

**PETUNJUK PRAKTIKUM DASAR SISTEM KENDALI DENGAN
SCILAB**



Disusun oleh:

Alfian Ma'arif, S. T., M. Eng.

**LABORATORIUM OTOMASI DAN SISTEM EMBEDDED
PRAGRAM STUDI TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS AHMAD DAHLAN
2020**

DAFTAR ISI

DAFTAR ISI	2
Praktikum DSK Unit 1 Pengenalan Scilab.....	3
Praktikum DSK Unit 2 Model Fungsi Alih dan Respons Sistem	14
Praktikum DSK Unit 3 Model Fungsi Alih Motor DC dengan Scilab	20
Praktikum DSK Unit 4 Pengenalan Aplikasi XCOS Scilab	24
Praktikum DSK Unit 5 Kendali Proporsional Integral Derivatif (PID) Motor DC dengan XCOS	29
DAFTAR PUSTAKA.....	42

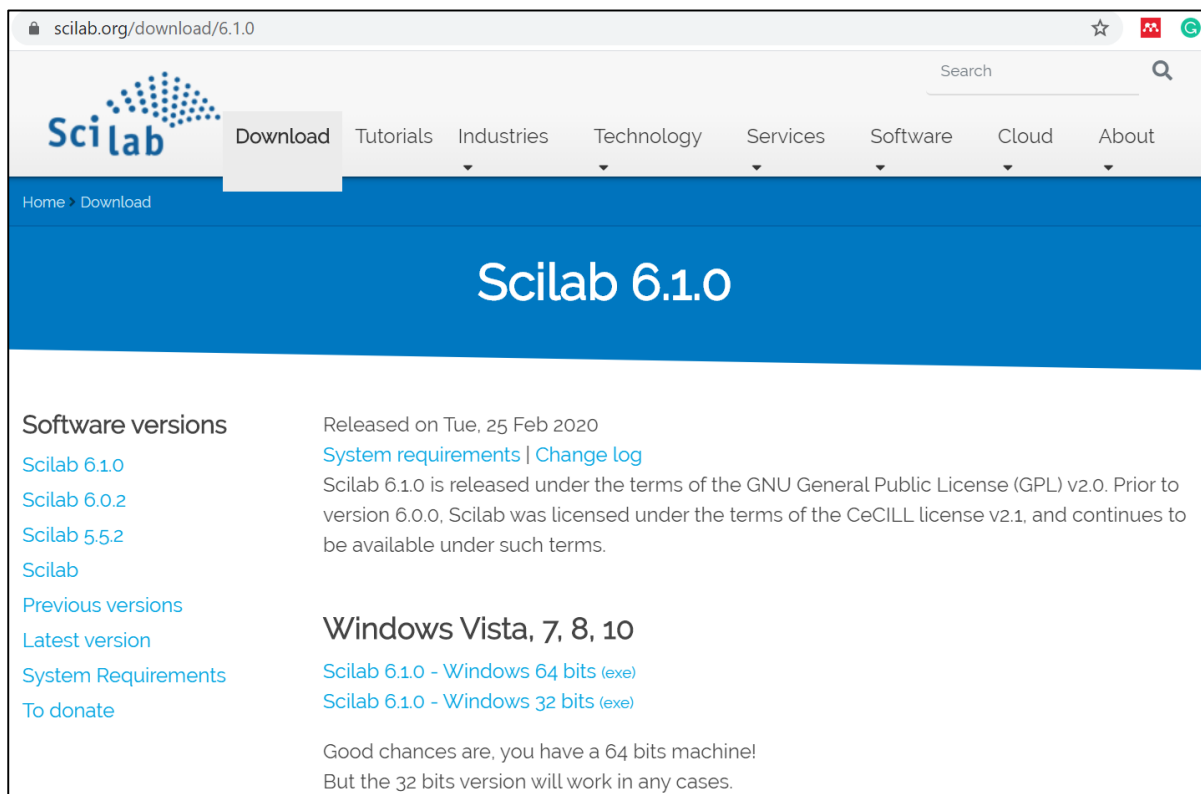
Praktikum DSK Unit 1 Pengenalan Scilab

oleh Alfian Ma'arif, M. Eng.

Praktikum Dasar Sistem Kendali menggunakan software Scilab. Praktikan diwajibkan untuk mendownload dan menginstall Scilab pada komputer atau laptop masing-masing. Praktikan mengakses halaman website Scilab pada alamat URL berikut

<https://www.scilab.org/download/6.1.0>.

Halaman website Download Scilab ditunjukkan pada Gambar 1.1. Pilih Scilab untuk Windows 7/8/10 dan 32/64 bits sesuai dengan komputer yang praktikan gunakan. Untuk mengetahui berapa bit processor pada komputer dapat diketahui dengan mengecek Propertis pada My Computer, perhatikan Gambar 1.2.



Gambar 1. 1. Website Download Scilab



Gambar 1. 2. Spesifikasi Windows dan Processor

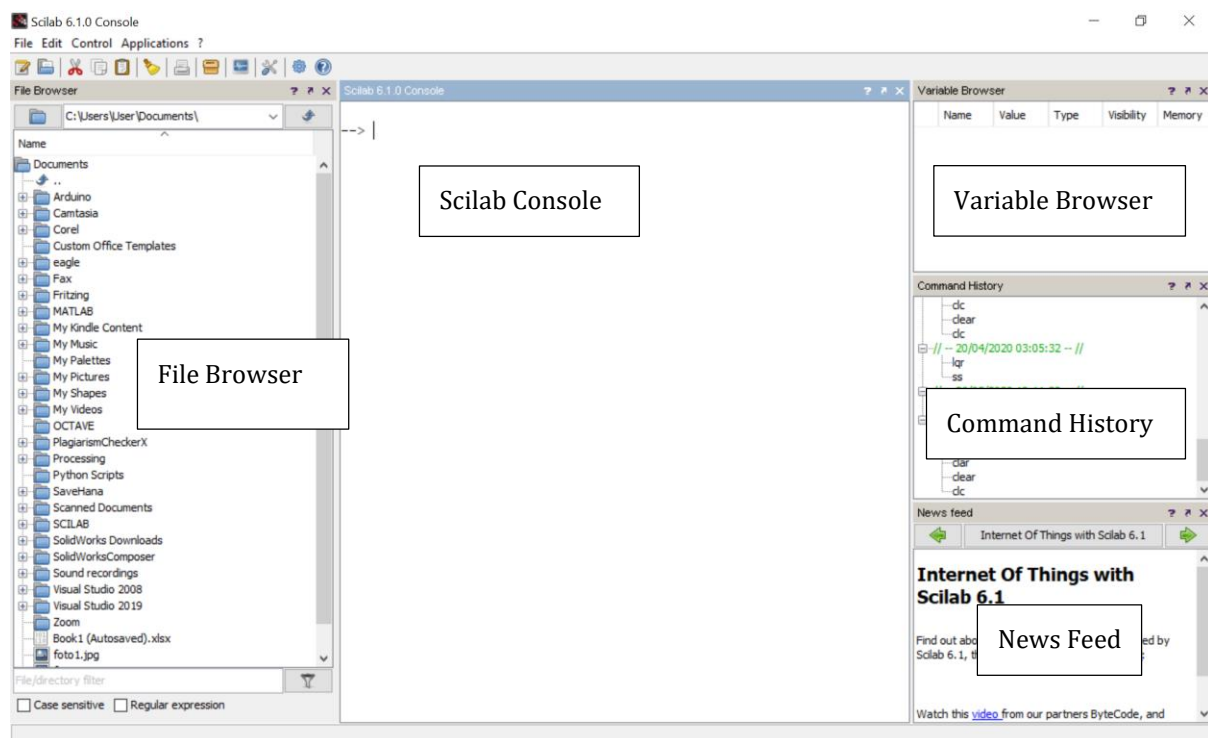
Tujuan Praktikum

Mengenalkan Scilab kepada Mahasiswa dan menjalankan fungsi-fungsi sederhana

Pengantar Scilab

Scilab adalah software alternatif untuk Matlab. Perbedaan Matlab dan Scilab adalah Matlab merupakan software berbayar dan memiliki file berukuran besar sementara Scilab merupakan software open source (gratis) dan memiliki file berukuran kecil. Kekurangan Scilab adalah tidak memiliki fitur selengkap Matlab atau memiliki fitur yang terbatas. Namun untuk keperluan praktikum dan mempelajari tentang dasar sistem kendali, Scilab sudah sangat cukup.

Setelah Mahasiswa mendownload, menginstall dan menjalankan Scilab, tampilan awal Scilab ditunjukkan pada Gambar 1.1. Terdapat beberapa bagian pada Scilab yaitu File Browser Scilab Console, Variable Browser, Command History, dan News Feed. File browser adalah folder tempat file file Scilab disimpan. Scilab Console (command editor) adalah tempat mengetikkan kode program. Variable browser adalah tempat variabel yang telah dijalankan disimpan sementara. Command History adalah tempat kode yang telah diketikkan sebelumnya dan news feed berisi tentang berita tentang Scilab.



Gambar 1. 3. Tampilan Awal Scilab

Pre-Test

Jawablah pertanyaan berikut dan tuliskan jawaban pada lembar laporan.

1. Apa itu software Scilab?
2. Apa perbedaan antara Scilab dan Matlab?

3. Apa kelebihan Software Scilab?
4. Pada bagian apa tempat mengetikkan perintah/ kode pada Scilab?
5. Pada bagian apa tempat variabel yang telah diketikkan disimpan pada Scilab?

Langkah Praktikum dan Post-Test

Ketikkan perintah-perintah setelah tanda (`-->`) pada Scilab (tanpa tanda `-->`). Lalu amati, perhatikan, pahami, analisis, modifikasi sesuai petunjuk dan tuliskan jawaban pertanyaan pada lembar laporan.

Perintah Bantuan dan Operasi Dasar

Di dalam Scilab sudah terdapat petunjuk dan definisi fungsi yang siap dioperasikan. Untuk mengetahui petunjuk atau bantuan Scilab dapat digunakan perintah `help`. Ketikkan Perintah `help` tanpa tanda (`-->`).

`--> help`

1. Apa fungsi perintah `help` pada Scilab?

`--> a=1`

`--> b=2`

`--> c=3;`

`--> d=4;`

2. Apa perbedaan antara perintah dengan dan tanpa tanda titik koma (;)?
3. Tuliskan variabel apa saja yang tersimpan pada bagian variable browser?

`--> clc`

4. Apa fungsi perintah `clc`?

`--> clear`

5. Amati bagian variable browser, apa fungsi perintah `clear`?

Operasi Matematika

Seperti halnya kalkulator, Scilab juga dapat mengerjakan operasi perhitungan sederhana. Operasi matematika ditunjukkan pada Tabel 1.1.

Tabel 1. 1. Operasi Matematika

$a+b$	Operasi Penjumlahan	a/b	Operasi Pembagian
$a-b$	Operasi pengurangan	$a \setminus b$	Operasi Pembagian
$a*b$	Operasi Perkalian	a^b	Operasi Pangkat

`--> 1+2+3+4`

`--> 10-1-2-3`

`--> 2*3*4`

--> 10/5

--> 10\5

--> 2^3

6. Berapa nilai hasil perhitungan? Apa fungsi perintah (+), (-), (*), (/), (\), (^) tersebut?

Variabel

Aturan Penamaan variabel dalam Scilab adalah

a. Penamaan variabel bersifat sensitif

Contoh: variabel dengan nama panjang, Panjang, panjang, PANJANG merupakan variabel yang berbeda

b. Nama variabel dimulai dengan huruf.

Contoh: panjang, lebar, tinggi

c. Nama variabel dapat berupa huruf dan angka

Contoh: panjang1, panjang2, panjang3

d. Nama variabel dapat menggunakan karakter garis bawah

Contoh: luas_persegi, luas_lingkaran, lebar_segitiga

Dalam Scilab terdapat variabel khusus yang ditunjukkan pada Tabel 1.1.

Tabel 1. 2. Variabel Khusus

%pi	3.1415927
%i	$i = \sqrt{-1}$
%e	epsilon= 2.7182818
%nan	bukan sebuah bilangan (not an number)
%inf	Tidak terhingga (infinity)

--> 0/0

--> 99/0

--> 0/99

7. Berapa nilai hasil perhitungan 0/0, 99/0 dan 0/99?

Fungsi Matematika

Scilab memiliki operasi umum matematika yang sudah disusun dalam fungsi-fungsi yang ditunjukkan pada Tabel 1.3.

Tabel 1. 3. Fungsi Matematika

--> abs(x)	fungsi untuk menghitung nilai mutlak dari suatu bilangan real atau nilai modulus dari suatu bilangan kompleks
--> round(x)	pembulatan ke terdekat
--> ceil(x)	pembulatan ke atas
--> floor(x)	pembulatan ke bawah

--> sqrt(x)	fungsi untuk menghitung akar pangkat dua
--> exp(x)	fungsi untuk menghitung nilai eksponensial
--> log(x)	fungsi untuk menghitung nilai logaritma

--> abs(-2)

--> round(5.67)

--> ceil(5.67)

--> floor(5.67)

--> sqrt(16)

8. Berapa nilai hasil perhitungan? Mengapa nilai perhitungan dapat memberikan hasil tersebut? Apa fungsi fungsi perintah abs, round, ceil, floor, sqrt?

Operasi Trigonometri

Daftar fungsi-fungsi trigonometri dan inverse trigonometri yang terdapat dalam Scilab diberikan pada Tabel 1.4.

Tabel 1. 4. Fungsi Trigonometri

--> sin(x)	Nilai sinus dalam radian
--> sind(x)	Nilai sinus dalam derajat
--> asin(x)	Nilai balik dari sinus dalam radian
--> asind(x)	Nilai balik dari sinus dalam derajat
--> cos(x)	Nilai cosinus dalam radian
--> cosd(x)	Nilai cosinus dalam derajat
--> acos(x)	Nilai balik dari cosinus dalam radian
--> acosd(x)	Nilai balik dari cosinus dalam derajat

--> sin(%pi/2)

--> sind(90)

--> asind(1)

9. Berapa nilai sinus 90 derajat dan sinus %pi/2 radian? Apakah sama?

10. Berapa nilai balik sinus (1) dengan perintah asind(1)?

Array

Array adalah kumpulan data yang memiliki tipe data yang sama dan dinyatakan dalam nama yang sama. Array digunakan untuk menyimpan banyak nilai dalam satu variabel. Konstruksi Array pada Scilab ditunjukkan pada Tabel 1.5.

Tabel 1. 5. Array

--> x=[awal:akhir]	membuat vektor baris x dengan elemen pertama adalah nilai awal dan elemen terakhir adalah nilai akhir dengan nilai penambahan 1 angka.
--------------------	--

--> x=[1:5]	membuat vektor baris x dengan elemen pertama adalah 1 dan elemen terakhir adalah 5 dengan penambahan 1 angka
--> x=[awal:undak:akhir]	membuat vektor baris x dengan elemen pertama adalah nilai awal dan elemen terakhir adalah nilai akhir dengan penambahan/pengurangan nilai undak
--> x=[1:2:5]	membuat vektor baris x dengan elemen pertama adalah 1 dan elemen terakhir adalah 5 dengan penambahan nilai undak 2 angka
--> x=linspace(awal,akhir, n)	membuat vektor baris dengan elemen pertama adalah awal dan elemen terakhir adalah akhir dengan jumlah elemen adalah n
--> x=linspace(1,5,10)	membuat vektor baris dengan elemen pertama adalah 1 dan elemen terakhir adalah 5 dengan jumlah elemen adalah 10 angka

--> x= [1:10]

--> y= [1:2:10]

--> z= [1:3:10]

11. Apa perbedaan antara ketiga perintah x= [1:10], y= [1:2:10], z= [1:3:10]? Kenapa hasil array bisa berbeda?

Catatan : Tekan tombol atas untuk mengulang perintah sebelumnya.

--> a= [10:-1:1]

--> b= [10:-2:1]

12. Bagaimana cara membuat array dari 10 sampai 1 dengan nilai pengurangan 3 angka?

13. Apa perbedaan antara hasil perintah a= [10:-1:1] dan b= [10:-2:1]?

--> deret1= linspace(1,5,10)

--> deret2= linspace(1,5,20)

14. Apa perbedaan antara bilangan deret1 dan deret2?

Elemen deret1 dan deret2 tersimpan pada variabel browser. Klik dua kali pada nama variabel untuk melihat isi elemennya.

Polinomial atau Bilangan Berpangkat

Polinomial pada Scilab direpresentasikan dalam bentuk vektor baris dari koefisien polinomial yang terkecil sampai yang terbesar. Koefisien yang paling kiri adalah koefisien pangkat terbesarnya dan koefisien paling kanan adalah pangkat terkecilnya. Lalu, jika terdapat nilai koefisien nol pada suatu pangkat tertentu, maka koefisien nol tersebut tetap harus dituliskan.

Contoh: Buatlah Polinomial $x^4 - 12x^3 + 25x + 116$ menggunakan Scilab

Jawab: Polinomial tersebut tidak memiliki orde/pangkat x^2 , oleh karena itu perlu dituliskan sebagai

$$x^4 - 12x^3 + 0x^2 + 25x + 116$$

Ketikkan perintah berikut untuk membuat persamaan polinomial dalam vektor baris (array).

```
--> p= [1 -12 0 25 116]
```

Untuk mencari nilai akar dari polinomial tersebut digunakan fungsi roots(p)

```
--> roots(p)
```

15. Berapa nilai akar dari polinomial $x^4 - 12x^3 + 25x + 116$?

16. Dengan menggunakan Scilab hitunglah berapa nilai akar dari polinomial $x^2 - 3x + 2$?

Membuat Grafik

Di dalam Scilab terdapat fasilitas untuk menggambar grafik, baik grafik 2 Dimensi atau 3 Dimensi. Grafik 2 Dimensi yaitu grafik hubungan 2 variabel dibuat dengan fungsi plot. Ketikkan Perintah berikut.

```
--> x=linspace(0,2*pi,30) ;
```

```
--> y=sin(x) ;
```

```
--> plot(x,y);
```

17. Gambarlah grafik sinus yang terbentuk.

```
--> xgrid;
```

18. Amati pada grafik. Apa perbedaan dengan grafik sebelumnya? Apa fungsi perintah xgrid tersebut?

```
--> title("Grafik Sinus");
```

19. Amati pada grafik. Apa perbedaan dengan grafik sebelumnya? Apa fungsi perintah title("Grafik Sinus") tersebut?

```
--> xlabel("Sudut (radian)");
```

```
--> ylable("Nilai Sinus");
```

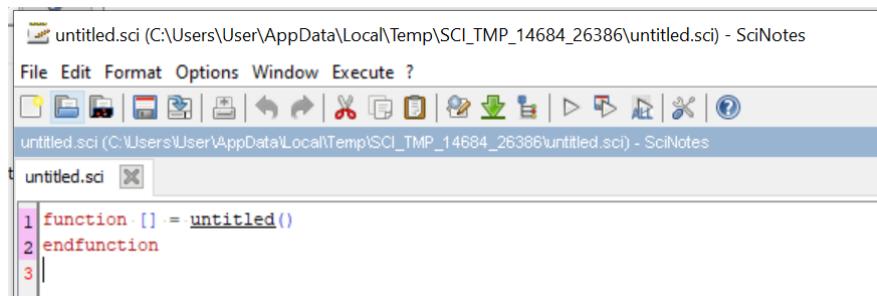
20. Amati pada grafik. Apa perbedaan dengan grafik sebelumnya? Apa fungsi perintah xlabel("Sudut (radian)") dan ylable("Nilai Sinus") tersebut?

Editor Scilab

Selain perintah dapat dituliskan langsung di Scilab Console, perintah tersebut juga dapat dituliskan dalam editor dan disimpan dalam file berektensi *.sci. File tersebut berisi deretan perintah yang ingin dieksekusi secara berurutan dengan cara sebagai berikut. Ketikkan perintah edit pada console Scilab, lalu Window Editor Scilab akan muncul seperti ditunjukkan pada Gambar 1.4.

Ketikkan perintah berikut untuk membuat program dengan editor Scilab.

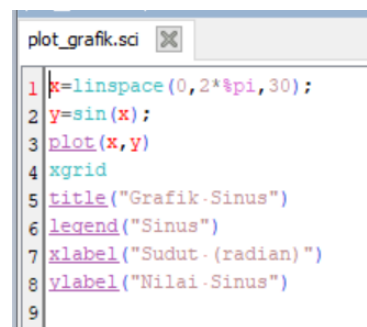
```
--> edit
```



Gambar 1. 4. Editor Scilab

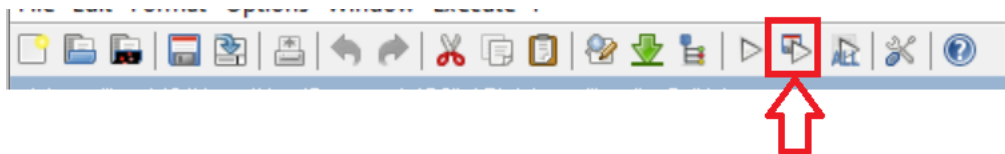
Pada jendela yang muncul hapus tulisan tersebut dan ketikkan perintah berikut. Lalu simpan dengan nama **plot_grafik.sci**

```
x=linspace(0,2*%pi,30);
y=cos(x);
plot(x,y)
xgrid
title("Grafik Cosinus")
xlabel("Sudut (radian)")
ylabel("Nilai Cosinus")
```



Gambar 1. 5. Editor Scilab

Untuk menjalankan file tersebut, pilih Execute lalu pilih Save and execute, atau dapat juga menggunakan tombol F5 atau menggunakan tombol Save and execute (Gambar 1.6) pada Window tersebut.



Gambar 1. 6. Tombol Save and Execute

21. Gambarkanlah grafik cosinus yang terbentuk.
22. Berikan analisis fungsi dari setiap baris program tersebut.

Membuat program perhitungan

Ketikkan perintah edit dan masukkan program berikut pada editor Scilab.

```
pensil=4;
kertas=5;
penghapus=2;
jumlah_barang=pensil+kertas+penghapus
total_pembelian=pensil*50+kertas*100+penghapus*25
```

23. Berapa nilai variabel jumlah_barang dan total_pembelian?

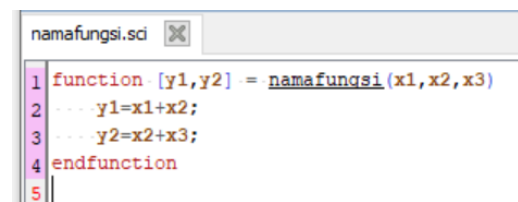
Membuat Fungsi di Scilab

Selain fungsi-fungsi standar yang sudah ada di dalam Scilab, pengguna bisa membuat sendiri fungsi-fungsi sesuai yang diperlukan. Dalam hal ini fungsi merupakan suatu program yang didefinisikan oleh pemakai. Fungsi tersebut dapat melibatkan fungsi yang sudah ada di dalam Scilab. Semua fungsi akan disimpan dalam file berektensi ***.sci**. Ketikkan perintah edit.

--> edit

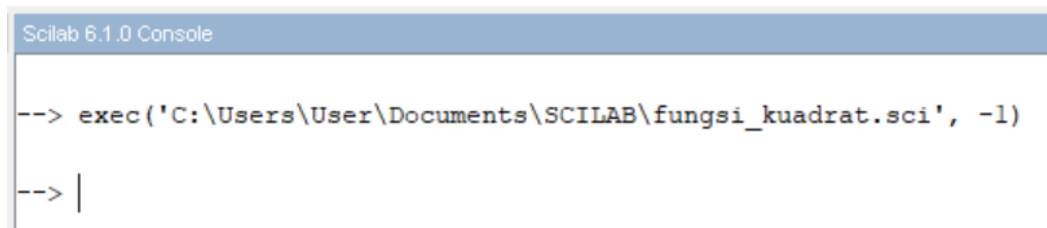
Pada jendela yang muncul modifikasi script tersebut sesuai script berikut (Gambar 1.7).

```
function [y1,y2] = fungsi1(x1,x2,x3)
    y1=x1+x2;
    y2=x2+x3;
endfunction
```



Gambar 1. 7. Editor Scilab

Simpan file dengan nama **fungsi1.sci**. Jalankan file tersebut dengan menekan tombol execute atau tombol F5 sampai muncul keterangan pada console Scilab seperti pada Gambar 1.8.



Gambar 1. 8. Keterangan Fungsi sudah Dieksekusi pada Editor Scilab

Untuk menjalankan fungsi tersebut ketikkan perintah berikut.

--> [y1,y2]=fungsi1(1,2,3)

24. Tuliskan hasil perintah tersebut. Berapa nilai y1 dan y2?

25. Dari mana nilai y1 dan y2 diperoleh?

Catatan: Fungsi pada Scilab tidak ada bisa bekerja jika belum dieksekusi atau dijalankan. Oleh karena itu pastikan file fungsi sudah di save dan di-execute (F5) sebelum digunakan atau dipanggil.

Perhatikan kembali fungsi berikut.

```
--> [y1,y2]=fungsi1(1,2,3)
```

```
function [y1,y2] = fungsi1(x1,x2,x3)
    y1=x1+x2;
    y2=x2+x3;
endfunction
```

Pada fungsi tersebut, variabel x1, x2, dan x3 adalah masukan fungsi (input) dan variabel y1, y2 adalah keluaran fungsi (output).

26. Berapa nilai input dari fungsi1 tersebut? Berapa nilai output dari fungsi tersebut? Berapa jumlah input dan output dari fungsi1 tersebut?

Membuat Grafik Fungsi Polinomial

Ketikkan perintah edit dan masukkan program berikut pada editor Scilab. Simpan program tersebut dengan nama **fungsi_kuadrat.sci** dan execute dengan menekan tombol F5.

```
function [y] = fungsi_kuadrat(x)
    y=x.^2-2.*x-3;
endfunction
```

Pada console scilab, jalankan perintah berikut.

```
--> x=-4:0.01:6;
--> y=fungsi_kuadrat(x);
--> plot(x,y)
--> xgrid
--> title("Grafik Fungsi Kuadrat")
--> xlabel("sumbu x")
--> ylabel("sumbu y")
```

27. Gambarkan hasil plot pada fungsi tersebut.

Laporan

Laporan dikerjakan dengan ditulis tangan menggunakan bolpoin tinta biru atau hijau. Laporan dikirimkan ke asisten dan dikirimkan ke google form. Laporan tidak memerlukan halaman muka atau halaman cover. Cantumkan nama praktikum, judul unit praktikum, nama dan NIM pada bagian atas dilanjutkan dengan jawaban Pre-Test dan Post-Test. Laporan dibuat 2 kolom dengan garis tepi sebesar 1cm pada bagian atas, bawah, kanan, dan kiri. Contoh laporan ditunjukkan pada contoh berikut.

Contoh Laporan Praktikum DSK

Praktikum DSK Unit 1 Pengenalan Scilab Nama (NIM)	
<u>Tujuan Praktikum</u> Mengenalkan Scilab kepada mahasiswa dan menjalankan fungsi-fungsi sederhana <u>Alat dan Bahan Praktikum</u> 1. Komputer/ Laptop 2. Software Scilab <u>Jawaban Pre-Test</u> 1. 2. 3.	<u>Jawaban Post-Test</u> 1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 11. 12.

Praktikum DSK Unit 2 Model Fungsi Alih dan Respons Sistem

oleh Alfian Ma'arif, M. Eng.

2.1. Tujuan Praktikum

Mahasiswa memahami tentang model fungsi alih dan respons sistem.

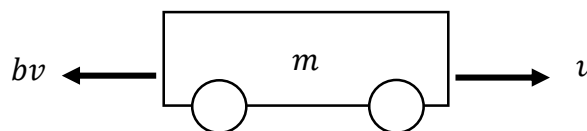
2.2. Model Fungsi Alih/ Transfer Function

Dalam mendesain pengendali suatu sistem, bagian pertama yang dilakukan adalah membuat model sistem. Model sistem adalah bentuk persamaan yang dapat menggambarkan karakteristik suatu sistem dinamik. Untuk memperoleh model sistem dilakukan analisis menggunakan hukum fisika dan dituliskan dalam persamaan matematis. Salah satu model dalam sistem kendali adalah model dalam bentuk Fungsi Alih (transfer function). Model fungsi alih adalah rasio perbandingan antara hasil transformasi laplace input dan output sistem dengan asumsi kondisi awal sama dengan nol. Bentuk umum model fungsi alih dapat dituliskan sebagai

$$G(s) = \frac{\text{num}(s)}{\text{den}(s)} = \frac{\text{output}}{\text{input}} = \frac{a_3s^3 + a_2s^2 + a_1s + a_0}{b_3s^3 + b_2s^2 + b_1s + b_0} \quad (2.1)$$

2.3. Membuat Model Fungsi alih

Model sistem kendaraan ditunjukkan pada Gambar 2.1. Variabel u adalah gaya dorong mobil, m adalah massa mobil dan bv adalah gaya gesek roda dengan jalan. Pada model tersebut, diasumsikan hambatan angin sama dengan nol sehingga tidak perlu dituliskan. Untuk membuat model fungsi alih, perlu didefinisikan terlebih dahulu variabel input dan outputnya. Pada sistem kendaraan inputnya adalah gaya dorong (u) dan outputnya adalah kecepatan linier (v).



Gambar 2.1. Model Sistem Kendaraan

Dengan analisis menggunakan hukum Newton II, persamaan sistem kendaraan dapat dituliskan sebagai

$$\sum F = m\dot{v}$$

$$u(t) - bv(t) = m\dot{v}(t) \quad (2.2)$$

Variabel $\dot{v}$ adalah percepatan linier dan $v(t)$ adalah kecepatan linier. Tanda titik atas menunjukkan turunan, titik atas satu berarti diturunkan satu kali. Persamaan dapat dituliskan ulang sebagai

$$u(t) = m\dot{v}(t) + bv(t)$$

$$u(t) = m \frac{dv}{dt}(t) + bv(t) \quad (2.3)$$

Persamaan (2.3) memiliki domain (t), untuk memperoleh dalam bentuk laplace (s) perlu ditransformasikan ke dalam bentuk domain frekuensi (s) dengan transformasi laplace sebagai

$$msV(s) + bV(s) = U(s) \quad (2.4)$$

Fungsi transformasi laplace secara umum adalah menghilangkan bagian integral dan diferensial. Jika terdapat bagian diferensial maka dikali dengan (s), jika terdapat bagian integral maka dibagi dengan (s).

Persamaan (2.4) dapat dituliskan ulang sebagai

$$V(s)(ms + b) = U(s) \quad (2.5)$$

Oleh karena itu dapat diperoleh persamaan sesuai dengan variabel input dan output yang telah ditentukan.

$$\frac{\text{output}}{\text{input}} = \frac{V(s)}{U(s)} = \frac{1}{ms + b} \quad (2.6)$$

Persamaan tersebut adalah model sistem kendaraan dalam bentuk fungsi alih.

Untuk menyusun model fungsi alih pada Scilab dapat digunakan perintah syslin atau lebih lengkapnya adalah

```
--> sys=syslin('dom', num, den)
```

Penjelasan Program

- dom : Terdapat beberapa domain yang digunakan yaitu 'c' untuk domain sistem waktu kontinyu, 'd' untuk domain waktu diskrit.
- num : Persamaan pada bagian pembilang
- den : Persamaan pada bagian penyebut

2.4. Respons Sistem

Sering kali, karakteristik kinerja sistem kendali ditentukan dengan respons transien terhadap masukan sinyal step, karena mudah untuk diperoleh. Respons transien sistem kendali sering menunjukkan osilasi teredam sebelum mencapai keadaan stabil. Karakteristik respons transien sistem kendali terhadap masukan step ditunjukkan pada Gambar 2.1. Karakteristik respons sistem secara umum adalah sebagai berikut,

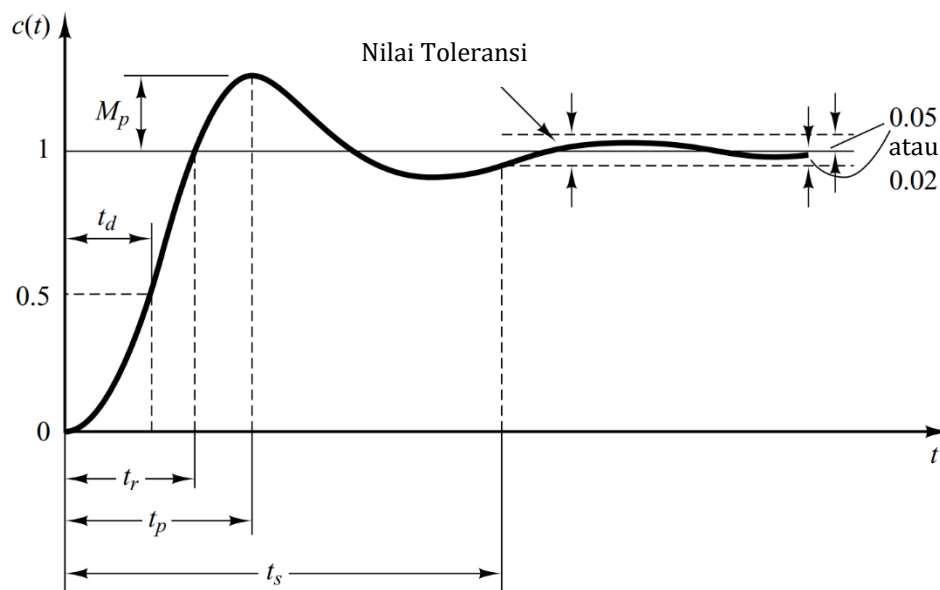
- a. Waktu tunda (Delay Time), t_d , adalah waktu yang dibutuhkan untuk respons mencapai setengah nilai final untuk pertama kali.,
- b. Waktu naik (Rise Time), t_r , adalah waktu yang diperlukan untuk respon naik dari 10% sampai 90%, 5% sampai 95%, 0% sampai 100% dari nilai final. Untuk sistem order 2 tidak teredam, waktu naik 0% sampai 100% biasanya digunakan. Untuk sistem yang teredam waktu naik 10% sampai 90% biasa digunakan.,
- c. Waktu puncak (Peak Time), t_p , adalah waktu yang diperlukan untuk respons mencapai puncak dari overshoot.,
- d. Overshoot maksimal, M_p , adalah nilai maksimum puncak lengkungan respons dihitung dari nilai satu. Jika nilai keadaan stabil respons kurang dari satu maka

umumnya digunakan nilai overshoot maksimal dengan persen. Nilai overshoot dalam persen adalah

$$M_p = \frac{c(t_p) - c(\infty)}{c(\infty)} \times 100\% \quad (2.7)$$

Dengan $c(t_p)$ adalah nilai overshoot maksimal dan $c(\infty)$ adalah nilai saat keadaan stabil. Jumlah maksimum overshoot mengindikasikan kestabilan sistem. Sistem stabil tidak memiliki overshoot lebih dari 10 persen.

- e. Waktu kestabilan (Settling Time), t_s , adalah waktu yang diperlukan untuk respons melengkung menuju dan tetap di rentang sekitar nilai final dalam spesifikasi persentase nilai final sebesar 2 sampai 5 persen.



Gambar 2.2. Respons Sistem

2.5. Mengamati Respon Sistem

Untuk mengamati respons sistem pada Scilab dapat digunakan perintah

```
--> y=csim('u', t, G,)
```

Penjelasan Program csim dapat dilihat dengan perintah help csim. Penjelasan singkat adalah

- u : Jenis masukan sistem dapat berupa 'step', 'impulse'.
- t : Variabel waktu
- G : Variabel fungsi alih

2.6. Pre Test

Jawablah pertanyaan berikut.

1. Apa itu model fungsi alih?
2. Tuliskan langkah-langkah untuk memperoleh model fungsi alih sistem kendaraan dengan input gaya dan output kecepatan linear.

3. Sebutkan apa saja karakteristik respons sistem beserta gambar dan penjelasannya!

2.7. Langkah Praktikum dan Post Test

Misal diketahui model fungsi alih sebagai

$$G(s) = \frac{\text{num}(s)}{\text{den}(s)} = \frac{25}{s^2 + 6s + 25} \quad (2.8)$$

Model fungsi alih persamaan tersebut pada Scilab, dapat dituliskan dengan Program 2.1.

Program 2.1.

```
s=%s
num=25
den=s^2 + 6*s + 25
G1=syslin('c', num, den)
```

1. Apakah model yang terbentuk pada console scilab sudah sama dengan model fungsi alih pada persamaan (2.8)?
 2. Tuliskan model fungsi alih yang ditampilkan pada Console Scilab.
 3. Tuliskan program tersebut pada lembar laporan lalu berikan analisis per baris pada sebelah kanan setiap perintah.
 4. Apa itu variabel "num" dan "den"? Dari mana variabel "num" dan "den" diperoleh?
 5. Apa fungsi perintah G1=syslin('c', num, den)?
- Catatan: s=%s memiliki fungsi seperti s=poly(0,"s") yaitu berfungsi untuk membuat persamaan polinomial

Grafik respon sistem pada persamaan (2.8) terhadap sinyal masukan step, dapat diketahui dengan Program 2.2.. Agar lebih mudah, bukalah editor Scilab dengan perintah edit, lalu tuliskan program tersebut pada editor. Tekan tombol save and execute untuk menjalankan program.

Program 2.2.

```
clear; clc;
s=%s
num=25;
den=s^2 + 6*s + 25;
G1=syslin('c', num, den);
t=0:0.1:10;
y=csim('step', t, G1)
plot(t,y)
xgrid
```

Perintah t=1:0.01:10 berfungsi untuk membuat array waktu simulasi dari detik 0 sampai 10 detik dengan nilai perbedaan array adalah 0,01.

6. Gambarkan grafik respons sistem untuk sistem pada persamaan (2.8) tersebut dan tuliskan bagian waktu naik, waktu tunda, waktu stabil, waktu naik dan overshoot. Hitunglah nilai respons sistem secara manual (menggunakan penggaris) berdasarkan grafik tersebut.
7. Apa fungsi perintah `G1=syslin('c', num, den)` pada Program 2.1?
8. Apa fungsi perintah `plot(t,y)` dan `xgrid`?

9. Diketahui model sistem kendaraan dalam bentuk fungsi alih adalah

$$\frac{\text{output}}{\text{input}} = \frac{V(s)}{U(s)} = \frac{1}{ms + b} \quad (2.9)$$

Dengan nilai $m = 1000$ dan $b = 50$, tuliskan model fungsi alih sistem tersebut.

10. Gambarkan grafik respons sistem beserta dengan bagian-bagian respons sistem untuk sistem pada persamaan (2.9) dengan waktu simulasi $t=0:0.1:300$.

2.8. Menghitung Respons Sistem

Diketahui fungsi alih suatu sistem sebagai

$$G = \frac{25}{s^2 + 6s + 25} \quad (2.10)$$

Untuk menghitung nilai waktu naik (rise time), waktu puncak (peak time) dan waktu kestabilan (settling time) dapat dilakukan dengan prosedur sebagai berikut. Bukalah editor Scilab dengan mengetikkan perintah edit. Lalu, Ketikkan Program 2.3 yang berfungsi untuk menghitung nilai respons sistem.

Program 2.3.

```
clear; clc;

s=%s;
num = 25;
den = 1*s^2 + 6*s + 25;
G=syslin('c',num,den)

printf("Model Fungsi Alih")
disp(G)

t = 0:0.005:5;
y = csim('step',t,G);
r = 1;

while y(r) < 1.0001
    r = r + 1;
end;

rise_time = (r - 1)*0.005;
printf("\n rise time = %f \n", rise_time)

[y_max,tp] = max(y);
```

```

peak_time = (tp - 1)*0.005;
printf("\n peak time = %f\n", peak_time)

max_overshoot = ymax-1
printf("\n max overshoot = %f\n", max_overshoot)

ii = 1001;
while y(ii) > 0.98 & y(ii) < 1.02
    ii = ii - 1;
end;

settling_time = (ii - 1)*0.005;
printf("\n settling time = %f\n", settling_time)

plot(t,y)
xgrid

```

11. Tuliskan berapa nilai rise time, peak time, dan settling time. Gambarlah grafik respons sistem yang terbentuk. Bandingkan apakah nilai rise time, peak time dan settling time sama dengan pengukuran secara manual.

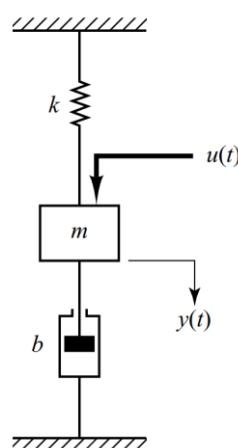
Diketahui persamaan sistem pegas dan peredam pada Gambar 2.3 adalah

$$m\ddot{y} + b\dot{y} + ky = u \quad (2.11)$$

Dengan m adalah massa, b adalah konstanta peredam, k adalah konstanta pegas, dan u adalah gaya dorong. Model fungsi alih dari sistem tersebut dengan input gaya dorong $U(s)$ dan output posisi $Y(s)$ adalah

$$\frac{Y(s)}{U(s)} = \frac{1}{ms^2 + bs + k} \quad (2.12)$$

12. Tuliskan proses untuk mendapatkan model fungsi alih tersebut.



Gambar 2.3. Sistem Pegas dan Peredam

13. Gambarkan grafik respons sistem beserta bagian-bagian respons sistem (rise time, peak time, settling time, overshoot) pada sistem pegas dan peredam jika diketahui nilai $m = 100$, $b = 30$, $k = 60$.

Praktikum DSK Unit 3 Model Fungsi Alih Motor DC dengan Scilab

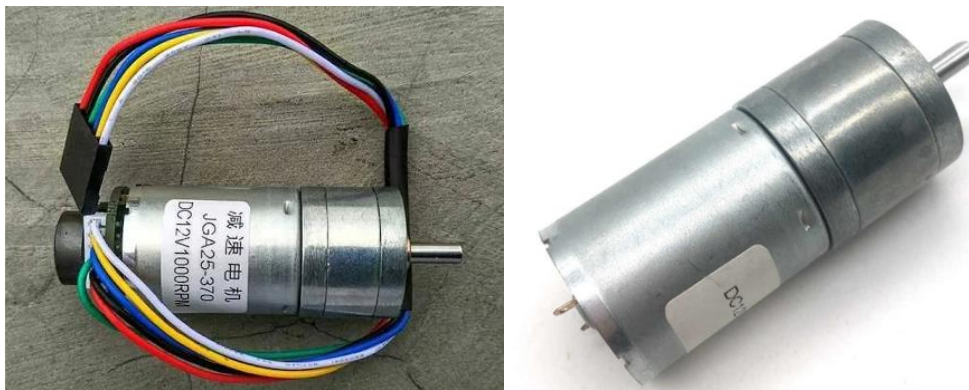
oleh Alfian Ma'arif, M. Eng.

3.1. Tujuan Praktikum

Tujuan praktikum adalah mahasiswa memahami tentang penurunan model fungsi alih motor DC dan memahami tentang karakteristik Motor DC berdasarkan pada respons sistem.

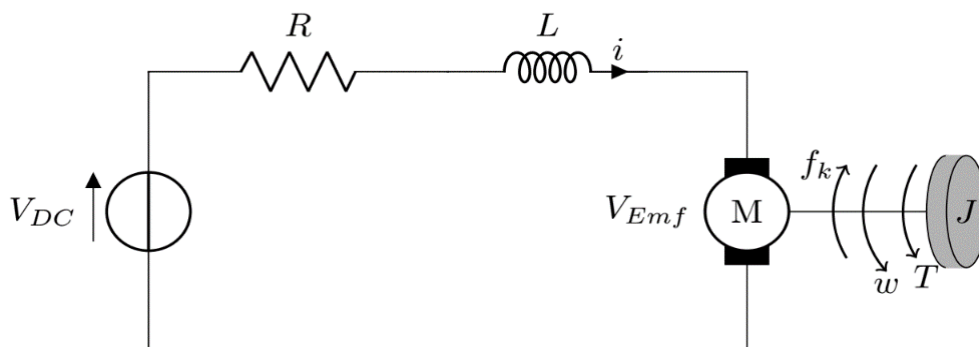
3.2. Motor DC

Motor Arus Searah atau disebut dengan Motor DC (Dirrect Current) adalah perangkat listrik yang berfungsi untuk mengubah energi listrik menjadi energi mekanik. Energi listrik yang digunakan adalah arus searah. Masukan Motor DC adalah tegangan DC dan keluaran Motor DC adalah posisi sudut, kecepatan sudut dan torsi. Motor DC memiliki aplikasi yang luas dalam bidang robotika dan industri. Contoh Motor DC ditunjukkan pada Gambar 3.1.



Gambar 3. 1. Diagram Motor DC (kiri) dengan sensor encoder (kanan) tanpa sensor encoder

Pada Gambar 3.1, terdapat 2 jenis motor DC, motor DC yang pertama memiliki sensor encoder yang sudah terpasang pada ujung belakang sementara Motor DC yang kedua tanpa sensor encoder. Kedua motor DC tersebut memiliki spesifikasi tegangan DC 12volt. Fungsi sensor encoder adalah untuk mengukur posisi sudut motor dan mengukur kecepatan sudut motor. Penampang Motor DC ditunjukkan pada Gambar 3.2.



Gambar 3. 2. Diagram Motor DC

Keterangan Gambar 3.2 adalah sebagai berikut, variabel R adalah nilai resistansi, L adalah nilai induktansi, V_{DC} adalah tegangan catu daya, i adalah arus yang mengalir dalam rangkaian, J adalah momen inersia motor, ω adalah kecepatan sudut motor. Variabel T adalah torsi motor yang berhubungan dengan arus sebagai

$$T = K_t i \quad (3.1)$$

Dengan K_t adalah konstanta torsi motor.

Variabel f_k adalah gaya gesek antara bagian rotor dan stator yang berhubungan dengan kecepatan sudut sebagai

$$f_k = K_b \omega \quad (3.2)$$

Dengan K_b adalah konstanta gaya gesek. Variabel V_{Emf} adalah tegangan balik motor yang berhubungan dengan kecepatan sudut

$$V_{Emf} = K_e \omega \quad (3.3)$$

Dengan K_e adalah konstanta tegangan balik.

Untuk memperoleh model fungsi alih motor DC, dilakukan analisis secara mekanik menggunakan Hukum Newton II tentang rotasi dan analisis secara elektrik menggunakan Hukum Kirchoff tentang tegangan.

Berdasarkan Hukum Newton II tentang rotasi diperoleh

$$\begin{aligned} \sum \tau &= I \alpha \\ T - f_k &= J \dot{\omega} \\ K_t i - K_b \omega &= J \dot{\omega} \end{aligned} \quad (3.4)$$

Berdasarkan Hukum Kirchoff tentang tegangan diperoleh

$$\begin{aligned} V_{beban} &= V_{sumber} \\ V_R + V_L + V_{Emf} &= V_{DC} \\ Ri + L \frac{di}{dt} + K_e \omega &= V_{DC} \end{aligned} \quad (3.5)$$

Oleh karena itu dapat diperoleh persamaan karakteristik Motor DC berdasarkan pada Persamaan (3.4) dan (3.5) sebagai

$$K_t i - K_b \omega = J \dot{\omega} \quad (3.6)$$

$$Ri + L \frac{di}{dt} + K_e \omega = V_{DC} \quad (3.7)$$

Kedua persamaan tersebut digunakan untuk membuat model fungsi alih motor DC. Prosedur pertama adalah kedua persamaan tersebut ditransformasi laplace sehingga diperoleh

$$K_t I(s) - K_b W(s) = JsW(s) \quad (3.8)$$

$$RI(s) + LsI(s) + K_e W(s) = V_{DC} \quad (3.9)$$

Dengan menggunakan pindah silang pada persamaan (3.8) dan (3.9), dapat diperoleh

$$K_t I(s) = (Js + K_b)W(s) \quad (3.10)$$

$$I(s) = \frac{V_{DC} - K_e W(s)}{(R + Ls)} \quad (3.11)$$

Lalu, persamaan (3.11) disubstitusikan ke persamaan (3.10), lalu dilanjutkan dengan proses pindah silang dalam bentuk rasio input dan output, sehingga diperoleh

$$\begin{aligned} K_t \frac{V_{DC} - K_e W(s)}{(R + Ls)} &= (Js + K_b)W(s) \\ K_t [V_{DC} - K_e W(s)] &= (R + Ls)(Js + K_b)W(s) \\ K_t V_{DC} - K_t K_e W(s) &= (R + Ls)(Js + K_b)W(s) \\ K_t V_{DC} &= (R + Ls)(Js + K_b)W(s) + K_t K_e W(s) \\ K_t V_{DC} &= [(R + Ls)(Js + K_b) + K_t K_e]W(s) \end{aligned} \quad (3.12)$$

Oleh karena itu dapat diperoleh model fungsi alih sebagai

$$V_{DC}(s) \frac{K_t}{[(R + Ls)(Js + K_b) + K_t K_e]} = W(s) \quad (3.13)$$

Atau

$$\frac{W(s)}{V_{DC}(s)} = \frac{K_t}{[(R + Ls)(Js + K_b) + K_t K_e]} \quad (3.14)$$

3.3. Pre Test

Jawablah pertanyaan berikut.

1. Apa itu Motor DC?
2. Gambarkan penampang motor DC beserta keterangannya.
3. Apa nama sensor yang digunakan untuk mengukur kecepatan sudut Motor DC?
4. Tuliskan proses untuk memperoleh model fungsi alih sistem Motor DC.

3.4. Langkah Praktikum dan Post Test

Kerjakan langkah praktikum berikut. Diketahui nilai parameter Motor DC ditunjukkan pada Tabel 3.1.

Tabel 3. 1. Nilai Parameter Motor DC

Nama Parameter	Nilai Parameter
Resistansi (R)	1
Induktansi (L)	0.5
Konstanta gaya gesek (K_b)	0.1
Konstanta torsi motor (K_t)	0.01
Konstanta tegangan balik (K_e)	0.01
Momen inersia (J)	0.01

Model fungsi alih motor DC adalah

$$\frac{W(s)}{V_{DC}(s)} = \frac{K_t}{[(R + Ls)(Js + K_b) + K_t K_e]} \quad (3.15)$$

1. Dengan menggunakan persamaan (3.15) hitunglah model fungsi alih motor DC dengan memasukkan nilai parameter pada Tabel 3.1..
2. Jika Anda memiliki sebuah motor DC, lalu diberikan tegangan 12 volt pada kedua kutubnya, apa yang terjadi pada motor DC tersebut? Bagaimana karakteristik kecepatan putarannya?

Buka kembali petunjuk Praktikum unit 2 untuk mengerjakan soal berikut.

3. Bukalah editor Scilab dan buatlah program untuk membuat model fungsi alih motor DC pada Scilab. Tuliskan program tersebut pada laporan dengan penjelasan setiap baris.
4. Tuliskan program untuk membuat grafik respons sistem fungsi alih motor DC dari detik ke nol sampai detik ke 10 ($t=0:0.1:10$) dan Gambarkan grafik tersebut pada laporan. Berikan penjelasan setiap baris pada program tersebut.

Grafik tersebut merupakan grafik kecepatan sudut motor DC dari detik nol sampai detik tertentu. Terlihat pada grafik tersebut bahwa kecepatan motor DC di detik nol adalah nol, lalu kecepatan terus naik dan stabil pada nilai tertentu. Grafik tersebut memiliki karakteristik yang sama ketika motor DC diberikan tegangan yaitu berputar dari kecepatan sudut nol sampai stabil pada kecepatan tertentu.

5. Hitunglah nilai rise time, setting time, peak time dan overshoot pada grafik tersebut secara manual menggunakan penggaris (tanpa program). Tuliskan nilai nol atau tidak ada, jika nilainya sama dengan nol atau tidak ada.

Diketahui persamaan motor DC adalah

$$K_t i - K_b \dot{\theta} = J \ddot{\theta} \quad (3.16)$$

$$Ri + L \frac{di}{dt} + K_e \dot{\theta} = V_{DC} \quad (3.17)$$

6. Tuliskan proses untuk mendapatkan model fungsi alih persamaan (3.18) dengan input adalah tegangan dan output adalah posisi sudut.

$$\frac{\theta(s)}{V_{DC}(s)} = \frac{K_t}{s[(R + Ls)(Js + K_b) + K_t K_e]} \quad (3.18)$$

7. Gambarkan respons sistem fungsi alih persamaan (3.18) beserta dengan nilai rise time, settling time, peak time dan overshoot secara manual.

Praktikum DSK Unit 4 Pengenalan Aplikasi XCOS Scilab

oleh Alfian Ma'arif, M. Eng.

4.1. Tujuan Praktikum

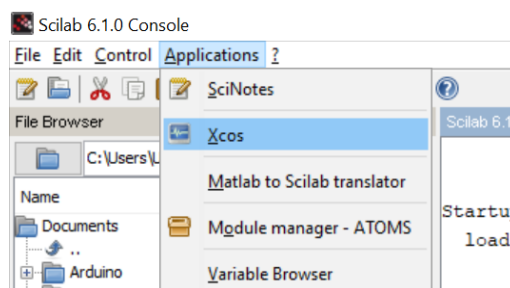
Tujuan praktikum adalah mahasiswa memahami tentang aplikasi xcos scilab dan membuat simulasi sederhana dengan xcos.

4.2. Xcos Scilab

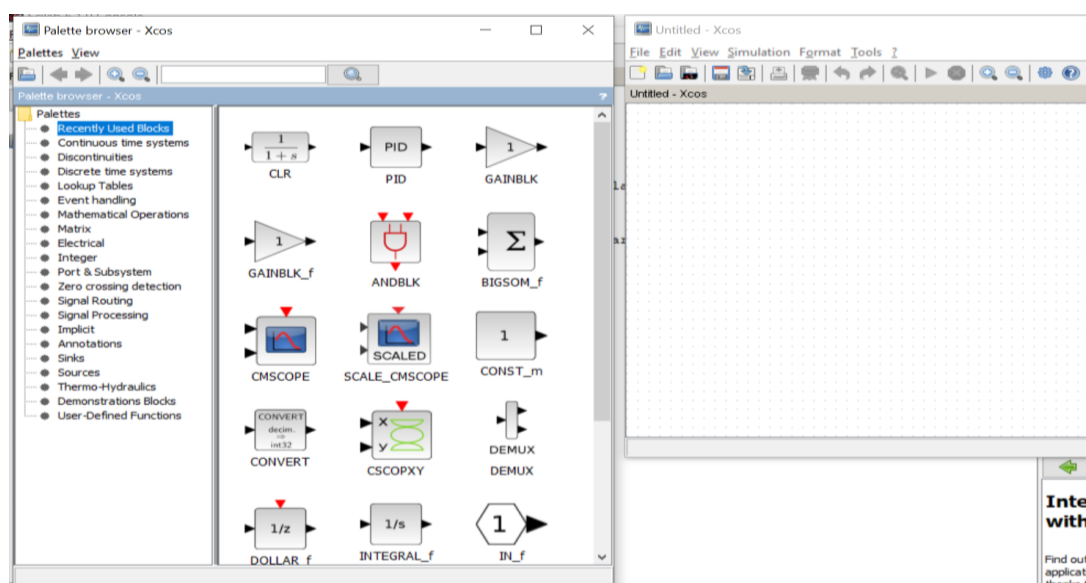
Xcos scilab adalah salah satu aplikasi dari scilab yang berfungsi untuk membuat simulasi sistem dalam bentuk blok-blok. Aplikasi ini sama dengan Simulink yang dimiliki oleh Matlab. Aplikasi ini digunakan dalam sistem kendali untuk membuat model sistem, mengendalikan sistem dan mengetahui karakteristik dinamik sistem.

Untuk membuka aplikasi Scilab, ketikkan perintah **xcos** pada console scilab untuk membuka fitur xcos Scilab. Halaman xcos juga dapat dibuka dengan menggunakan menu scilab bagian application, perhatikan Gambar 4.1. Tampilan xcos ditunjukkan pada Gambar 4.2. Terdapat 2 bagian pada xcos yaitu bagian palettes yang berisi blok-blok yang berfungsi untuk membuat sistem dan halaman kerja yang berisi sistem yang dirancang.

--> XCOS



Gambar 4. 1. Xcos Scilab



Gambar 4. 2. Tampilan xcos Scilab

Di dalam jendela palettes terdapat beberapa folder yang berisi blok. Blok-Blok ini dikelompokkan sesuai dengan foldernya. Beberapa folder yang terdapat pada xcos adalah sebagai berikut.

1. Folder recently used blocks: Berisi tentang blok yang baru saja ditambahkan.
2. Folder continuous time systems: Berisi tentang blok sistem waktu kontinu seperti blok fungsi alih, blok model state space dan blok PID.
3. Folder Sinks: Berisi tentang blok yang berfungsi untuk menampilkan grafik.
4. Folder Source: Berisi tentang blok yang berfungsi untuk menghasilkan sinyal masukan seperti blok step dan ramp.

4.3. Pre Test

Jawablah pertanyaan dan tuliskan jawaban pertanyaan berikut pada lembar laporan.

1. Apa itu aplikasi xcos scilab?
2. Sebutkan 4 folder yang ada di dalam palettes xcos beserta nama blok di dalamnya minimal 2 buah blok.

4.4. Langkah Praktikum dan Post Test

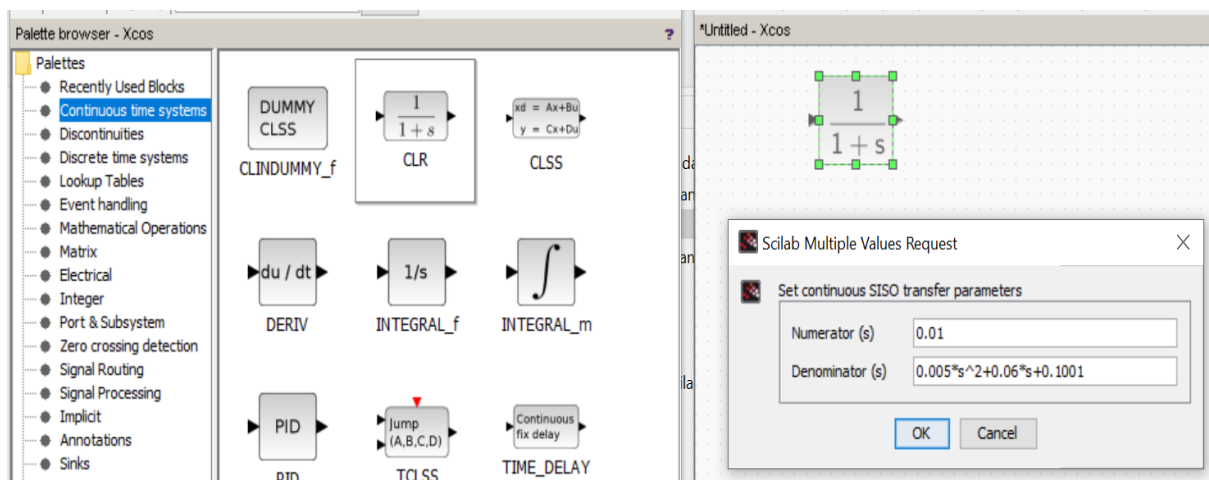
Ikutilah langkah praktikum berikut.

Membuat Simulasi Open Loop Sistem

Buka palettes pada bagian Continuous time systems lalu masukkan Blok CLR untuk membuat fungsi alih motor DC dengan cara klik dua kali pada blok CLR atau tarik blok tersebut ke halaman kerja. Klik dua kali blok CLR masukkan nilai numerator dan denominator dari fungsi alih motor DC sebagai

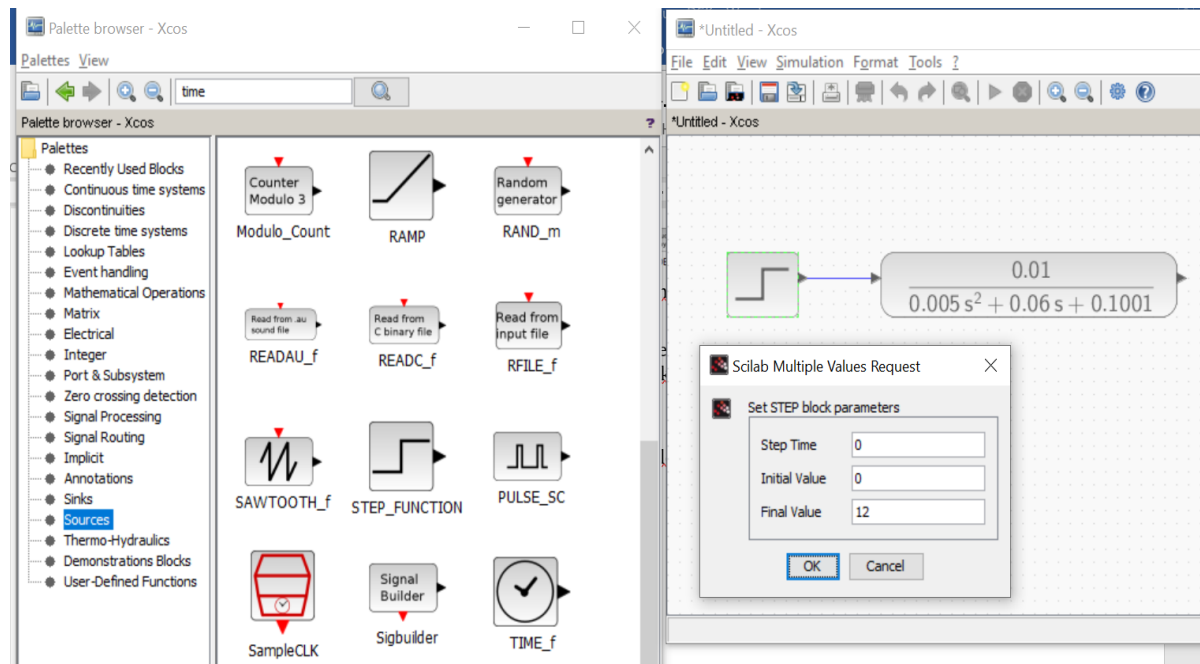
$$W(s) = \frac{0.01}{[0.005s^2 + 0.06s + 0.1001]} V_{DC}(s)$$

Perhatikan Gambar 4.3 untuk mengetahui proses memasukkan blok CLR.



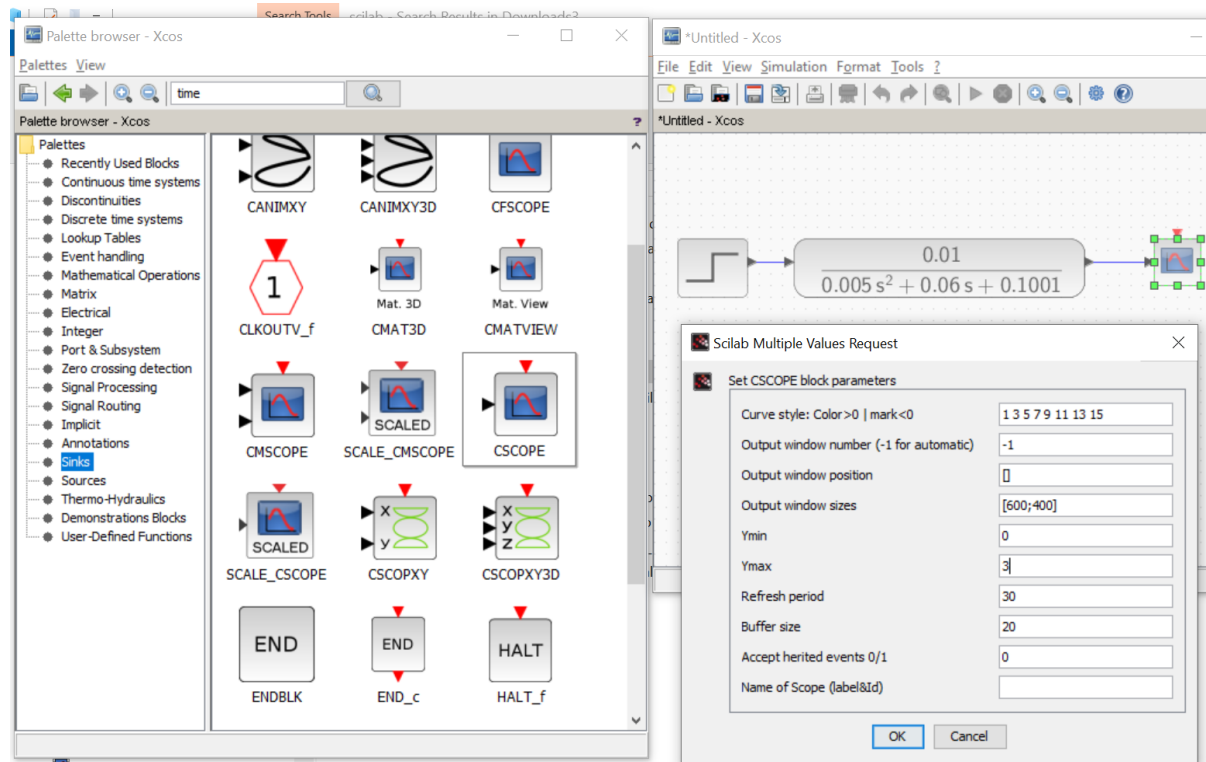
Gambar 4. 3. Memasukkan blok CLR sebagai blok fungsi alih

Buka palettes pada bagian folder Sources dan masukkan blok step_function ke halaman kerja. Klik dua kali blok step_function dan masukkan nilai 0 step time dan initial value, dan nilai 12 pada final value. Lalu hubungkan blok step_function dengan blok fungsi alih. Perhatikan Gambar 4.4.



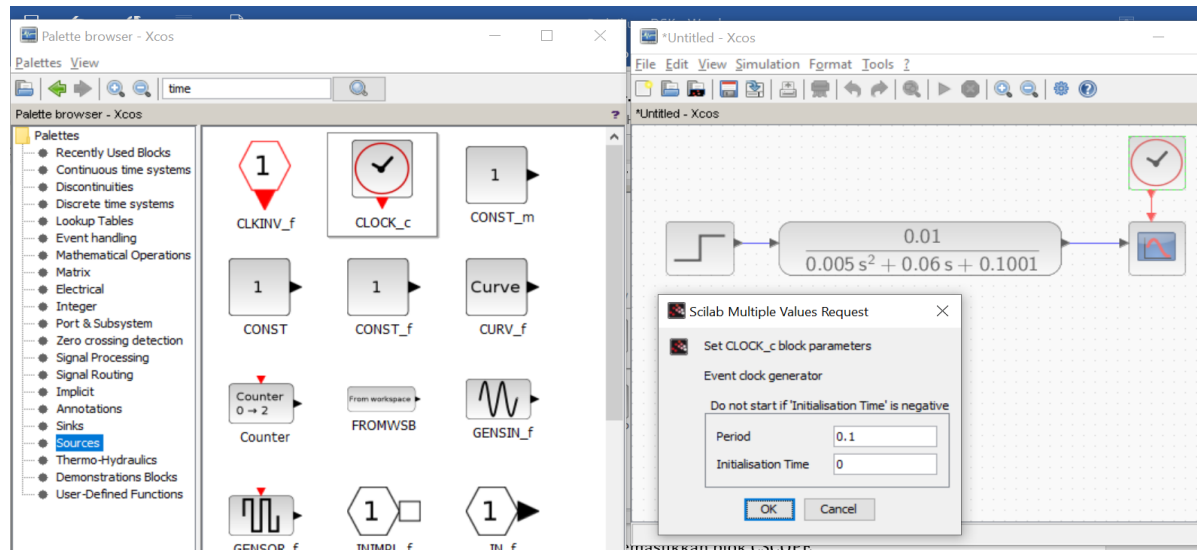
Gambar 4. 4. Memasukkan blok fungsi step sebagai nilai referensi

Buka palettes pada folder Sinks, masukkan blok CSCCOPE dan klik pada blok tersebut. Isikan nilai Ymin adalah 0 dan Ymax adalah 2 (Tuliskan 2 walau pada Gambar tertera 3). Lalu hubungkan dengan fungsi alih. Perhatikan Gambar 4.5.



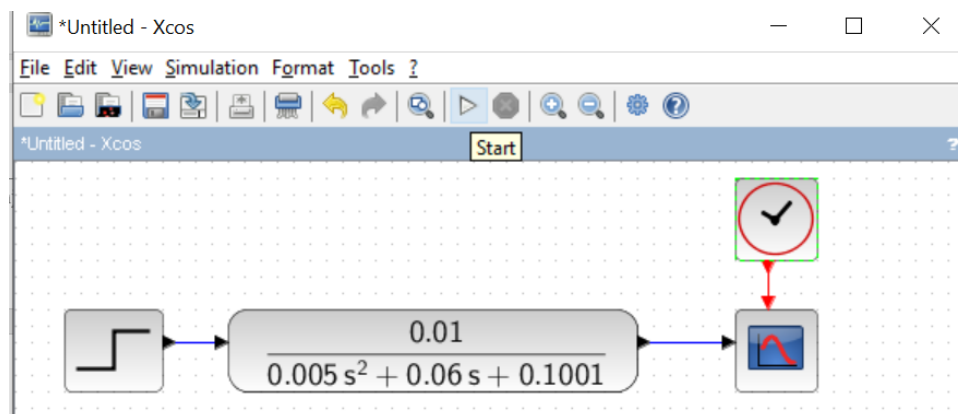
Gambar 4. 5. Memasukkan blok CSCCOPE

Buka palettes pada folder Sources dan masukkan Blok CLOCK_c. Klik dua kali pada blok Clock_c dan masukkan nilai 0 pada Initialization Time lalu klik OK. Hubungkan blok Clock_c dengan Blokc Cscope. Perhatikan Gambar 4.6.

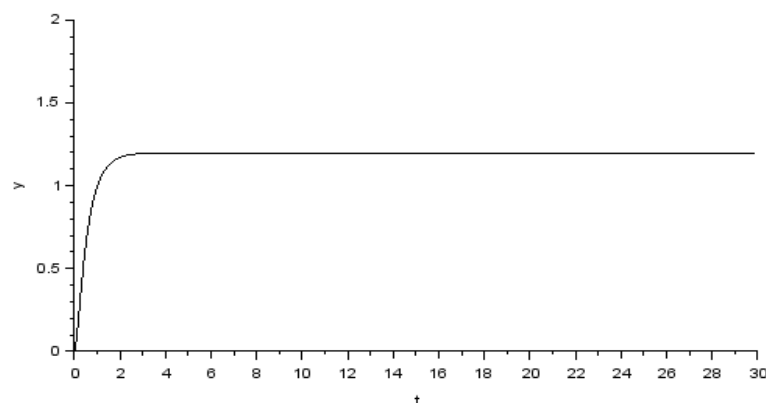


Gambar 4. 6. Memasukkan blok Clock_c

Tekan tombol Start dengan simbol segitiga untuk menjalankan simulasi, perhatikan pada Gambar 4.7. Hasil grafik yang terbentuk ditunjukkan pada Gambar 4.8.

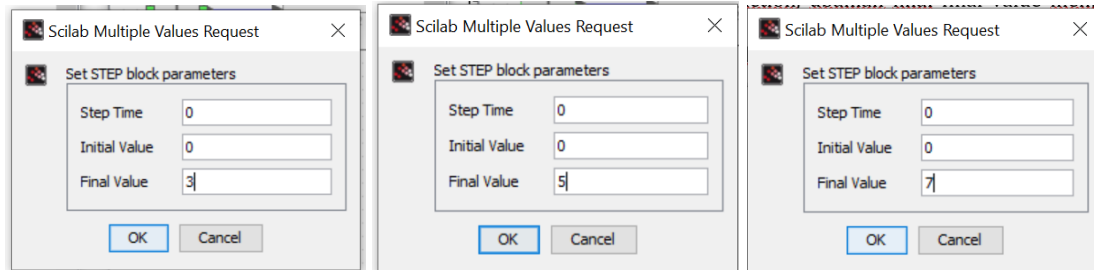


Gambar 4. 7. Menjalankan simulasi pada xcos



Gambar 4. 8. Grafik hasil simulasi

1. Gambarkan grafik yang terbentuk pada laporan.
2. Klik dua kali blok `step_function`, ubahlah nilai final value menjadi 3, lalu jalankan simulasi. Gambarkan grafik yang terbentuk. Lakukan proses yang sama untuk nilai final value dengan nilai 5, 7, dan 9 dalam satu grafik. Perhatikan Gambar 4.9.

Gambar 4. 9. Variasi nilai `step_function`

3. Nilai 3, 5, 7, dan 9 pada final value blok `step_function` tersebut menurut Anda merupakan nilai Apa? Lalu, grafik yang ditampilkan tersebut merupakan grafik Apa? Coba hubungkan dengan persamaan fungsi alih sistem motor DC.
4. Dengan prosedur yang sama dengan simulasi open loop motor dc, buatlah simulasi open loop untuk sistem pegas dan bamper dengan nilai final value blok `step_function` sebesar 3. Fungsi alih sistem pegas dan bamper adalah

$$\frac{Y(s)}{U(s)} = \frac{1}{(100s^2 + 30s + 60)} \quad (2.13)$$

Ubahlah nilai $Y_{\max}$ dari blok `CScope` menjadi 1; 0,5 dan 0,1 kemudian amati hasil grafiknya. Apa fungsi bagian $Y_{\max}$ pada blok `CScope` terhadap hasil grafik yang terbentuk.

5. Gambarkan grafik open loop sistem pegas dan bamper pada laporan.
6. Ubahlah nilai final value pada blok `unit_step` menjadi 5; 10 dan 15. Amati apa perbedaan pada grafik yang terbentuk. Ubahlah nilai $Y_{\max}$ menjadi lebih besar jika grafik tidak terlihat.
7. Nilai 3, 5, 10, dan 15 pada final value blok `step_function` tersebut menurut Anda merupakan nilai Apa? Lalu, grafik yang ditampilkan tersebut merupakan grafik Apa? Coba hubungkan dengan persamaan fungsi alih sistem pegas dan bamper.
8. Carilah pengertian tentang simulasi open loop pada buku referensi. Apa fungsi dari simulasi open loop tersebut? Apakah sistem motor DC dan sistem pegas bamper merupakan sistem yang stabil? Berikan penjelasan pada jawaban yang dipilih.

Praktikum DSK Unit 5 Kendali Proportional Integral Derivatif (PID) Motor DC dengan XCOS

oleh Alfian Ma'arif, M. Eng.

5.1. Tujuan Praktikum

Tujuan praktikum adalah mahasiswa memahami tentang aplikasi xcossclab dan membuat simulasi sederhana dengan xcoss.

5.2. Dasar Teori

Terdapat dua dasar teori pada bagian berikut. Dasar teori pertama adalah dasar teori tentang Motor DC dan dasar teori kedua adalah tentang pengendali proporsional Integral Derivatif atau biasa disebut sebagai pengendali PID.

5.2.1. Motor DC

Motor DC merupakan salah satu aktuator yang sering digunakan dalam industri, robotika dan contoh sistem kendali. Hal tersebut disebabkan karena sistem tersebut sederhana dan tidak terlalu susah untuk dikendalikan. Walaupun begitu sistem ini sangat baik digunakan untuk mempelajari sistem kendali selain karena banyak juga digunakan di industri dan robotika.

Contoh sistem di industri yang menggunakan motor DC adalah konveyor. Sementara pada bidang robotika motor DC merupakan aktuator pada robot line follower, line maze solving robot, balancing robot dan sebagainya.

Model motor DC dalam fungsi alih dengan masukan tegangan dan keluaran kecepatan sudut dapat dituliskan sebagai

$$W(s) = \frac{0.01}{[0.005s^2 + 0.06s + 0.1001]} V_{DC}(s)$$

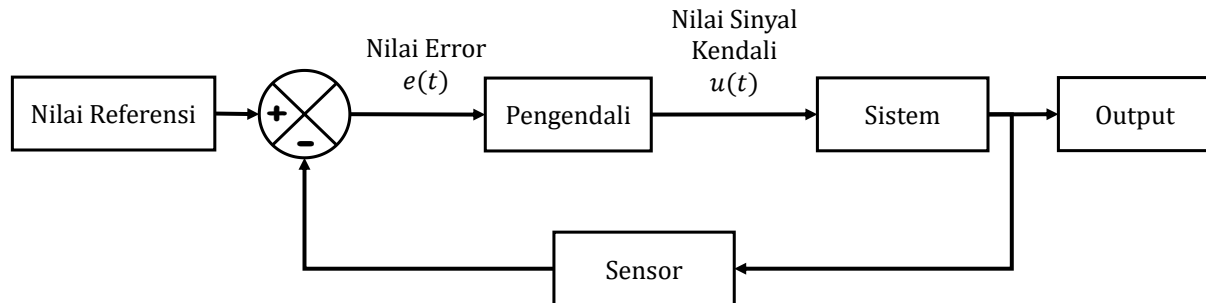
5.2.2. Pengendali Proporsional Integral Derivatif (PID)

Sebuah pengendali berfungsi untuk mengendalikan suatu sistem, memperbaiki respons sistem dan memperbaiki kinerja sistem agar dapat bekerja dengan baik. Salah satu jenis pengendali adalah Pengendali Proporsional Integral Derivatif atau bisa disebut dengan Pengendali PID adalah pengendali yang terdiri atas pengendali Proporsional, Pengendali Integral dan Pengendali Derivatif. Ketiga pengendali tersebut memiliki kelebihan dan kekurangan masing-masing yang mempengaruhi respons sistem seperti rise time, settling time dan overshoot. Gabungan ketiga pengendali tersebut membuat suatu pengendali yang saling meniadakan kekurangan sehingga didapat pengendali dengan kelebihan dari masing-masing pengendali.

Pengendali PID adalah pengendali yang paling banyak digunakan di industri dan robotika sampai saat ini. Padahal usia pengendali tersebut sudah lebih dari 1 abad, yaitu telah ditemukan pada Tahun 1922. Hal tersebut disebabkan karena pengendali PID mudah untuk dimengerti, mudah untuk dipahami, dan memiliki struktur yang sederhana namun bisa memberikan kinerja sistem yang baik.

Diagram Blok Sistem Kendali

Diagram blok sistem kendali umumnya menggunakan sistem kendali kalang tertutup. Diagram blok sistem kendali ditunjukkan pada Gambar 5.1. Nilai error adalah selisih antara nilai error dengan nilai dari sensor (nilai terukur). Nilai sinyal kendali adalah nilai keluaran dari pengendali. Output adalah nilai yang dikendalikan. Sensor berfungsi untuk mengukur variabel yang dikendalikan.



Gambar 5. 1. Diagram Blok Sistem Kendali

Pengendali Proporsional

Persamaan Pengendali Proporsional adalah

$$u(t) = K_p e(t)$$

Dengan K_p adalah konstanta proporsional, $u(t)$ adalah sinyal kendali dan $e(t)$ adalah nilai error antara nilai referensi dan nilai umpan balik. Esensi dari pengendali proporsional adalah mengalikan semua nilai error dengan suatu penguatan sehingga diperoleh nilai error yang lebih besar. Jika nilai error bernilai negatif maka nilai sinyal kendalinya juga semakin negatif (semakin besar). Kebalikannya, jika nilai error bernilai positif maka nilai sinyal kendali juga akan semakin positif (semakin besar).

Pengendali Proporsional Integral

Persamaan Pengendali Proporsional Integral adalah

$$u(t) = K_p e(t) + \frac{K_p}{T_i} \int e(t)$$

Dengan T_i adalah konstanta waktu integral. Arti bagian integral tersebut adalah pengendali integral menjumlahkan semua nilai error dari pertama kali sistem dijalankan sampai sistem dimatikan.

Pengendali tersebut dapat dituliskan dalam bentuk yang lebih familiar sebagai

$$u(t) = K_p e(t) + K_i \int e(t)$$

Dengan

$$K_i = \frac{K_p}{T_i}$$

Pengendali Proporsional Derivatif

Persamaan Pengendali Proporsional Derivatif adalah

$$u(t) = K_p e(t) + K_p T_d \frac{de(t)}{dt}$$

Dengan T_d adalah konstanta waktu Derivatif. Arti dari bagian diferensial tersebut adalah pengendali derivatif mencari nilai selisih antara nilai error sekarang dengan nilai error sebelumnya.

Pengendali tersebut dapat dituliskan dalam bentuk yang lebih familiar sebagai

$$u(t) = K_p e(t) + K_d \frac{de(t)}{dt}$$

Dengan

$$K_d = K_p T_d$$

Pengendali Proporsional Integral Derivatif

Persamaan pengendali PID adalah

$$u(t) = K_p e(t) + \frac{K_p}{T_i} \int e(t) + K_p T_d \frac{de(t)}{dt}$$

Pengendali PID dapat dituliskan dalam bentuk lain sebagai

$$u(t) = K_p e(t) + K_i \int e(t) + K_d \frac{de(t)}{dt}$$

Dengan

$$K_i = \frac{K_p}{T_i} \quad K_d = K_p T_d$$

5.3. Pre Test

Jawablah pertanyaan berikut.

1. Jelaskan dan Gambarkan diagram blok sistem kendali kalang tertutup?
2. Apa fungsi sebuah Pengendali?
3. Apa itu Pengendali PID?
4. Apa Kelebihan Pengendali PID?
5. Tuliskan Persamaan Pengendali PID.

5.4. Langkah Percobaan

Ketikkan Perintah xcos untuk membuka jendela xcos untuk membuat simulasi pengendali Motor DC.

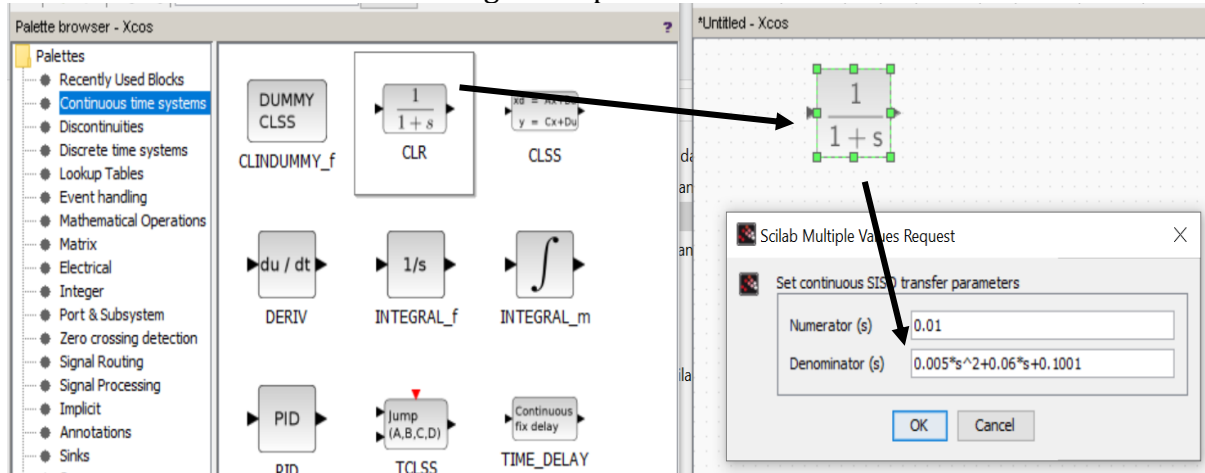
--> xcos

Pengendali Proporsional Motor DC

Buka palettes pada bagian Continuous time systems lalu masukkan Blok CLR untuk membuat fungsi alih motor DC dengan cara klik dua kali pada blok CLR atau tarik blok tersebut ke halaman kerja. Klik dua kali blok CLR masukkan nilai numerator dan denominator dari fungsi alih motor DC sebagai

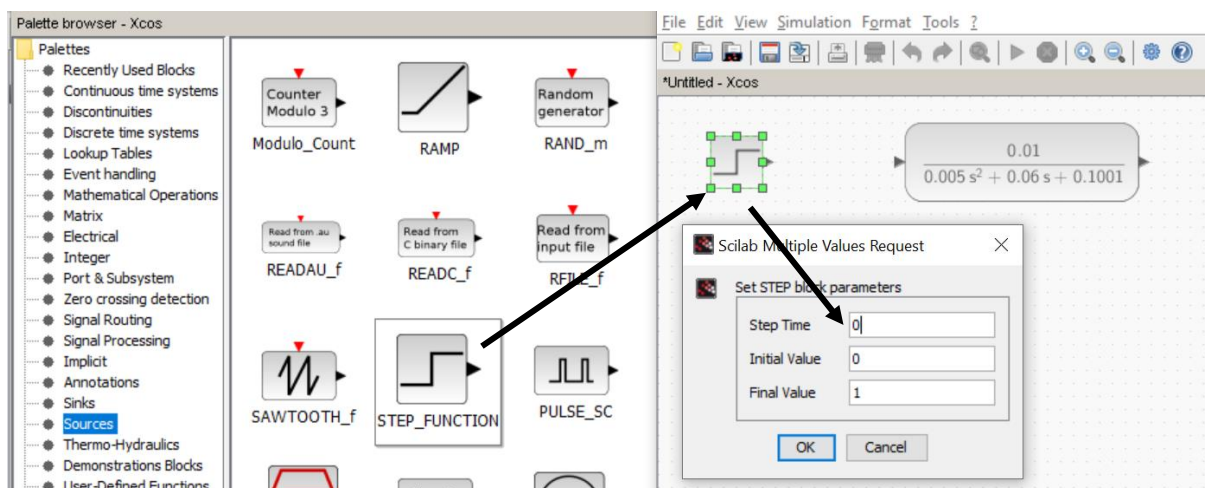
$$W(s) = \frac{0.01}{[0.005s^2 + 0.06s + 0.1001]} V_{DC}(s)$$

Perhatikan Gambar 5.2 untuk mengetahui proses memasukkan blok CLR.



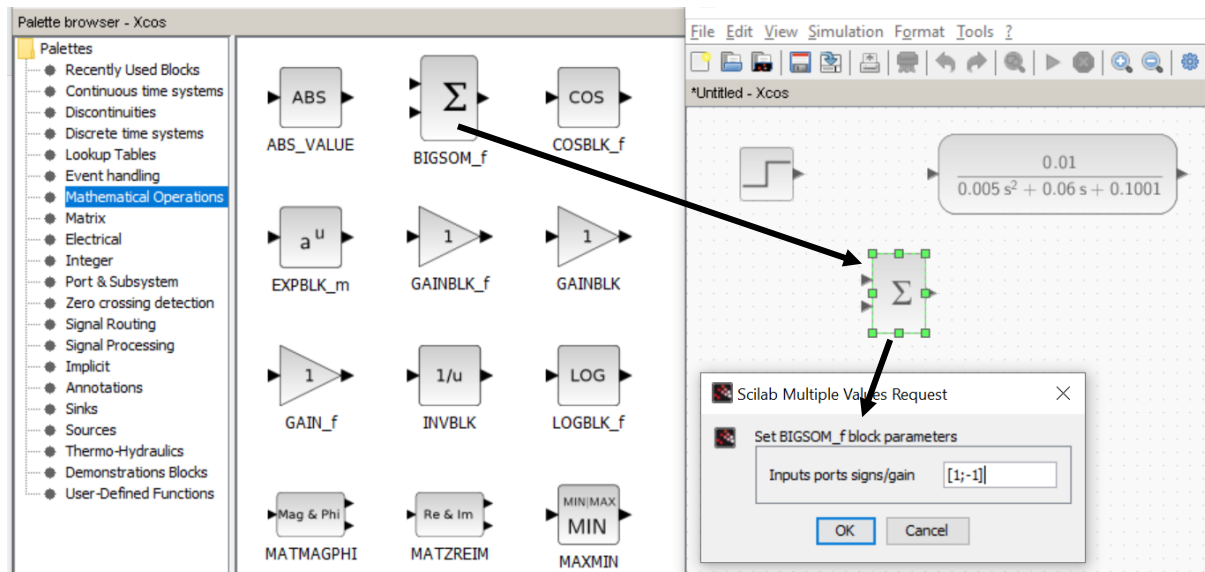
Gambar 5. 2. Memasukkan blok CLR sebagai blok fungsi alih

Buka palettes pada bagian folder Sources dan masukkan blok step_function ke halaman kerja. Klik dua kali blok step_function dan masukkan nilai 0 step time dan initial value, dan nilai 12 pada final value. Lalu hubungkan blok step_function dengan blok fungsi alih. Perhatikan Gambar 5.3.



Gambar 5. 3. Memasukkan blok fungsi step sebagai nilai referensi

Buka palettes pada bagian folder Mathematical Operations dan masukkan blok BIGSOM_f ke halaman kerja. Klik dua kali blok BIGSOM_f dan masukkan nilai [1;-1]. Perhatikan Gambar 5.4.

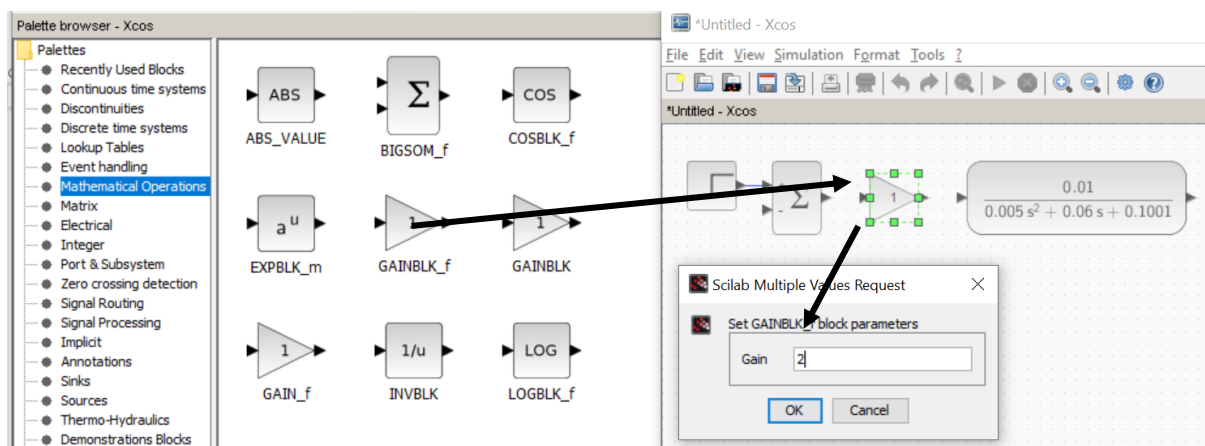


Gambar 5. 4. Memasukkan blok sum untuk menghitung nilai Error

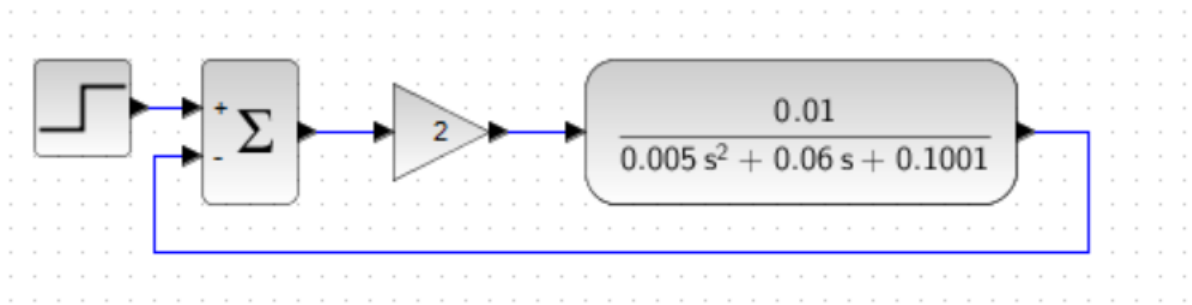
Persamaan Pengendali Proporsional adalah

$$u(t) = K_p e(t)$$

Untuk membuat persamaan tersebut, nilai K_p dapat diwakili dengan blok penguat (gain) pada xcos. Prosedurnya adalah sebagai berikut. Buka palettes pada bagian folder Mathmeatical Operations dan masukkan blok GAINBLK_f ke halaman kerja. Klik dua kali blok GAINBLK_f dan masukkan nilai 2. Lalu hubungkan blok GAINBLK_f dengan BIGSOM_f dan blok fungsi alih. Perhatikan Gambar 5.5 dan Gambar 5.6.

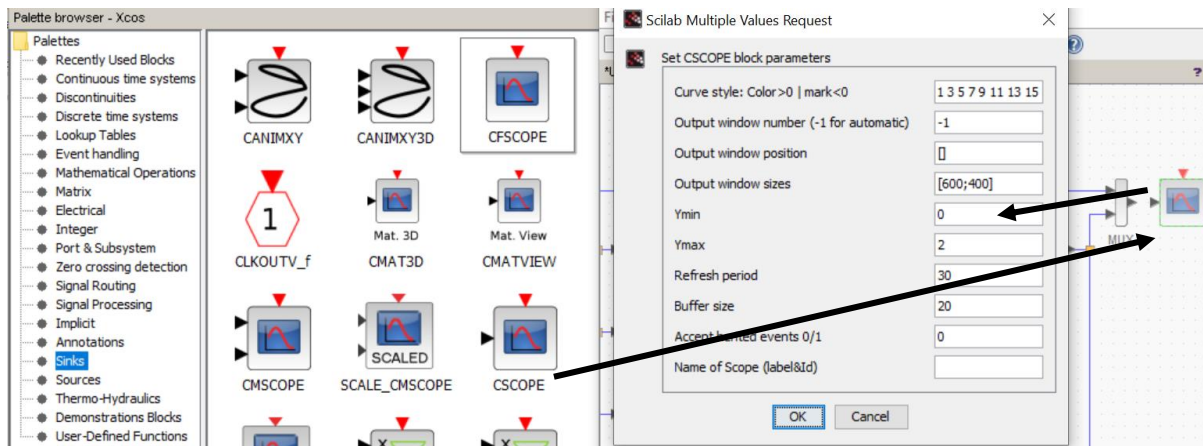


Gambar 5. 5. Memasukkan blok GAINBLK_f sebagai Pengendali Proporsional



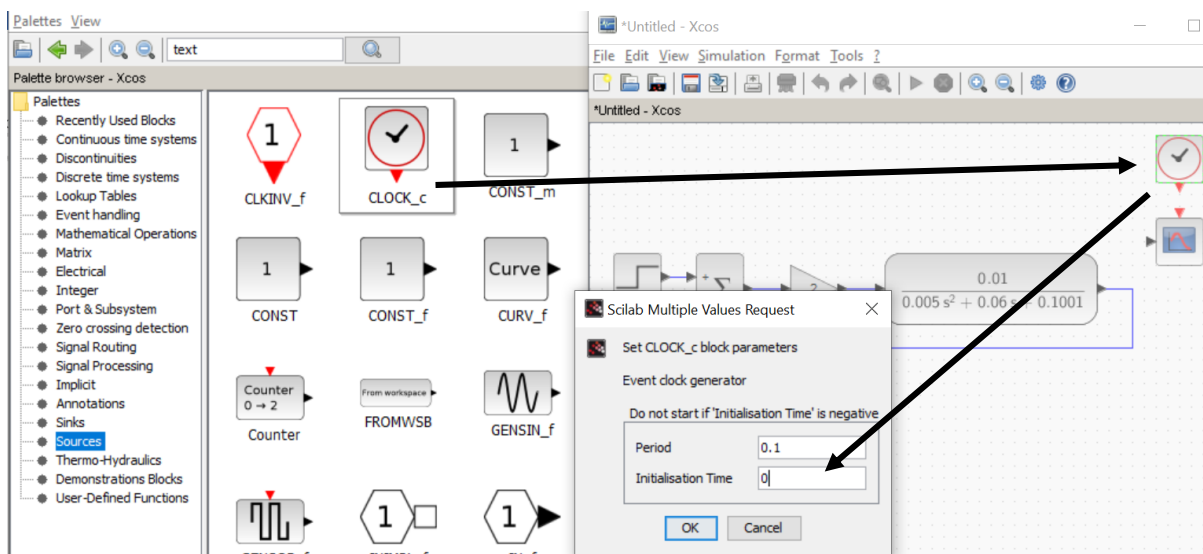
Gambar 5. 6. Menghubungkan kabel pada xcoss

Buka palettes pada folder Sinks, masukkan blok CSCOPE dan klik dua kali pada blok tersebut. Isikan nilai Ymin adalah 0 dan Ymax adalah 2. Perhatikan Gambar 5.7.



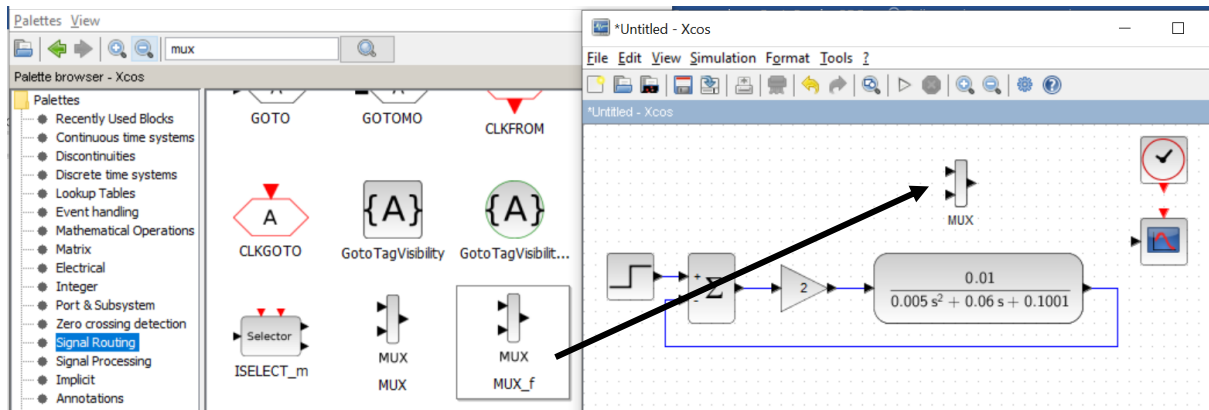
Gambar 5. 7. Memasukkan blok CSOPE

Buka palettes pada folder Sources dan masukkan Blok CLOCK_c. Klik dua kali pada blok Clock_c dan masukkan nilai 0 pada Initialization Time lalu klik OK. Perhatikan Gambar 5.8.



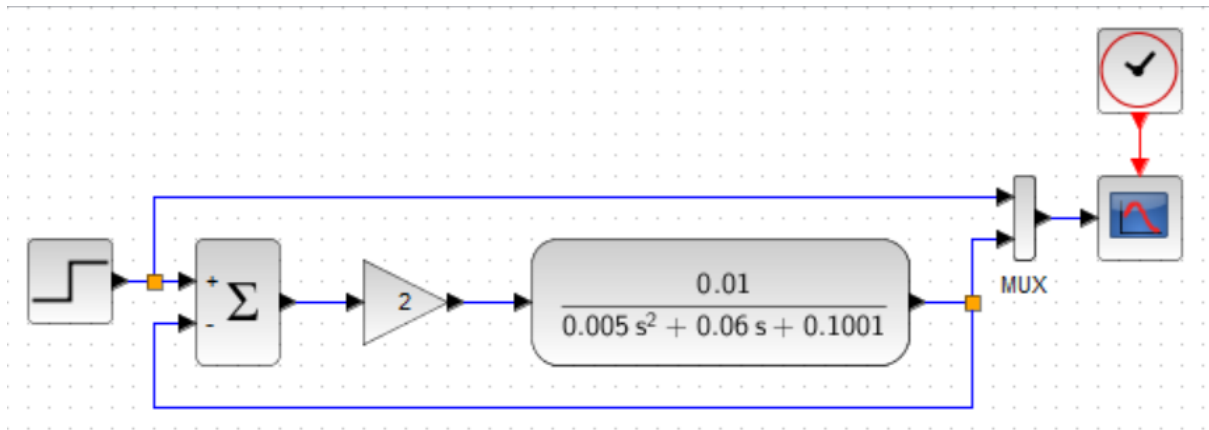
Gambar 5. 8. Memasukkan blok Clock_c

Buka palettes pada folder Signal Routing dan masukkan Blok MUX_f. Perhatikan Gambar 5.9.



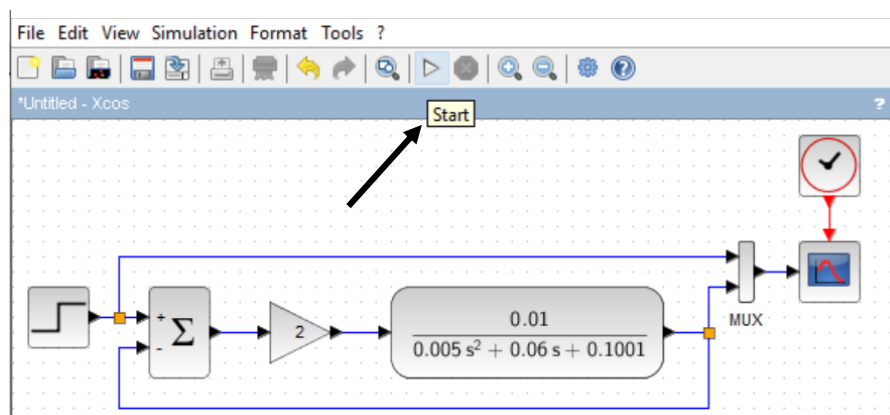
Gambar 5. 9. Memasukkan blok MUX_f

Susun simulasi sesuai pada Gambar 5.10. Hubungkan garis sesuai dengan gambar tersebut.

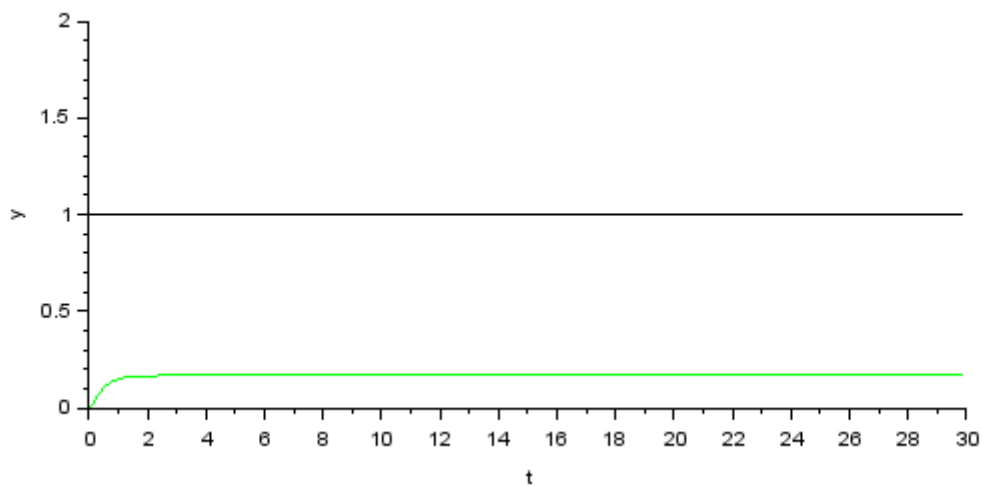


Gambar 5. 10. Susunan Simulasi Pengendali Proporsional

Tekan tombol start untuk menjalankan pengendali proporsional pada motor DC ditunjukkan pada Gambar 5.11. Hasil simulasi pengendali Proporsional ditunjukkan pada Gambar 5.12.



Gambar 5. 11. Menjalankan simulasi dengan klik pada Tombol Start



Gambar 5. 12. Hasil simulasi pengendali Proporsional kecepatan sudut motor DC dengan nilai Penguatan 2

Penjelasan Gambar 5.11 dan Gambar 5.12 adalah sebagai berikut. Gambar 5.12 adalah hasil grafik kecepatan motor DC (berwarna hijau) dan nilai referensi (warna hitam). Pada gambar tersebut terlihat nilai kecepatan motor DC belum mencapai nilai referensi (Fungsi pengendali untuk membuat sistem motor DC mencapai nilai referensi).

Fungsi blok mux pada Gambar 5.11 berfungsi untuk membuat grafik menampilkan dua buah grafik seperti ditunjukkan pada Gambar 5.11.

1. Gambarkan grafik untuk nilai penguatan proporsional dengan nilai seperti ditunjukkan pada Tabel 5.1. Amati grafik, apa pengaruh penambahan nilai proporsional terhadap respon sistem?

Tabel 5. 1. Pengaruh penambahan nilai Pengendali Proporsional

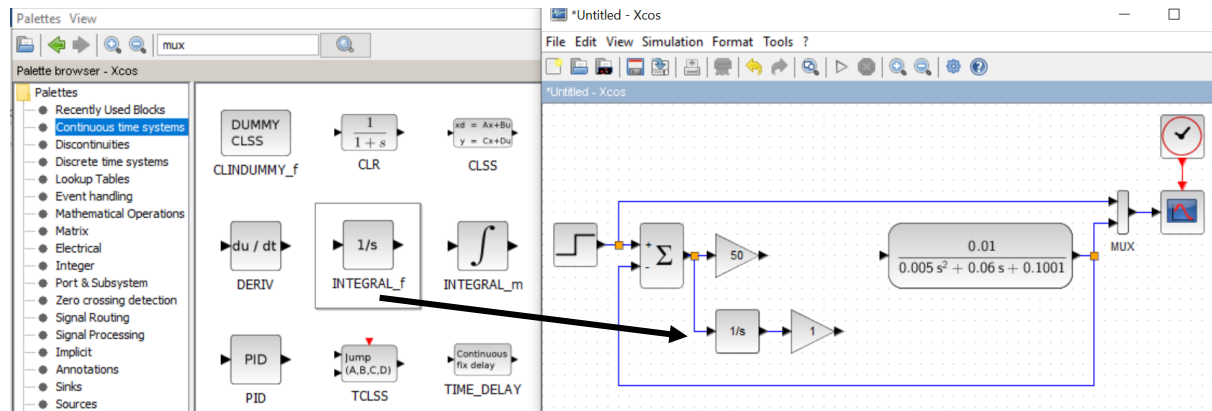
Nilai K_p	Gambar Grafik
10	
20	
30	
40	
50	

Pengendali Proporsional Integral Motor DC

Persamaan Pengendali Proporsional Integral adalah

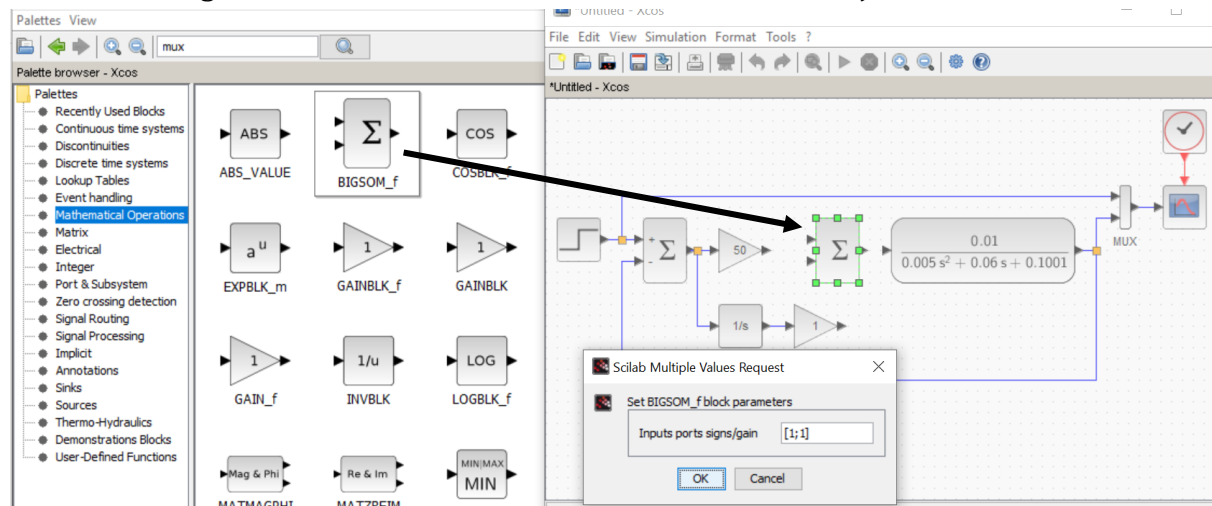
$$u(t) = K_p e(t) + K_i \int e(t)$$

Untuk menambahkan pengendali integral, buka palettes pada folder continuous time systems, dan masukkan blok INTEGRAL_f. Tambahkan juga blok penguat (gain) dengan blok GAINBLK_f dari folder Mathematical Operations. Isikan nilai 5 untuk GAINBLK_f pada pengendali integral. Perhatikan Gambar 5.13.

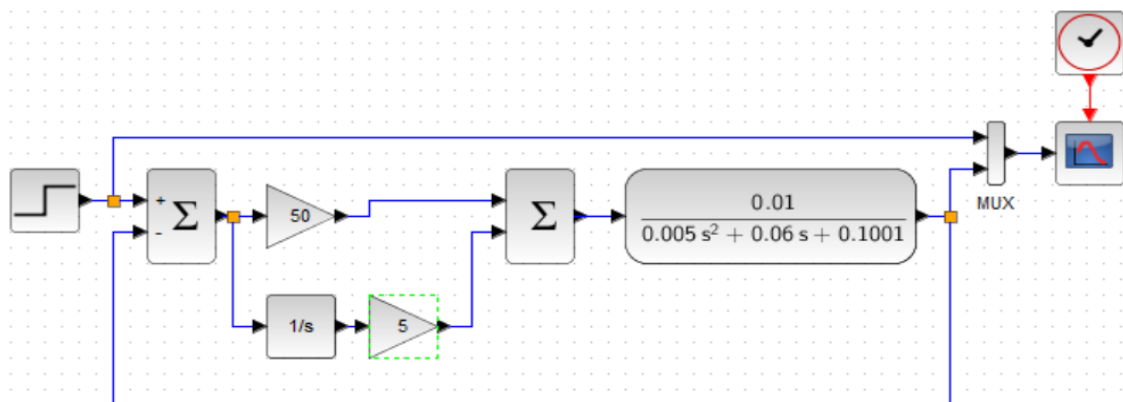


Gambar 5. 13. Menambahkan blok pengendali integral

Buka palettes pada folder mathematics Operations dan masukkan Blok BIGSOM_f. Perhatikan Gambar 5.14. Susunlah simulasi pengendali proporsional integral pada motor DC sesuai dengan Gambar 5.15. Tekan tombol start untuk menjalankan simulasi.

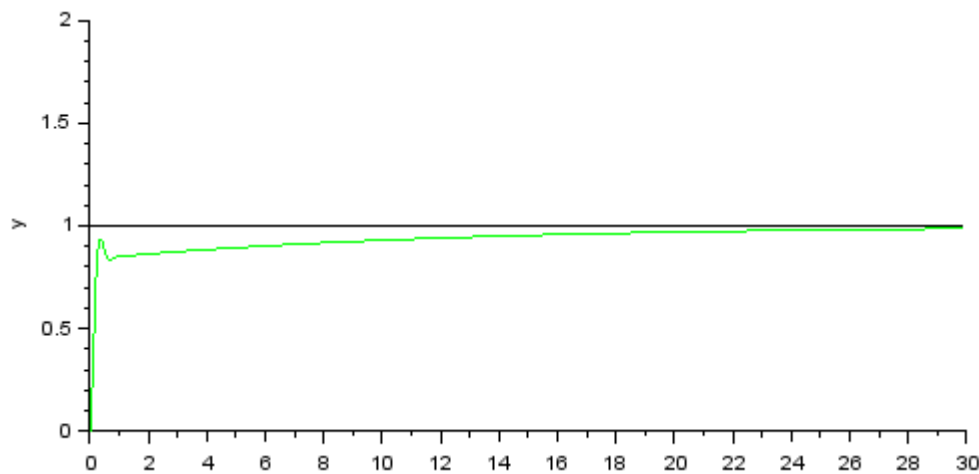


Gambar 5. 14. Menambahkan blok BIGSOM_f



Gambar 5. 15. Simulasi Pengendali Proporsional Integral pada Motor DC

Hasil simulasi ditunjukkan pada Gambar 5.16. Amati hasil grafik tersebut dengan grafik hanya dengan pengendali Proporsional. Terlihat bahwa dengan pengendali integral respons sistem mulai mendekati nilai referensi.



Gambar 5. 16. Hasil simulasi pengendali proporsional integral kecepatan sudut motor DC

2. Gambarkan grafik untuk nilai penguatan integral dengan nilai seperti ditunjukkan pada Tabel 5.2. Amati grafik, apa pengaruh penambahan nilai integral terhadap respon sistem?

Tabel 5. 2. Pengaruh penambahan nilai Pengendali Integral

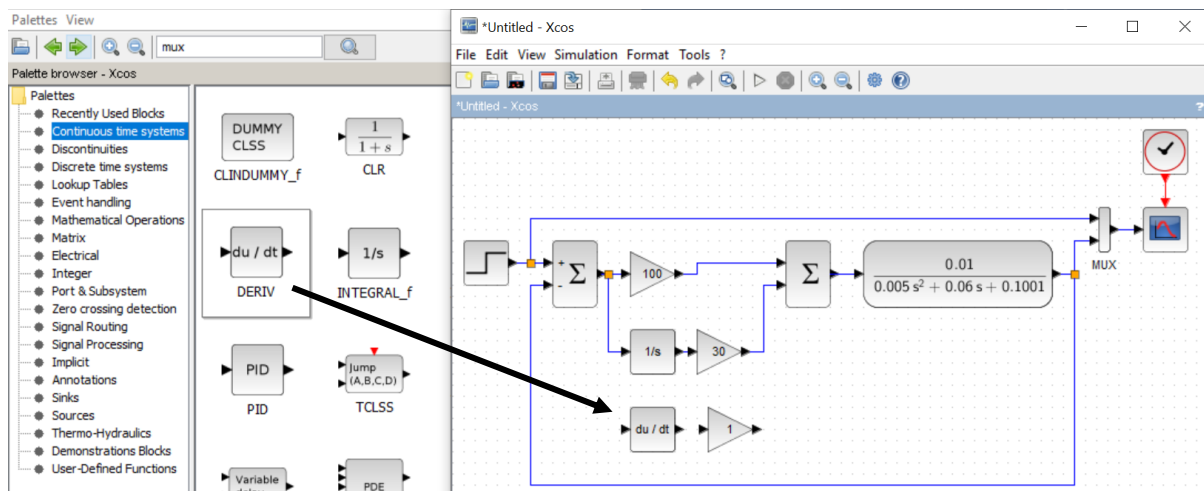
Nilai K_p	Nilai K_i	Gambar Grafik
50	10	
50	15	
50	20	
50	25	
50	30	

Pengendali Proporsional Integral Derivatif Motor DC

Persamaan Pengendali Proporsional Integral Derivatif adalah

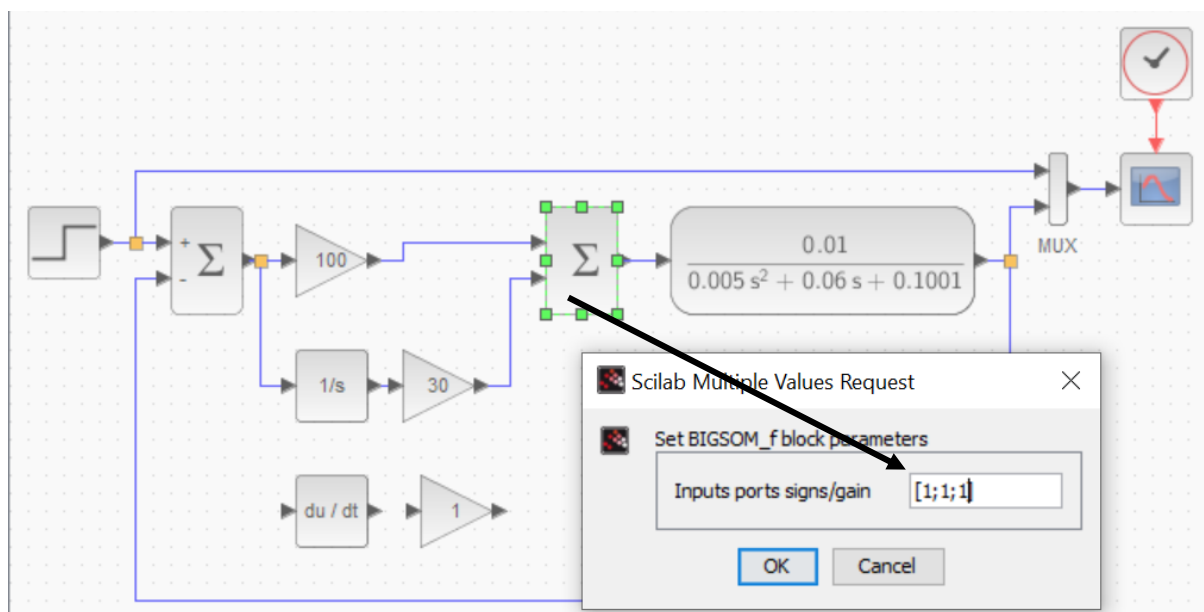
$$u(t) = K_p e(t) + K_i \int e(t) + K_d \frac{de(t)}{dt}$$

Untuk menambahkan pengendali derivatif, buka palettes pada folder continuous time systems, dan masukkan blok DERIV. Tambahkan juga blok penguat (gain) dengan blok GAINBLK_f dari folder Mathematical Operations. Isikan nilai 1 untuk GAINBLK_f pada pengendali derivatif. Perhatikan Gambar 5.17.

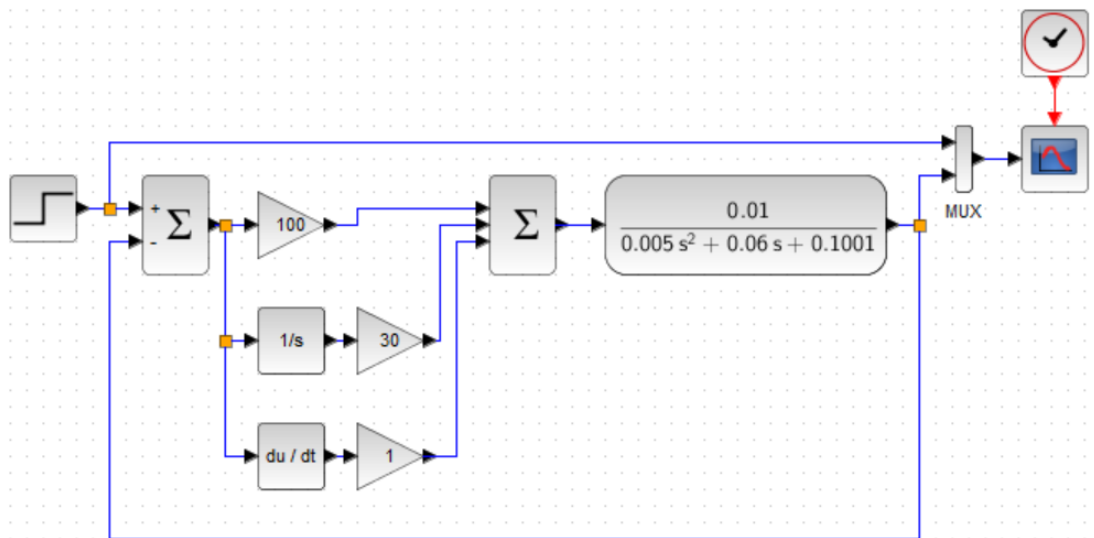


Gambar 5. 17. Menambahkan blok pengendali Derivatif

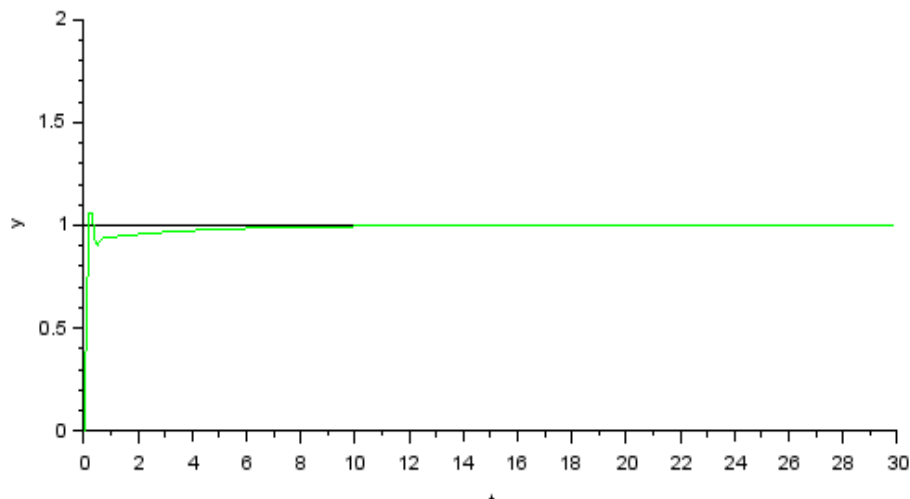
Klik dua kali pada Blok BIGSOM_f dan modifikasi agar memiliki masukan tiga buah input dengan matriks $[1;1;1]$. Perhatikan pada Gambar 5.18. Susun simulasi seperti Gambar 5.19. Tekan tombol start untuk menjalankan simulasi. Hasil simulasi ditunjukkan pada Gambar 5.20.



Gambar 5. 18. Modifikasi Blok BIGSOM_f



Gambar 5. 19. Simulasi Pengendali Proporsional Integral Derivatif pada Motor DC



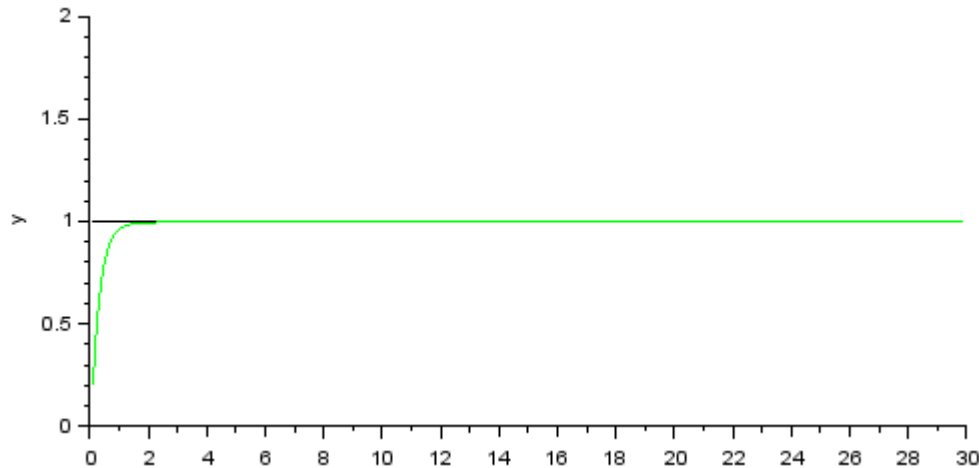
Gambar 5. 20. Hasil simulasi pengendali proporsional integral derivatif kecepatan sudut motor DC

3. Gambarkan grafik untuk nilai penguatan derivatif dengan nilai seperti ditunjukkan pada Tabel 5.3. Amati grafik, apa pengaruh penambahan nilai derivatif terhadap respons sistem?

Tabel 5. 3. Pengaruh penambahan nilai Pengendali Derivatif

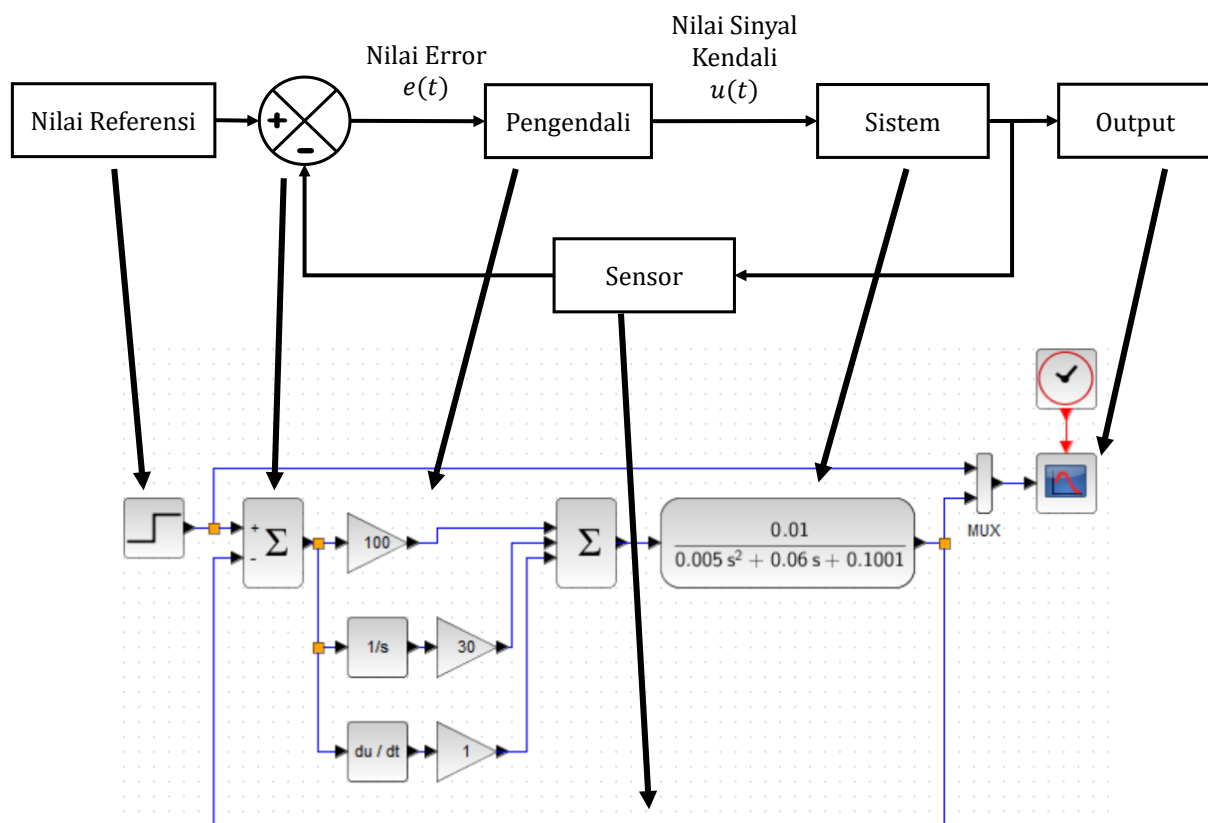
Nilai K_p	Nilai K_i	Nilai K_d	Gambar Grafik
100	30	2	
100	30	4	
100	30	6	
100	30	8	
100	30	10	

4. Carilah nilai K_p , K_i dan K_d yang menghasilkan grafik respons yang paling baik yaitu tidak ada overshoot, rise time cepat dan settling time cepat. Lalu gambarkan pada laporan respons paling baik yang Anda dapatkan. Contoh grafik respons yang baik ditunjukkan pada Gambar 5.21.



Gambar 5. 21. Hasil simulasi pengendali proporsional integral derivatif kecepatan sudut motor DC yang tidak ada overshoot, rise time cepat dan settling time juga cepat

5. Analogi simulasi sistem kendali kecepatan motor DC dengan diagram blok sistem kendali kalang tertutup ditunjukkan pada Gambar 5.22. Jelaskan perbandingan kedua blok tersebut.



Gambar 5. 22. Analogi Diagram blok sistem kendali dengan Simulasi

DAFTAR PUSTAKA

Katsuhiko Ogata, Modern Control Engineering, Pearson, 2010.
<https://www.scilab.org/>