LAPORAN PRAKTIKUM

PEMODELAN SISTEM MOTOR DC

TF3020 Laboratorium Instrumentasi dan Kontrol

Hari, Tanggal Praktikum : Selasa, 26 September 2023

Kelompok: 10

Nama Anggota:

13320026 Yosep Putra Setiyanto

13320032 Dewi Ashiila Hikmawan

13320045 Zayna Mosa Kalila

13320052 Daffa Auliya Ulhaq Sulistiyan

---

1 Latar Belakang

Sistem motor banyak digunakan di dunia industri, seperti pada sistem *compressor, fan, blower*, peralatan *heavy-duty*, sistem HVAC, *crusher*, pompa, *dryer*, dan lain-lainnya. Motor juga banyak digunakan dalam perangkat-perangkat listrik sehari-hari, seperti *vibrator* ponsel, peralatan industri dan rumah tangga, *disk driver*, bor listrik, dan kipas. Uji coba hubungan masukan dan keluaran sistem secara kalang terbuka diperlukan untuk desain sebuah pengendali kecepatan motor. Setelah melihat karakteristik sistem tersebut, dilakukan kompensasi agar hubungan masukan dan keluaran lebih optimal.

Pada praktikum TF4022 Sistem Kontrol Diskrit modul 1, praktikan akan memodelkan sistem motor DC dengan sistem kalang terbuka (*open loop*). Praktikum dilakukan dengan cara mengakuisisi data dari sensor potensiometer dan tachometer, untuk kemudian ditentukan parameter-parameter hubungan keluaran dan masukan motor. Hasil pengamatan ini nantinya akan digunakan untuk menentukan fungsi kompensator, yang dapat digunakan untuk mengkoreksi hasil keluaran dari motor DC. Yang terakhir adalah dilakukan pemodelan sistem berdasarkan responsnya.

2 Tujuan Praktikum

Tujuan yang ingin dicapai pada modul praktikum ini adalah sebagai berikut:

1 Mengakuisisi data posisi sudut putaran motor dari sensor potensiometer dan data kecepatan putar motor dari tachometer

2 Mengetahui parameter hubungan keluaran dan masukan pada motor untuk melakukan perbaikan (kompensasi) terhadap parameter hasil keluaran sensor yang diperoleh

3 Menggunakan dinamika respons sistem terhadap perubahan masukan untuk memodelkan sistem yang diperoleh.

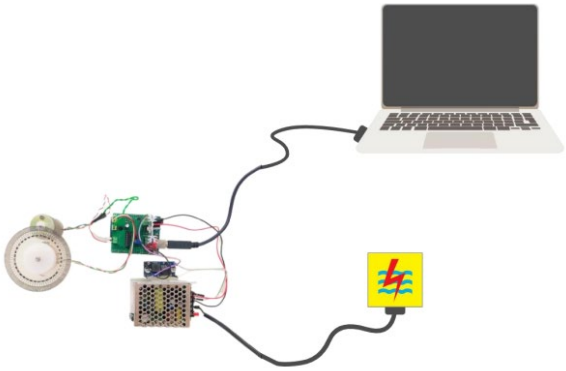
3 Alat dan Bahan

Tabel 1 Alat dan Bahan

No	Alat dan bahan	Jumlah	Keterangan
1	Laptop	1	Disiapkan sendiri
2	Kit Praktikum	1	Disediakan asisten
3	Modul Praktikum Pertama	1	Disediakan, dapat diunduh di kuliah2021 dan Edunex
4	Dokumentasi Kit Praktikum	1	Disediakan, dapat diunduh di kuliah2021 dan Edunex
5	Software Arduino IDE	1	Instal sendiri, unduh di https://www.arduino.cc/en/Main/Software
6	Software MATLAB	1	Dapat menggunakan Matlab online
7	Program SKD23.ino	1	Disediakan, dapat diunduh di kuliah2021 dan Edunex
8	Program Simulasi.m	1	Disediakan, dapat diunduh di kuliah2021 dan Edunex

4 Prosedur Praktikum

Kit praktikum berupa rangkaian motor DC dengan mikroprosesor Arduino sebagai pengendali. Terdapat juga potensiometer dan tachometer. Kit praktikum disambungkan dengan sumber saya listrik sendirinya. Arduino kemudian disambungkan dengan laptop untuk pengambilan data dan pemograman. Praktikum ini menggunakan library Arduino Adafruit_MCP4725.



Gambar 1 Ilustrasi kit praktikum

4.1 Percobaan 1 : Akuisisi Data Sensor

Pada percobaan akuisisi data sensor, diambil data posisi dari potensiometer dan data kecepatan dari tachometer. Diputar potensiometer dengan manual tanpa memberikan masukan kepada motor untuk melakukan akuisisi potensiometer. Potensiometer diputar sampai posisi yang terbaca pada program sama dengan 10, 50, dan 90. Lalu, diberikan masukan kepada motor melalui serial monitor untuk melakukan akuisisi data kecepatan. Masukkan yang dicoba adalah 0% dan 75%. Dilakukan variasi waktu sampling yaitu 500 dan 1000 ms.

4.2 Percobaan 2 : Kalang Terbuka

Praktikan melakukan pengamatan pada nilai kecepatan yang muncul pada Serial Monitor Arduino IDE serta putaran motor di kit praktikum. Kedua hal tersebut menunjukkan hubungan antara masukan motor dan kecepatan tunak terukur serta spesifikasi awal kit praktikum. Pada percobaan ini, motor diberi masukan yang nilainya naik secara bertahap, dimulai dari angka 0 sampai dengan 100. Sistem diberi waktu sampling sebanyak 3000 ms untuk menunggu sistem mencapai keadaan tunak. Praktikan melakukan pengamatan pada nilai kecepatan yang muncul pada Serial Monitor Arduino IDE serta putaran motor di kit praktikum. Kedua hal tersebut menunjukkan hubungan antara masukan motor dan kecepatan tunak terukur serta spesifikasi awal kit praktikum. Hubungan antara masukan motor dan kecepatan tunak terukur dapat diamati pada **Tabel 2 Hubungan masukan motor dan kecepatan tunak terukur** yang dapat diakses melalui file Excel terlampirkan. Kami menggunakan data dari metode ketiga yang mengambil beberapa data dari pin sensor, merata-ratakan data tersebut dan mengkonversikannya menjadi persen. Tipe data dalam bentuk float. Sementara itu, spesifikasi awal kit praktikum dapat diamati pada **Tabel 3** di bawah ini.

Tabel 3 Spesifikasi awal kit praktikum berkaitan dengan hubungan masukan dan kecepatan

Spesifikasi	Nilai	Satuan
Nomor kit	10	
Masukan minimum putar motor (positif)	21	Persen
Masukan minimum putar motor (negatif)	-28	Persen
Masukan minimum terdeteksi (positif)	32	Persen
Masukan minimum terdeteksi (negatif)	-34	Persen
Masukan minimum putaran maksimal (positif)	97	Persen
Masukan minimum putaran maksimal (negatif)	-100	Persen
Kecepatan putar maksimum (positif)	79	Persen
Kecepatan putar maksimum (negatif)	78	Persen

Data yang didapatkan ditampilkan dalam bentuk grafik input terhadap output sebagai berikut:

Gambar 2 Grafik Kecepatan Input vs Kecepatan yang Terukur Sistem Kalang Terbuka

Pada Gambar 2 Grafik Kecepatan Input vs Kecepatan yang Terukur Sistem Kalang Terbuka, terlihat terdapat *deadzone* yang disebabkan pembacaan nol oleh sensor. Untuk itu, dibuat kompensator. Kompensator adalah komponen dalam sistem kontrol yang digunakan untuk mengatur sistem lain. Biasanya, hal ini dilakukan dengan mengkondisikan input atau output ke sistem tersebut. Kompensasi pada tab compensation yang kami gunakan adalah sebagai berikut:

```
float deadzoneCompensator(float inputSerial){
  float inputCompensated;
  if(inputSerial < 0){
```

```
          inputCompensated = (-34)+((inputSerial-(0))/((-100)-0))*((-
100)-(-34));
        }
        else if (inputSerial > 0){
          inputCompensated = (32)+((inputSerial-(0))/((100)-
0))*((100)-(32));
        }
        else if (inputSerial == 0){
          inputCompensated = 0;
        }
        return inputCompensated;
      }
```

Pada tab compensation dibuat kode seperti diatas, pertama didefinisikan variabel deadzoneCompensator sebagai kompensasi untuk deadzone yang terjadi. Lalu dilakukan pendefinisian variable inputCompensated, pada grafik tampak deadzone berada pada wilayah masukan negative sampai positif, maka digunakan fungsi *if* dan *else if* untuk mendapat kompensasi yang sesuai. Digunakan metode sebagai berikut.

Untuk kompensasi di wilayah positif:

$$\frac{A-32}{100-32}=\frac{B-0}{100-0}$$

$B = nilai\ masukan\ di\ serial\ monitor$

$A = nilai\ masukan\ yang\ telah\ terkompensasi$

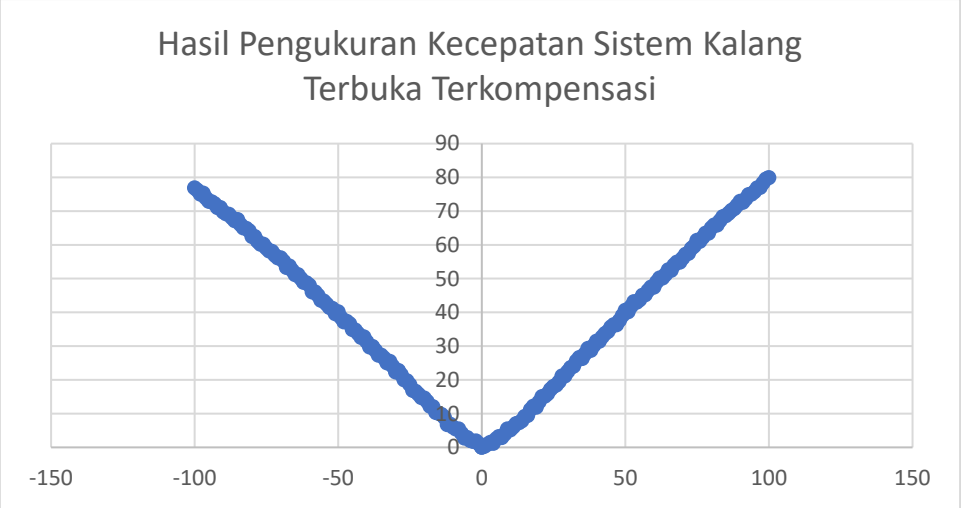
$$inputCompensated = 32+\frac{inputSerial-0}{100-0}\times(100-32)$$

Untuk kompensasi di wilayah negatif:

$$inputCompensated = -34+\frac{inputSerial-0}{-100-0}\times(-100-(-34))$$

4.3 Percobaan 3 : Kalang Terbuka Terkompensasi

Fungsi kompensasi yang telah dirancang pada percobaan 2 digunakan untuk mengurangi deadzone. Diambil ulang data seperti pada percobaan 2. Berikut hasil data yang didapatkan:



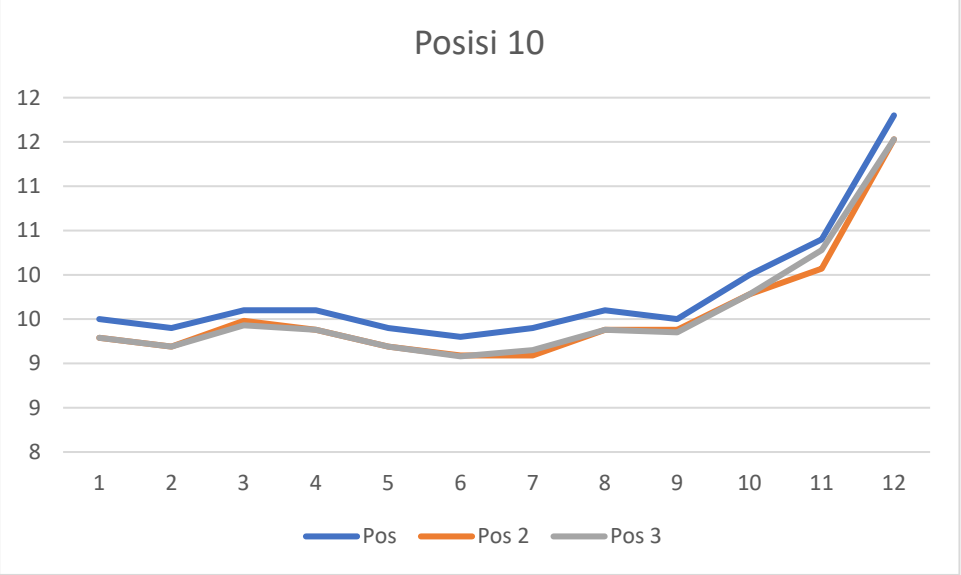
Gambar 3 Grafik Kecepatan Input vs Kecepatan yang Terukur Sistem Kalang Terbuka Terkompensasi

Setelah itu diambil data dinamika perubahan kecepatan motor hingga tunak ketika diberikan perubahan masukan (setiap kenaikan masukan 5). Data dinamika yang diperoleh kemudian digunakan untuk melakukan pemodelan kecepatan.

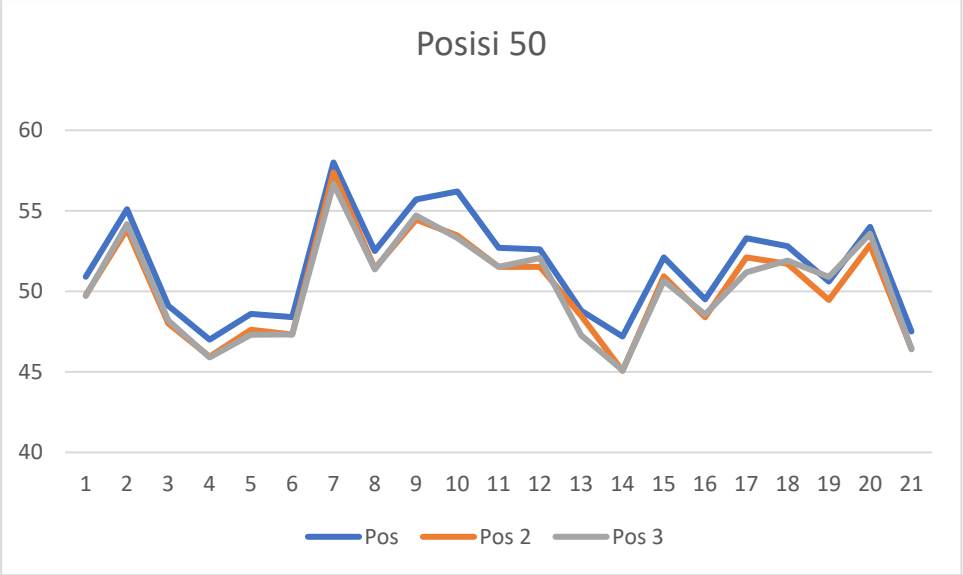
5 Analisis Hasil Percobaan

5.1 Percobaan 1: Akuisisi Data Sensor

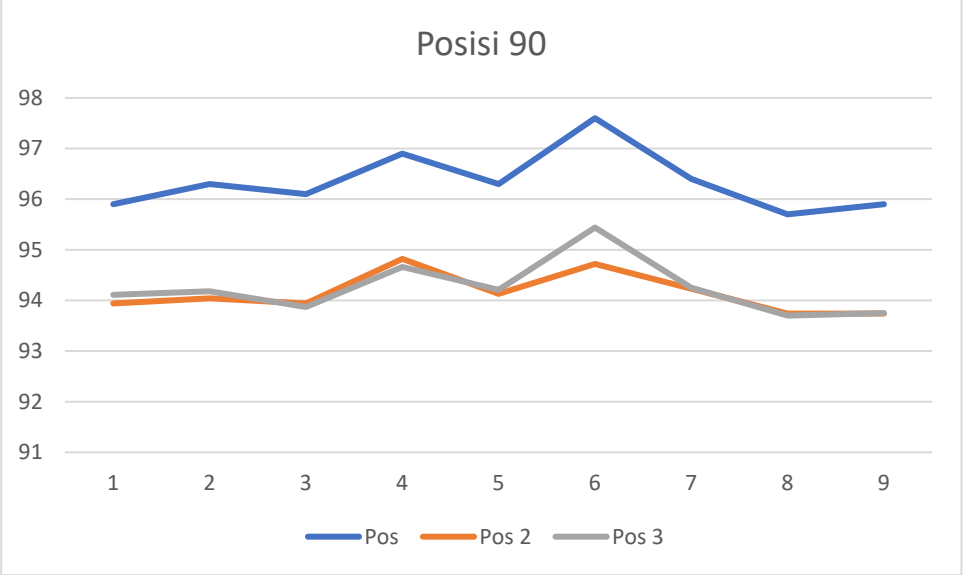
Berikut adalah beberapa grafik yang mendemonstrasikan akuisisi posisi dari sensor potensiometer:



Gambar 4 Grafik Hasil Pengukuran Posisi dengan Input 10 Terhadap Waktu



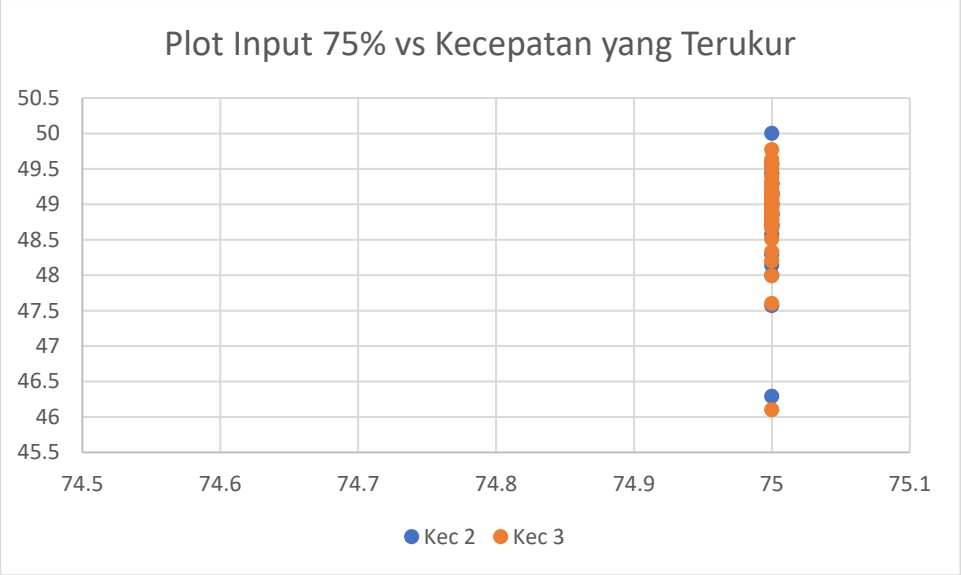
Gambar 5 Grafik Hasil Pengukuran Posisi dengan Input 50 Terhadap Waktu



Gambar 6 Grafik Hasil Pengukuran Posisi dengan Input 90 Terhadap Waktu

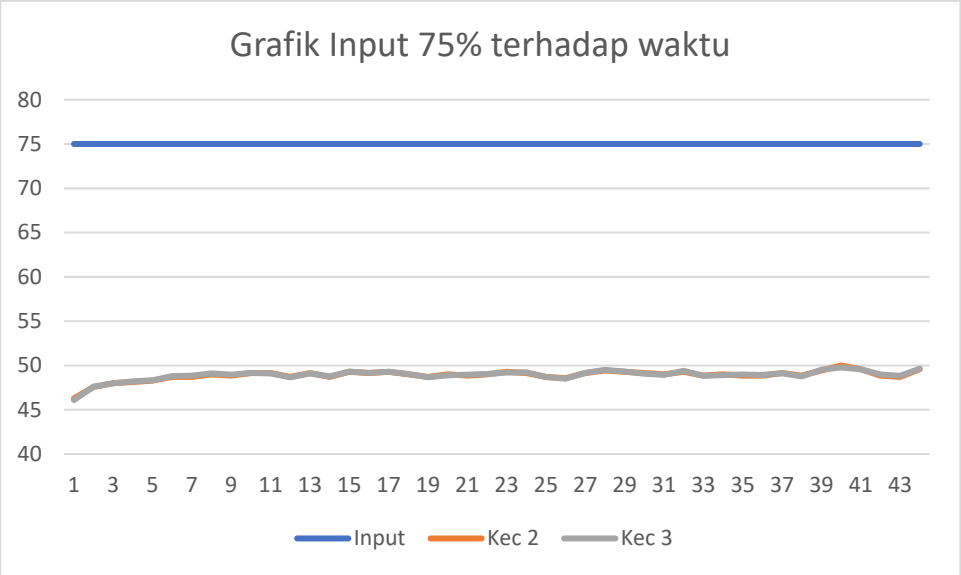
Terlihat semakin besar posisi, sensor semakin tidak akurat, namun tetap presisi.

Berikut adalah beberapa grafik yang mendemonstrasikan akuisisi kecepatan dari sensor tachometer:



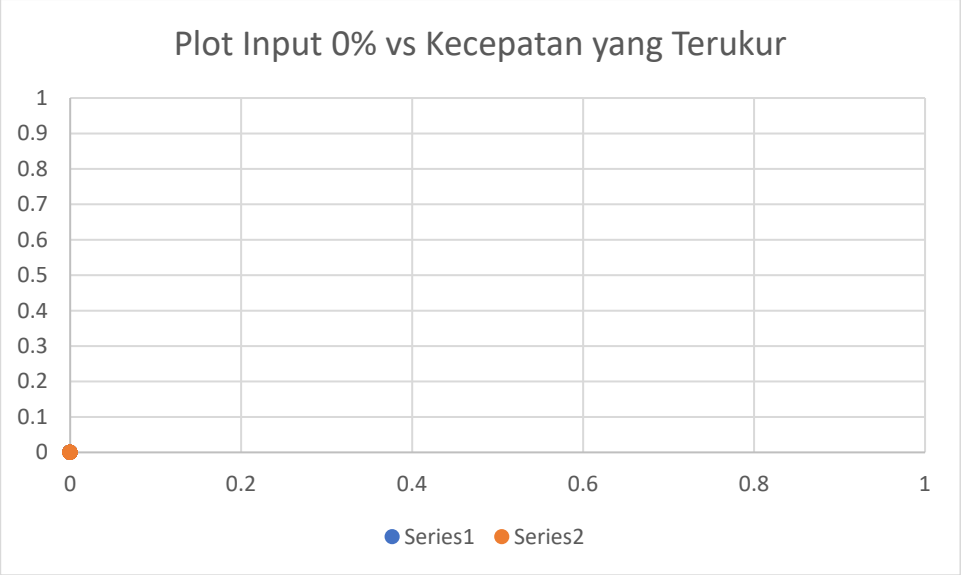
Gambar 7 Plot Input 75% vs Kecepatan yang Terukur

Terlihat pada Gambar 7, terlihat saat input kecepatan 75%, kecepatan yang terukur tidak presisi.



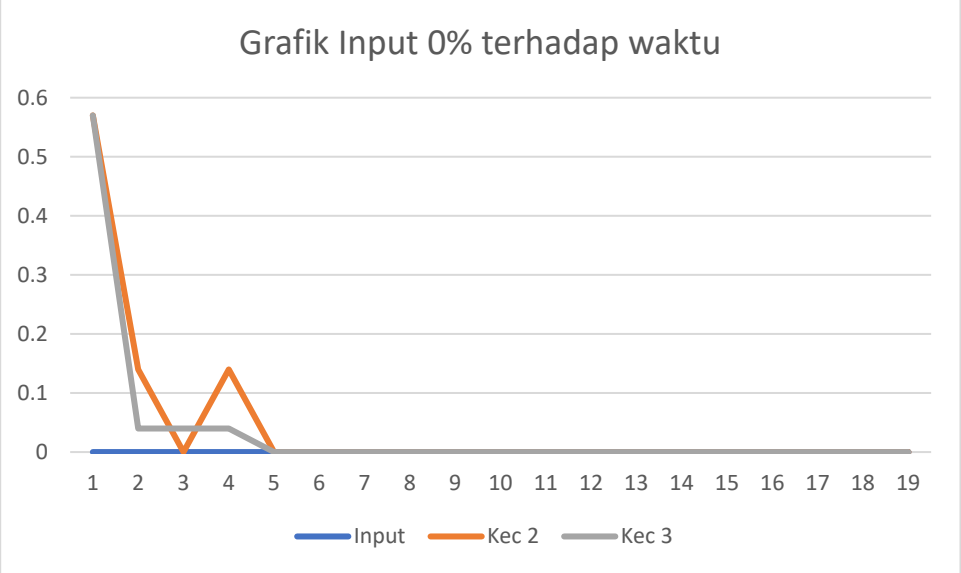
Gambar 8 Grafik Input Kecepatan 75% dan Kecepatan yang Terukur terhadap waktu

Terlihat pada Gambar 8, terlihat saat input kecepatan 75%, kecepatan yang terukur tidak akurat.



Gambar 9 Plot Input 0% vs Kecepatan yang Terukur

Terlihat pada Gambar 9, terlihat saat input kecepatan 0%, kecepatan yang terukur presisi.



Gambar 10 Grafik Input Kecepatan 0% dan Kecepatan yang Terukur terhadap waktu

Terlihat pada Gambar 10, terlihat saat input kecepatan 0%, kecepatan yang terukur cukup akurat, namun pada awalnya tidak langsung terukur 0.

Fungsi sampling digunakan pada metode kedua untuk mengakuisisi data sensor Poten2 dan Tacho2. Metode ini mengambil data langsung dari pin sensor, kemudian data dikonversi ke dalam satuan persen dan menjadi bertipe data float. Sementara itu, fungsi smooth digunakan pada metode ketiga untuk mengakuisisi data sensor yang

ketiga Poten3 dan Tacho3. Metode ini mengambil beberapa data dari pin sensor, merata-ratakannya, lalu mengonversi ke dalam satuan persen dan bertipe data float juga. Jumlah data yang dirata-ratakan diatur pada ukuran window. Pada fungsi "smooth_pin" ukuran window diatur pada variabel "SizeRead". Program akan menjuemlahkan terlebih dahulu semua data hasil pembacaan yang ada dalam window mennggunakan looping. Kemudian hasil penjumlahan tersebut di akhir akan dibagi lagi oleh ukuran window. Hasil pembagian tersebut tidak lain merupakan hasil rata-rata data yang ada pada window dan dijadikan sebgai hasil akhir untuk pengukuran posisi maupun kecepatan.

Nilai yang terbaca pada sensor potensiometer adalah data yang diambil langsung dari pin sensor dan dikonversi ke dalam satuan persen. Dengan demikian, rentang nilai 0 s.d. 100 pada Serial Monitor menunjukkan persentase putaran potensiometer, di mana nilai 0% menunjukkan 0deg dan 100% menunjukkan 360deg, sehingga hasil pengukuran menunjukkan penambahan nilai dari 0 langsung ke 100 ataupun sebaliknya.

Berdasarkan *source code* yang digunakan pada Arduino, dapat diketahui bahwa hal-hal yang memengaruhi waktu proses antara lain adalah variabel Time, StartTime, LastTime, DeltaTime, Delay, dan TimeSampling. [Time – StartTime] menunjukkan lama waktu terhitung program memasuki loop pertama kali dalam satuan milidetik. DeltaTime, yang dihitung sebagai [Time – LastTime], menunjukkan lama waktu eksekusi satu loop program dalam satuan milidetik. TimeSampling adalah waktu yang dibutuhkan untuk mencuplik satu data. Dalam hal ini, nilai waktu Delay dibuat sama dengan TimeSampling.

5.2 Percobaan 2 : Kalang Terbuka

Pada percobaan 2, keluaran bernilai positif meskipun masukan yang diberikan negatif ataupun positif karena sensor yang digunakan berupa tachometer yang menggunakan optocoupler (pengirim dan pendeteksi sumber cahaya) yang mengukur kecepatan bilah memutar pada motor. Sensor akan menerima pantulan sinar dari *transmitter* yang dipantulkan oleh benda di depannya dan menghitung kecepatan benda tersebut bedasarkan intensitas cahaya terpantulkan yang diperoleh. Tidak terdapat parameter dari cahaya pantulan tersebut yang dapat memberikan informasi arah putaran benda (tachometer mengukur kecepatan absolut). Namun, masukan positif ataupun

negatif pada program mempengaruhi arah putar motor. Arah putaran motor (searah atau berlawanan arah jarum jam) dapat diatur dengan membalik polaritas tegangan yang masuk ke terminal daya motor.

Ketika masukan yang diberikan kepada motor kecil, motor tidak bergerak karena motor memiliki spesifikasi berupa *range*. *Range* adalah rentang minimal dan maksimal dari suatu alat, baik *input* maupun *output*. Motor baru dapat mulai berputar apabila masukan sudah mencapai nilai tertentu.

Saat percobaan dilakukan, motor mulai bergerak saat masukan sama dengan 21% atau -28%. Terlihat pada Gambar 2 Grafik Kecepatan Input vs Kecepatan yang Terukur Sistem Kalang Terbuka bahwa saat masukan masih di bawah atau sama dengan nilai 32% atau -34%, motor berputar dengan sangat pelan dan dibaca oleh sensor bergerak dengan kecepatan 0. Hal ini disebabkan saat motor bergerak dengan sangat pelan, cahaya yang dipantulkan masih dalam rentang frekuensi yang dianggap nol oleh sensor. Dan seperti halnya motor, sensor juga memiliki *range* tertentu.

Dilakukan pembagian grafik menjadi beberapa daerah linier dengan membuat perbandingan output / input. Nilai perbandingan tersebut dilihat pada satu angka pentingnya. Semua yang memiliki nilai perbandingan yang sama dimasukkan dalam rentang daerah linier yang sama.

Tabel 6 Pembagian daerah linier kalang terbuka kit praktikum Sistem Kontrol Diskrit

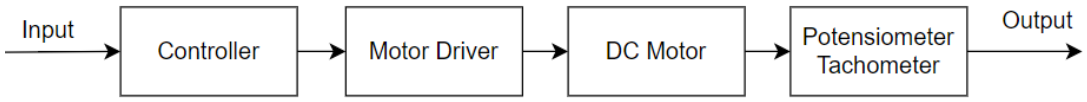
Rentang Masukan	Rentang Keluaran	Keterangan
-100 – -95	-77.86 – -72.00	Nilai output / input = 0,8
-94 – -76	-69.71 – -49.43	Nilai output / input = 0,7
-75 – -65	-47.86 – -35.86	Nilai output / input = 0,6
-64 – -58	-34.43 – -27.43	Nilai output / input = 0,5
-57 – -51	-25.43 – -18.29	Nilai output / input = 0,4
-50 – -46	-17.00 – -12.14	Nilai output / input = 0,3
-45 – -42	-10.14 – -06.57	Nilai output / input = 0,2
-41 – -37	-05.29 – -02.00	Nilai output / input = 0,1
-36 – 34	-00,86 – 01,14	Nilai output / input = 0,0
35 – 38	02,57 – 05,14	Nilai output / input = 0,1
39 – 43	06,00 – 10,57	Nilai output / input = 0,2
44 – 46	12,00 – 14,71	Nilai output / input = 0,3
47 – 53	16,57 – 23,43	Nilai output / input = 0,4
54 – 58	25,71 – 31,00	Nilai output / input = 0,5
59 – 67	32,57 – 42,86	Nilai output / input = 0,6
68 – 83	44,29 – 61,86	Nilai output / input = 0,7

84 – 100	64,00 – 78,43	Nilai output / input = 0,8

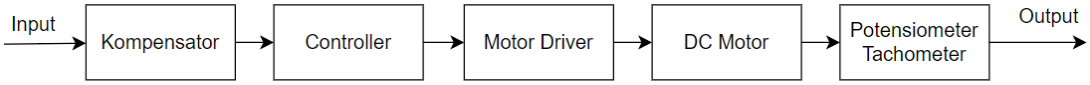
[Untuk penjelasan kompensasator ada pada bab prosedur (4.2)]

5.3 Percobaan 3 : Kalang Terbuka Terkompensasi

Praktikum ini dilakukan dengan kalang terbuka terkompensasi, berikut diagram blok untuk percobaan 2 dan percobaan 3.



Gambar 11 Diagram blok kalang terbuka



Gambar 12 Diagram blok kalang terbuka terkompensasi

Perbedaan hanya terletak pada ada dan tidaknya komponen kompensator, setelah dilakukan percobaan terjadi *deadzone* pada masukan -36 sampai 34, seperti pada **Tabel 6.** Komponen kompensator mengkompensasi nilai input yang masuk.

Pada percobaan kedua didapatkan harga spesifikasi awal kit praktikum juga didapat grafik perbandingan masukan motor tanpa kompensasi dan kecepatan tunak terukur seperti pada **Gambar 2 Grafik Kecepatan Input vs Kecepatan yang Terukur Sistem Kalang Terbuka**. Setelah itu didapat persamaan yang digunakan untuk mengkompensasi nilai masukan (dapat dilihat pada subbab **4.2**), dan didapat hasil perbandingan masukan terkompensasi dengan kecepatan seperti pada **Gambar 3 Grafik Kecepatan Input vs Kecepatan yang Terukur Sistem Kalang Terbuka Terkompensasi**. Dengan melihat grafik tersebut terbukti bahwa fungsi kompensasi yang telah dijalankan berhasil mengurangi deadzone yang terjadi.

Dari grafik tersebut diperoleh dan dibuat beberapa daerah linear, sebagai berikut:

Tabel Pembagian daerah linier kalang terbuka terkompensasi kit praktikum Sistem Kontrol Diskrit

Rentang Masukan	Rentang Keluaran	Keterangan
(-100) – (-26)	(-76.86) – (-19.71)	Nilai output / input = 0,8
(-25) – (-13)	(-18.57) – (-8.86)	Nilai output / input = 0,7
(-12) – (-3)	(-6.71) – (1.71)	Nilai output / input = 0,6

0 – 4	0 – 1	Nilai output / input = 0,3
5 – 11	2 – 6	Nilai output / input = 0.5
12 – 16	7 – 9	Nilai output / input = 0,6
17 – 30	11 – 22	Nilai output / input = 0,7
31 – 100	24 – 80	Nilai output / input = 0,8

Jika dibandingkan dengan **Tabel 6 Pembagian daerah linier kalang terbuka kit praktikum Sistem Kontrol Diskrit**, terjadi perubahan daerah linnier, dan perubahan paling signifikan terjadi pada rentang -36 sampai 34 dimana terjadi deadzone pada masukan nonkompensasi. Dan pada terkompensasi daerah linier tidak sebanyak pada non-kompensasi.

Dilakukan pemodelan untuk tiap respon dinamik masukan positif menggunakan model orde satu dengan waktu tunda. Pemodelan dapat dilihat pada file Excel terlampir. Berikut ringkasan hasil parameter model dari setiap masukan:

Tabel 7 Identifikasi parameter model kit praktikum Sistem Kontrol Diskrit

Masukan	Keluaran Tunak	Gain	Konstanta Waktu	Waktu Tunda
5	5.1586	1.0317	0.0371	0.1750
10	3.1103	0.6221	0.0573	0.0583
15	5.3632	1.0726	0.0836	0.0149
20	3.4656	0.9594	0.0695	0.0500
25	5.2831	1.0566	0.0887	0.0086
30	3.8541	0.7708	0.0824	0.0130
35	3.7639	0.7528	0.0916	0.0129
40	4.7970	0.9594	0.0896	0.0125
45	3.5482	0.7096	0.0757	0.0144
50	4.8160	0.9632	0.0985	0.0063
55	3.0259	0.6052	0.0841	0.0182
60	3.5194	0.7039	0.1052	0.0250
65	4.6157	0.9231	0.0296	0.1000
70	3.5927	0.7185	0.0988	0.0067
75	4.5117	0.9023	0.0839	0.0131
80	2.9832	0.5966	0.0996	0.0125
85	3.3938	0.6788	0.1129	0.0000
90	4.1656	0.8331	0.0902	0.0185
95	2.7874	0.5575	0.0897	0.0212
100	4.1471	0.8294	0.1292	0.0000
AVERAGE		0.812	0.085	0.029
VARIANS		0,027	0,001	0,002

Waktu cacah yang digunakan pada percobaan ini adalah 25 milisecond atau 0.025 detik. Angka tersebut didapat dan digunakan berdasarkan hasil *trial and error*. Untuk menghasilkan model yang dapat diamati karakteristik dinamiknya diperlukan data yang baik. Dengan *trial and error* didapat bahwa waktu cacah 25 ms mampu menghasilkan data yang baik yang dapat diamati nilai karakteristik dinamiknya, dapat dilihat pada lampiran Data Praktikum Ms.Excel, terdapat hasil percobaan yang telah dimodelkan dengan identifikasi sistem orde 1.

Nilai gain, konstanta waktu, waktu tunda yang digunakan adalah nilai rata-rata dari seluruh hasil parameter dari percobaan per 5 data yang didapatkan sebelumnya. Putusan ini berdasarkan nilai varians dari data yang sangat kecil (<0,03) yang berarti data tidak begitu bervariasi atau fluktuatif.

5.4 Tugas Tambahan

5.4.1 *Penalaan Parameter Pengontrol Sistem dengan Metode Ziegler Nichols dan Cohen Coon*

Salah satu jenis kontroler yang sering digunakan dalam sistem kontrol proses adalah kontroler PID (Proportional-Integral-Derivative). Algoritma ini terdiri dari tiga komponen yaitu P (proporsional), I (integral), dan D (derivatif), yang bekerja sama untuk mencapai tujuan kontrol yang diinginkan. Komponen P menghasilkan keluaran kontrol proporsional terhadap perbedaan antara set point dan variabel proses saat ini. Komponen I menambahkan keluaran kontrol yang proporsional terhadap integral dari perbedaan antara set point dan variabel proses selama waktu tertentu. Sedangkan, komponen D menambahkan keluaran kontrol yang proporsional terhadap perubahan cepat dalam variabel proses. Dengan mengombinasikan ketiga komponen ini, kontroler PID dapat mencapai stabilitas kontrol yang tinggi dan respons kontrol yang cepat pada berbagai jenis sistem kontrol.

Parameter PID harus disesuaikan dengan sistem kontrol yang sedang diatur untuk memastikan kinerja kontrol yang optimal. Pengontrol PID dalam beberapa penggunaannya menyesuaikan terhadap dinamika sistem. Metode penalaan PID yang paling banyak digunakan yaitu dengan aturan penalaan Ziegler Nichols dan Cohen Coon yang mengusulkan penentuan nilai-nilai dari

proportional gain Kp, integral time Ti, dan derivative time Td berdasarkan karakteristik respons transien dari sebuah sistem. Penalaan dengan Ziegler Nichols berdasarkan respons step dari sistem orde 1 dapat menggunakan tabel aturan penalaan Ziegler Nichols first method yang dapat dilihat sebagai berikut:

Tabel perhitungan Ziegler Nichols first method

Kontrol	*P*	*I*	*D*
P	$\frac{1}{K}\left(\frac{\tau}{T_d}\right)$	-	-
PI	$\frac{0.9}{K}\left(\frac{\tau}{T_d}\right)$	$0.33\ T_d$	-
PID	$\frac{1.2}{K}\left(\frac{\tau}{T_d}\right)$	$2\ T_d$	$0.5\ T_d$

Metode penalaan Cohen-Coon merupakan metode penyetelan parameter PID yang hampir mirip dengan metode Ziegler Nichols first method namun dalam persamaannya lebih kompleks dan metode Cohen-Coon memberikan waktu transien yang lebih cepat. Persamaan dalam menentukan parameter PID menggunakan metode Cohen-Coon ditampilkan sebagai berikut:

Tabel perhitungan Cohen Coon first method

Pengontrol	*P*	*im*	T_D
P	$\frac{1}{K}\frac{\tau}{T_d}\left(1+\frac{T_d}{3\tau}\right)$	-	-
PI	$\frac{1}{K}\frac{\tau}{T_d}\left(0.9+\frac{T_d}{12\tau}\right)$	$T_d\frac{30+3T_d/\tau}{9+20T_d/\tau}$	-
PID	$\frac{1}{K}\frac{\tau}{T_d}\left(\frac{4}{3}+\frac{T_d}{4\tau}\right)$	$T_d\frac{32+6T_d/\tau}{13+8T_d/\tau}$	$T_d\frac{4}{11+2T_d/\tau}$

Dari pemodelan sistem praktikum terakhir yang diperoleh (Tabel 8), dilakukan penalaan Ziegler Nichols dan Cohen Coon dengan bantuan Ms. Excel:

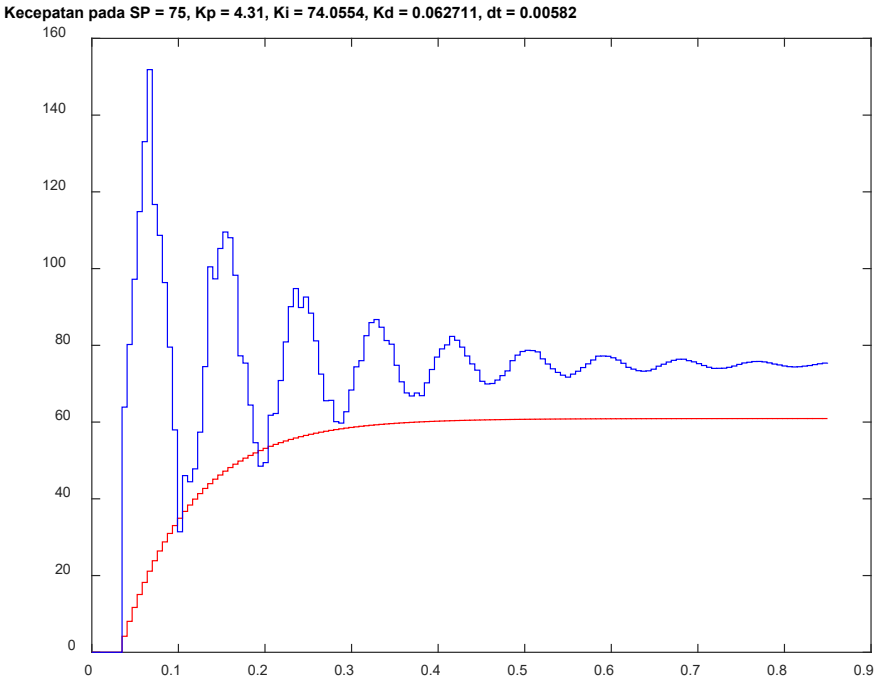
Tabel Parameter PID Dari Model Sistem Praktikum

Tabel X - Ziegler Nichols				**Tabel X - Cohen Coon**			
Kontroller	**P**	**I**	**D**	**Kontroller**	**P**	**I**	**D**

P	3.5954			**P**	4.0057		
PI	3.2358	0.0095		**PI**	3.3384	0.0568	
PID	4.3145	0.0581	0.01	**PID**	5.1016	0.0628	0.010

5.4.2 *Simulasi Respons Sistem dengan MATLAB*

Program Simulasi.m yang sudah ada diubah dengan mengubah nilai parameter (Gain, Tau, dan DeadTime) sesuai dengan Tabel8. Dengan menjalankan program Simulasi.m pada perangkat lunak Matlab, hasil yang diperoleh adalah sebagai berikut:

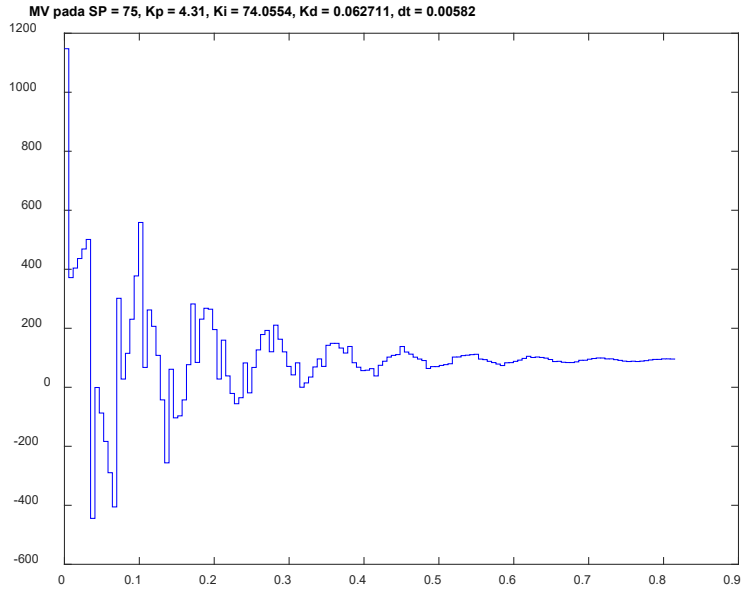


Gambar 13 Plot "Kecepatan pada SP"

Plot "Kecepatan pada SP" menunjukkan grafik *stairs* atau tangga, yakni grafik yang memetakan elemen y sesuai nilai x, di mana x dan y berupa vektor atau matriks dengan ukuran yang sama. Sumbu-x dibuat dari nol hingga FinalTime. Sumbu-y dibuat dari min(Velocity1) hingga max(Velocity1) untuk grafik berwarna merah dan min(Velocity2) hingga max(Velocity2) untuk grafik berwarna biru.

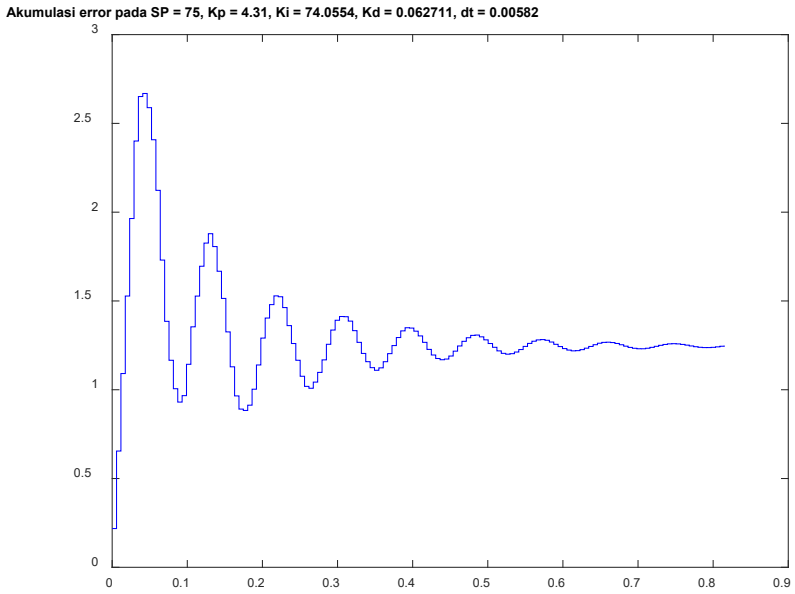
Plot merah yaitu Velocity1 menunjukkan kenaikan nilai hingga akhirnya tunak pada nilai tertentu. Velocity1 menunjukkan nilai kecepatan pada sistem kalang terbuka (*open loop system*). Sementara itu, plot biru yaitu Velocity2

berfluktuasi dan menunjukkan pola pengulangan. Velocity2 menunjukkan nilai kecepatan pada sistem kalang tertutup (*closed loop system*).



Gambar 14 Plot "MV pada SP"

Plot kedua menunjukkan grafik ManipulatedVariable (MV) terhadap waktu t. MV adalah penjumlahan dari nilai Mp, Mi, dan Md. Seperti yang dapat dibaca pada *source code Simulasi.m,* Mp adalah Kp*Error, Mi adalah Ki*AccumulationError(i), dan Md adalah Kd*RateError. Nilai Error diperoleh dari SetPoint dikurangi dengan Velocity2(i).



Gambar 15 Plot Akumulasi Error

Plot ketiga adalah plot nilai akumulasi error (AccumulationError) pada Set Point mulai dari rentang waktu t=1 hingga t=end-dlay-1. AccumulationError diberi nilai awal yakni nol, kemudian seiring penambahan waktu mengalami penambahan nilai.

5.4.3 *Penentuan Jenis Pengontrol (P, PI, atau PID) yang Paling Tepat*

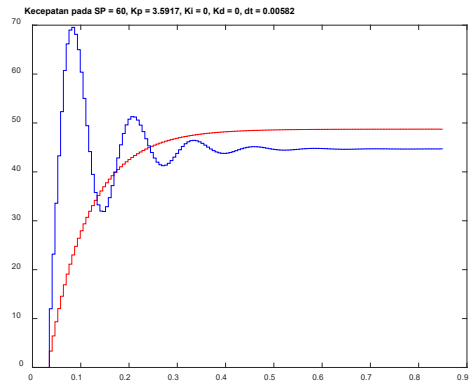
Menggunakan program simulasi seperti pada bagian sebelumnya, kini dicoba menggunakan 3 jenis pengontrol yang berbeda, yaitu pengontrol Proporsional, pengontrol Proporsional-Integral dan juga pengontrol Proporsional-Integral-Derivatif. Pada program simulasi.m kita dapat menukar jenis pengontrol dengan mengganti variabel type, yang nanti nya akan menentukan state dari switch case, seperti pada cuplikan kode berikut ini.

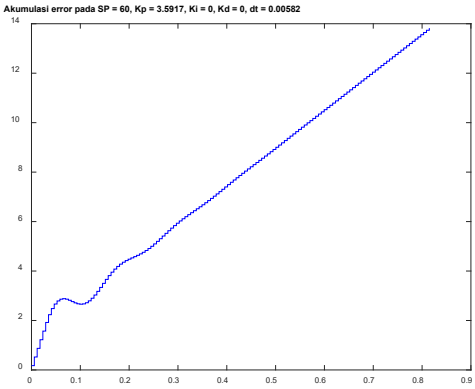
```
% Controller Design : Ziegler Nichols FOPDT
Type = 3; % 1:P, 2:PI, 3:PID, otherwise:Your Parameter Control
switch Type
    case 1
        % Proportional Controller
        ...
    case 2
        % Proportional-Integral Controller
        ...
    case 3
        % Proportional-Integral-Derivative Controller
        ...
    otherwise
        % Custom Control Parameter
```

```
...
end
```

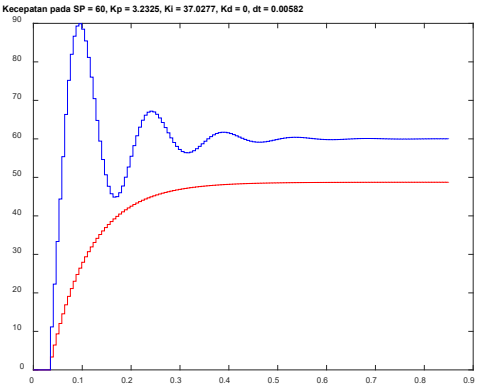
Berikut ini merupakan hasil simulasi untuk ketiga jenis pengontrol.

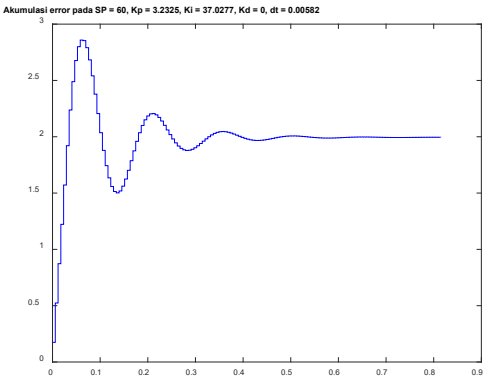
- Pengontrol P



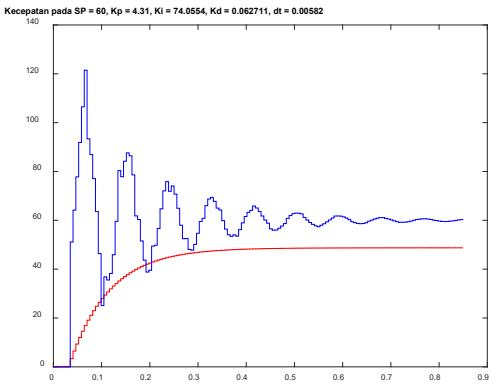


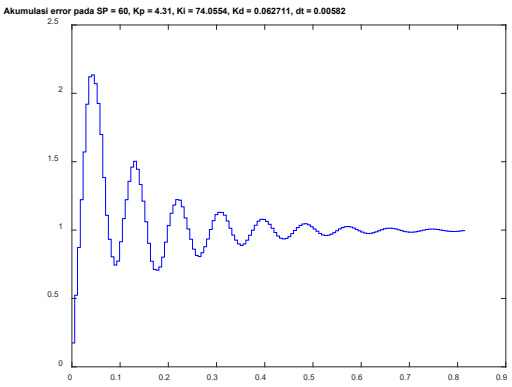
- Pengontrol PI





- Pengontrol PID





Berdasarkan hasil simulasi di atas, untuk pengontrol jenis proporsional, pada gambar sebelah kiri, hasil steady state nya masih meimiliki offset dengan

nilai set point, sehingga pada gambar sebelah kanan, error akan tetap ada dan bila dikumulatifkan akan terus meningkat. Hal tersebut menunjukkan bahwa pengontrol jenis proporsional masih belum cukup untuk dapat mengontrol kecepatan motor.

Selanjutnya untuk pengontrol jenis Proporsional Integral, pada grafik sebelah kiri, terlihat bahwa pengontrol sudah cukup untuk membuat kecepatan motor berjalan sesuai setpoint yang diinginkan. Pada grafik disebelah kanan, kita juga dapat melihat bahwa error semakin menurun dengan cepat ketika kecepatan motor mulai mendekati nilai steady state.

Selanjutnya untuk pengontrol jenis Proporsional Integral Derivativ, pada grafik sebelah kiri, terlihat bahwa pengontrol sudah cukup untuk membuat kecepatan motor berjalan sesuai setpoint yang diinginkan, walaupun begitu terlihat osilasi yang cukup banyak, sehingga menyebabkan settling time yang lama untuk sistem. Pada grafik disebelah kanan, kita juga dapat melihat bahwa error semakin konvergen ketika kecepatan motor mulai mendekati nilai steady state, namun terlihat osilasi yang cukup banyak, sehingga menyebabkan settling time yang lama untuk sistem.

Berdasarkan analisis di atas, maka jenis pengontrol yang paling baik untuk mengontrol kecepatan motor adalah jenis pengontrol Proportional Integral karena dapat membuat kecepatan motor sesuai dengan setpoint yang diinginkan, dan membuat settling time juga cepat.

5.4.4 *Pembuatan Fungsi Kontrol PID pada Pemograman*

Dibuat fungsi kontrol pada tab control, fungsi ini bertujuan untuk menghitung aksi kontrol PID. Berikut ini cuplikan kode program implementasi kontrol PID:

```
int P,I,D,LastError;
int kp = 0;
int ki = 0;
int kd = 0;

int controlPID(int feedback, int setPoint){
  int Error =  setPoint - feedback;
  P = kp*Error;
  I = I + (Error*ki);
  D = kd*(Error-LastError);
  LastError = Error;
  return P+I+D;
}
```

Bila kita melihat program tersebut, fungsi menerima 2 buah masukan yaitu setpoint yang akan menjadi referensi kecepatan motor akan dikontrol oleh pengontrol, selain itu juga menerima masukan feedback yang merupakan pembacaaan realtime dari tachometer terkait kecepatan motor saat ini. Pertama dihitung error yang merupakan selisih dari kedua inputan fungsi. Berdasarkan error tersebut, difromulasikan untuk menghitung parameter P,I,dan D.

Untuk parameter P dihitung dengan cara mengalikan konstanta proporsional (kp) dengan error. Untuk parameter I dihitung dengan cara melakukan sumasi atau penjumlahan antara parameter I sebelumnya dengan perkalian antara konstanta integral dengan error. Untuk parameter D dihitung dengan cara mengalikan error dengan selisih antara error saat ini dan error sebelumnya.

Setelah itu, untuk transisi loop berikutnya, nilai error saat ini dijadikan nilai last error atau error sebelumnya. Maka terakhir kita bisa mendapatkan hasil akhir dari pengontrol dengan menambahkan semua parameter P,I dan juga D.

6 Kesimpulan

Telah berhasil dilakukan akuisisi data posisi sudut putaran motor dari sensor potensiometer dan data kecepatan putar motor dari sensor tachometer. Untuk masing-masing besaran yang diukur, terdapat tiga metode yang digunakan. Metode pertama menggunakan fungsi analogRead sehingga data diambil langsung dari pin sensor. Metode kedua menggunakan fungsi sampling, yakni data diambil dari pin sensor kemudian dikonversi ke dalam satuan persen. Metode terakhir menggunakan fungsi smooth, yakni

beberapa data diambil dari pin sensor, dirata-ratakan, baru kemudian dikonversi menjadi persen. Pada plot grafik, hasil pengukuran dari ketiga metode

Untuk mengetahui parameter hubungan keluaran dan masukan pada motor DC, pertama-tama motor diberi masukan berupa angka 0 s.d. 100 (masukan positif) dan 0 s.d. -100 (masukan negatif). Keluaran yang diperoleh adalah kecepatan tunak terukur pada sistem kalang terbuka. Diketahui bahwa motor DC pada kit praktikum nomor 10 ini baru mulai berputar saat nilai masukan 21% (positif) dan –28% (negatif). Pada nilai masukan ini, nilai kecepatan yang terukur masih nol. Meski begitu, nilai kecepatan tunak terukur baru akan terdeteksi oleh sensor tachometer dan nilainya muncul pada Serial Monitor pada saat nilai masukan minimum 32% (positif) dan –34% (negatif). Sementara itu, nilai masukan yang menghasilkan putaran motor maksimal adalah 97% (positif) dan –100% (negatif). Kecepatan putaran motor maksimal tersebut adalah 79% (positif) dan 78% (negatif).

Dari hasil pengukuran, dipilih data dari metode ketiga. Selanjutnya, dibuat grafik hubungan masukan motor dan kecepatan tunak. Pada grafik, didapati adanya *deadzone.* Selain itu, grafik tersebut juga dibagi menjadi beberapa daerah linier dengan cara membuat perbandingan keluaran/masukan. Pertimbangan yang digunakan untuk menentukan pembagian daerah linear adalah nilai perbandingan tersebut dilihat pada satu angka pentingnya. Semua yang memiliki nilai perbandingan yang sama dimasukkan dalam rentang daerah linier yang sama. Dengan demikian, sebanyak 17 daerah linear dihasilkan.

Sebelumnya, pada grafik hubungan masukan motor dan kecepatan tunak terukur, didapati adanya *deadzone* yang harus diperbaiki dengan fungsi kompensasi. Dengan menggunakan fungsi kompensasi berikut, *deadzone* berhasil diperbaiki:

Untuk kompensasi di wilayah positif:

$$inputCompensated = 32 + \frac{inputSerial - 0}{100 - 0} \times (100 - 32)$$

Untuk kompensasi di wilayah negatif:

$$inputCompensated = -34 + \frac{inputSerial - 0}{-100 - 0} \times \big(-100 - (-34)\big)$$

Langkah selanjutnya adalah melakukan pemodelan untuk tiap respons dinamik masukan positif menggunakan model orde satu dengan waktu tunda (*dead time*). Untuk

masing-masing nilai masukan mulai dari 0 s.d. 100 dengan beda antarnilai 5, diperoleh parameter model kit praktikum SKD seperti keluaran tunak, gain, konstanta waktu, dan waktu tunda. Sebagai model final kit praktikum, dipilih nilai parameter untuk memodelkan sistem orde satu, yang persamaannya adalah sebagai berikut:

$$G(s) = \frac{0.844\, e^{-0.208s}}{0.098s\ +\ 1}$$

7 Referensi

[1] K. Ogata , *Modern Control Engineering*, Prentice Hall, 2010.

[2] N. S. Nise, *Control Systems Engineering*, John Wiley & Sons, 2020.

8 Lampiran

Data Praktikum TF4022_Kel10_Modul1.xlsx

9 Kontribusi

Tabel 5 Kontribusi Anggota Kelompok

Nama	NIM	Kontribusi Anggota
Yosep Putra Setiyanto	13320026	Melakukan coding saat praktikum Percobaan 1 - Prosedur Percobaan 2 - Prosedur Percobaan 3 - Prosedur Penjelasan coding Percobaan 3 - Analisis 4 - Kecepatan 0, 5, 10, 15, 20, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100 Tugas tambahan - 3 Tugas tambahan - 4
Dewi Ashiila H	13320032	Percobaan 1 - Analisis 1, 2 Percobaan 2 - Analisis 1 Percobaan 2 - Analisis 2 Percobaan 2 - Analisis 3 Percobaan 2 - Analisis 4 Percobaan 2 - Analisis 5 Percobaan 2 - Analisis 6 Percobaan 3 - Analisis 6 Tugas tambahan - 1 Percobaan 3 - Analisis 4 - Kecepatan 25, 30, 35, 40, 45
Zayna Mosa Kalila	13320045	Kesimpulan percobaan 1 Kesimpulan percobaan 2 Percobaan 1 - Analisis 3

		Percobaan 1 - Analisis 4 Percobaan 1 - Analisis 5 Percobaan 1 - Analisis 6 Tugas tambahan - 2
Daffa Auliya U S	13320052	Percobaan 2 - Analisis 7 Percobaan 3 - Analisis 1 Percobaan 3 - Analisis 2 Percobaan 3 - Analisis 3 Percobaan 3 - Analisis 5 Kesimpulan percobaan 3