PERHITUNGAN FUNGSI ALIH DAN KONTROL PID MOTOR PG45 PADA ROBOT DIPO-MECARO (MEDICAL ASSISTANT ROBOT) DI PT PERMALAT BERDIKARI JAYA

Ahmad Didik Setiyadi[1)], Hadha Afrisal, S.T., M.Sc.

Program Studi Sarjana Departemen Teknik Elektro, Universitas Diponegoro
Jl. Prof. Soedarto, SH, Kampus Undip Tembalang, Semarang, 50275, Indonesia
[1)]*e-mail: didik@student.undip.ac.id*

Abstrak

Fungsi alih dari suatu komponen merupakan perbandingan antara output transformasi laplace dengan input transformasi laplace dengan kondisi awal sama dengan nol. Dengan mengetahui fungsi alih suatu komponen kita dapat memprediksi keluaran dari komponen tersebut terhadap masukan yang diberikan. Begitu juga dengan fungsi alih motor DC, dengan mengetahui fungsi alih motor DC maka kita bisa mengetahui keluaran motor DC tersebut terhadap masukan yang kita berikan. Dipo-Mecaro (Medical Assistant Robot) merupakan sebuah omni-wheel mobile robot dengan empat motor PG45 yang dapat dioperasikan secara nirkabel melalui koneksi Wi-Fi sehingga dapat mengurangi kontak antara pasien Covid-19 dan tenaga medis. Perusahaan yang membuat motor PG45 yang digunakan Dipo-Mecaro tidak memberikan spesifikasi secara rinci, sehingga diperlukan pencarian fungsi alih yang berguna untuk digunakan perhitungan metode kontrol berikutnya. Pencarian fungsi alih dilakukan menggunakan masukan sinyal PWM acak dan keluaran kecepatan motor yang kemudian diolah dengan tool sistem identifikasi MATLAB. Hasil perhitungan yang dilakukan menghasilkan fungsi alih motor PG45 (45.71S + 0.6757)/(S^3 + 〖14.15S〗^2+24.59S + 10.93) dengan best fit 97.54% dan sistem teruji stabil. Perhitungan parameter Kontrol PID dihasilkan KP = 32.9279740191956, KI = 196.610205687394, dan KD = 1.20746780266902.
Kata Kunci: Fungsi Alih, Kontrol PID, Motor DC PG45

Abstract

The transfer function of a component is the ratio between the laplace transform output and the laplace transform input with the initial condition equal to zero. By knowing the transfer function of a component, we can predict the output of that component against the given input. Likewise with the DC motor transfer function, by knowing the DC motor transfer function, we can find out the DC motor output to the input we provide. Dipo-Mecaro (Medical Assistant Robot) is an omni-wheel mobile robot with four PG45 motors that can be operated wirelessly via a Wi-Fi connection so as to reduce contact between Covid-19 patients and medical personnel. The company that made the PG45 motorbike used by Dipo-Mecaro did not provide detailed specifications, so it was necessary to find a transfer function that was useful for calculating the next control method. The search for the transfer function is carried out using random PWM signal input and motor speed output which is then processed using the MATLAB identification system tool. The results of the calculations carried out resulted in the motor transfer function PG45 (45.71S + 0.6757)/(S^3 + 〖14.15S〗^2+24.59S + 10.93) with best fit 97.54% and the system was tested stable. Calculation of PID control parameters resulted in KP = 32.9279740191956, KI = 196.610205687394, and KD = 1.20746780266902.
Keywords: Transfer Function, PID Control,Motor DC PG45

1. Pendahuluan

1.1 Latar Belakang

Mahasiswa saat ini mendapatkan informasi dari berbagai sumber baik di bangku perkuliahan maupun di luar perkuliahan. Mahasiswa program sarjana mendapatkan program perkuliahan selama 4 tahun. Selama program perkuliahan, banyak hal yang dilakukan agar mahasiswa mampu mendapatkan ilmu secara maksimal. Beberapa prodi mewajibkan mahasiswa melakukan praktek kerja lapangan untuk mendapatkan ilmu penerapan yang selama ini dipelajari di bangku perkuliahan. Hal ini bertujuan agar kelak mahasiswa dapat menjadi tenaga ahli yang kompeten dan profesional. Salah satu media yang digunakan untuk menambah pengalaman mahasiswa dan siap menghadapi dunia kerja adalah dengan Kerja Praktik.

Kerja praktik merupakan salah satu mata kuliah wajib di Fakultas Teknik Universitas Diponegoro. Tujuan dari kerja praktik adalah memberikan fasilitas kepada mahasiswa untuk mengaplikasikan ilmu yang telah didapatkan selama berada di bangku perkuliahan, membuktikan kebenaran teori-teori dan dapat menimba ilmu pengetahuan guna peningkatan dan penguasaan ilmu pengetahuan dan teknologi. Pembuatan laporan ini didasari oleh pengalaman penulis selama melakukan kerja praktik di PT. Permalat Berdikari Jaya, Divisi MEP (Mechanical, Electrical, and Plumbing).

PT. Permalat Berdikari Jaya memiliki salah satu divisi utama yaitu Divisi MEP (Mechanical, Electrical, and Plumbing). yang memiliki fungsi sebagai perancangan dan pengawasan pada kegiatan yang berhubungan dengan bidang mechanical, electrical, dan plumbing pada proyek yang dijalankan di PT. Permalat Berdikari Jaya. Perancangan dan pengawasan dilakukan pada proyek yang dikerjakan oleh vendor saat kontrak diambil dengan PT.

Permalat Berdikari Jaya sehingga sesuai dengan keinginan customer.

Dipo-Mecaro (*Medical Assistant Robot*) merupakan proyek riset yang tengah dikerjakan oleh PT. Permalat Berdikari Jaya saat kerja praktik dilaksanakan. Customer dari proyek ini merupakan sebuah rumah sakit yang berkerjasama dengan PT. Permalat Berdikari Jaya. Dipo-Mecaro (*Medical Assistant Robot*) merupakan robot yang dapat dioperasikan secara nirkabel melalui koneksi Wi-Fi sehingga dapat mengurangi kontak antara pasien Covid-19 dan tenaga medis. Robot Dipo-Mecaro menggunakan empat motor DC PG45 sebagai penggerak utamanya. Namun perusahaan pemproduksi motor DC PG45 yang digunakan tidak memberikan spesifikasi fungsi alih dari motor, sehingga dalam perancangan kontrol menggunakan *trial error* tanpa perhitungan matematis. Sehingga perlu identifikasi sistem dari motor DC PG45 yang digunakan oleh robot Dipo-Mecaro.

Berdasarkan latar belakang tersebut, pada kerja praktik ini penulis mengambil judul "Perhitungan Fungsi Alih an Kontrol PID Motor PG45 Pada Robot Dipo-Mecaro (Medical Assistant Robot) Di PT. Permalat Berdikari Jaya".

1.2 Tujuan

Tujuan dari pelaksanaan Kerja Praktik ini adalah untuk mengetahui fungsi alih motor PG45 dan cara mengontrol kecepatan motor PG45 dengan metode PID untuk Robot Dipo-Mecaro (Medical Assistant Robot) di PT. Permalat Berdikari Jaya.

1.3 Batasan Masalah

Pembatasan ruang lingkup pembahasan laporan kerja praktik ini yaitu berfokus perhitungan fungsi alih fungsi alih motor PG45 dan cara mengontrol kecepatan motor PG45 dengan metode PID pada Robot Dipo-Mecaro (*Medical Assistant Robot*) di PT. Permalat Berdikari Jaya.

2. Kajian Pustaka

2.1 Motor DC

Motor DC adalah motor listrik yang memerlukan suplai tegangan arus searah (DC) pada kumparan medan untuk diubah menjadi energi gerak mekanik. Kumparan medan pada motor dc disebut stator (bagian yang tidak berputar) dan kumparan jangkar disebut rotor (bagian yang berputar). Motor arus searah, sebagaimana namanya, menggunakan arus langsung yang tidak langsung/direct-unidirectional [1]. Robot Dipo-Mecaro menggunakan empat buah motor DC PG45 mekanum sebagai penggerak utamanya. Motor PG45 pada robot dikontrol dengan mikrokontroller dengan bantuan *driver motor* EMS 30A H-Bride. Mikrokontroler mengatur kecepatan motor PG45 dengan cara mengatur besar pulsa PWM (*Pulse Width Modulation*). Yang mana penelitian yang sudah dilakukan sebelumnya, kecepatan motor DC dapat diatur menggunakan kerapatan pulse PWM yang terdapat dalam mikrokontroler ATMega2560 [2].

Gambar 1 Motor DC PG45

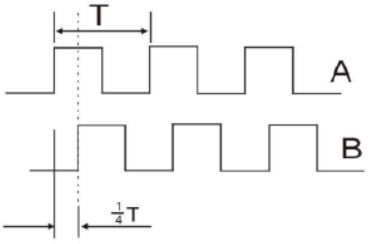
Motor DC PG45 yang digunakan pada kerja praktik ini memiliki spesidikasi sebagai berikut:

Tabel 1 Sepsifikasi Motor DC PG45

Fitur Teknis	Spesifikasi
Motor Type	*Brushed Motor*
Voltage	24V
Power Rated	100W
No Load Current	<1.5 A
Torque	15kgcm
No Load Speed	468 (±10%) Rpm
Rated Load Speed	398 (±10%) Rpm
Internal Encoder	37PPR (datasheet) 150PPR (praktis)

2.2 *Rotary Encoder*

Rotary Encoder atau encoder poros adalah perangkat elektromekanis yang mengubah sudut posisi/gerakan poros atau mengubah poros ke kode analog ataupun digital (eitel, 2014). Ada dua jenis utama rotary encoder, yaitu : absolut dan incremental. Sebuah *rotary encoder incremental* menyediakan output siklus (hanya) ketika encoder diputar. *Rotary encoder* dapat berupa mekanik atau optik. Jenis pada benda mekanik membutuhkan *debouncing* dan biasanya digunakan sebagai potensiometer digital pada peralatan termasuk perangkat konsumen.



Gambar 2 Keluaran Sinyal *Rotary Enocder*

Dalam kerja praktik ini menggunakan sesnor encoder internal dari motor DC PG45 yang mempunyai spesifikasi 37PPR, namun dalam praktisnya setelah dites memiliki ±150PPR

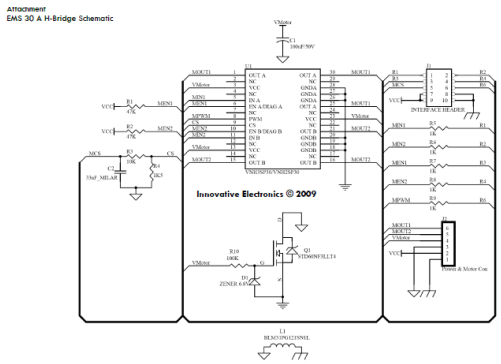
2.3 *Driver Motor H-Bridge*

Motor DC tidak bisa dikontrol secara langsung tanpa menggunakan *driever motor*. Driver motor ini dipasaran ada beberapa jenis, ada yang menggunakan IC, realy, ataupun mosfet. Driver motor jenis ini hanya memiliki satu keluaran, sehingga jika untuk mengontrol motor DC maka

hanya bisa maju atau berhenti saja. Ada jenis driver motor yang lebih *advance* dalam penggunaanya, yaitu *driver motor H-Bride*, driver motor ini memiliki dua keluaran, sehingga dalam pengaplikasiannya, motor DC dapat dikontrol untuk maju dan mundur. Dalam kerja praktik ini digunakan *driver motor EMS 30A H-Bridge* yang artinya bisa mengeluarkan arus sebesar 30A secara kontinu dalam penggunaanya. Spesifikasi driver motor yang digunakan bisa dilihat pada tabel 3.2.

Tabel 2 Spesifikasi *EMS 30A H-Bridge*

Fitur Teknis	Sepsifikasi
Driver	VNH2SP30
Tegangan Load	5.5 – 36 V
Arus	30A (kontinu)
Frekuensi PWM maksimum	10KHz
Keamanan	Arus pendek, kelebihan dan kekurangan tegangan, baterai terbalik, *overheating*
Mode	3 mode (Maju, Mundur, dan Normal/Diam)



Gambar 3 *Wiring EMS 30A H-Bridge*

Fungsi dari driver motor ini adalah untuk mengatur PWM ke motor PG45. Dalam pengaplikasian kontrol motor, PWM teknik modulasi dengan mengubah lebar pulsa (duty cycle). Dengan mengubah lebar pulsa maka kecepatan motor akan berubah dan kecepatan motor sebanding dengan lebar pulsa.

2.4 Mikrokontroler

Mikrokontroler merupakan sebuah chip IC yg dapat di program sesuai dengan kebutuhan penggunannya. Untuk mengisi program yang sesuai dengan kebutuhan pengguna dengan tipe data heksa (Hex file) dengan berisikan intruksi atau perintah untuk menjalankan sistem kontrol [3]. Dalam kerja praktik ini menggunakan mikrokontroler ATMEGA 2560 dengan sistem minimum Arduino Mega. Untuk mengisi program Arduini Mega dapat menggunakan Arduino IDE yang merupakan program open-source yang dibuat khusus untuk sistem minimum berbasis Arduino. Arduino Mega digunakan untuk mengambil data *sample* dari kecepatan motor PG45 terhadap masukan yang berupa PWM (*Pulse Width Modulation*) sehingga data masukan dan keluaran tersebut menjadi dataset yang akan diolah pada proses selanjutnya.

Gambar 4 Arduino Mega 2560

Dalam pengumpulan dataset, mikrokontroler membangkitkan sinyal PWM ke driver motor, kemudian driver motor mengolah sinyal PWM menjadi tegangan yang dilanjutkan ke motor PG45. Ketika motor PG45 mendapat arus listrik searah maka motor berputar dan sensor encoder internal dari motor PG45 membaca pergerakan kemudian dikirim ke dan diolah mikrokontroler ATMEGA 2560. Papan mikrokontroler yang digunakan memiliki spesifikasi seperti pada tabel 3.2.

Tabel 3 Spesifikasi Arduino Mega2560

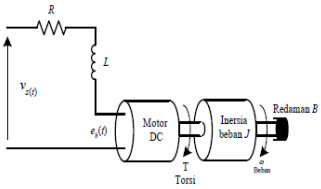
Fitur Teknis	Sepsifikasi
Tegangan Kerja	5V
I/O Pin Digital	54 (15 PWM) Pin
O/O Pin Analog	16 Pin
Arus per I/O	20 mA
Flash Memory	256 KB
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Dimensi	10,16 cm x 5,3 cm
Berat	37 Gram

2.5 Fungsi Alih

Dalam teori kontrol, fungsi yang disebut "fungsi alih" sering digunakan untuk mencirikan hubungan masukan dan keluaran dari sistem linier parameter konstan. Konsep fungsi alih hanya digunakan pada sistem linier parameter konstan, walaupun dapat diperluas untuk suatu sistem kontrol nonlinier [4]. Dengan fungsi alih kita bisa menganalisa dan menghitung keluaran sistem secara matematis tanpa menghidupkan komponen tersebut, dengan begitu kita bisa mengetahui keluaran suatu komponen hanya dengan perhitungan matematis. Dalam penggunaannya untuk motor PG45, kita bisa menghitung secara matematis percepatan, kecepatan, dan posisi dari motor. Secara tidak langsung fungsi alih ini sangat penting untuk mengetahui karakteristik suatu komponen sehingga mempermudah dalam pengontrolan komponen tersebut.

2.6 Fungsi Alih Motor DC

Secara umum motor DC memiliki komponen yang perlu diperhitungkan dalam perhitunngan matematis, komponen tersebut : resistansi jangkar, Induktansi jangkar, arus medan, momen inersia, dan koefisien gesek. Dari semua komponen tersebut dapat digambarkan hubungan antar komponen seperti pada **Gambar 5** dibawah.



Gambar 5 Model Motor DC

Dari rangkaian listrik dapat dibuat persamaan tegangan menurut hukum Kirchhoff tegangan seperti dinyatakan oleh persamaan berikut:

$$V_s(t) = Ri_a(t) + L\frac{di_a(t)}{dt} + e_b(t)$$

2.7 Fungsi Alih Motor DC

Untuk mengidentifikasi parameter motor DC ada beberapa metode yang bisa digunakan. Pada tahun 2012 Setiawan menghitung fungsi alih menggunakan *system identifier toolbox Matlab* dengan data masukan PWM acak dan keluaran kecepatan motor menghasilkan *best fit* 93% [2]. Pada tahun yang sama Wu berhasil menghitung persamaan fungsi alih motor DC menggunakan perhitungan matematis dengan *speed step respones* [5]. Pada tahun 2018 Septiarini berhasil menghitung pemodelan matematika kecepatan motor DC menggunakan identifikasi dengan metode 2s dan hasil perhitungan tersebut digunakan untuk perhitungan kontrol PID [6]. Dalam kerja praktik ini penulis menggunakan metode pertama untuk menghitung fungsi alih motor DC PG45 robot Dipo-Mecaro (*Medical Assistant Robot*). Pada tahun 2017 Tang berhasil melakukan identifikasi sistem dan *PID Tunning* dengan menggunakan *system identifier toolbox Matlab* dan *PID Tunner Matlab* menghasilkan *best fit* sebesar 68.71% [7]. Dalam kerja praktik ini penulis menggunakan metode yang sama dilakukan oleh Tang pada tahun 2017 yaitu menggunakan *system identifier toolbox Matlab* dan *PID Tunner Matlab* karena *system identifier toolbox Matlab* mampu mengestimasi *best fit* hasil yang diperoleh dan *PID Tunner Matlab* mampu mensimulasi hasil fungsi alih yang diperoleh untuk membuat sistem dengan karakteristik sesuai kebutuhan hanya dengan menggeser *time respon*-nya.

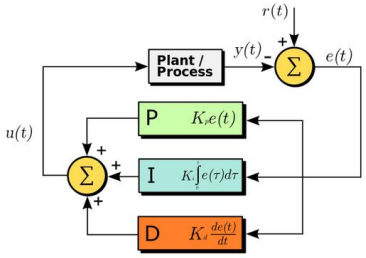
2.8 Kestabilan Sistem

Fungsi alih dikatakan apabila semua akar-akarnya berada disebelah kiri sumbuk y (bernialai negatif) [8]. Dalam kerja praktik ini penulis mengubah fungsi alih yang dihasilkan menjadi loop tertutup terlebih dahulu, karena fungsi alih yang akan dihasilkan akan digunakan dalam metode kontrol tertentu. Sehingga sistem yang digunakan adalah loop tertutup dan akar-akar dari loop tertutup tersebut yang menjadi acuan uji kestabilan sistem.

2.9 Kontrol PID

Kontrol PID adalah merupakan kontroler untuk menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya umpan balik pada sistem tesebut. Persamaan PID merupakan kontrol yang paling banyak digunakan dalam dunia industri [9]. PID sesuai memiliki parameter kontrol proposrsional (KP), kontrol integratif (KI), dan kontrol derivatif (KD). Pada tahun 2018 Septiarini berhasil mengontrol motor DC dengan perhitungan fungsi alih yang telah dihasilkan menggunakan metode 2s [6]. Dalam kerja praktik ini menggunakan metode *trial error* dengan *PID Tunner Matlab* menggunakan fungsi alih dari motor DC PG45 yang dihasilkan. Adapun persamaan kontrol PID sebagai berikut:

$$mv(t) = K_p(e(t) + K_i\int_0^t e(t)dt + K_d\frac{de(t)}{dt})$$



Gambar 6 Blok Diagram Kontrol PID

3. Perhitungan Fungsi Alih dan Parameter PID Motor DC PG45 Pada Robot Dipo-Mecaro (*Medical Assistant Robot*) Di PT. Permalat Berdikari Jaya

3.1 Robot Dipo-Mecaro (*Medical Assistant Robot*)

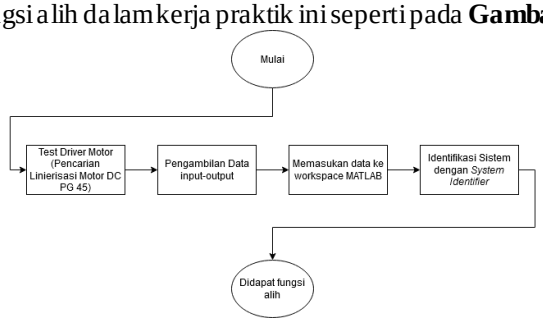
Robot Dipo-Mecaro (*Medical Assistant Robot*) merupakan salah satu riset yang berfungsi untuk membantu tenaga medis di kondisi pandemi Covid-19 sedang berlangsung. Robot ini mampu melakukan beberapa kegiatan yang dilakukan oleh perawat pada umumnya, seperti mengantarkan makanan dan obat-obatan, melakukan pemerikasaan sederhana seperti mengukur tekanan darah, serta mampu berkomunikasi dengan pasien. Robot Dipo-Mecaro merupakan mobile robot yang menggunakan konfigurasi omni-wheel yang dapat memungkinkan robot dapat bergerak kesegala arah.

Gambar 7 Robot Dipo-Mecaro (*Medical Assistant Robot*)

Riset ini dilakukan oleh salah bidang dari PT. Permalat Berdikari Jaya yaitu dengan "Media Geometri (MG)" yang bergerak dibidang produksi alat medis.

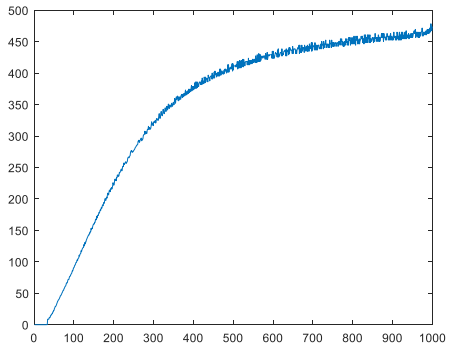
3.2 Pengujian *Driver*

Tahap pertama dalam pencarian fungsi alih adalah pengujian *driver motor*. Langkah-langkah dalam pencarian fungsi alih dalam kerja praktik ini seperti pada **Gambar 8.**



Gambar 8 Alur Pencarian Fungsi Alih Motor DC PG45

Pengujian driver motor bertujuan untuk melihat range linierisasi dari motor PG45. Driver motor yang digunakan adalah EMS H-Bridge 30A yang dikontrol menggunakan Arduino Mega. Pengujian pada kerja praktik ini menggunakan PWM 1KHz yang akan dikonversi dengan fungsi *mapping* menjadi 0-255 sesuai spesifikasi dari Arduino. Pengujian ini bertujuan untuk mengetahui linieritas dari motor DC yang diuji dengan cara mengirim sinyal PWM dari nilai minimum hingga maksimum dan memonitor hasil kecepatan dari pengaruh perubahan PWM tersebut, sehingga diketahui linieritas dari motor DC. Percobaan ini menghasilkan data masukan PWM dan keluaran kecepatan motor seperti pada **Gambar 9.**



Gambar 9 Grafik Kecepatan Motor DC PG45 Dengan Masukan PWM

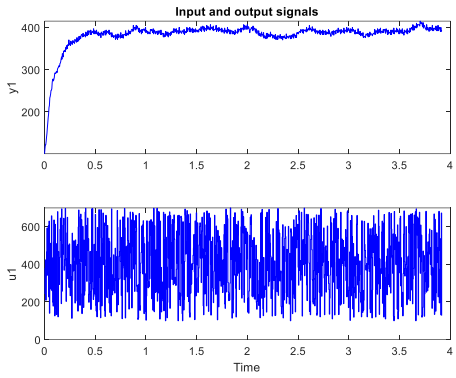
Bisa dilihat bahwa kecepatan motor mulai naik secara signifikan dari PWM 100-700. Dengan demikian linieritas dari motor berada pada PWM 100-700.

3.3 Pengambilan Data Dengan PWM Acak

Setelah diketahi batasan linieritas motor, maka tahap selanjutnya adalah mengambil data kecepatan motor PG45 terhadap masukan PWM dengan nilai acak rentang nilai 100-700. Pengambilan data kerja praktik ini dilakukan menggunakan perangkat yang sama seperti test driver motor. Namun berbeda program, sehingga mikrokontroler menjalankan program yang berbeda.

Program menggunakan fungsi random dengan nilai minimum 100 dan nilai maksimum 700, nilai random tersebut dikonversi menjadi 0-255 dengan fungsi *map*. Pengambilan data ini dilakukan dengan *loopingi* setiap 4ms. Pencuplikan dengan 4ms ini didasari dengan kontrol robot Dipo-Mecaro yang mempunyai *loop* setiap kerjanya 4ms yang artinya semua *task* pada robot Dipo-Mecaro dieksekusi setiap 4ms. Sehingga percobaan ini diatur sedemikian rupa dengan *loop* 4ms.

Data hasil dari percobaan ini didaptakan data sebagaimana pada **Gambar 10.**

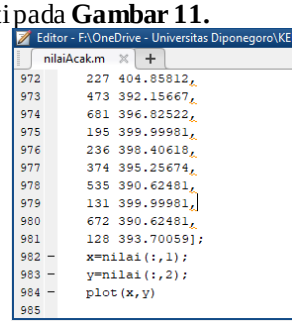


Gambar 10 Grafik Kecepatan Motor PG45 Dengan Masukan PWM Acak 10-70%

Data yang didapat diatas kemudian diproses menggunakan *System Identifer* di *Matlab*.

3.4 Identifikasi Sistem Dengan *System Identifier Toolbox Matlab*

Data yang sudah didapatkan pada proses seblumnya diolah menggunakan Matlab. Data tersebut dimasukan ke *Workspsace Matlab*. *Listing program* memasukan data tersebut seperti pada **Gambar 11.**



Gambar 11 Memsukan Hasil Data Ke *Workspace Matlab*

Pada gamabar 4.6 data sebelah kiri adalah masukan PWM acak dan yang sebelah kanan adalah feedback kecepatan motor DC PG45 yang dideteksi dengan sensor encoder. Memisahkan matriks data masukan dan keluaran yang masing-masing berjummlah 980 data menggunakan program:

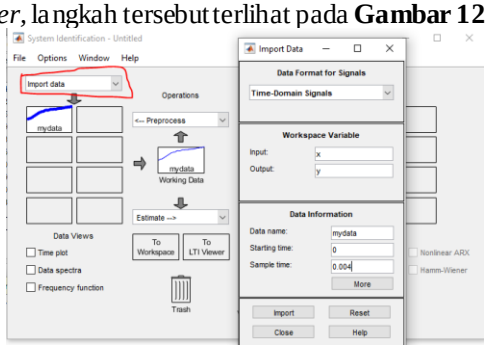
```
X=nilai(:,1);
Y=nilai(:,2);
```

Dengan sintaks diatas dipisahkan data masukan PWM dan keluaran kecepatan motor DC PG45 yang mana variabel x berisi semua masukan PWM dengan mengambil

semua kolom 1 dan variabel y berisi semua keluaran kecepatan motor DC PG45 dengan mengambil semua kolom 2 dari matrik nilai. Dari data variabel x dan y, selanjutnya diproses menggunakan *System Identifier Toolbox Matlab*.

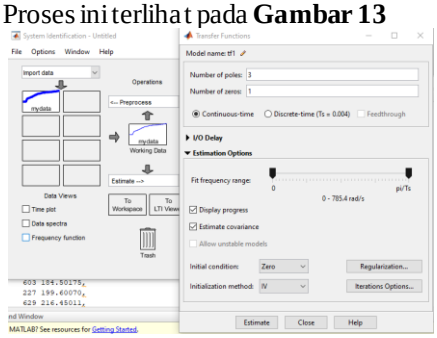
Setelah *dataset* berhasil dimasukan kedalam *workspace* proses selanjutnya adalah mengidentifikasi menggunakan *System Identifer Matlab*. Untuk memanggilnya bisa menggunakan sintask 'ident' untuk Matlab versi lama dan 'SystemIdentifier' pada untuk Matlab versi yang terbaru pada *command window*. Selain menggunakan *command windows* bisa meng-klik '*System Identifier* pada header APP di Matlab.

Dengan *system identifier* memasukan *dataset* dari *workingspace* Matlab ke dalam *working data* pada *system identifier*, langkah tersebut terlihat pada **Gambar 12**



Gambar 12 Memasukan Data ke *System Identifier Toolbox*

Pada proses ini memilih *import data* pada **Gambar 12** yang terlingkar warna merah dengan pilihan *Time Domain Signals*, karena dalam percobaan kerja praktik ini bekerja pada domain waktu. Kemudian memasukan variabel x sebagai *input* dan y sebagai *output* yang mana merupakan *datset* dari *working space*. Menggunakan *Starting time 0* karena data diambil dari waktu ke nol dan menggunakan *sample time 0.004* karena waktu pencuplikan atau pengambilan tiap data adalah 4ms. Setelah *dataset* berhasil dimasukan kedalam *working data system identifier* proses selanjutnya adalah mengidentifikasi dengan fungsi *Estimate*. Proses ini terlihat pada **Gambar 13**



Gambar 13 Estimasi Fungsi Alih Dengan *System Identifier Toolbox*

Dalam proses ini dilakukan estimasi fungsi alih dari data di working data dengan variabel mydata yang sudah dimasukan sebelumnya. Seperti pada **Gambar 13**, klik '*Estimate*' yang sudah dilingkari merah, kemudian pilih '*Transfer Function Model*' karena yang ingin dicari adalah model fungsi alihnya, maka akan muncul jendela 'Transfer Function' seperti pada **Gambar 13** diatas. Dalam kerja praktik ini menggunakan dataset data masukan dan keluaran. Disini penulis memilih 3 pole dan 1 zero dengan initial condition zero. Pemilihan pole dan zero ditentukan berdasarkan penelitian sebelumnya, salah satunya adalah penelitian yang dilakukan oleh Setiawan pada tahun 2012 tentang kontrol kecepatan motor DC dengan metode PID menggunakan Visual Basic 6.0 dan mikrokontroler ATmega 16 berhasil menghasilkan best fit 93.35% dengan 3 pole dan 2 zero [2].

Proses estimasi parameter pada kerja praktik ini dilakukan sebanyak 5 kali dengan target minimal menghasilkan *best fit* yang sama atau lebih tinggi dari penelitian sebelumnya yang dilakukan oleh Setiawan pada tahaun 2012 dan Tang pada tahun 2017. Hasil percobaan pertama mengikuti parameter yang dilakukan oleh Setiawan yaitu 3 pole dan 2 zero menghasilkan dengan dataset sebanyak 10000 data menghasilkan *best fit* 62.42% dengan sistem yang tidak stabil, percobaan kedua menggunakan menggunakan data yang sama dan mengurangi zero menjadi 1 menghasilkan *best fit* 62.41% dengan sistem yang stabil. Percobaan ketiga menggunakan data yang lebih sedikit, dataset sebanyak 1000 data dengan 3 pole dan 1 zero menghasilkan *best fit* 89.3% dengan sistem yang stabil. Disini bisa dilihat bahwa penggunaan dataset 1000 data menghasilkan best fit yang lebih baik. Percobaan 4 dan 5 menggunakan dataset sebanyak 980 data dengan mengabaikan data awal motor saat pertama berakselerasi. Pada percobaan 4 menggunakan 3 pole dan 2 zero menghasilkan *best fit* 97.54%. percobaan 5 menggunakan data yang sama namun dengan 3 pole dan 1 zero menghasilkan *best fit* sebesar 97.64% dengan persamaan fungsi alih (45.71S + 0.6757)/(S^3 + 〖14.15S〗^2+24.59S + 10.93). Sehingga bisa dikatakan fungsi alih dari motor DC PG45 dengan *driver motor EMS 30A H-Bridge* pada robot Dipi-Mecaro adalah (45.71S + 0.6757)/(S^3 + 〖14.15S〗^2+24.59S + 10.93). Hasil ini lebih baik dari penelitian terbaru identifiksi sistem dengan metode dan kontroler yang sama, yaitu dengan Arduino Mega2560. Pada tahun 2017 Tang melakukan penelitian identifikasi motor DC menggunakan System Identifer Tool Matlab dengan Arduino Mega2560 menghasilkan *best fit* hanya sebesar 68.71% dengan 1 zero dan 2 pole [7]. Hasil final estimasi dapat dilihat pada **Gambar 14**. Secara *default* masukan dan keluaran variabel akan beganti nama menjadi u1 sebagai masukan dan y1 sebagai keluaran yang mana sebelumnya bernama x dan y. Semua dataset kerja praktik ini bisa diakses di https://github.com/ahmaddidiks/SystemIdentifier/tree/master/matlab.

```
From input "u1" to output "y1":
        45.71 s + 0.6757
  ---------------------------------
  s^3 + 14.15 s^2 + 24.59 s + 10.93
Name: 1zero3pole
Continuous-time identified transfer function.

   Use "tfdata", "getpvec", "getcov" for parameters and

Status:
Estimated using TFEST on time domain data "mydata".
Fit to estimation data: 97.54% (stability enforced)
FPE: 11.79, MSE: 11.68
```

Gambar 14 Fungsi Alih Motor PG45

3.5 Analisa Kestabilan Sistem

Dalam kerja praktik ini penulis menggunakanmetode ini untuk melihat kestabilan fungsi alih yang sudah didapatkan sebelumnya. Percobaan ini dilakukan menggunakan Matlab dengan listing program sebagai berikut:

```
num = [45.71 0.6757];
den = [1 14.15 24.59 10.93];
%fungsi alih open loop
disp("Fungsi alih open loop")
sys = tf(num,den)
%fungsi alih close loop
disp("Fungsi alih close loop")
sys1 = feedback(sys,1)
%akar-akar persamaan karakteristiknya
disp("Akar-akar pers.
karakteristiknya")
damp(sys1)
%Posisi akar2 pes. karakteristik
sgrid
pzmap(sys1)
grid on
```

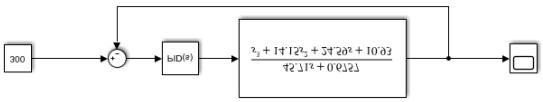
Karena fungsi alih akan digunakan dalam metode kontrol maka fungsi alih yang sudah didapatkan dibuat menjadi loop tertutup terlebih dahulu dengan nama variable sys1. Dengan begitu akar-akar persamaan karakteristik yang dicari adalah dari fungsi alih yang sudah dibuat menjaid loop tertutup. Fungsi alih loop tertutup yang dihasilkan adalah $\frac{45.71S + 0.6757}{S^3 + 14.15S^2 + 70.3S + 11.61}$ dengan akar-akarnya berada pada -0.171, -6.99 + 4.37i dan -6.99 – 4.37i. Semua akar-akarnya minus yang berarti bahwa fungsi alih loop tertutup tersebut stabil sehingga tidak perlu memodifikasi fungsi alih yang sudah di dapatkan agar stabil.

```
sys1 =

         45.71 s + 0.6757
  ---------------------------------
  s^3 + 14.15 s^2 + 70.3 s + 11.61

Continuous-time transfer function.

Akar-akar pers. karakteristiknya

        Pole              Damping       Frequency      Time Constant
                                      (rad/seconds)      (seconds)

 -1.71e-01                1.00e+00      1.71e-01          5.85e+00
 -6.99e+00 + 4.37e+00i    8.48e-01      8.24e+00          1.43e-01
 -6.99e+00 - 4.37e+00i    8.48e-01      8.24e+00          1.43e-01
>>
```

Gambar 15 Analisa Kestabilan Fungsi Alih Loop Tertutup

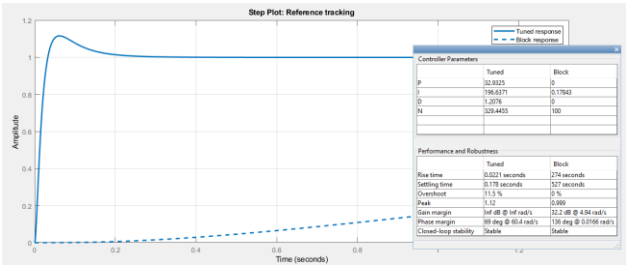
3.6 Pencarian Parameter PID

Setelah dipastikan fungsi alih yang didapatkan stabil, kontrol kecepatan motor DC PG45 dengan metode kontrol. Dalam kerja praktik ini dilakukan dengan metode kontrol PID. Pencarian parameter KP, KI, dan KD dilakukan secara trial error menggunakan PID Tunner Matlab. Teknik ini digunakan menggunakan Simulink dengan fungsi alih yang sudah didapatkan sebelumnya. Masukan dengan nilai nilai konstan dan keluaran dipantau dengan scope. Rangakain simulasi simulink yang dilakukan dapat dilihat pada **Gambar 16**



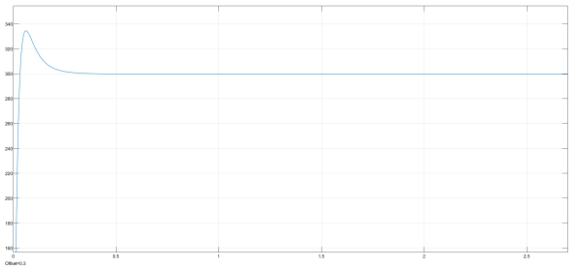
Gambar 16 Rangkaian Simulasi Simulink

Penentuan parameter menggunakan *PID Tunner, trial error* dengan menggeser *time respons*e untuk mendapatkan karakteristik kontrol yang dikehendaki. Pemilihan parameter KP, KI, KD pada kerja praktik ini didasari pada penggunaan motor DC PG45 secara langsung di robot Dipo-Mecaro. Pengendalian motor menggunakan setpoin yang selalu berubah-ubah setiap waktu sesuai keluarandari algoritma *plath planning,* Sehingga dibutuhkan kontrol yang mampu mencapai *steady state* secepat mungkin. Pencarian parameter dilakukan hanya sekali menggunakan fungsi alih yang sudah didapatkan sebelumnya dengan menggeser time respone secepat mungkin namun mempertimbangkan *overshoot* yang dihasilkan. Hasil terbaik yang didapat adalah dengan KP = 32.9279740191956, KI = 196.610205687394, dan KD = 1.20746780266902. karakteristik kontrol dari PID tersebut adalah dengan *Rise time* = 0.0221 detik atau 22.1ms, *Settling time* = 0.178 detik atau 178ms, *overshoot* = 11.5%.



Gambar 17 Hasil Parameter PID dengan *PID Tunner*

Selanjutnya dilakuakn simulasi dengan memasukan nilai setpoint 300RPM pada Simulink menghasilkan hasil simulasi seperti pada **Gambar 18**



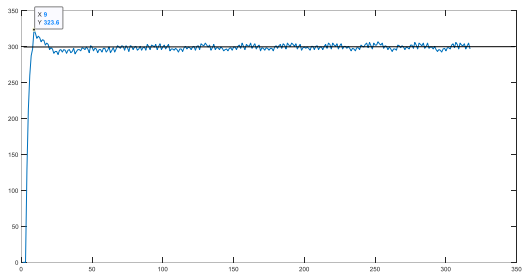
Gambar 18 Hasil Simulasi Dengan Setpoin = 300, KP = 32.9279740191956, KI = 196.610205687394, dan KD = 1.20746780266902

Dari hasil simulasi memperlihatkan terjadi overshoot yang tinggi sampai hampir menyentuh 340 dari setpoin 300, namun setelah itu langsung mencapai steady state sebelum 0.5 detik.

3.7 Aplikasi Hasil PID Kontrol Kecepatan Motor DC PG45

Hasil perhitungan parameter PID sebelumnya diaplikasikan langsung ke motor DC PG45. Rangkaian yang digunakan sama seperti tahap sebelumnya dari pengujain *driver motor*. Pada pengujain ini dilakukan dengan memberi setpoin pada program dan melihat performa dari kontrol PID yang sudah ditentukan sebelumnya.

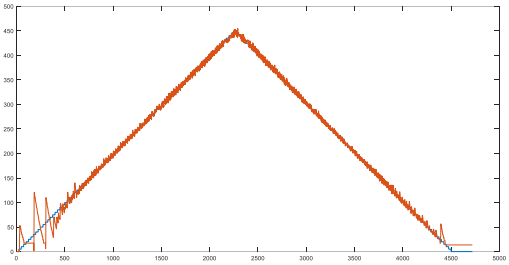
Dalam kerja praktik ini menggunakan *library PID* dari Arduino untuk memudahkan pemograman, tentunya dengan parameter PID yang telah ditemukan sebelumya. Keluaran sinyal PID diatur dengan rentang nilai 0-1000 yang kemudian di-mapping menjadi 0-255 karena limitasi PWM Arduino. Dilakukan pengujian dengan setpoin konstan 300RPM.



Gambar 19 Performa PID Dengan Setpoin 300 RPM

Hasil dari percobaan dengan setpoin 300 RPM terlihat pada **Gambar 19** yang mana seperti hasil simulasi, performa dari PID mempunyai *overshoot* yang cukup tinggi, pada percobaan mencapai 323.8 RPM (7.93%) dari setpoin 300 RPM, namun keadaan langsung stabil sebelum mencapai 50ms. Hasil ini memiliki sistem yang mencapai *steady state* lebih cepat dari penelitian sebelumnya yang dilakukan oleh Djalal pada tahun 2017 menggunakan PID berbasis *Bee Colony* mencapai *steady state* pada waktu 100ms [1]. Hasil ini juga lebih cepat dari penelitian sebelumnya yang dilakukan oleh Masrukhan pada tahun 2016 menggunakan PID *Tunning Ant Colony* (ACO) mencapai *steady state* pada waktu 500ms [10].

Dibutuhkan sistem yang mampu mencapai *steady state* karena dalam pengaplikasiannya, setpoin kecepatan robot Dipo-Mecaro selalu berubah sesuai keluaran algoritma *path planning*. Selanjutnya pengujian dengan setpoin yang berubah ±5RPM setiap 100ms dari nol mencapai 450 RPM dan kembali ke nol lagi, dihasilkan data seperti pada **Gambar 19** dibawah.



Gambar 20 Performa PID Dengan Setpoin Berubah-ubah

Dari **Gambar 20** garis warna biru adalah setpoin yang diberikan dan garis warna coklat adalah kecepatan dari motor PG45. Bisa dilihat pada awal gerak setpoin kecepatan dibawah 100RPM kontrol PID tidak terlalu bagus mengikuti setpoin, disaat di jangkauan yang sama saat berhenti performa kontrol kecepatan bisa mengikuti dengan baik. Setalah setpoin diatas 100RPM performa kontrol kecepatan sangat baik dengan perubahan kecepatan tiap 100ms mampu secara cepat menuju *steady state*.

4. Penutup

4.1 Kesimpulan

Dari Kerja Praktik yang telah dilakukan di PT. Permalat Berdikari Jaya, dapat diambil kesimpulan sebagai berikut:

1. Diketahui bahwa dapat dilakukan perhitungan identifikasi sistem model matematis fungsi alih dari motor DC PG45 dengan data masukan PWM acak dan keluaran kecepatan motor DC PG45 menggunakan System Identifier Toolbox Matlab menghasilkan best fit 97.64% dengan persamaan fungsi alih (45.71S + 0.6757)/(S^3 + 〖14.15S〗^2+24.59S + 10.93)
2. Diketahui bahwa fungsi alih yang dihasilkan telah teruji stabil dengan menguji akar-akar persamaan karakteristiknya menghasilkan fungsi alih loop tertutupnya adalah (45.71S + 0.6757)/(S^3 + 〖14.15S〗^2+70.3S + 11.61) dengan akar-akarnya berada pada -0.171, -6.99 + 4.37i dan -6.99 – 4.37i
3. Kontrol kecepatan motor DC PG45 dengan PID menggunakan fungsi alih pada poin 1 menghasilkan parameter KP = 32.9279740191956, KI = 196.610205687394, dan KD = 1.20746780266902
4. Keandalan sistem kontrol PID pada poin 3 memiliki overshoot yang cukup tinggi namun mampu menuju steady state yang sangat cepat. Pada percobaan setpoin 300RPM overshoot yang dihasilkan mencapai 23.8 (7.93%) namun mencapai steady state kurang dari 100ms.

4.2 Saran

1. Dalam pengukuran kecepatan motor menggunakan sensor encoder internal memiliki PPR yang kecil sehingga keakurasiannya tidak tinggi, perlu adanya sensor encoder eksternal dengan spesifikasi PPR yang tinggi untuk memperbaiki akuisisi data
2. Perlu adanya perbaikan kontrol pada saat motor diatur pada setpoin kecepatan rendah (100RPM)
3. Menggunakan metode kontrol yang lebih baik dan modern, dengan adanya fungsi alih maka sistem bisa digunakan untuk membangun kontrol dengan metode kontrol yang lebih baik, misalnya dengan kontrol fuzzy atau dengan jaringan syaraf tiruan.

Referensi

[1] Djalal, M.R., Nurohmah, H., Imran, A. and Yunus, M.Y. (2017). Aplikasi Metode Cerdas untuk Optimasi Controller PID Motor DC Berbasis Firefly Algorithm. Jurnal Nasional Teknik Elektro, 6(2), pp.76-83.

[2] Setiawan, D. (2017). Sistem Kontrol Motor Dc Menggunakan PWM Arduino Berbasis Android System. Jurnal Sains dan Teknologi Industri, 15(1), pp.7-14.

[3] Desnanjaya, I.G.M.N. and Iswara, I.B.A.I. (2018). Trainer Atmega32 Sebagai Media Pelatihan Mikrokontroler Dan Arduino. Jurnal RESISTOR (Rekayasa Sistem Komputer), 1(1), pp.55-64

[4] Sumardi. (2019). Buku Ajar: Sistem Kontrol Multivariabel, UNDIP Press, Semarang.

[5] Wu, W. (2012). DC motor parameter identification using speed step responses. Modelling and Simulation in Engineering.

[6] Septiarini, A.D. (2018). Pemodelan Matematika Kecepatan Motor DC Menggunakan Identifikasi Dengan Metode 2S (Doctoral dissertation, Institut Teknologi Sepuluh Nopember).

[7] Tang, W.J., Liu, Z.T. dan Wang, Q. (2017) , July. Dc motor speed control based on system identification and pid auto tuning. In 2017 36th Chinese Control Conference (CCC) (pp. 6420-6423). IEEE.

[8] Triwiyatno, A. (2019). *Buku Ajar: Sistem Kontrol Analog*, UNDIP Press, Semarang

[9] Setiawan, I. (2015). Kontrol PID Untuk Proses Industri, PT Elex Media Komputindo, Surabaya

[10] Masrukhan, M.N., Mulyo, M.P., Ajiatmo, D. and Ali, M. (2016). Optimasi Kecepatan Motor DC Menggunakan Pid Dengan Tuning Ant Colony Optimization (ACO) Controller. SENTIA 2016, 8(2).

Biodata

Ahmad Didik Setiyadi, lahir di Boyolali, 2 Mei 1998. Lulus pendidikan SDN Pakang, SMP Negeri 1 Andong, dan SMA Negeri 1 Gemolong. Pada tahun 2017 penulis melanjutkan studi di Program Studi Sarjana Teknik Elektro Universitas Diponegoro dengan konsentrasi Teknik Kontrol dan Instrumentasi dan saat ini masih menempuh pendidikan Strara 1 (S1).

Saya menyatakan bahwa segala informasi yang tersedia di makalah ini adalah benar, merupakan hasil karya sendiri, bebas dari plagiat, dan semua karya orang lain telah dikutip dengan benar

Ahmad Didik Setiyadi
NIM. 21060117120024

Pengesahan

Telah disetujui untuk diajukan pada seminar Kerja Praktik.

Semarang, 21 Desember 2020

Mengetahui,
Dosen Pembimbing

Hadha Afrisal, S.T., M.Sc.
NIP H.7. 199104172018071002

Listing Program Arduino:

```
/* program for collecting dataset PG45
   Author     : Ahmad Didik Setiyadi
   Last edited  : 4 Des 2020

*/

#include <AutoPID.h>
int a;
int b=0;
const float pi = 3.14159267;
double setPoint = 0;
double pwm, speedInRPM;
//waktu inc 0-1000 = 10000*10 (sampling time = 10ms) = 10000
#define waktuAmbilData 15000 //berapa lama ambil datanya (ms)

#define KP 32.9279740191956//2.0086510034733
#define KI 196.610205687394//29.8109534766749
#define KD 1.20746780266902
#define OUTPUT_MIN 0
#define OUTPUT_MAX 1000

#define LED_PIN 13
#define motor 5
#define pinA 6
#define pinB 7
#define ClockPin 2 // Must be pin 2 or 3
#define DataPin 9 // can be any other pin
      // My Encoder has 150 Clock pulses per revolution
      // note that 399999.8 = (60 seonds * 1000000 microseconds)microseconds in a
 minute / 150 pulses in 1 revolution)
      // change the math to get the proper multiplier for RPM for your encoder
#define Multiplier 399999.8//150000.0 // don't forget a decimal place to make thi
s number a floating point number

long nilaiRandom;
unsigned long waktuAwal,waktuAkhir;

volatile long EncoderCounter = 0;

//input/output variables passed by reference, so they are updated automatically
AutoPID myPID(&speedInRPM, &setPoint, &pwm, OUTPUT_MIN, OUTPUT_MAX, KP, KI, KD);
```

```
void setup() {
  // put your setup code here, to run once:
pinMode(motor, OUTPUT);
pinMode(pinA, OUTPUT); digitalWrite(pinA, LOW);
pinMode(pinB, OUTPUT); digitalWrite(pinB, HIGH); //FORWARD
pinMode(ClockPin, INPUT);
pinMode(DataPin, INPUT);
pinMode(LED_PIN, OUTPUT);
attachInterrupt(0,onPin2CHANGECallBackFunction,RISING);


myPID.setBangBang(200);
myPID.setTimeStep(1);

Serial.begin(57600);
Serial.println("Serial Test");
delay(500);
for (int i=0;i<=5;i++){
  Serial.print("Collecting dataset in "); Serial.println(5-i);
  delay(1000);
        }
waktuAwal = millis();

waktuAkhir = waktuAwal+waktuAmbilData;
}

void loop() {
  // put your main code here, to run repeatedly:
static unsigned long SpamTimer;

while(SpamTimer <= waktuAkhir ){
  if ( (unsigned long)(millis() - SpamTimer) >= (3)) {

    SpamTimer = millis();


    //run this code for motor driver test
    //motorDriverTest();
    //Run this code for collecting data
    //nilaiAcak();

    //Run this code for PID Test
    //a++;
    //time = a*4ms (estimated)
    //if(a==25) { a=0;
```

```
    //            b+=1;
    //            setPoint = sin(b)*180/pi;
    //            if (b==180) b=180;
    //            }
    PIDTest();
    }
  }
  if(SpamTimer >= waktuAkhir) analogWrite(motor,0); //motor off
}

void onPin2CHANGECallBackFunction(uint32_t Time, uint32_t PinsChanged, uint32_t P
ins){
    static uint32_t lTime; // Saved Last Time of Last Pulse
    uint32_t cTime; // Current Time
    cTime = micros(); // Store the time for RPM Calculations
    int32_t dTime; // Delt in time

// Encoder Code
    bool DataPinVal = digitalRead(DataPin);
// We know pin 2 just went high to trigger the interrupt
// depending on direction the data pin will either be high or low
    EncoderCounter += (DataPinVal) ? 1 : -1; // Should we step up or down?
// End Encoder Code

// calculate the DeltaT between pulses
    dTime = cTime - lTime;
    lTime = cTime;
    speedInRPM = Multiplier / ((DataPinVal) ? dTime: (-
1 * dTime)); // Calculate the RPM Switch DeltaT to either positive or negative to
 represent Forward or reverse RPM
}

void motorDriverTest(){
     nilaiRandom += 1;
     analogWrite(motor,map(nilaiRandom, 0, 1000, 0, 255)); //set pwm motor as ran
dom number
      //show nilaiRandom value as a PWM
     Serial.print(nilaiRandom);
     Serial.print(" ");

     //show Encoder value
    //Serial.println(EncoderCounter);
    //Serial.print(" ");

    //show speed in RPM
```

```
    Serial.print(speedInRPM, 5);
    Serial.println(",");
}

void nilaiAcak(){
     nilaiRandom = random(100,700); //10-70%
     analogWrite(motor,map(nilaiRandom, 0, 1000, 0, 255)); //set pwm motor as ran
dom number
      //show nilaiRandom value as a PWM
     Serial.print(nilaiRandom);
     Serial.print(" ");

     //show Encoder value
    //Serial.println(EncoderCounter);
    //Serial.print(" ");

    //show speed in RPM
    Serial.print(speedInRPM, 5);
    Serial.println(",");
}

void PIDTest(){
    myPID.run(); //call every loop, updates automatically at certain time interva
l
    analogWrite(motor,map(pwm, 0, 1000, 0, 255)); //use PID Lib

    //show PWM value
    Serial.print(setPoint);
    Serial.print(" ");

    //show Encoder value
    //Serial.println(EncoderCounter);
    //Serial.print(" ");

    //show speed in RPM
    Serial.print(speedInRPM, 5);
    Serial.println(",");
    digitalWrite(LED_PIN, myPID.atSetPoint(10));//light up LED when we're at setp
oint +-10 RPM
}
```

Semua data dan program Arduino maupun Matlab bisa diakses di:
https://github.com/ahmaddidiks/SystemIdentifier