Komponen-komponen utama kemudian disusun menjadi suatu sistem yang ditunjukkan pada diagram sistem seperti yang sudah ditunjukkan di awal.



Konfigurasi Pin-Pin Arduino Mega

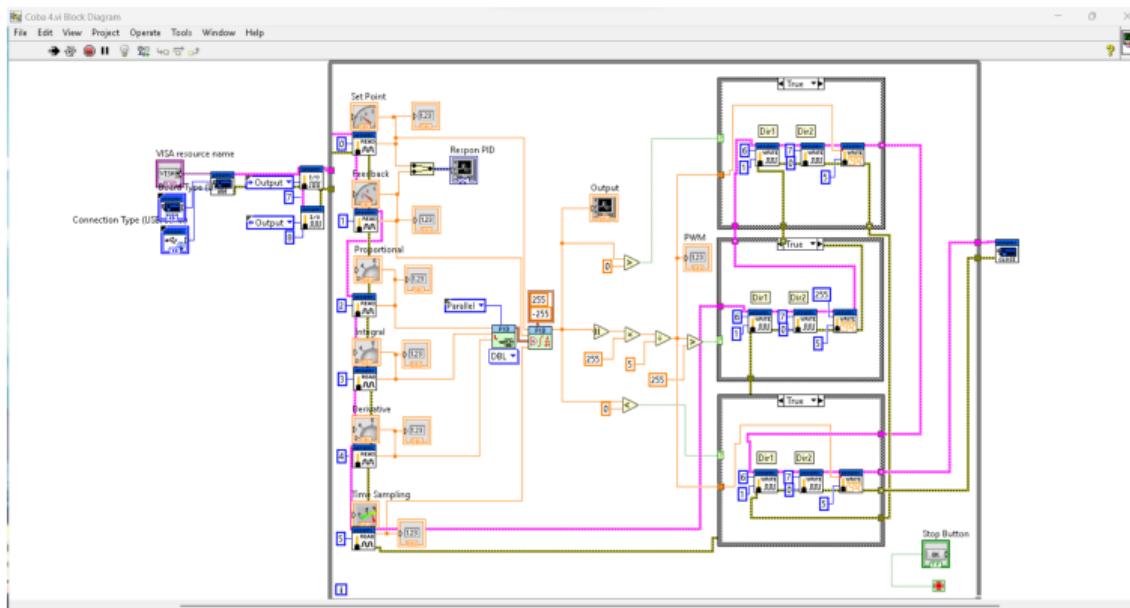


Kontroler

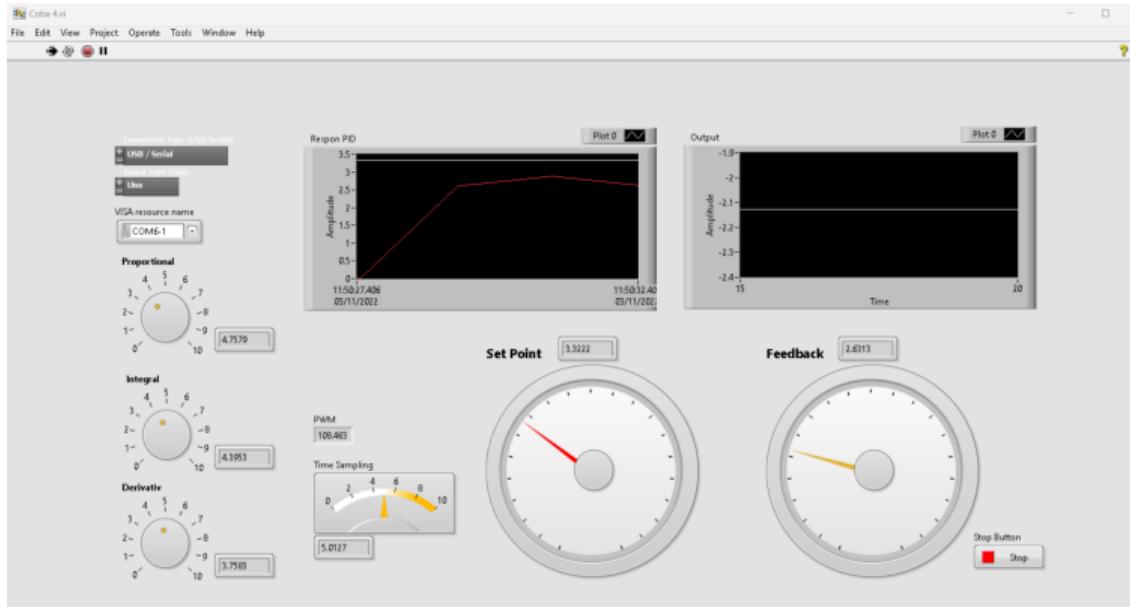
Input-Output	Fungsi	Pin
SP	Set Poin	A7
P	Proporsional	A5
I	Integral	A8
D	Derivatif	A6
SEL	Selector	34, 32, 30
dt	<i>Time Sampling</i>	A4
LCD	LCD	0, 1
KEY	Keypad	52, 50, 48, 46, 44, 42, 40, 38

Desain GUI

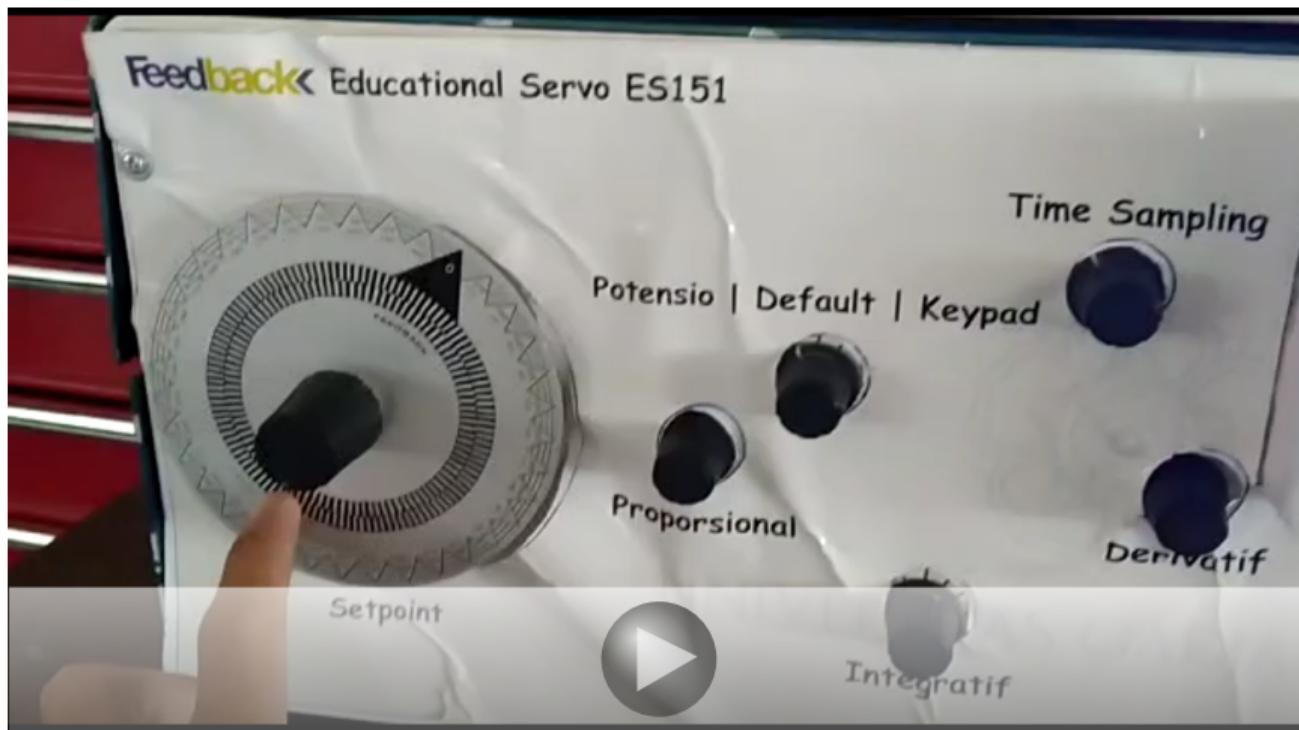
Diagram Labview



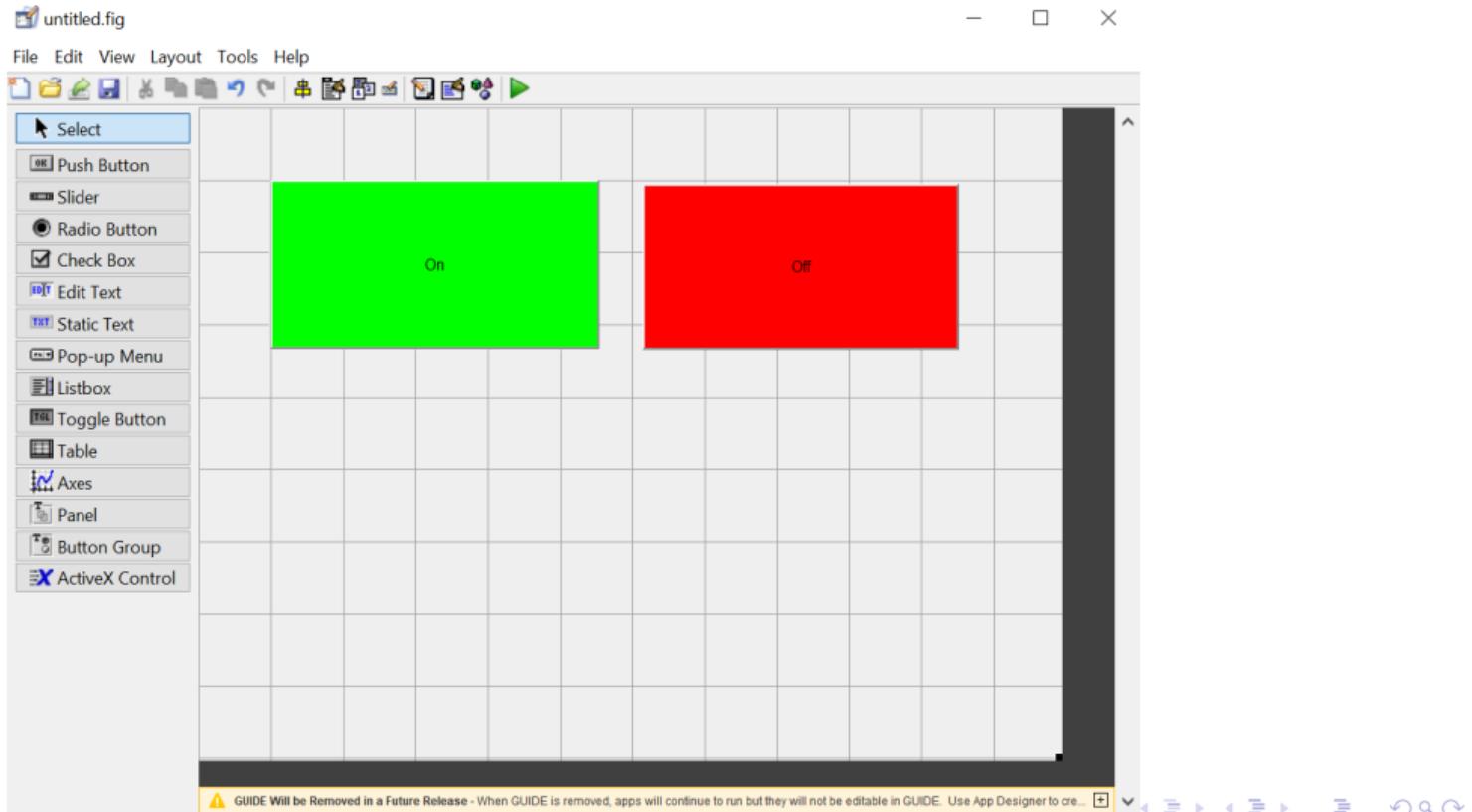
Tampilan GUI Labview



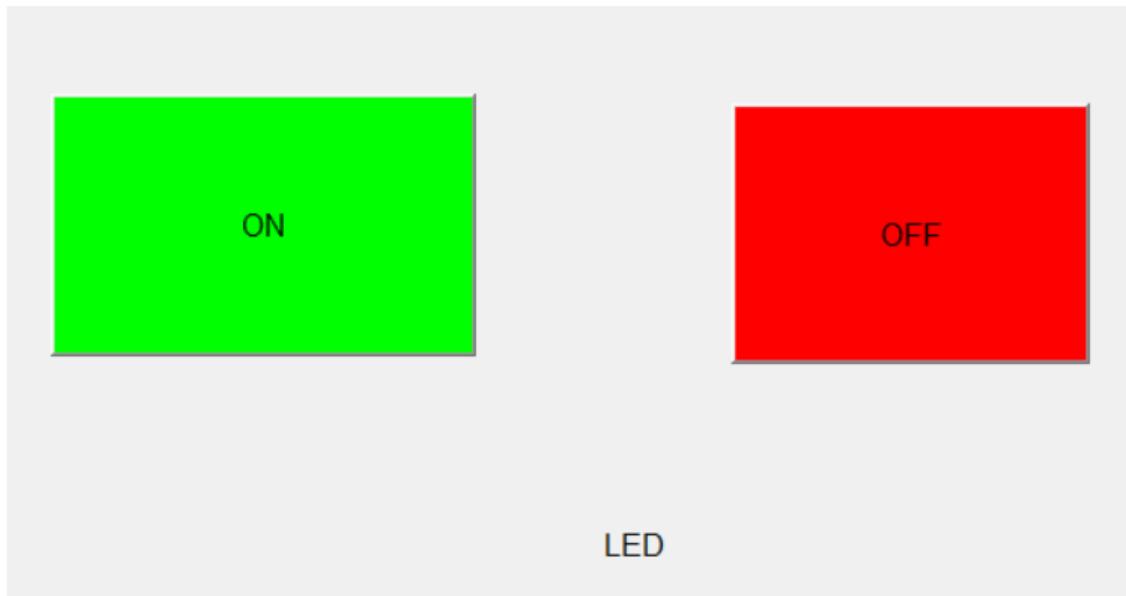
Video LabView



GUI Matlab-Arduino



Tampilan GUI Matlab-Arduino



Kode Program Pada Matlab

```
function varargout = LEDlagi2(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',      mfilename, ...
    'gui_Singleton',   gui_Singleton, ...
    'gui_OpeningFcn', @LEDlagi2_OpeningFcn, ...
    'gui_OutputFcn',  @LEDlagi2_OutputFcn, ...
    'gui_LayoutFcn',  [], ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

% --- Executes just before LEDlagi2 is made visible.
function LEDlagi2_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);
```

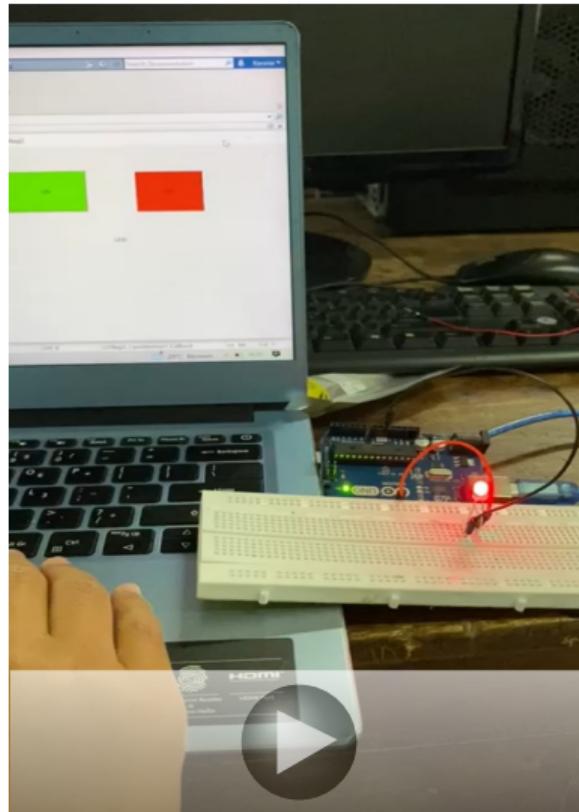
Kode Program Pada Matlab

```
% --- Outputs from this function are returned to the command line.
function varargout = LEDlagi2_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
clear all;
global a;
a = arduino;

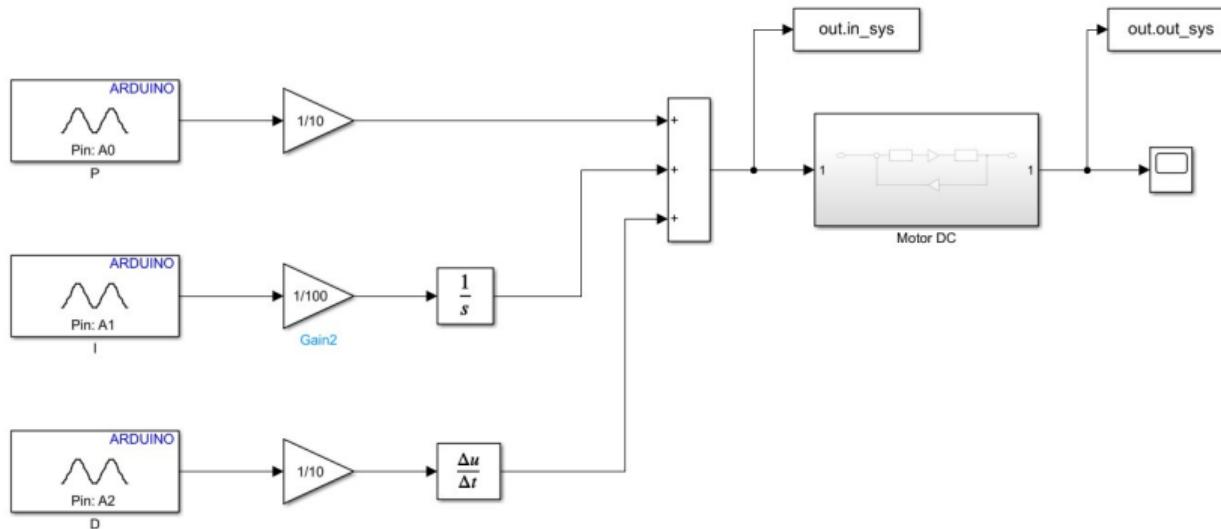
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
global a;
writeDigitalPin(a, 'D9',1);

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
global a;
writeDigitalPin(a, 'D9',0);
```

Video Simulasi GUI Matlab-Arduino

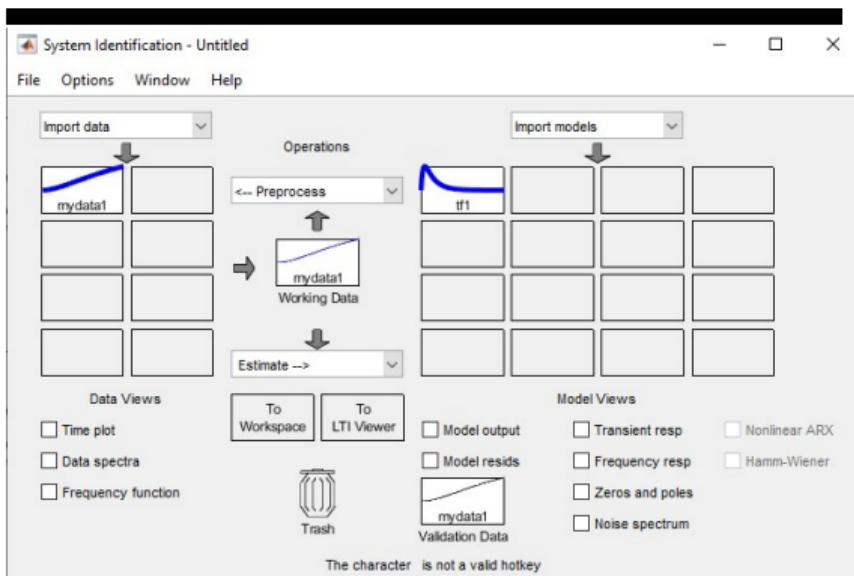


Identifikasi Sistem Pada Simulasi Arduino-Simulink

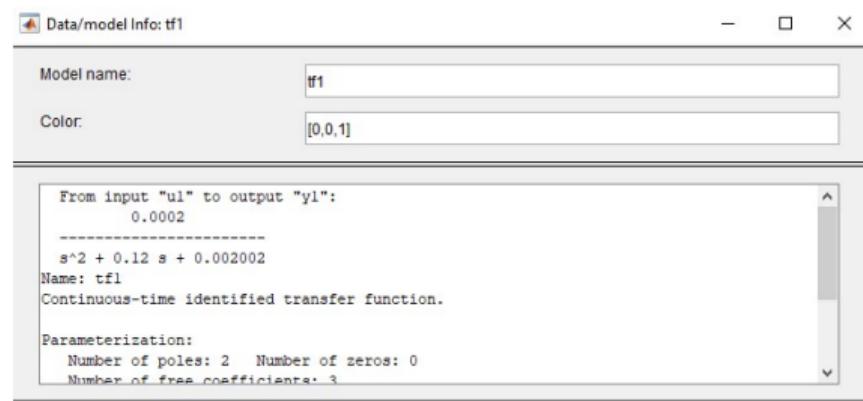


Proses Identifikasi Sistem

Identifikasi Sistem

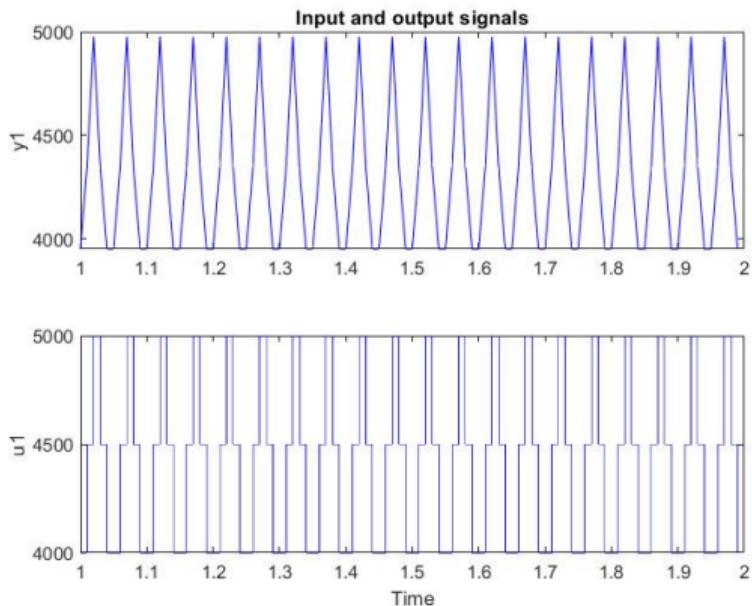


Hasil Transfer Function



Percobaan Identifikasi Sistem

Grafik Input dan Output



Transfer Function

```
tf1 =
```

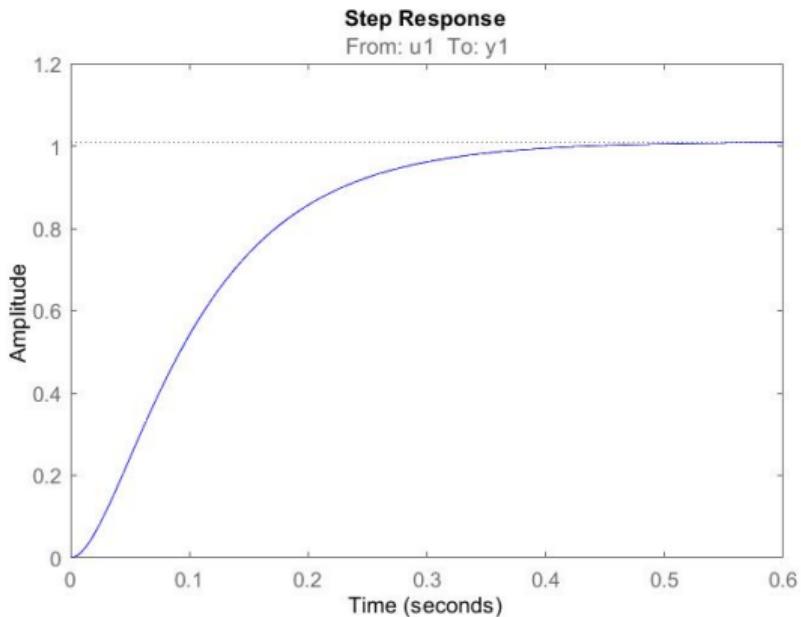
From input "u1" to output "y1":

386.7

 $s^2 + 44.73 s + 382.8$

Percobaan Identifikasi Sistem

Step Response



State Space

```
A =          x1           x2           x3           x4           x5           x6           x7
x1   328.2        487.8       24.99      -3364  -1.031e-30  4.167e-31  -1.444e-76
x2   -997.6       -940.2      -244.3       1556  4.623e-31  -1.869e-31  8.794e-77
x3   591.9        342.8       195.8       3612  6.666e-31  -2.694e-31  1.058e-76
x4   117.9        98.34      19.18      -28.59  -3.142e-32  1.27e-32   3.932e-78
x5    0            0            0            0       -200         0            0
x6    0            0            0            0         0       -200         0
x7    0            0            0            0         0         0       -200

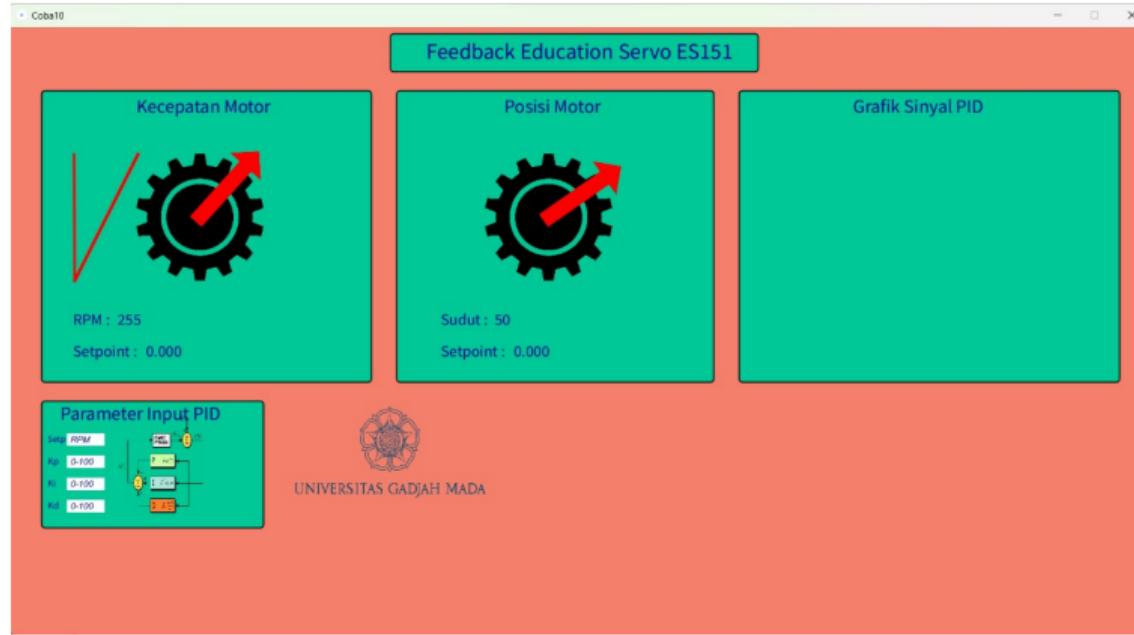
B =          u1
x1  -5.354e+31
x2  4.768e+31
x3  5.667e+31
x4  -4.095e+30
x5  -3.525e+20
x6  -9.512e+19
x7  -2.624e+19

C =          x1           x2           x3           x4           x5           x6           x7
y1  1.319e-28  9.893e-29  3.302e-29  1.962e-28  3.728e-61  -1.507e-61  8.734e-108

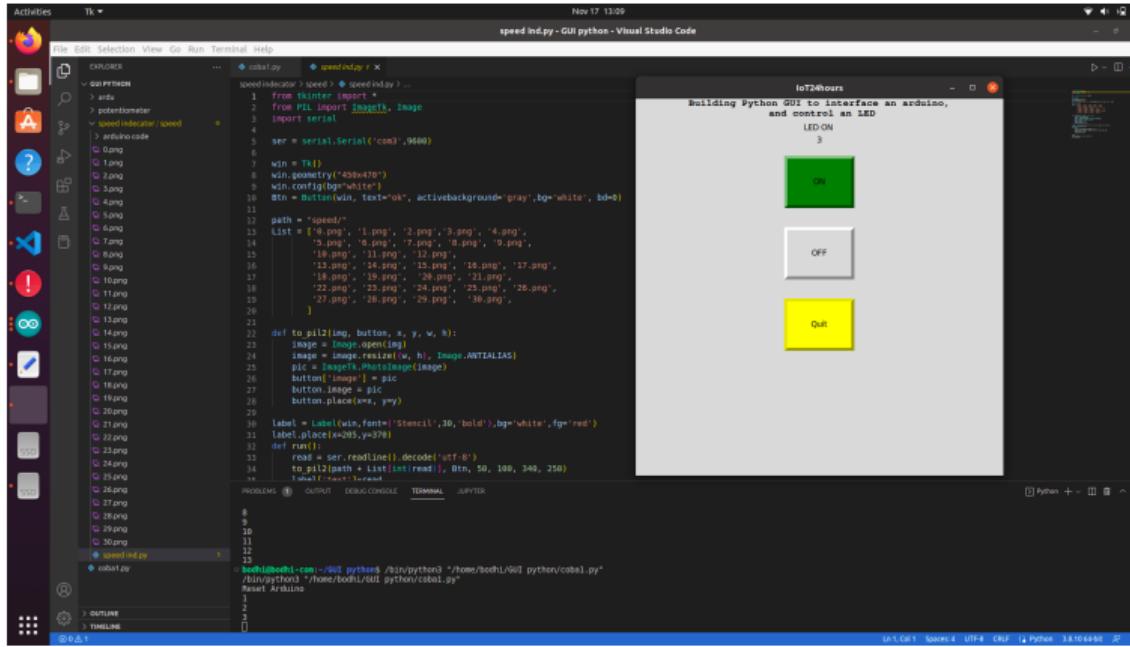
D =          u1
y1  0

E =          y1
x1  4.147e+31
x2  1.061e+31
x3  -3.959e+31
x4  -3.638e+30
x5  1.243e-32
x6  2.517e-79
x7  0
```

GUI Processing



GUI Python TKinter



The screenshot shows a Linux desktop environment with several windows open:

- Visual Studio Code (Left Window):** The title bar says "speed.indy - GUI python - Visual Studio Code". The code editor displays Python code for a Tkinter application named "speed.indy.py". The code uses PIL to load images and Tkinter to create a window with three buttons: "ON", "OFF", and "QUIT". It also includes a serial port connection to an Arduino.
- Tkinter Application (Right Window):** The title bar says "IoT24hours". The window contains the text "Building Python GUI to interface an arduino, and control an LED". Below this, there are three buttons labeled "LED ON", "OFF", and "QUIT".
- Terminal (Bottom):** The terminal window shows the command "bohdihedih@bohdihedih:~/PyQt\$ python3 ./bin/python3 /home/bohdihedih/GUI python/cobal.py" being run.

Temporary page!

\LaTeX was unable to guess the total number of pages correctly. As there was some unprocessed input that should have been added to the final page this extra page has been added to receive it. If you rerun the document (without altering it) this surplus page will go away, because \LaTeX knows how many pages to expect for this document.