DESAIN KONTROLER PI PADA SISTEM KONTROL KECEPATAN SEPEDA LISTRIK DENGAN METODE ROOT LOCUS

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI TEKNIK KONTROL

Ditujukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

AHMAD HANIF AZHAR
NIM. 125060307111015

UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2016

UNIVERSITAS BRAWIJAYA

LEMBAR PENGESAHAN

DESAIN KONTROLER PI PADA SISTEM KONTROL KECEPATAN SEPEDA LISTRIK DENGAN METODE ROOT LOCUS

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI TEKNIK KONTROL

Ditujukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

AHMAD HANIF AZHAR
NIM. 125060307111015

Skripsi ini telah direvisi dan disetujui oleh dosen pembimbing
Pada tanggal 4 Agustus 2016

Pembimbing I	Pembimbing II
Rahmadwati S.T., M.T., Ph.D	**Goegoes Dwi Nusantoro S.T., M.T**
NIP. 19771102 200604 2 003	**NIP. 19711013 200604 1 001**

Mengetahui,
Ketua Jurusan Teknik Elektro

M. Aziz Muslim, ST., MT., Ph.D
NIP. 19741203 200012 1 001

JUDUL SKRIPSI :

DESAIN KONTROLER PI PADA SISTEM KONTROL KECEPATAN SEPEDA LISTRIK DENGAN METODE ROOT LOCUS

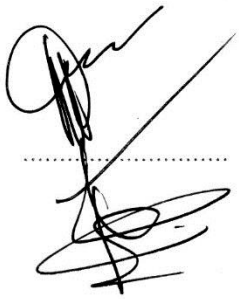
Nama Mahasiswa	: Ahmad Hanif Azhar
NIM	: 125060307111015
Program Studi	: Teknik Elektro
Konsentrasi	: Teknik Kontrol

KOMISI PEMBIMBING :

Ketua : Rahmadwati, S.T., M.T., Ph.D .....

Anggota : Goegoes Dwi Nusantoro., S.T., M.T .....

TIM DOSEN PENGUJI :



Dosen Penguji 1 : Ir. Purwanto, M.T .....

Dosen Penguji 2 : Ir. Retnowati, M.T .....

Dosen Penguji 3 : Dip. Ing. Ir. Mochammad Rusli .....

Tanggal Ujian : 29 Juli 2016

SK Penguji : No. 916/UN10.6/SK/2016

Teriring Ucapan Terima Kasih kepada:

Bapak dan Ibu tercinta

UNIVERSITAS BRAWIJAYA

PERNYATAAN ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya dan berdasarkan hasil penelusuran berbagai karya ilmiah, gagasan dan masalah ilmiah yang diteliti dan diulas di dalam Naskah Skripsi ini adalah asli dari pemikiran saya. Tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu Perguruan Tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah Skripsi ini dapat dibuktikan terdapat unsur-unsur jiplakan, saya bersedia Skripsi dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, pasal 25 ayat 2 dan pasal 70).

Malang, 4 Agustus 2016

Mahasiswa,

AHMAD HANIF AZHAR

NIM. 125060307111015

UNIVERSITAS BRAWIJAYA

RINGKASAN

Ahmad Hanif Azhar, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Juni 2016, *Desain Kontroler PI Pada Sistem Kontrol Kecepatan Sepeda Listrik Dengan Metode Root Locus,* Dosen Pembimbing: Rahmadwati, S.T., M.T., Ph.D., dan Goegoes Dwi Nusantoro, S.T., M.T.

Sepeda listrik merupakan salah satu transportasi alternatif yang dapat menggurangi penggunaan bahan bakar fosil dan ramah lingkungan. Dimana penggerak utama dari sepeda listrik ini yaitu motor *Brushless Direct Current* (BLDC). Motor BLDC ini mempunyai keuntungan yaitu memiliki kecepatan tinggi, respon yang cepat, efisiensi yang tinggi, lebih tahan lama pemakaian, tidak menimbulkan kebisingan, dan juga memungkinkan untuk mendapatkan berbagai kontroler kecepatan. Tetapi motor ini masih memiliki *error steady state*.

Pada penelitian ini menggunakan kontroler *Proportional Integral* (PI) untuk kontrol kecepatan motor BLDC. Gabungan dari kedua kontroler ini diharapkan agar mendapatkan keluaran sistem yang mempunyai *settling time* cepat, tidak ada *overshoot*, dan tidak ada *error steady state*. Sehingga metode yang digunakan untuk mencari parameter PI yaitu metode *root locus*. Dimana metode *root locus* diharapkan untuk mendapatkan respon *plant* yang baik. Oleh karena itu metode ini digunakan untuk mencari parameter dan didapatkan nilai parameter Kp = 2.7301 dan Ki = 8. Dari parameter tersebut kemudian diimplementasikan pada motor BLDC.

Hasil *output* motor BLDC pada implementasi tanpa diberi gangguan dengan *setpoint* 100 rpm, 120 rpm, dan 140 rpm memiliki *error steady state* sebesar 5%, 4.17%, dan 2.14%. sedangkan *settling time* sebesar 27 detik, 24 detik, dan 20 detik. Dan terdapat *overshoot* pada *setpoint* 100 rpm sebesar 7%. Berdasarkan hasil implementasi dengan diberi gangguan pada *setpoint* 100 rpm, 120 rpm, dan 140 rpm, *output* akan mengalami perlambatan dan percepatan. Kemudian respon *plant* kembali keadaan *steady state* dengan *recovery time* sebesar 38 detik, 39 detik, dan 40 detik.

Kata Kunci: Sepeda Listrik, Motor BLDC, Kontroler Kecepatan, Kontrol PI, *Root Locus*.

SUMMARY

Ahmad Hanif Azhar, Department of Electrical Engineering, Faculty of Engineering University of Brawijaya, June 2016, *The Design Of PI Controller In Speed Control System Electric Bike Using Root Locus Method*, Academic Supervisor: Rahmadwati, S.T., M.T., Ph.D., dan Goegoes Dwi Nusantoro, S.T., M.T.

Electric bicycle is one of the alternative transportation that can menggurangi the use of fossil fuels and environmentally friendly. Where is the prime mover of the electric bicycle motor Brushless Direct Current (BLDC). BLDC motor has the advantage that is has high speed, fast response, high efficiency, more durable usage, does not cause noise, and also allows to obtain various speed controller. But this bike still has the steady state error.

This research uses an Integral Proportional controllers (PI) to BLDC motor speed control. A combination of both of these controllers are expected to obtain the output system has fast settling time, no overshoot, and no steady state error. So that the method used to find the parameters of the root locus method of PI. Where the root locus method is expected to get the response of a plant. Therefore the method is used to find the parameters and parameter values are obtained by Kp = 2.7301 and Ki = 8. Of the parameters are then implemented in BLDC motor..

BLDC motor output results in implementation without disruption with setpoint 100 rpm, 120 and 140 rpm, rpm has a steady state error of 5%, 4.17% and 2.14%. While the settling time of 27 minutes, 24 seconds, and 20 seconds. And there is an overshoot at 100 rpm setpoint amounted to 7%. Based on the results of the implementation with the given disorder on setpoint 100 rpm, rpm, 120 and 140 rpm, the output will experience a deceleration and acceleration. Then the response of plant back State-state with a recovery time of 38 minutes, 39 seconds, and 40 seconds.

Keywords: *Electrical Bicycle, BLDC Motor, Speed Controller, PI Control, Root Locus*

PENGANTAR

Bismillahirrohmanirrohim. . . .

Alhamdulillah, puji syukur penulis panjatkan kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi yang berjudul "Desain Kontroler PI Pada Sistem Kontrol Kecepatan Sepeda Listrik Dengan Metode *Root Locus*" dengan baik. Tak lepas shalawat serta salam tercurahkan kepada junjungan kita Nabi Muhammad SAW yang telah menjadi suri tauladan bagi yang mengharapkan rahmat dan hidayah-Nya.

Penulis menyadari bahwa penyusunan skripsi ini tidak lepas dari bantuan, bimbingan serta dorongan dari berbagai pihak. Pada kesempatan ini penulis menyampaikan rasa terima kasih yang sebesar – besarnya kepada:

- Allah SWT yang telah memberikan kelancaran, kemudahan dan hidayah-Nya.
- Keluarga tercinta, kedua orang tua H Fatchan Qorib B.SC dan Hj Chanifah yang selalu memberikan kasih sayang dan doanya yang tiada akhir. Serta kakak tercinta Rifqi, mbak Tutik, mbak nila dan mas iskamto yang selalu memberikan dukungan.
- Bapak M. Aziz Muslim, ST., MT., Ph.D. selaku Ketua Jurusan Teknik Elektro Universitas Brawijaya.
- Bapak Hadi Suyono, ST., MT., Ph.D. selaku Sekretaris Jurusan Teknik Elektro Universitas Brawijaya
- Bapak Ali Mustofa, ST., MT. selaku Ketua Program Studi S1 Jurusan Teknik Elektro Universitas Brawijaya
- Ibu Rahmadwati S.T., M.T., Ph.D. sebagai dosen pembimbing yang telah banyak memberikan kesempatan, nasehat, pengarahan, motivasi, saran dan masukan yang telah diberikan.
- Bapak Goegoes Dwi Nusantoro S.T., M.T. sebagai dosen pembimbing yang telah banyak memberikan kesempatan, nasehat, pengarahan, motivasi, saran dan masukan yang telah diberikan.
- Esti Hardiyanti dan Keluarga besar, terima kasih atas waktu, pengertian, semangat, bantuan dan kesabarannya yang telah diberikan.
- Pranata Laboratorium, Pak Dulhadi dan Keluarga besar asisten Laboratorium Sistem Digital, Rasyid, Adit, Ilham, Firman, Ricki, Juli, Cio, Wuri, Fara, Laila, Machfud, Alec,

Abyan, Mukti, Arsyil, Fatah, Doni, Chandra, Dicka, Nadia, dan Esti terima kasih telah memberikan banyak bantuan, dukungan dan canda tawa.

- Sahabat-sahabat, Suro, Nora, Zakiya, Faizal, Hesa, Hanif, Gabriel, Tyo, dan Odi terima kasih telah memberikan banyak bantuan, dukungan dan canda tawa.
- Teman-teman SMA, Rafi, Tegar, Miftah, Sulis, Fahmi, Irfan, Hasyim, dan Agung yang telah memberikan semangat dan dukungan.
- Sahabat-sahabat SMP, Alvi, Ita, Qisti, Fani, Zamroni terima kasih telah memberikan banyak bantuan, dukungan dan canda tawa.
- Keluarga besar Sistem Kontrol angkatan 2012 dan teman-teman angkatan 2012 "Voltage" atas do'a, semangat, serta dukungan yang diberikan pada penulis.
- Semua pihak yang telah memberikan bantuan serta dukungan baik secara langsung maupun tidak langsung atas penyusunan skripsi ini.

Dalam penyusunan skripsi ini, penulis menyadari bahwa skripsi ini belumlah sempurna, karena keterbatasan ilmu dan kendala–kendala lain yang terjadi selama pengerjaan skripsi ini. Oleh karena itu, penulis berharap kritik dan saran untuk penyempurnaan tulisan di masa yang akan datang. Semoga tulisan ini dapat bermanfaat dan dapat digunakan untuk pengembangan lebih lanjut.

Malang, 4 Agustus 2016

Penulis

DAFTAR ISI

Halaman

DAFTAR TABEL

No. Judul Halaman

UNIVERSITAS BRAWIJAYA

DAFTAR GAMBAR

No. Judul Halaman

DAFTAR LAMPIRAN

No. Judul Halaman

BAB I
PENDAHULUAN

1.1 LATAR BELAKANG

Kemajuan teknologi masa kini berkembang sangat pesat. Hal ini dapat dibuktikan dengan banyaknya inovasi-inovasi yang telah diciptakan di dunia. Salah satu perkembangan teknologi saat ini adalah bermunculannya sepeda listrik. Sepeda listrik merupakan teknologi terbaru pada kendaraan roda dua yang memanfaatkan sumber listrik sebagai bahan bakarnya dan motor listrik sebagai penggeraknya. Sepeda listrik menjadi alat transportasi alternatif untuk mengurangi beban kerja pada manusia. Saat ini di negara jepang, belanda, dan cina merupakan pengguna sepada terbesar didunia (Ahmad, 2011).

Sepeda listrik merupakan salah satu alternatif karena tidak menghasilkan emisi gas buang. Sistem penggerak utama sepeda listrik ini yaitu menggunakan motor *Bruhsless Direct Current* (BLDC). Motor BLDC dapat menghasilkan efisiensi tinggi, torsi tinggi, kecepatan tinggi dan biaya perawatan yang rendah (Adva, 2015).

Motor BLDC merupakan aktuator yang banyak digunakan pada sepeda listrik. Motor BLDC dapat bekerja, diperlukan adanya medan putar magnet stator. Untuk medan magnet putar stator diperlukan sumber tegangan *Alternative Current* (AC) 3 fasa pada stator motor. Oleh karena itu digunakan *inverter* 3 fasa untuk merubah tegangan *Direct Current* (DC) menjadi AC 3 fasa. Motor BLDC memiliki efisiensi yang tinggi, kecepatan tinggi, dan respon yang cepat, namun masih memiliki *error steady state* (*offset*). Motor BLDC dapat menyediakan sebuah torsi awal yang tinggi dan juga memungkinkan untuk mendapatkan berbagai kontrol kecepatan (Irham, 2015). Salah satu kontroler yang dapat menghilangkan *error steady state* adalah kontroler *Proportional Integral* (PI).

Kontroler PI adalah kontroler yang merupakan gabungan dari kontroler *Proportional* dan kontroler *Integral*. Gabungan dari kedua kontroler ini diharapkan dapat menghilangkan *error steady state* (*offset*), *overshoot,* dan mempercepat *settling time*. (Rozi, 2015).

Pada skripsi ini akan dibuat suatu desain kontrol kecepatan sepeda listrik dengan menggunakan kontroler PI dan juga dilengkapi dengan metode *root locus* untuk menentukan parameternya. Hal tersebut bertujuan untuk mendapatkan *error steady state* sekecil mungkin dan mendapatkan *settling time* yang cepat.

1.2 RUMUSAN MASALAH

Berdasarkan latar belakang, maka dapat disusun rumusan masalah sebagai berikut:

1. Bagaimana menentukan fungsi alih pada *plant* motor BLDC.
2. Bagaimana mendesain kontroler PI pada sistem kontrol kecepatan sepeda listrik dengan metode *root locus*.

1.3 BATASAN MASALAH

Dengan mengacu pada permasalahan yang telah dirumuskan, maka hal-hal yang berkaitan dengan perancangan akan diberikan batasan sebagai berikut:

1. Plant yang digunakan adalah sepeda listrik dengan terhubung motor *Brushless* DC (BLDC) dan *driver* tiga fasa.
2. Sensor yang digunakan adalah Sensor *Rotary Encoder*.
3. Kontroler yang digunakan adalah Mikrokontroler *Arduino Mega 2560.*
4. Pembahasan ditekankan pada penggunaan metode *Root Locus*.
5. Pengujian dilakukan dengan sepeda yang dibalik.
6. Kinerja rangkaian *driver* dan elektronika tidak dibahas mendalam.
7. Gangguan yang digunakan adalah rem pada roda sepeda.

1.4 TUJUAN

Tujuan skripsi ini adalah :

1. Menentukan fungsi alih pada *plant* motor *Brushless* DC.
2. Menghasilkan sebuah desain kontroler PI pada sistem kontrol kecepatan sepeda listrik dengan metode *Root Locus*.

1.5 MANFAAT

Manfaat skripsi ini adalah dapat dipahaminya sistem kontroler PI pada sistem kontrol kecepatan sepeda listrik dengan metode *Root Locus*.

BAB II
TINJAUAN PUSTAKA

Bab ini menguraikan teori-teori pendukung skripsi, yang terdiri atas :

2.1. Mikrokontroler Arduino Mega 2560

Mikrokontroler adalah sebuah alat pengendali berukuran mikro atau sangat kecil yang dikemas dalam bentuk *chip*. Mikrokontroler pada dasarnya bekerja seperti sebuah mikroprosesor pada komputer. Keduanya memiliki sebuah *Central Processing Unit* (CPU) yang menjalankan instruksi program, melakukan logika dasar, dan pemindahan data. Mikrokontroler telah memiliki memori dan *interface input output* didalamnya, bahkan beberapa mikrokontroler memiliki unit *analog to digital converter* (ADC) yang dapat menerima masukkan sinyal *analog* secara langsung. Karena berukuran kecil, murah, dan menyerap daya yang rendah, mikrokontroler merupakan alat kontrol yang paling tepat untuk diterapkan pada berbagai peralatan (Artanto, 2009).

Arduino Mega 2560 adalah papan mikrokontroler berdasarkan *ATmega328* (lihat Gambar 2.1). *Board* ini memiliki 54 digital *input/output* pin (14 pin dapat digunakan sebagai *output* PWM), 16 *input* analog, 16 MHz osilator kristal, USB koneksi, jack listrik, *header* ICSP, dan tombol reset.

Gambar 2.1 Arduino Mega 2560
Sumber : www.electroschematics.com

Arduino Mega 2560 dapat diaktifkan melalui koneksi USB atau dengan catu daya *eksternal*. *Eksternal* (non-USB) daya dapat berasal baik dari AC ke adaptor DC atau baterai. Adaptor ini dapat dihubungkan dengan menancapkan *plug jack* pusat-positif ukuran 2.1 mm konektor *power*. Ujung kepala dari baterai dapat dimasukkan kedalam Gnd dan Vin pin *header* dari konektor *power*. *Arduino* dapat beroperasi dengan catu daya *eksternal* 6 sampai 20 volt. Namun jika menggunakan lebih dari 12V, regulator tegangan bisa panas dan merusak papan. Kisaran yang disarankan adalah 7 sampai 12 volt.

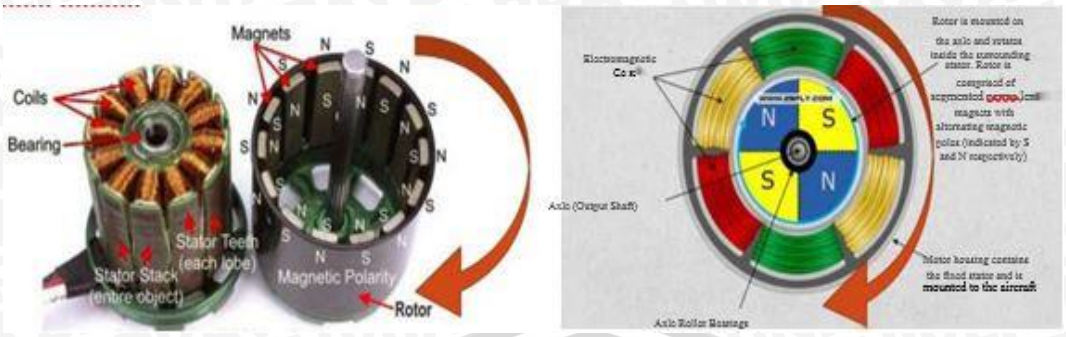
2.2. Motor *Brushless Direct Current* (BLDC)

Motor *Brushless Direct Current* (BLDC) adalah jenis motor yang memiliki karakteristik dan kinerja yang lebih baik dari motor DC. Motor BLDC menggunakan sumber arus searah sebagai sumber energi utama yang kemudian diubah menjadi tegangan AC 3 (fasa) dengan menggunakan *inverter* 3 (tiga) fasa. Tujuan dari pemberian tegangan AC 3 (tiga) fasa pada stator motor BLDC agar menghasilkan medan magnet putar stator untuk menarik magnet rotor.

Motor BLDC telah digunakan pada berbagai macam aplikasi karena memiliki beberapa keunggulan. Keunggulan yang dimiliki BLDC motor seperti karakteristik torsi terhadap kecepatan yang lebih baik, respon dinamik tinggi, efisiensi dan kehandalan tinggi, masa operasi lama, operasi tak bersuara, dapat beroperasi dengan *range* kecepatan yang lebar, dan pengurangan *interferensi elektromagnetik* (EMI). Selain itu, rasio torsi terhadap ukuran motor lebih tinggi, sehingga bermanfaat dalam aplikasi di mana ruang dan berat merupakan faktor penting, terutama aplikasi ruang angkasa (Hamdi, I.T, 2015).

Motor BLDC sudah digunakan di beberapa peralatan listrik seperti sepeda listrik, penggerak printer, fan laptop dll. Dengan ditemukannya komponen-komponen baru menambah banyaknya aplikasi, sehinggga dapat menggantikan sikat komutator mekanik menjadi elektronik. Penambahan sensor posisi misalnya *IC hall effect* dan *rotary encoder* dibutuhkan sebagai umpan balik. Motor ini mempunyai efisiensi yang tinggi, usia kerja yang panjang dan konsumsi daya yang lebih rendah. Motor BLDC merupakan jenis motor sinkron. Medan magnet yang dihasilkan stator dan medan magnet dari rotor memiliki frekuensi putar.

Motor BLDC terdiri dari dua jenis konstruksi yaitu *inrunner* dan *outrunner*. Pada konstruksi *inrunner*, rotor/magnet permanen terletak di sisi dalam sedangkan stator terletak di sisi bagian luar. Motor BLDC *inrunner* memiliki karateristik kecepatan yang tinggi, torsi rendah, efisiensi lebih tinggi dari *outrunner*, lebih rentan rusak, dan menimbulkan suara bising. Sedangkan untuk konstruksi *outrunner*, bagian rotor/maget permanen terletak di sisi luar dan stator terletak di bagian dalam. Motor BLDC *outrunner* memiliki karateristik kecepatan yang rendah, torsi tinggi, mudah digunakan, dan motor lebih tenang. Jenis motor *inrunner* dan *outrunner* ditunjukkan dalam Gambar 2.2.



Gambar 2.2 Bentuk rotor dan stator motor BLDC *(a)outrunner (b)inrunner*
Sumber: Eric Wahl (2014)

2.2.1. Cara Kerja Motor *Brushless Direct Current* (BLDC)

Motor BLDC dapat bekerja ketika stator yang terbuat dari kumparan diberikan arus 3 fasa. Akibat arus yang melewati kumparan pada stator timbul medan magnet (B):

$$B = \frac{\mu i N}{2a} \qquad (2-1)$$

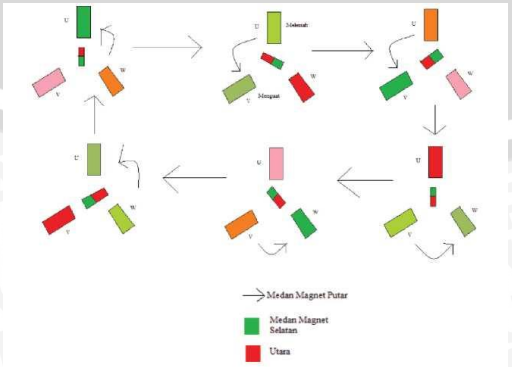
Di mana N merupakan jumlah lilitan, i merupakan arus, a merupakan jari-jari lilitan dan μ merupakan permeabilitas bahan.

Karena arus yang diberikan berupa arus AC 3 fasa sinusoidal, nilai medan magnet dan polarisasi setiap kumparan akan berubah-ubah setiap saat. Akibat yang ditimbulkan dari adanya perubahan polarisasi dan besar medan magnet tiap kumparan adalah terciptanya medan putar magnet dengan kecepatan.

$$n_s = \frac{120f}{p} \qquad (2-2)$$

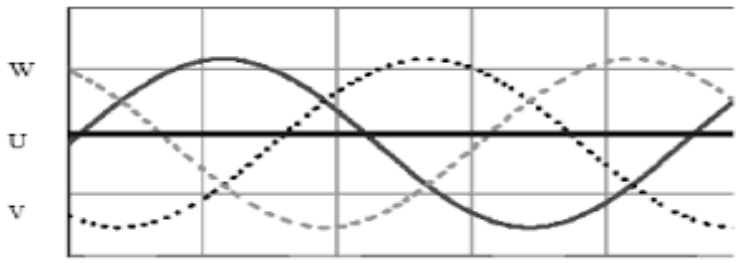
Dengan :

Ns = putaran stator
f = frekuensi
p = jumlah kutub



Gambar 2.3 Medan putar stator dan putaran rotor
Sumber: Abe dharmawan (2009)

Berdasarkan Gambar 2.3, medan putar magnet stator timbul akibat adanya perubahan polaritas pada stator U, V, dan W. Perubahan polaritas ini terjadi akibat adanya arus yang mengalir pada stator berupa arus AC yang memiliki polaritas yang berubah-ubah.



Gambar 2.4 Analisa medan putar
Sumber: Abe dharmawan (2009)

Berdasarkam Gambar 2.4, ketika stator U diberikan tegangan negatif maka akan timbul medan magnet dengan polaritas negatif sedangkan V dan W yang diberikan tegangan positif akan memiliki polaritas positif. Akibat adanya perbedaan polaritas antara medan magnet kumparan stator dan magnet rotor, sisi positif magnet rotor akan berputar mendekati medan magnet stator U, sedangkan sisi negatifnya akan berputar mengikuti medan magnet stator V dan W. Akibat tegangan yang digunakan berupa tegangan AC sinusoidal, medan magnet stator U, V, dan W akan berubah – ubah polaritas dan besarnya mengikuti perubahan tegangan sinusoidal AC. Ketika U dan V memiliki medan magnet negatif akibat mendapatkan tegangan negatif dan W memiliki medan magnet positif akibat tegangan positif, magnet permanen rotor akan berputar menuju ke polaritas yang bersesuaian yakni bagian negatif akan berputar menuju medan magnet stator W dan sebaliknya bagian positif akan berputar menuju medan magnet stator U dan V. Selanjutnya ketika V memiliki medan magnet negatif dan U serta W memiliki medan magnet positif, bagian positif magnet permanen akan berputar menuju V dan bagian negatif akan menuju U dari kumparan W. Karena tegangan AC sinusoidal yang digunakan berlangsung secara kontinu, proses perubahan polaritas tegangan pada stator ini akan terjadi secara terus menerus sehingga menciptakan medan putar magnet stator dan magnet permanen rotor akan berputar mengikuti medan putar magent stator ini. Hal inilah yang menyebabkan rotor pada BLDC dapat berputar.

2.3. *Driver* Motor Tiga Fasa

Driver motor berfungsi sebagai mengubah sinyal *pulse width modulation* (PWM) dari mikrokontroler menjadi tegangan. Dalam aplikasinya *driver* motor biasannya tersusun dari

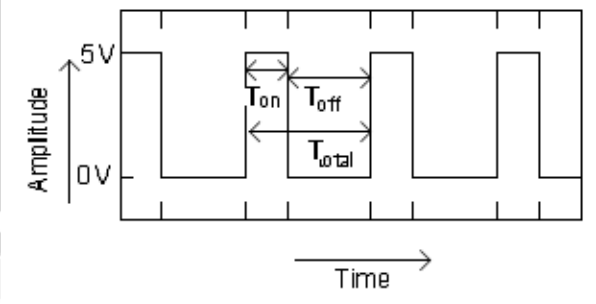
rangkaian transistor-transistor yang tersusun sedemikian rupa sehingga mampu mengendalikan arah putar dan kecepatan motor berdasarkan arah *loop* dan tegangan kutub motor. *Driver* yang akan digunakan adalah *driver* motor tiga fasa dapat dilihat dalam Gambar 2.5.

Gambar 2.5 *Driver* motor tiga fasa
Sumber : www.bogipower.com

2.4. *Pulse Width Modulation* (PWM)

Pulse Width Modulation (PWM) adalah metode yang dapat digunakan untuk mengatur kecepatan dari motor BLDC. Dimana kecepatan motor BLDC tergantung pada besarnya *duty cycle* yang diberikan pada motor BLDC tersebut.

Pada sinyal pwm, frekuensi sinyal konstan sedangkan *duty cycle* bervariasi dari 0%-100%. *Duty cycle* adalah besarnya sinyal kontrol yang diberikan pada motor. Persamaan untuk perhitungan *duty cycle* ditunjukkan pada persamaan (2-3) dengan Ton adalah periode logika tinggi, dan T adalah periode keseluruhan. Sinyal PWM secara umum dapat dilihat dalam Gambar 2.6.



Gambar 2.6 Sinyal PWM
Sumber: www.8051projects.net

$$Duty\ Cycle = \frac{T_{on}}{T} x\ 100\% \qquad (2-3)$$

dimana :

Ton = lebar pulsa saat logika tinggi

T = periode pulsa

Sedangkan frekuensinya dapat ditentukan dengan rumus sebagai berikut:

$$fOCn = \frac{fclk\ \ I/O}{N} \qquad (2-4)$$

dimana:

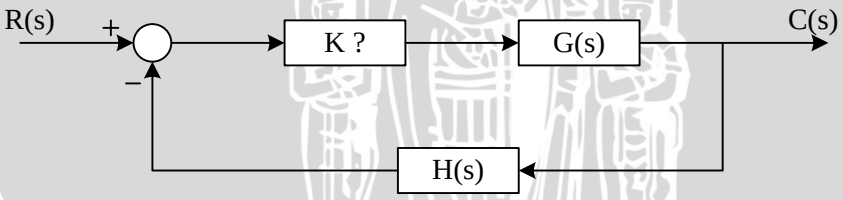
$fclk\ \ I/O$ = frekuensi I/O

Timer atau *counter* yang digunakan pada PWM ini yaitu *timer* atau *counter* 1 (16 bit) dengan metode fast PWM dan *prescaler factor* (N) yaitu 256.

2.5. Root Locus

Metode sistem dengan parameter sistem berubah pada lingkup tertentu, misalnya perubahan parameter *Root Locus* / letak kedudukan akar digunakan untuk meneliti perilaku penguat K. Di dalam analisis sistem, penguatan K dipilih sedemikian rupa agar sistem stabil serta memberikan respon yang baik. Rancangan dimaksudkan agar letak *pole* dan *zero* dari fungsi alih *loop* tertutup terletak pada daerah yang ditentukan. Agar sistem stabil, *pole* dan *zero* harus terletak pada bidang s sebelah kiri sumbu imajiner.

Metode letak kedudukan akar ini memberikan informasi penguatan K jika penguatan K diubah dari nol menjadi tak terhingga. Metode ini memungkinkan untuk mencari *pole loop* tertutup dan zero loop terbuka dengan penguatan sebagai parameter.



Gambar 2.7 Sistem loop tertutup
Sumber: Ogata K (1997)

Fungsi alih *loop* tertutup secara umum adalah sebagai berikut

$$\frac{C(s)}{R(s)} = \frac{G(s)}{1+G(s)H(s)} \qquad (2-5)$$

akar-akar karakteristik yang memenuhi persamaan karakteristik:

$$1+G(s)\,H(s)=0 \qquad (2-6)$$

Suatu sistem *loop* tertutup dalam Gambar 2.7 mempunyai persamaan karakteristik sebagai berikut

$$1+K\,G(s)\,H(s)=0$$

atau

$$K\,G(s)\,H(s) = -1 \qquad (2-7)$$

maka akar karakteristik adalah harga s yang memenuhi syarat berikut ini:

syarat sudut

$$\angle G(s)\,H(s) = 180^{\circ}(2K+1); \qquad K = 0,1,2,3,..$$

syarat magnitud

$$|G(s)H(s)| = 1$$

Bila *K* berubah, maka letak *pole-pole* nya juga berubah. Untuk mengetahui kawasan letak *pole-pole* persamaan karakteristik sistem tersebut terhadap kemungkinan kombinasi nilai K, maka diperlukan sebuah metode yang disebut dengan *Root Locus*. *Root* berarti akar dan *Locus* berarti tempat kedudukan. Maka *Root Locus* adalah tempat kedudukan akar-akar persamaan karakteristik dari sebuah sistem pengendalian proses dengan *K = 0* sampai *K= tak hingga*. Ini dapat digunakan untuk menentukan stabilitas sistem tersebut selalu stabil atau ada batas kestabilan, sehingga dalam merancang sistem kendali, kita bisa mendapatkan hasil *step respon* sesuai yang diinginkan dengan mengubah nilai-nilai *K*.

2.6. Kontroler

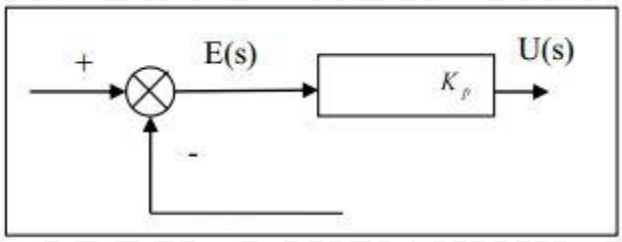
Kontroler seringkali juga disebut dengan istilah kompensator atau pengontrol. Kontroler adalah suatu sistem dinamis yang sengaja ditambahkan untuk mendapatkan karakteristik sistem keseluruhan yang diinginkan (Ogata K., 1997).

Salah satu fungsi kontroler adalah meminimumkan sinyal *error steady state*, sinyal *error* adalah perbedaan nilai *setpoint* dengan nilai *output plant*. *Setpoint* adalah nilai referensi atau nilai yang diinginkan, sedangkan *output plant* adalah nilai aktual yang terukur pada *output plant*. Semakin kecil nilai sinyal *error* maka kinerja sistem kontrol dinilai semakin baik.

Prinsip kerja kontroler adalah membandingkan nilai *output plant* dengan nilai *setpoint*, menentukan nilai kesalahan dan akhirnya menghasilkan sinyal kontrol untuk meminimumkan *error steady state*, *settling time*, dan *overshoot*.

2.6.1. Kontroler Proporsional (P)

Kontroler proporsional (P) adalah sebuah kontroler yang memiliki karakteristik mempercepat *output*. Hubungan anatara *output* kontroler U(s) dan sinyal *error* E(s) ditunjukan dalam Gambar 2.8 dan persamaan $(2-8)$ berikut :



Gambar 2.8 Diagram blok kontroler proporsional (P)
Sumber : Ogata K (1997)

$$U(s) = Kp\, E(s) \qquad (2-8)$$

dimana:

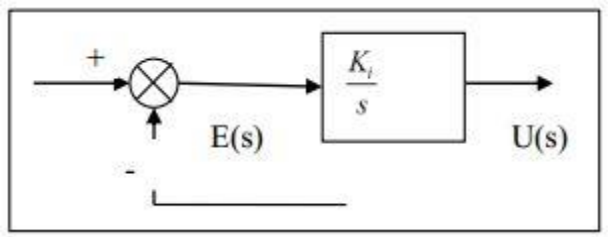
Kp = adalah penguatan proporsional

$E(s)$ = sinyal *error*

$U(s)$ = *output* kontroler

2.6.2. Kontroler Integral (I)

Kontroler integral (I) adalah sebuah kontroler yang memiliki kemampuan untuk mengurangi *offset* yang diakibatkan oleh kontroler proporsional. *Output* kontroler U(s) diubah dengan laju yang sebanding denga sinyal *error* E(s) ditunjukan dalam Gambar 2.9 dan persamaan $(2-9)$ berikut :



Gambar 2.9 Diagram blok kontroler integral (I)
Sumber : Ogata K (1997)

$$\frac{dU(s)}{dt} = Ki\, E(s)$$

$$U(s) = Ki \int E(s)\, dt$$

$$\frac{U(s)}{E(s)} = \frac{K_i}{s} \qquad (2-9)$$

dimana:

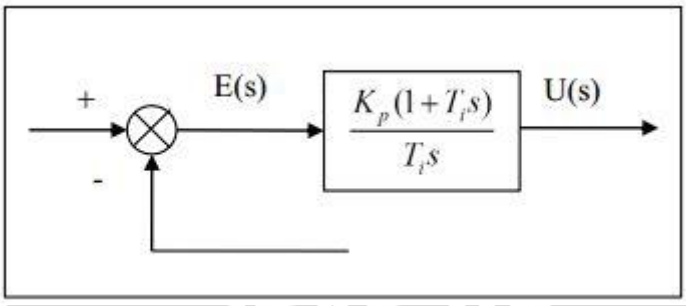
Ki = adalah penguatan integral

$E(s)$ = sinyal *error*

$U(s)$ = *output* kontroler

2.6.3. Kontroler Proporsional Integral (PI)

Kontroler proporsional integral (PI) memiliki kemampuan untuk mempercepat *output* dan mengurangi *offset*. Gambar 2.10 menunjukan diagram blok kontroler proporsional integral dan persamaan $(2-10)$.



Gambar 2.10 Diagram blok kontroler proporsional dan integral
Sumber : Ogata K (1997)

$$U(s) = Kp\,.E(s) + \frac{K_p}{T_i}\int_0^t E\,(s)dt$$

Adapun fungsi alihnya adalah

$$\frac{U(s)}{E(s)} = K_p\left(1+\frac{1}{T_i s}\right)$$

$$= \frac{K_p(1+T_i s)}{T_i s} \qquad (2-10)$$

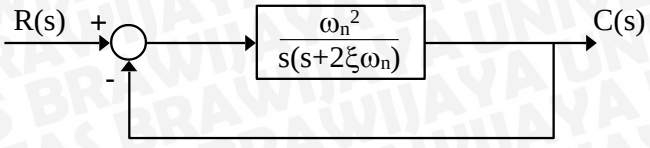
dimana:

Kp = adalah penguatan proporsional
Ti = adalah waktu integral
$E(s)$ = sinyal *error*
$U(s)$ = *output* kontroler

Dengan K_p penguatan proporsional dan T_i disebut waktu integral, yang keduanya dapat ditentukan. Waktu integral mengatur aksi kontrol integral sedangkan perubahan nilai K_p berakibat pada bagian aksi kontrol proporsional maupun integral.

2.7. Sistem Orde Dua

Diagram blok sistem orde dua (lihat Gambar 2.11), dengan fungsi alihnya adalah sebagai berikut.



Gambar 2.11 Sistem orde dua

$$\frac{C(s)}{R(s)} = \frac{\frac{\omega_n^2}{s(s+2\xi\omega_n)}}{1+\frac{\omega_n^2}{s(s+2\xi\omega_n)}}$$

$$= \frac{\omega_n^2}{s(s+2\xi\omega_n)+\omega_n^2}$$

$$= \frac{\omega_n^2}{s^2+2\xi\omega_n s+\omega_n^2} \qquad (2-11)$$

Akar-akar penyebut fungsi alih atau persamaan karakteristik adalah

$$s_{12} = \frac{-2\xi\omega_n \pm \sqrt{(2\xi\omega_n)^2-4\omega_n^2}}{2}$$

$$s_{12} = -\xi\omega_n \pm \omega_n\sqrt{\xi^2-1}$$

$$= -\xi\omega_n \pm j\omega_n\sqrt{1-\xi^2}$$

$$s_{12} = -\xi\omega_n \pm j\omega_d \qquad (2-12)$$

dimana

ξ = rasio peredaman sistem (*damping ratio*)

ω_n = frekuensi natural/alamiah tak teredam

ω_d = frekuensi natural/alamiah teredam

Kelakuan dinamik sistem orde dua dapat digambarkan dalam suku dua parameter ξ dan ω_n. Jika $(0<\xi<1)$, maka *pole loop* tertutup merupakan konjugat kompleks dan berada pada bidang s sebelah kiri. Dalam hal ini, sistem dikatakan dalam peredaman dan tanggapan peralihan berosilasi. Jika $(\xi=1)$, maka sistem dikatakan teredam kritis. Sistem terlalu teredam berhubungan dengan $(\xi>1)$. Tanggapan peralihan sistem teredam kritis dan sistem terlalu teredam tidak berosilasi. Jika ξ=0, tanggapan peralihan tidak muncul.

Pada sistem orde dua seperti terlihat dalam Gambar 2.11, berdasarkan *output* sistem dengan masukan *unit step* akan terdapat tiga keadaan yang berbeda yaitu keadaan teredam $(0<\xi<1)$**,** teredam kritis $(\xi=1)$, dan sistem terlalu teredam $(\xi>1)$**.**

2.7.1. Teredam Kurang / *Underdamp* ($0 < \zeta < 1$)

$$\frac{C(s)}{R(s)} = \frac{\omega_n^2}{(s^2+2\xi\omega_n\, s+\omega_n^2)} \qquad (2-13)$$

Jika sistem diberi *input* berupa *unit step* atau $R(s)=\frac{1}{s}$, maka:

$$C(s) = \frac{\omega_n^2}{(s^2 + 2\xi\omega_n\, s + \omega_n^2)s} \qquad (2-14)$$

Dari Tabel Transformasi *Laplace* didapatkan

$$c(t) = 1 - \frac{1}{\sqrt{1-\xi^2}} e^{-\xi\omega_n t} \sin(\omega_n \sqrt{1-\xi^2}\, t + \phi)$$

$$\phi = \arctan\frac{\sqrt{1-\xi^2}}{\xi}$$

Jika $\omega_d = \omega_n\sqrt{1-\xi^2}$; maka

$$c(t) = 1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}} \sin\left(\omega_d t + \arctan\frac{\sqrt{1-\xi^2}}{\xi}\right) \quad t \geq 0 \qquad (2-15)$$

Output sistem tersebut juga bisa diperoleh dengan menggunakan Transformasi *Laplace* balik jika C(s) ditulis dalam bentuk berikut:

$$C(s) = \frac{1}{s} - \frac{s + 2\xi\omega_n}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

$$= \frac{1}{s} - \frac{s + \xi\omega_n}{(s+\xi\omega_n)^2 + \omega_d^2} - \frac{\xi\omega_n}{(s+\xi\omega_n)^2 + \omega_d^2}$$

$$\mathcal{L}^{-1}\left[\frac{s+\xi\omega_n}{(s+\xi\omega_n)^2+\omega_d^2}\right] = e^{-\xi\omega_n t}\cos\omega_d t$$

$$\mathcal{L}^{-1}\left[\frac{\xi\omega_n}{(s+\xi\omega_n)^2+\omega_d^2}\right] = e^{-\xi\omega_n t}\sin\omega_d t \qquad (2-16)$$

oleh karena itu, transformasi *laplace* balik dari persamaan

$$C(s) = \frac{\omega_n^2}{(s^2 + 2\xi\omega_n\, s + \omega_n^2)s}$$

diperoleh sebagai

$$\mathcal{L}^{-1}[C(s)] = c(t)$$

$$c(t) = 1 - e^{-\xi\omega_n t}\left(\cos\omega_d t + \frac{\xi}{\sqrt{1-\xi^2}}\sin\omega_d t\right)$$

$$c(t) = 1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}} \sin\left(\omega_d t + \arctan\frac{\sqrt{1-\xi^2}}{\xi}\right) \quad t \geq 0 \qquad (2-17)$$

Sinyal kesalahan / *error* adalah e(t) = r(t) – c(t), dimana r(t) = 1 dan

$$c(t) = 1 - e^{-\xi\omega_n t}\left(\cos\omega_d t + \frac{\xi}{\sqrt{1-\xi^2}}\sin\omega_d t\right)$$

sehingga

$$e(t) = e^{-\xi\omega_n t}\left(\cos\omega_d t + \frac{\xi}{\sqrt{1-\xi^2}}\sin\omega_d t\right) \qquad t \geq 0$$

jika $\xi = 0 \quad \Rightarrow \quad c(t) = 1 - \cos\omega_n t$

2.7.2. Teredam Kritis / *Critically Damped* ($\zeta = 1$)

Dalam hal ini apabila dua *pole* $C(s)/R(s)$ hampir sama, maka sistem dapat didekati dengan bentuk teredam kritis. Jika *input* berupa *unit step* dimana R(s) = 1/s maka:

$$C(s) = \frac{\omega_n^2}{(s+\omega_n)^2 s} \qquad (2-18)$$

$$c(t) = 1 - e^{-\omega_n t}(1+\omega_n t) \qquad t \geq 0$$

2.7.3. Terlalu Teredam / *Overdamped* ($\zeta > 1$)

Dalam hal ini *pole* $C(s)/R(s)$ adalah bilangan nyata / *real* negatif yang tidak sama. Jika *input* berupa *unit step* dimana R(s) = 1/s dan C(s) dapat ditulis dengan :

$$C(s) = \frac{\omega_n^2}{\left(s+\xi\omega_n+\omega_n\sqrt{\xi^2-1}\right)\left(s+\xi\omega_n-\omega_n\sqrt{\xi^2-1}\right)s}$$

$$c(t) = 1 + \frac{\omega_n}{2\sqrt{\xi^2-1}}\left(\frac{e^{-s_1 t}}{s_1} - \frac{e^{-s_2 t}}{s_2}\right) \qquad t \geq 0 \qquad (2-19)$$

Dengan

$$s_1 = \left(\xi + \sqrt{\xi^2-1}\right)\omega_n$$

$$s_2 = \left(\xi - \sqrt{\xi^2-1}\right)\omega_n \qquad (2-20)$$

Tanggapan c(t) terdiri dari dua suku eksponensial menurun.

2.8. Tanggapan Peralihan

Sistem dengan tenaga tidak dapat memberikan tanggapan seketika dan akan menunjukkan tanggapan peralihan walaupun diberi masukan ataupun gangguan. Karakteristik unjuk kerja sistem kontrol yang diinginkan dicirikan oleh suku tanggapan

peralihan terhadap masukan *unit step* karena hal itu mudah dilakukan dan cukup drastis. Jika tanggapan terhadap masukan *unit step* diketahui, secara matematis dapat dihitung tanggapan untuk masukan yang lain.

Tanggapan peralihan sistem kontrol selalu menunjukkan osilasi teredam sebelum mencapai keadaan mantapnya, hal ini juga menunjukkan bahwa sistem tersebut mempunyai rasio peredaman $(0 < \xi < 1)$ yang juga berarti bahwa sistem tersebut merupakan sistem yang kurang teredam / *underdamped.*

Tanggapan peralihan sistem kontrol terhadap masukan *unit step* umumnya dikelompokkan sebagai berikut (lihat Gambar 2.12).

1) *Delay Time* / Waktu Tunda, t_d

   Waktu yang dibutuhkan oleh *output* untuk mencapai setengah harga akhir pada saat lonjakan pertama.

2) *Rise Time* / Waktu Naik, t_r

   Waktu yang dibutuhkan oleh *output* agar bertambah dari 10% menjadi 90% dari nilai akhir.

3) *Peak Time* / Waktu Puncak, t_p

   Waktu yang dibutuhkan oleh *output* untuk mencapai puncak pertama lonjakan (maksimum).

4) *Maximum Overshoot* / Lonjakan Maksimum, M_p

   Merupakan nilai puncak kurva *output* diukur dari satu.

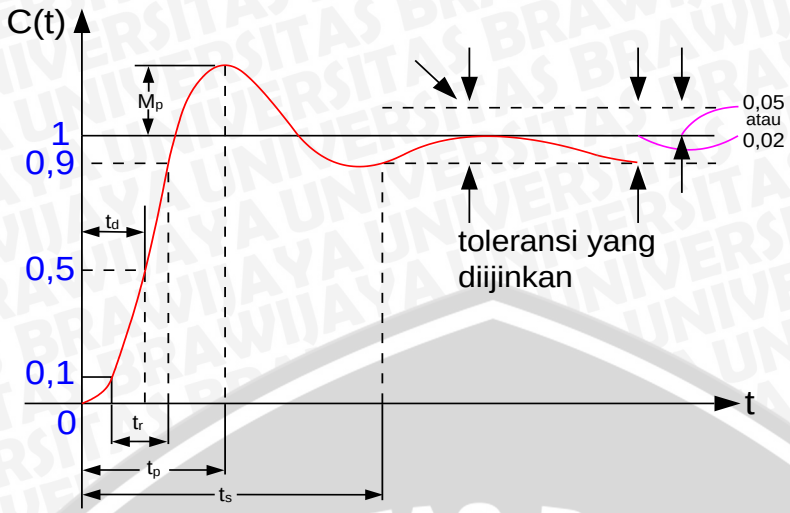
   $$Mp = \frac{c(t_p) - c(\infty)}{c(\infty)} \times 100\%$$

   dengan $c(t_p)$ = nilai *output* pada saat lonjakan maksimum.

   $c(\infty)$ = nilai *output* pada saat keadaan mantap.

5) *Settling Time* / Waktu Penetapan, t_s

   Waktu yang dibutuhkan oleh *output* untuk mencapai harga tertentu dan tetap dalam *range* nilai akhir (biasanya 5% atau 2%).



Gambar 2.12 *Output unit* step sistem orde dua
Sumber : Ogata K (1997)

2.9. Diskritisasi

Banyak cara yang dapat digunakan untuk proses diskritisasi (mengubah bentuk *analog* menjadi diskrit), tiga diantaranya yang banyak digunakan dalam bidang kontrol adalah *backward difference*, *forward difference* (metode *Euler*) dan *bilinear transformation*. Semua metode yang digunakan hanya merupakan pendekatan (*approximations*), sehingga hasilnya tidak akan persis sama dengan bentuk *analog*. Hal ini dikarenakan bentuk diskrit selalu ada sebagian informasi yang hilang, yaitu informasi yang terjadi antara satu cuplikan dengan cuplikan lainnya walaupun frekuensi cuplik yang digunakan tinggi dan karena formula matematika yang digunakan dalam proses diskritisasi juga diturunkan berdasarkan pendekatan agar lebih mudah digunakan. Berikut tahapan diskritisasi yang dapat digunakan :

1. Tulis algoritma *analog* dalam bentuk trasformasi *laplace*.
2. Lakukan diskritisasi menjadi bentuk transformasi Z dengan mengganti operator s dengan menggunakan salah satu dari tiga metode diskritisasi, yaitu:
   - *Backward difference* :

     $$s = \frac{1 - z^{-1}}{T_s} \qquad (2-21)$$

   - *Forward difference* :

     $$s = \frac{1 - z^{-1}}{T_s z^{-1}} \qquad (2-22)$$

   - *Bilinear transform* :

$$s = \frac{2(1 - z^{-1})}{T_s(1 + z^{-1})} \qquad (2 - 22)$$

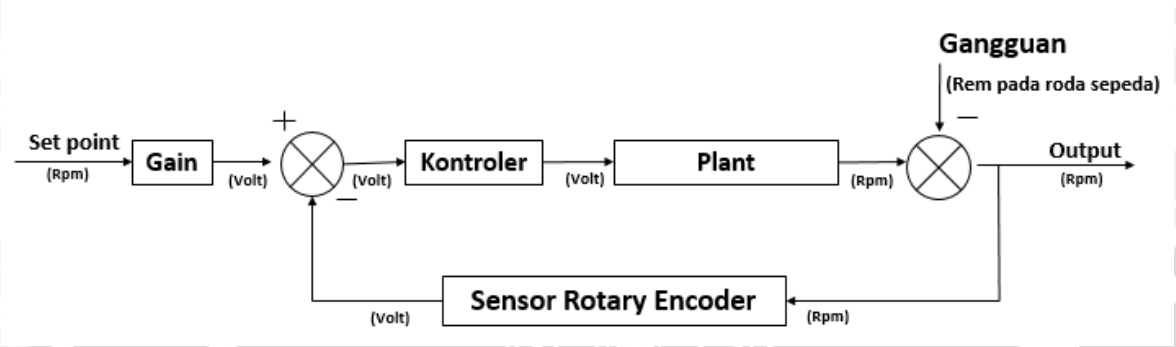
Dimana T_s adalah periode cuplik

Hingga tahap 2, algoritma sudah didapat dalam bentuk diskrit yang dinyatakan dalam transformasi Z. Pada implementasinya, bentuk transformasi Z tersebut perlu diubah menjadi *time domain* (persamaan beda), yaitu dengan mengubah operator z^nmenjadi n kali waktu *delay* (z^{-1} berarti 1 kali waktu delay, z^{-2} berarti 2 kali waktu *delay* dan seterusnya).

BAB III
METODE PENELITIAN

Penyusunan skripsi ini didasarkan dalam masalah yang bersifat aplikatif, yaitu perencanaan dan perealisasian sistem agar dapat bekerja sesuai dengan yang direncanakan. Langkah-langkah yang perlu dilakukan untuk merealisasikan sistem yang dirancang adalah.

3.1 Perancangan Blok Diagram Sistem



Gambar 3.1 Blok digram sistem *loop* tertutup

Keterangan :

1. *Setpoint* sistem berupa putaran motor yang memiliki tiga nilai yaitu 100 rpm, 120 rpm, dan 140 rpm.
2. *Gain* berfungsi sebagai konversi dari kecepatan (Rpm) ke tegangan (V). Dimana nilai *gain* tersebut sama dengan nilai *gain* sensor *rotary encoder*.
3. Kontroler yang digunakan adalah kontroler *Proportional Integral* (PI) menggunakan perangkat keras *Arduino Mega 2560.*
4. *Plant* yang digunakan adalah sepeda listrik yang terhubung dengan motor *brushless* DC dan *driver* motor tiga fasa.
5. Sensor *Rotary Encoder* sebagai pembaca putaran motor digunakan untuk *feedback* sistem.
6. Gangguan pada motor berupa rem sepeda.

3.2 Spesifikasi Desain

Desain yang diinginkan pada perancangan pengontrolan kecepatan motor *Brushless* DC (BLDC) mempunyai spesifikasi yaitu :

1. *Error Steady State* <5%

   Error Steady State <5%, karena sistem yang baik memiliki *output* dengan batas nilai akhir 5% dari *setpoint*.

2. $\xi = 1$

   $\xi = 1$, agar output tidak berosilasi dan dapat mencapai *setpoint*.

3. *Settling time* < 20 detik

   Settling time < 20 detik, karena diharapkan penggunaan kontroler PI mampu mempercepat *Settling time* sistem kurang dari 20 detik.

4. *Output* tanpa *overshoot*

   Penggunaan kontroler PI diharapkan mampu meminimalkan *overshoot*.

3.3 Karakteristik Setiap Blok

3.3.1 Karakteristik Motor *Brushless* DC (BLDC)

a. Tujuan

   Mengetahui karakteristik motor BLDC pada setiap kenaikan *duty cycle (%)*.

b. Peralatan yang digunakan

   1. *Baterai* atau Aki Motor.
   2. Motor BLDC.
   3. *Tachometer* digital.
   4. Perangkat komputer.
   5. Kabel penghubung.

c. Langkah Pengujian

   1. Hubungkan *output* tegangan *Baterai* dengan motor BLDC.
   2. Atur *duty cycle* sinyal PWM pada Arduino Mega 2560 dengan nilai 0%-100%.
   3. Gunakan *tachometer* digital untuk mendapatkan nilai putaran motor.
   4. Amati dan catat hasil pengukuran putaran motor.

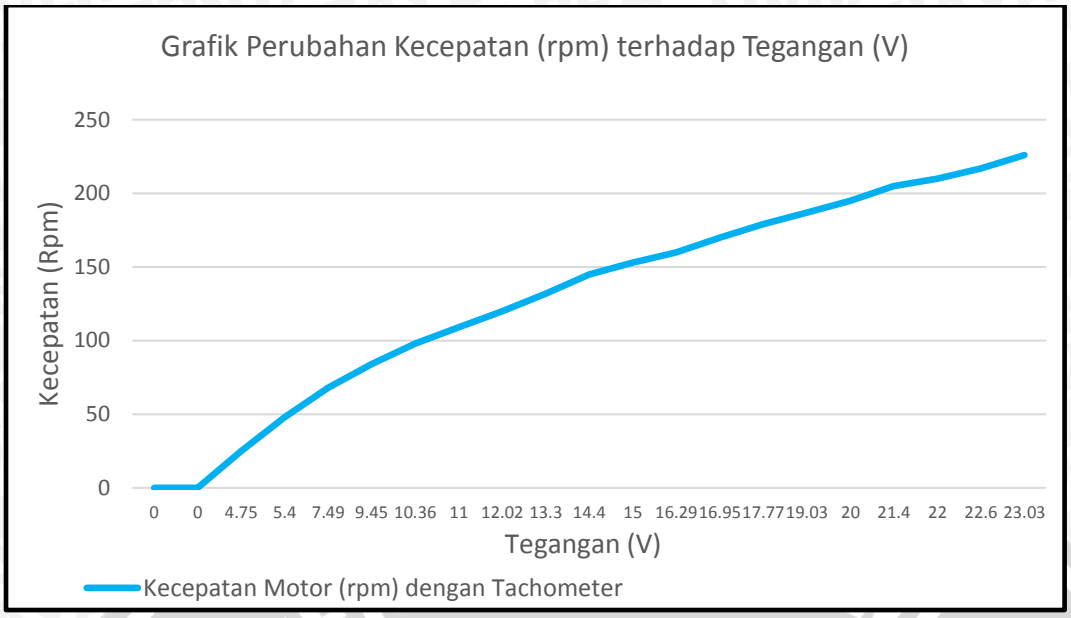
d. Hasil Pengujian

   Data hasil pengujian kecepatan dengan perubahan tegangan pada motor BLDC ditunjukan dalam Tabel 3.1.

Tabel 3.1 Data pengujian kecepatan motor BLDC (rpm) terhadap tegangan (V)

Duty Cycle (%)	*Tegangan (V)*	*Kecepatan Motor (RPM) dengan Tachometer*
0	0	0
5	0	0
10	4,75	25
15	6,3	48
20	7,49	68
25	9,03	84
30	10,36	98
35	11,5	109
40	12,5	120
45	13,3	132
50	14,4	145
55	15	153
60	16,29	160
65	16,95	170
70	17,77	179
75	19,03	187
80	20	195
85	21	205
90	21,79	210
95	22,6	217
100	23,03	226

Pada *duty cycle* 0% dan 5%, terlihat bahwa motor BLDC tidak berputar dan tegangan tidak keluar. Pada daerah ini dapat disebut dengan daerah *dead time*. Berdasarkan Tabel 3.1 dengan mengambil data *duty cycle* dimulai dari 10%. Maka akan didapatkan kurva kecepatan motor (rpm) terhadap tegangan (V) seperti ditunjukan pada Gambar 3.1.



Gambar 3.2 Grafik perubahan kecepatan motor BLDC (rpm) terhadap tegangan (v)

Dari Gambar 3.2 didapatkan *gain* motor BLDC adalah :

$$m = \frac{y_{20} - y_{16.29}}{x_{20} - x_{16.29}} = \frac{195 - 160}{20 - 16.29} = \frac{35}{3.71} = 9.433$$

3.3.2 Karakteristik *Driver* Motor Tiga Fasa

a. Tujuan

Mengetahui kinerja dan *output* rangkaian *driver* motor tiga fasa dengan membandingkan *output* tegangan efektif *driver* dengan masukan *duty cycle* sinyal PWM yang diberikan oleh *Arduino Mega 2560.*

b. Peralatan yang digunakan

1. Komputer atau PC.
2. *Baterai* atau Aki Motor.
3. *Driver* Motor Tiga Fasa.
4. Multimeter.
5. *Arduino Mega 2560.*
6. Kabel Pengujian.

c. Langkah pengujian

1. Hubungkan *baterai* pada *input driver* motor tiga fasa.
2. Hubungkan *input* tegangan *driver* motor tiga fasa dengan *pin output* PWM di *Arduino Mega 2560.*
3. Hubungkan *output* tegangan *driver* motor tiga fasa dengan multimeter.

4. Atur *duty cycle* sinyal PWM pada *Arduino Mega 2560* dengan nilai 0% - 100%.
5. Amati dan catat hasil pembacaan multimeter disetiap kenaikan 5%.

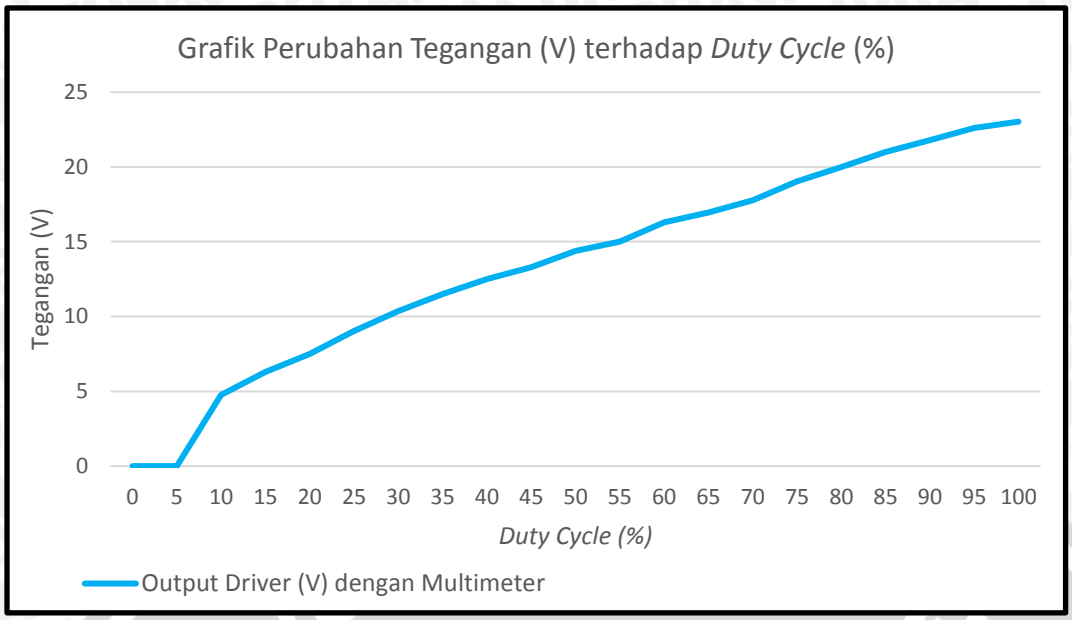
d. Hasil pengujian

Data pengujian *driver* motor tiga fasa ditunjukjan dalam Tabel 3.2.

Tabel 3.2 Data pengujian *driver* motor tiga fasa

Duty Cycle %	Output Driver (V) dengan Multimeter
0	0
5	0
10	4,75
15	6,3
20	7,49
25	9,03
30	10,36
35	11,5
40	12,5
45	13,3
50	14,4
55	15
60	16,29
65	16,95
70	17,77
75	19,03
80	20
85	21
90	21,79
95	22,6
100	23,03

Berdasarkan Tabel 3.2 akan didapatkan kurva *output* dengan tegangan *driver* (V) terhadap masukan *duty cycle* sinyal PWM (%) seperti ditunjukan pada Gambar 3.3.



Gambar 3.3 Grafik perubahan tegangan *output driver* (V) terhadap *duty cycle* (%)

Dari Gambar 3.3 didapatkan *gain driver* tiga fasa adalah :

$$m = \frac{y_{45} - y_{35}}{x_{45} - x_{35}} = \frac{13.3 - 11.5}{45 - 35} = \frac{1.8}{10} = 0.18$$

3.3.3 Karakteristik Sensor *Rotary Encoder*

a. Tujuan

Mengetahui tingkat kelinieran dari *rotay encoder* dalam pembacaan putaran motor.

b. Peralatan yang digunakan

1. Komputer atau PC.
2. *Baterai* atau Aki Motor.
3. *Rotary Encoder.*
4. *Driver* Motor Tiga fasa.
5. Motor BLDC.
6. *Arduino Mega 2560.*
7. Kabel Penghubung.

c. Langkah pengujian

1. Hubungkan *baterai* pada *input driver* motor tiga fasa dan *input rotary encoder* ke *arduino mega 2560.*
2. Hubungkan *pin output rotary encoder* pada *pin interrupt eksternal Arduino Mega 2560.*
3. Atur *duty cycle* sinyal PWM pada *Arduino Mega 2560* dengan nilai 0% - 100%.

4. Amati dan catat hasil pembacaan *rotary encoder* disetiap kenaikan 5%.

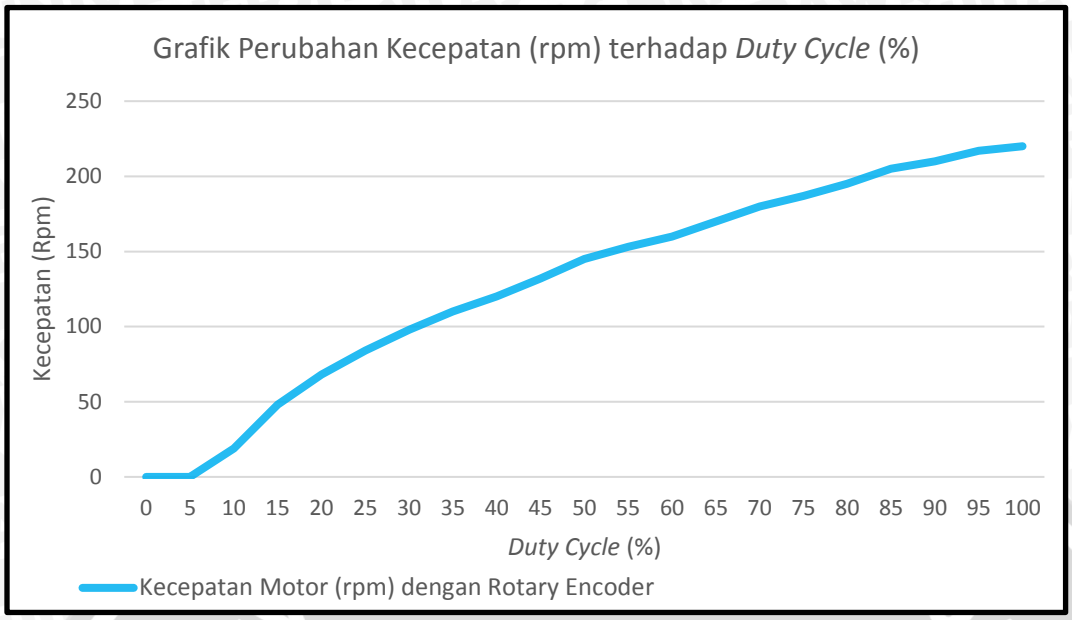
d. Hasil pengujian

Data hasil pengujian sensor *rotary encoder* ditunjukan dalam Tabel 3.3.

Tabel 3.3 Data pengujian *Rotary Encoder*

Duty Cycle %	Kecepatan Motor (rpm) dengan *Rotary Encoder*
0	0
5	0
10	19
15	48
20	68
25	84
30	98
35	110
40	120
45	132
50	145
55	153
60	160
65	170
70	180
75	187
80	195
85	205
90	210
95	217
100	220

Berdasarkan Tabel 3.3 akan didapatkan kurva kecepatan motor dengan menggunakan *rotary encoder* (rpm) terhadap masukan *duty cycle* sinyal PWM (%) seperti ditunjukkan pada Gambar 3.4.



Gambar 3.4 Grafik perubahan *output* kecepatan motor BLDC (rpm) terhadap *duty cycle* (%)

Dari Gambar 3.4 didapatkan *gain* sensor *roatry encoder* adalah :

$$m = \frac{y_{70} - y_{60}}{x_{70} - x_{60}} = \frac{180 - 160}{179 - 153} = 0.8$$

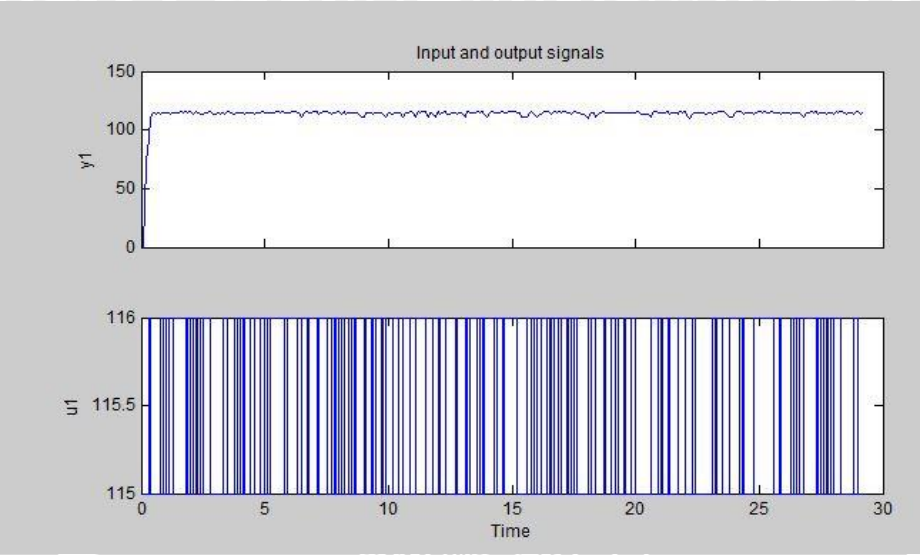
3.4 Penentuan Fungsi Alih Motor *Brushless* DC (BLDC)

Untuk mengendalikan motor BLDC digunakan *Arduino Mega 2560* sebagai pengolah dan memberikan data berupa *Pulse Width Modulaion* (PWM) agar motor bergerak. Motor BLDC yang digunakan pada perancangan ini tidak diketahui karakteristiknya, sehingga yang perlu dilakukan adalah melakukan pengujian dengan menggunakan sensor *rotary encoder*. Untuk mendapatkan karakteristik motor BLDC pada perancangan ini diberikan masukan *unit step*.

Untuk mendapatkan fungsi alih dari motor dilakukan pemodelan dengan cara membangkitkan sinyal *Pseudo Random Binary Sequence* (PRBS). Langkah yang dilakukan untuk membangkitkan sinyal PRBS adalah sebagai berikut :

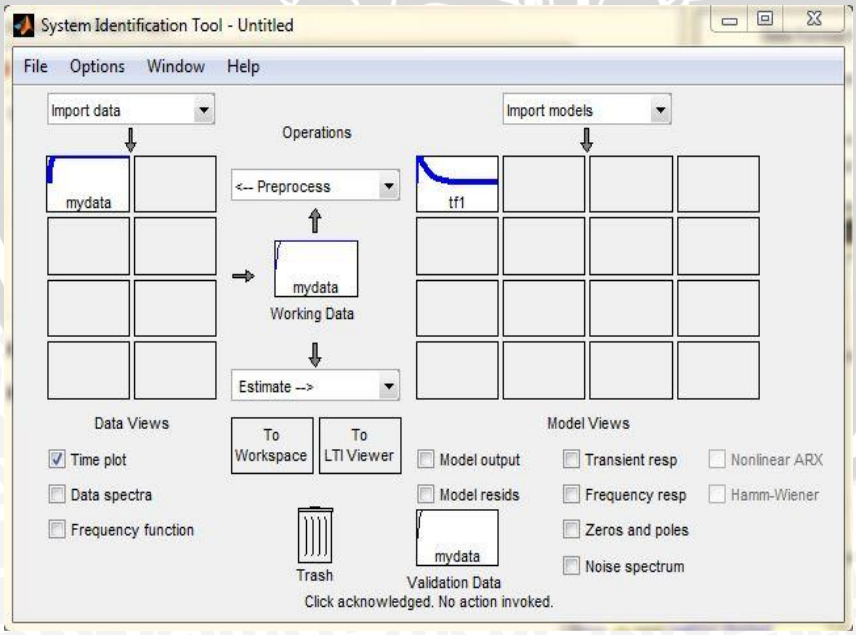
1. Mencari nilai yang linier dari hasil kecepatan motor terhadap *duty cycle* sinyal PWM.
2. Memasukkan nilai batas atas dan bawah berdasarkan nilai yang linier untuk membangkitkan sinyal PRBS.
3. Sinyal PRBS yang telah dibangkitkan kemudian digunakan sebagai masukan motor BLDC.

4. Setelah didapatkan data sinyal PRBS dan data kecepatan motor BLDC (lihat Gambar 3.5), selanjutnya adalah melakukan identifikasi dengan menggunakan *software Matlab*.



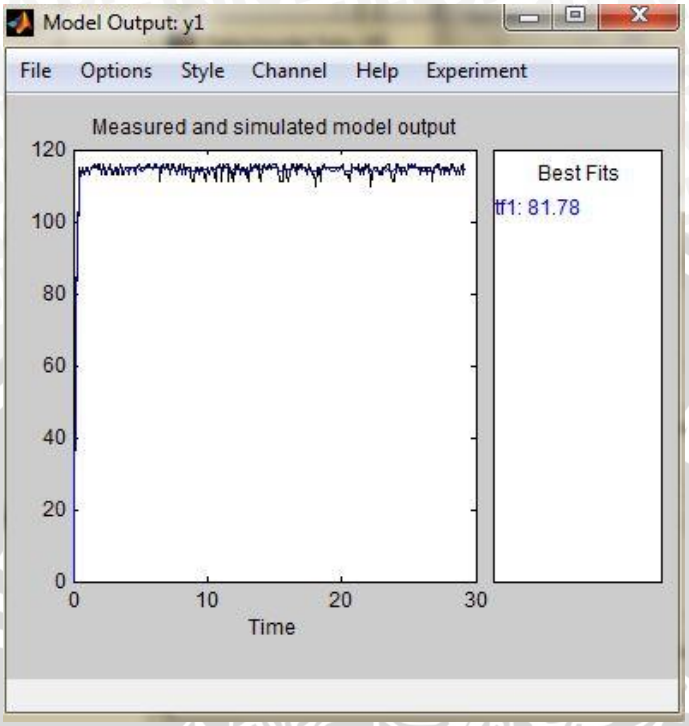
Gambar 3.5 *Input* dan *output* sinyal PRBS

5. Dengan menggunakan sintaks *ident* pada *command window* pada *matlab*, data sinyal PRBS dan data kecepatan motor yang telah disimpan kemudian di *import* pada blok *System Identification Toolbox* (lihat Gambar 3.6).



Gambar 3.6 Tampilan aplikasi ident di *software Matlab*

6. Setelah melakukan beberapa estimasi model berdasarkan data yang telah di *import* didapatkan fungsi alih motor dengan *best fits* sebesara 81.78 seperti dalam Gambar 3.7.



Gambar 3.7 Hasil estimasi model

7. Dari hasil identifikasi, fungsi alih motor yang didapat adalah

$$\frac{Y(s)}{U(s)} = \frac{843.5}{s^2 + 99.27s + 851.1} \qquad (3-1)$$

$$F(s) = \frac{843.5}{s^2 + 99.27s + 851.1}$$

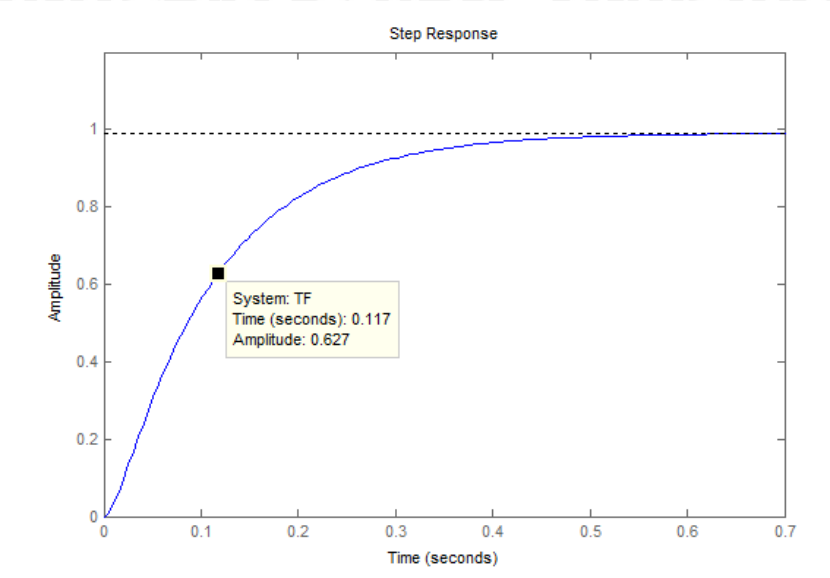
$$F(s) = \frac{{\omega_n}^2}{s^2 + 2\xi\omega_n s + {\omega_n}^2} \qquad (3-2)$$

$$2\xi\omega_n = 99.27 \qquad (3-3)$$

$$\omega_n = \sqrt{843.5} = 29.043 \qquad (3-4)$$

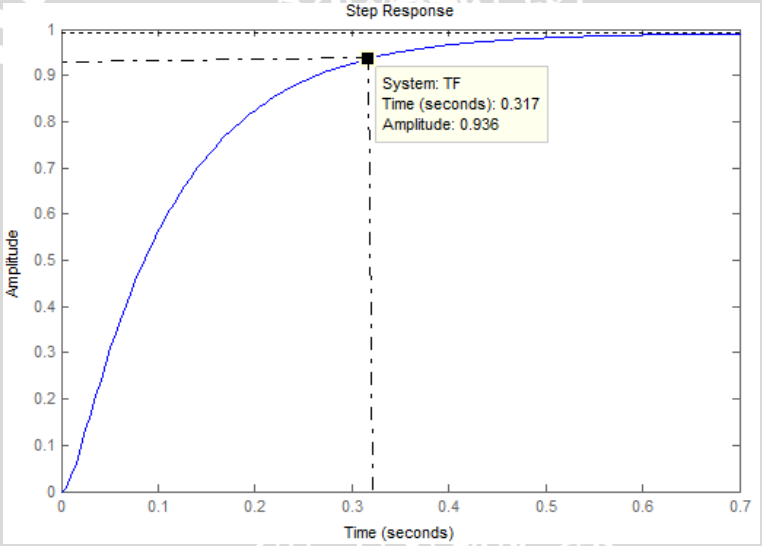
$$\xi = \frac{99.27}{2 \text{ x } 29.043} = 1.70901 \qquad (3-5)$$

8. Dengan memberikan masukan *unit step* pada program *Matlab* didapatkan *output* dengan nilai *time constant* yang merupakan waktu yang dibutuhkan untuk mencapai 63.2% dari nilai *steady state* yaitu 0.117 detik seperti dalam Gambar 3.8.



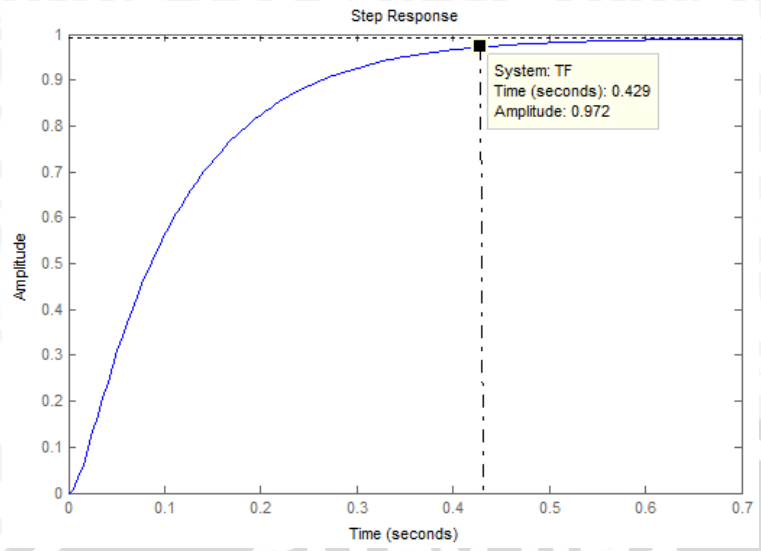
Gambar 3.8 Nilai *time constant output* motor BLDC dengan *input unit step*

Dengan memberikan *input unit step* pada program *Matlab* didapatkan nilai *rise time* adalah 0.317 detik seperti dalam Gambar 3.9.



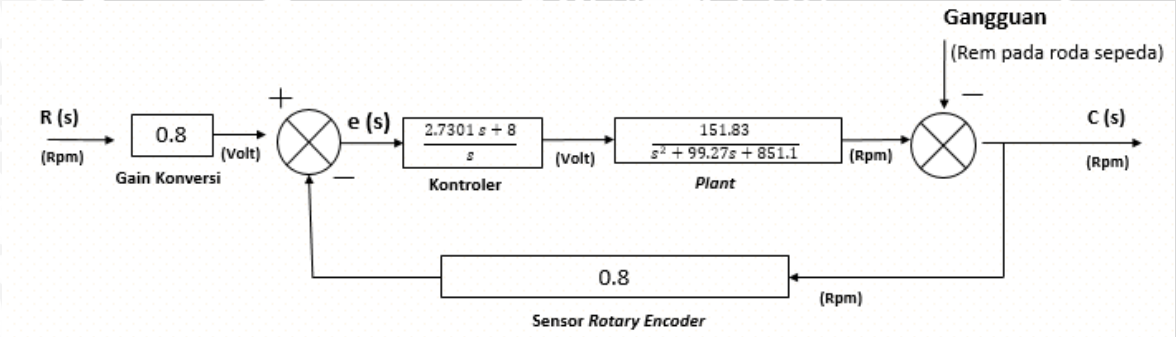
Gambar 3.9 Nilai *rise time output* motor BLDC dengan *input unit step*

Dengan memberikan *input unit step* pada program *Matlab* didapatkan nilai *settling time* adalah 0.429 detik seperti dalam Gambar 3.10.



Gambar 3.10 Nilai *settling time output* motor BLDC dengan *input unit step*

9. Setelah didapatkan karakteristik dari masing-masing blok, maka didapatkan diagram blok perancangan sistem seperti dalam Gambar 3.11.



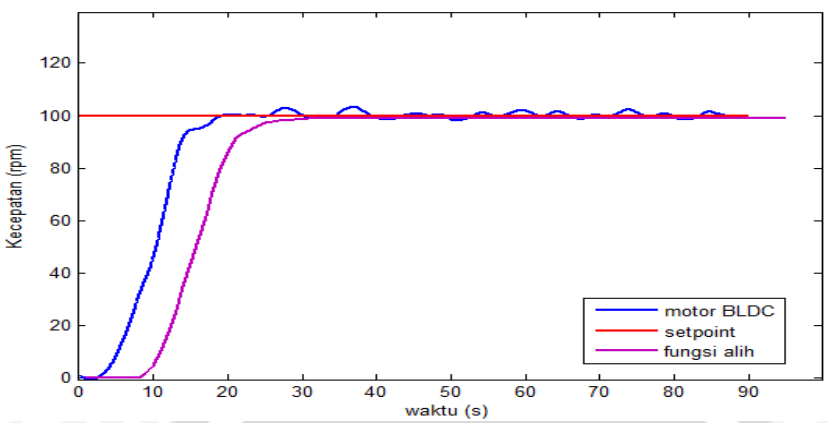
Gambar 3.11 Diagram blok sistem keseluruhan

Dengan :

R(s) = *input* kecepatan
Gain Konversi = sebagai pengubah kecepatan (Rpm) menjadi tegangan (V)
e(s) = *error*
Gangguan = rem pada roda belakang sepeda
C(s) = *output* kecepatan

3.5 Validasi Fungsi Alih Motor *Brushless* DC (BLDC)

Dalam melakukan validasi fungsi alih motor dilakukan dengan cara membandingkan *output plant* dan *output* kecepatan motor BLDC yang dapat dari pembacaan *rotary encoder* dengan memberikan masukan pulsa *unit step*. Berikut perbandingan kedua *output* yang didapat dengan menggunakan Matlab (lihat Gambar 3.12).



Gambar 3.12 Validasi fungsi alih dengan *output* motor BLDC

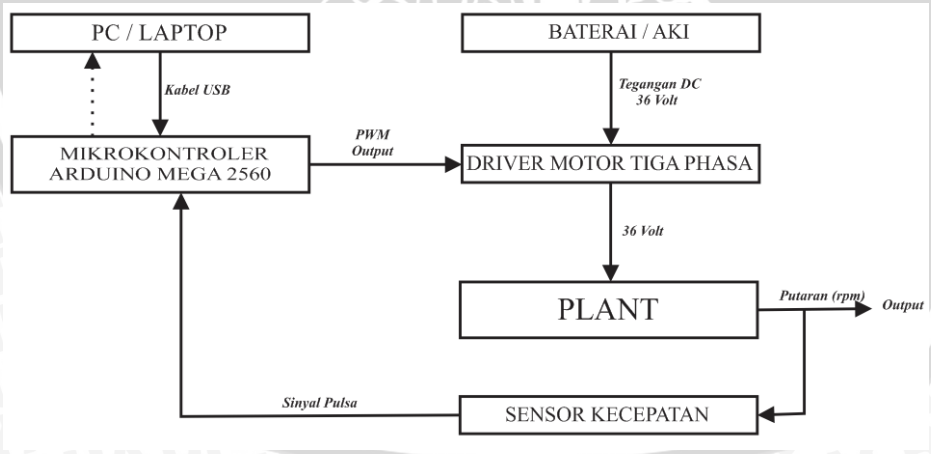
Dari Gambar 3.12 dapat dilihat bahwa *output* sistem yang telah didapat dari proses identifikasi hampir menyerupai *output* kecepatan motor BLDC. Jadi fungsi alih yang telah didapatkan dianggap dapat mewakili pemodelan *plant* motor BLDC.

3.6 Pembuatan Perangkat Keras

Pembuatan perangkat keras dilakukan sebagai langkah awal sebelum terbentuknya suatu sistem beserta pemrogramannya, hal ini dimaksudkan agar Desain kontroler PI pada system kontrol kecepatan sepeda listrik dengan metode *root locus* dapat berjalan sesuai deskripsi awal yang telah direncanakan.

Pembuatan perangkat keras yang dilakukan meliputi:

1. Skema pembuatan perangkat keras.



Gambar 3.13 Skema pembuatan perangkat keras

2. Penentuan modul elektronik yang digunakan meliputi :
   - Komputer atau PC yang sudah *terinstal software Arduino* dan *Matlab.*

Gambar 3.14 Komputer atau PC

- *Baterai* atau aki berfungsi sabagai sumber catu daya motor *Brushless* DC dimana memiliki *range* 0-36 Volt.

Gambar 3.15 Baterai atau Aki

- Mikrokontroler *Arduino Mega 2560* berfungsi sebagai penyimpan algoritma kontroler sistem, pengendali *driver* motor, dan mengolah data yang dikirim sensor.

Gambar 3.16 *Arduino Mega 2560*

- *Driver* motor *Brushless* DC yang digunakan adalah *driver* motor tiga fasa.

Gambar 3.17 Driver tiga fasa

- Motor yang digunakan adalah motor *Brushless* DC (BLDC) dengan catu daya maksimal sebesar 48 V. Motor BLDC berfungsi sebagai aktuator untuk menggerakan sepeda listrik.

Gambar 3.18 Motor *bruhsless* DC

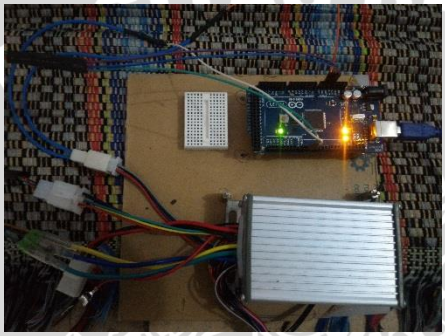
- Sensor yang digunakan adalah sensor *Rotary Encoder* merupakan sebuah sensor kecepatan. Sensor *Rotary Encoder* membutuhkan suplai daya sebesar 5 V yang diperoleh dari mikrokontroler.

Gambar 3.19 Sensor *rotary encoder*

- Pembuatan modul Motor BLDC yang telah dikopel dengan *Rotary Encoder* dapat dilihat pada Gambar 3.20.

Gambar 3.20 Pembuatan modul motor BLDC dan *rotary encoder*

- Pembuatan modul Mikrokontroler *Arduino Mega 2560* yang terhubung dengan *driver* tiga fasa dapat dilihat pada Gambar 3.21.



Gambar 3.21 modul *Arduino Mega 2560* dan *drive* tiga fasa motor

3.7 Prinsip Kerja Sistem

Prinsip kerja sistem adalah sebagai berikut :

1. *Baterai* atau aki motor diseri kemudian memiliki catu 36 Volt.
2. Aki motor catu 36 Volt sebagai *supply* rangkaian *driver* tiga fasa motor BLDC.
3. Catu daya *Arduino Mega 2560* berupa tegangan 5 VDC yang didapat dari kabel USB *(plug A to plug B)* yang dihubungkan pada komputer.
4. Keluaran *rotary encoder* berupa pulsa dengan tegangan 0 V sampai 5 V sebagai masukan digital pada pin *external interrupt Arduino Mega 2560* yang kemudian di konversi menjadi nilai pembacaan kecepatan motor.
5. Sinyal kontrol dari *Arduino Mega 2560* masuk ke *driver* tiga fasa motor.
6. Motor BLDC yang berputar telah dikopel dengan *rotary encoder* sebagai pembaca putaran motor BLDC.
7. Mencari *output* kecepatan motor BLDC terhadap perubahan sinyal *Pulse Width Modulation* (PWM).

8. Mencari fungsi alih motor BLDC dengan memberikan sinyal *Pseudo* Random *Binary Sequence* (PRBS).
9. Mencari parameter kontroler PI menggunakan metode *root locus*.
10. Menguji parameter kontroler PI yang didapat dalam simulasi *simulink Matlab*.
11. Membandingkan *output* sistem *simulink* dengan *output* sistem motor BLDC yang didapat.
12. Mengimplementasikan hasil desain perancanan pada sistem.

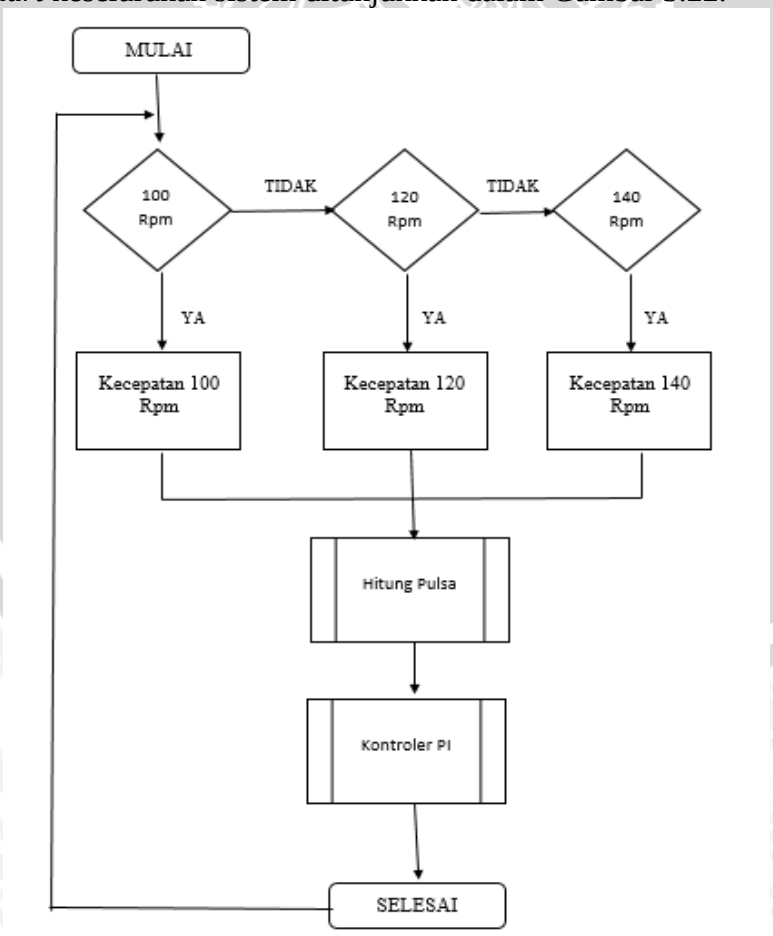
3.8 Perancangan Perangkat Lunak (software)

Perancangan perangkat lunak terbagi menjadi 3 bagian yaitu :

1. *Flowchart* sistem secara keseluruhan
2. *Flowchart* hitung pulsa.
3. *Flowchart* PI.

3.8.1 *Flowchart* Sistem Secara Keseluruhan

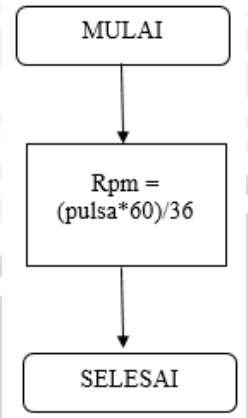
Flowchart keseluruhan sistem ditunjukkan dalam Gambar 3.22.



Gambar 3.22 *Flowchart* sistem secara keseluruhan

3.8.2 *Flowchart* Hitung Pulsa

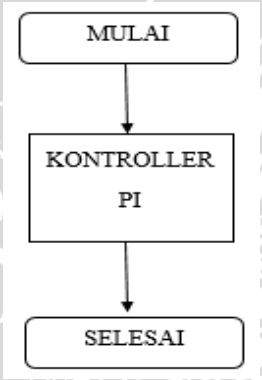
Flowchart perhitungan pulsa kecepatan putar motor dapat dilihat pada Gambar 3.23



Gambar 3.23 *Flowchart* perhitungan pulsa kecepatan putar motor

3.8.3 *Flowchart* Kontrol Proporsional Integral (PI)

Flowchart Kontrol Proporsional Integral (PI) dapat dilihat pada Gambar 3.24.



Gambar 3.24 *Flowchart* kontrol proporsional integral (PI)

3.9 Perancangan Algoritma

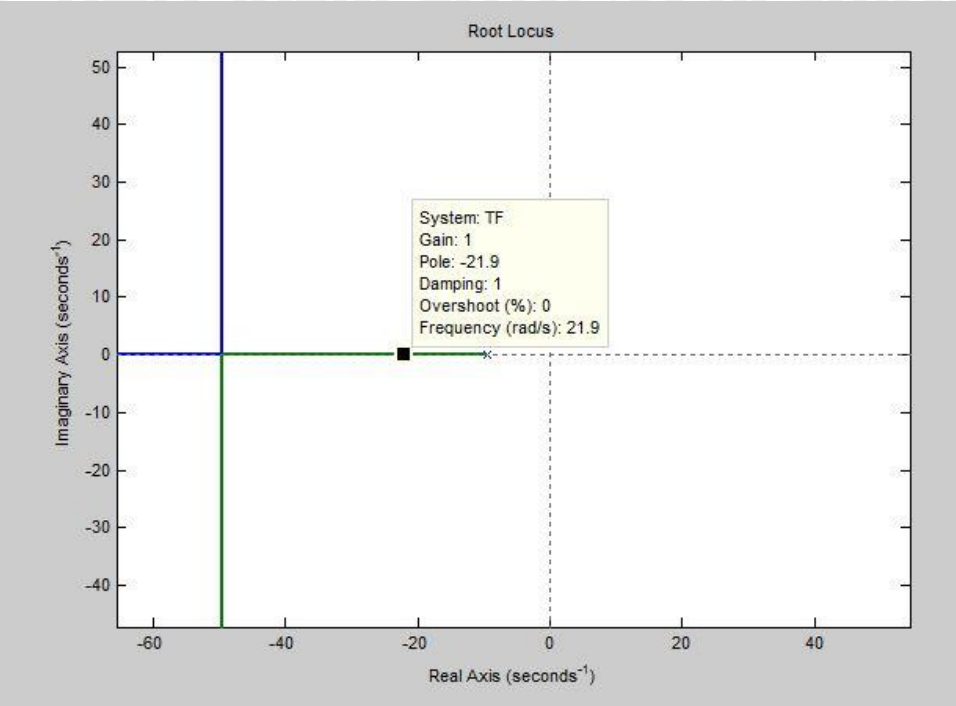
Dalam perancangan perangkat lunak, hal pertama yang dilakukan adalah mengetahui karakteristik motor *Brushless* DC. Setelah itu dilakukan perhitungan untuk mencari parameter kontroler PI dengan metode *root locus*. Setelah didapatkan parameter kontroler kemudian didiskritisasi agar dapat diolah oleh mikrokontroler.

3.9.1 Penentuan parameter kontroler PI dengan metode *root locus*

Untuk memenuhi tujuan performansi *loop* yang diinginkan, maka perlu ditambahkan kontroler pada sistem tersebut. Kontroler yang dipilih ialah kontroler Proporsional Integral (PI). Setelah didapatkan fungsi alih sistem yaitu $\mathbf{F(s)} = \frac{843.5}{s^2+99.27s+851.1}$. Selanjutnya adalah menentukan letak simpul *loop* tertutup.

Berdasarkan fungsi alih *loop* tertutup dapat diketahui sistem berorde dua. Nilai parameter PI ditentukan oleh pemilihan *pole* pada diagram *root locus*. Pada penelitian

ini digunakan s_1 = -21.9. pada pole s_1 didapatkan nilai *gain* = 1, rasio redaman = 1, dan *overshoot* = 0. Penentuan letak *pole* pada diagram *root locus* terlihat dalam Gambar 3.25.



Gambar 3.25 Letak *pole* pada diagram *root locus* (Perancangan)

Setelah ditentukan letak *pole* yang diinginkan. Kemudian dengan menstubtitusi nilai s_1 pada nilai fungsi alih sistem dan memvariasikan nilai Ki akan didapatkan parameter PI dalam Tabel 3.4. Pencarian parameter Kp dan Ki dengan menggunakan *Matlab 2013* ditunjukkan pada *listing* program berikut :

```
%nilai pole yang ditentukan dari Gambar 3.25 root locus
    s1= -21.9
    KI=[1 2 3 5 8]
    plant_num=[0 0 843.5];
    plant_den=[1 99.27 851.1];

    s1mag = abs(s1)
    beta = angle(s1)
    plant_a1 = polyval(plant_num,s1)/polyval(plant_den,s1);
    plants1mag = abs(plant_a1)
    psi = angle(plant_a1)
    t=0:1:20:300;

    for k =1:5

      KP=-sin(beta+psi)/(plants1mag*sin(beta))-2*KI(k)*cos(beta)/s1mag
      nilai_KI = KI(k)
```

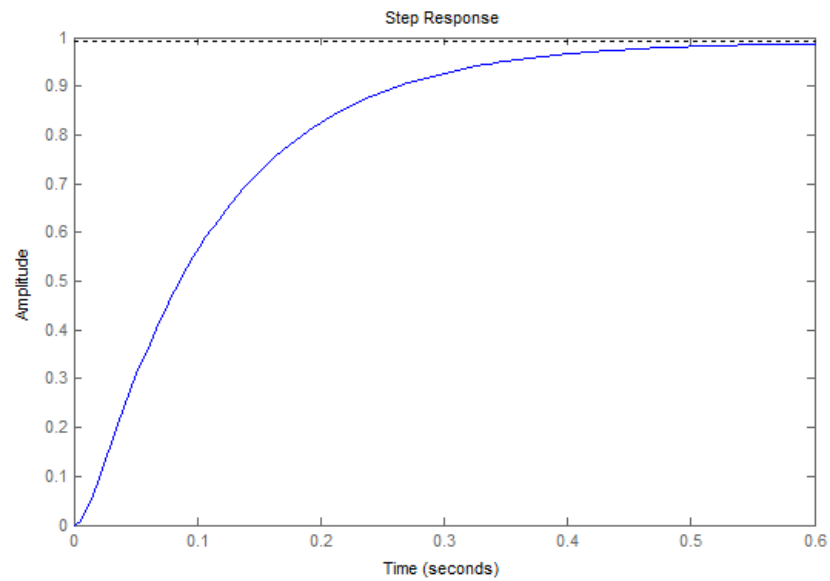
```
    Gcnum = [KP KI (k)];
    Gcden = [1 0];

    Tnum = conv(plant_num,Gcnum);
    Tden = conv(plant_den,Gcden)+conv(plant_num,Gcnum);

    r = roots(Tden)
    step (Tnum,Tden,t)
  hold on
end

   hold off
   figure, rlocus(Tnum,Tden)
```

Hasil pencarian parameter Kp dan Ki dari perhitungan pada program diatas ditunjukkan dalam Tabel 3.4

Tabel 3.4 Parameter PI dengan s_1= -21.9

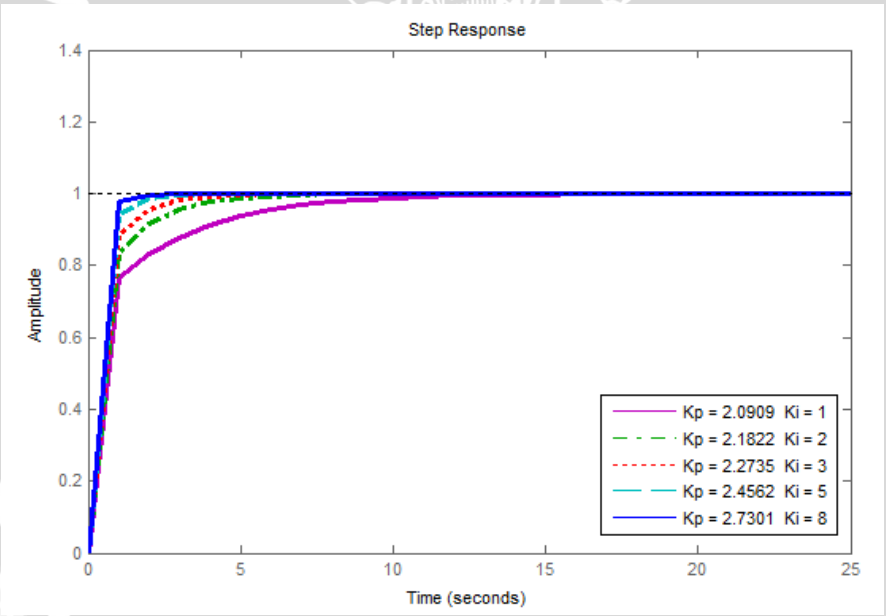
No	Kp	Ki	Pole		
			1	2	3
1	2.0909	1	-117.3074	-21.9000	-0.3283
2	2.1822	2	-118.7457	-21.9000	-0.6487
3	2.2735	3	-120.1918	-21.9000	-0.9614
4	2.4562	5	-123.1063	-21.9000	-1.5643
5	2.7301	8	-127.5306	-21.9000	-2.4161

Setelah didapatkan nilai Kp dan Ki hasil perhitungan, kemudian dilakukan pengujian terhadap sistem, dan parameter yang sesuai dengan sistem. Pada Gambar 3.36 berikut adalah grafik *output plant*.



Gambar 3.26 *Output plant*

Pada Gambar 3.27 berikut adalah grafik *output plant* dengan menggunakan kontroler dan memvariasikan nilai Kp dan Ki. Seperti dalam tabel 3.4



Gambar 3.27 *Output plant* dengan kontroler kecepatan

Dari Gambar 3.26 dapat diketahui bahwa *output plant* tanpa menggunakan kontroler PI lebih lama menuju *steady state*. Sedangkan pada Gambar 3.27 dengan digunakannya parameter PI hasil perhitungan dengan *Root Locus* didapatkan *output* yang lebih cepat dari pada *output* tanpa menggunakan PI. Dari 5 jenis parameter PI yang didapat dipilih nilai PI yang memiliki *output* terbaik yaitu: Kp = 2.7301 dan Ki = 8.

3.9.2 Diskritisasi

Dari 5 jenis parameter PI yang didapatkan diatas dipilih nilai PI yang memiliki *output sistem* yang terbaik yaitu Kp = 2.7301 dan Ki = 8.

Dari peroleh nilai Kp dan Ki maka diperoleh persamaan transformasi *laplace* kontroler PI yang ditunjukkan pada (persamaan 3-6).

$$C_{(s)} = \left(K_p + \frac{K_I}{s}\right) E_{(s)} \qquad (3-6)$$

persamaan diatas belum bisa dimasukan ke dalam program pada *Arduino* maka dari itu persamaan kontinyu harus diubah ke dalam bentuk diskrit melalui transformasi z. Dalam transformasi z dibutuhkan periode sampling (Ts). Digunakan metode *Backward Difference* dikarenakan metode ini memiliki sensitifitas yang lebih rendah dari metode lainnya sehingga sistem tidak mudah mengalami suatu perubahan atau osilasi saat diberi gangguan dan tetap mempertahankan performansinya. Metode *Backward Difference* mengganti nilai notasi s pada *laplace* setara dengan :

$$s = \frac{1 - z^{-1}}{Ts} \qquad (3-7)$$

Maka (persamaan 3-7) disubstitusikan ke (persamaan 3-6) menjadi :

$$C_{(z)} = \left(K_p + \frac{K_i}{\frac{1 - z^{-1}}{Ts}}\right) E_{(z)} \qquad (3-8)$$

$$= \left(K_p + \frac{K_i \, x \, Ts}{1 - z^{-1}}\right) E_{(z)} \qquad (3-9)$$

Lalu didapatkan :

- Kontroler Proporsional : $C_{p(z)} = K_p E_{(z)}$ (3 – 10)
- Kontroler Integral : $C_{i(z)} = \frac{K_I \, x \, Ts}{1 - z^{-1}} E_{(z)}$

$$C_{i(z)} = \left[C_{i(z)} z^{-1} + K_i \; Ts \, E_{(z)}\right] \qquad (3-11)$$

Sehingga

$$C_{(z)} = K_p \, E_{(z)} + \left[C_{i(z)} z^{-1} + K_i \; Ts \, E_{(z)}\right] \qquad (3-12)$$

Dari persamaan di atas kemudian diubah ke dalam persamaan beda sehingga didapatkan persamaan berikut :

$$C_{(K)} = K_p \, E_{(K)} + \left[C_i(K-1) + K_i \, x \, Ts \, E_{(K)}\right] \qquad (3-13)$$

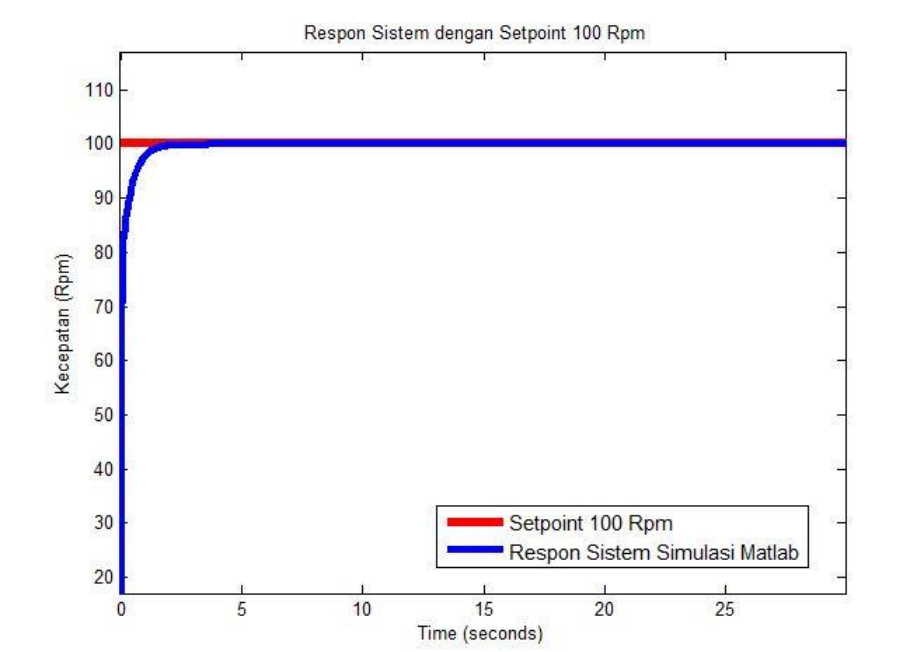
Dimana (K-1) adalah kondisi sebelumnya. Persamaan diatas lalu dimasukan dalam program mikrokontroler.

BAB IV
HASIL DAN PEMBAHASAN

Hasil dan pembahasan dilakukan dengan melakukan pengujian keseluruhan sistem. Pengujian keseluruhan sistem dilakukan untuk menguji apakah parameter yang sudah didapatkan diaplikasikan pada simulasi *Matlab* dan alat sudah sesuai dengan *input* yang diinginkan serta mengetahui hasil *outputnya*. Pengujian keseluruhan dibagi menjadi beberapa bagian yaitu :

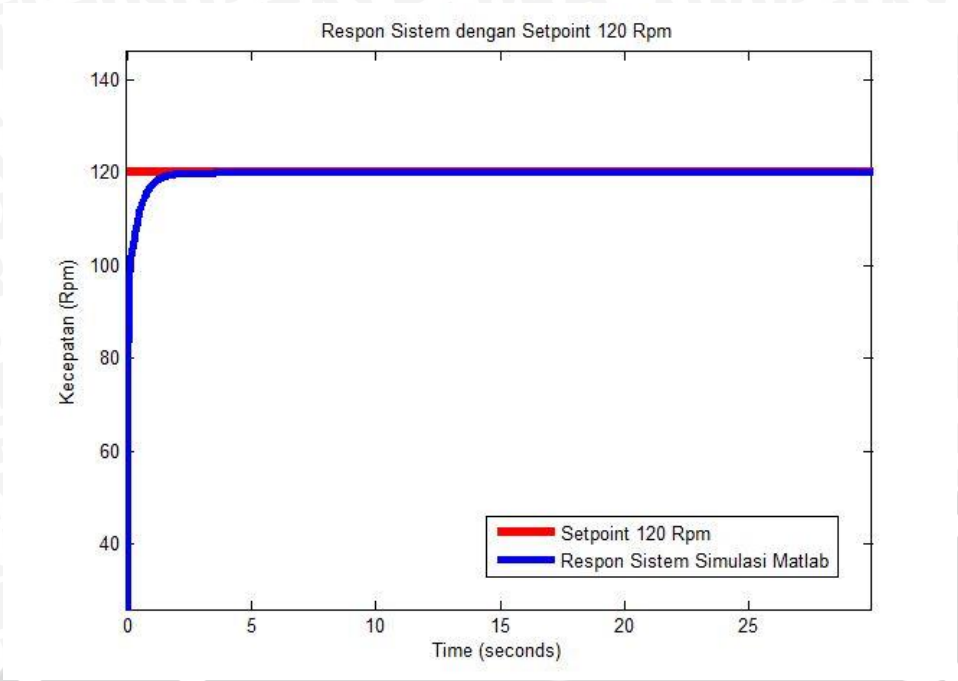
4.1 Pengujian Sistem pada Simulasi Aplikasi *Matlab*

Pengujian ini dilakukan untuk mengetahui apakah nilai parameter yang didapatkan sudah sesuai dengan *setpoint*. Pada pengujian ini *setpoint* yang diberikan sebesar 100 rpm, 120 rpm, dan 140 rpm. Hasil dari pengujian sistem ditunjukkan dalam Gambar 4.1, Gambar 4.2, dan Gambar 4.3.



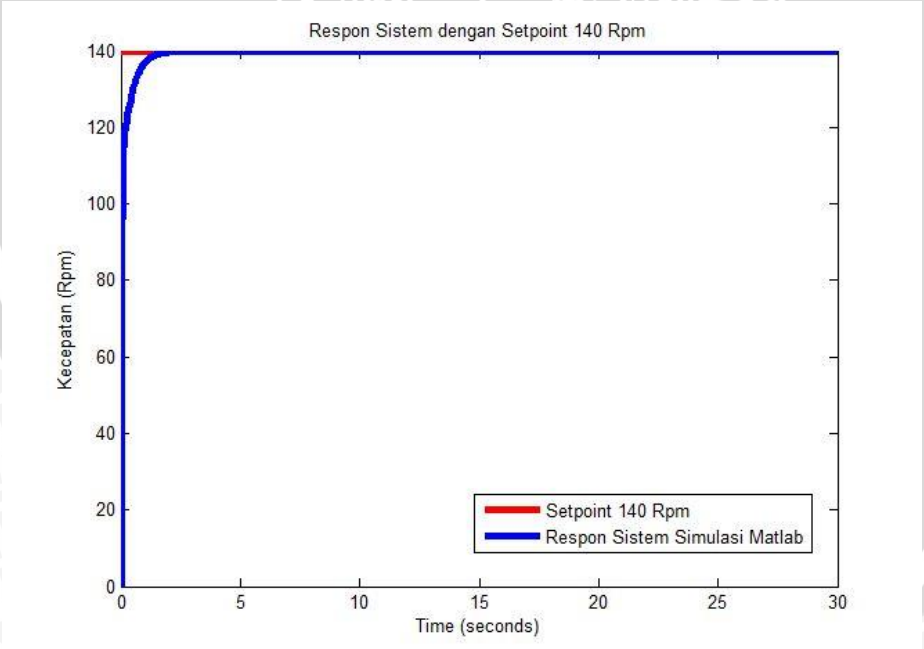
Gambar 4.1 *Output* sistem simulasi *Matlab* dengan *setpoint* 100 Rpm

Dari *output* yang ditunjukan dalam Gambar 4.1 dapat disimpulkan bahwa sistem tidak terdapat *error steady state*, tidak ada *overshoot*, dan *settling time* sebesar 2 detik.



Gambar 4.2 *Output* sistem simulasi *Matlab* dengan *setpoint* 120 Rpm

Dari *output* yang ditunjukan dalam Gambar 4.2 dapat disimpulkan bahwa sistem tidak terdapat *error steady state*, tidak ada *overshoot*, dan *settling time* sebesar 1.5 detik.



Gambar 4.3 *Output* sistem simulasi *Matlab* dengan *setpoint* 140 Rpm

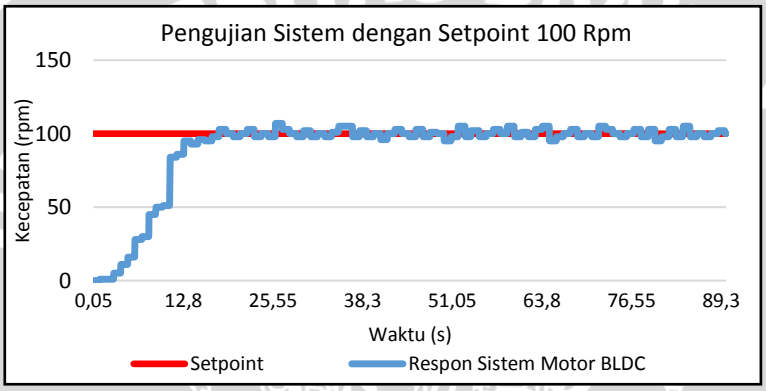
Dari *output* yang ditunjukan dalam Gambar 4.3 dapat disimpulkan bahwa sistem tidak terdapat *error steady state*, tidak ada *overshoot*, dan *settling time* sebesar 1.3 detik.

4.2 Pengujian Sistem pada Alat

Pengujian ini dilakukan untuk mengetahui apakah *output* yang didapat dalam simulasi *Matlab* sama dengan *output* yang didapat dari alat dengan nilai parameter kontroler dan *setpoint* yang sama.

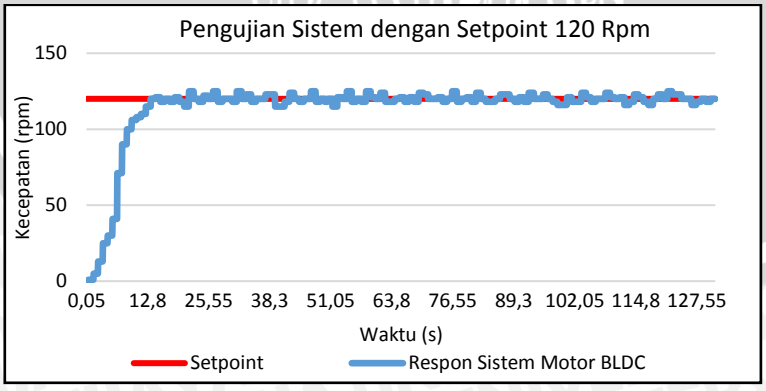
4.2.1 Pengujian Sistem pada Alat tanpa Gangguan

Pengujian ini dilakukan untuk mengetahui bagaimana *output* sistem dengan menggunakan kontroler PI. Pada pengujian ini *setpoint* yang diberikan sebesar 100 rpm, 120 rpm, dan 140 rpm. Hasil dari pengujian sistem ditunjukkan dalam Gambar 4.4, Gambar 4.5, dan Gambar 4.6.



Gambar 4.4 *Output* motor BLDC dengan *setpoint* 100 Rpm

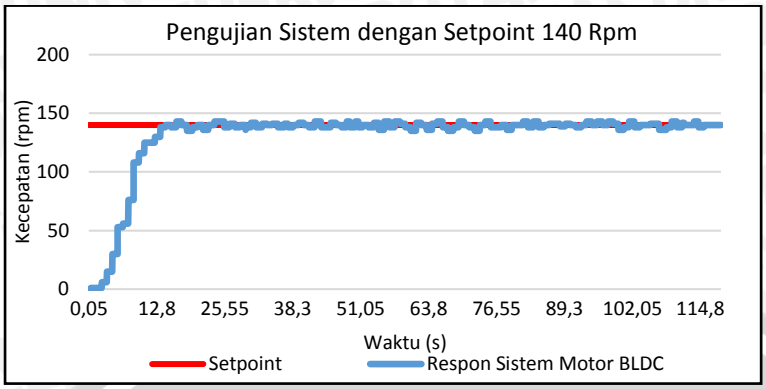
Dari *output* sistem yang ditunjukkan dalam Gambar 4.4 dapat disimpulkan bahwa sistem terdapat *error steady state* sebesar 5 %, *settling time* sebesar 27 s, dan terdapat *overshoot* sebesar 7 %. Sehingga *output* sistem pada alat tidak sama dengan *output* simulasi *Matlab*.



Gambar 4.5 *Output* motor BLDC dengan *setpoint* 120 Rpm

Dari *output* sistem yang ditunjukkan dalam Gambar 4.5 dapat disimpulkan bahwa sistem terdapat *error steady state* sebesar 4.17 %, *settling time* sebesar 24 s,

dan tidak terdapat *overshoot.* Sehingga *output* sistem pada alat tidak sama dengan *output* simulasi *Matlab*.

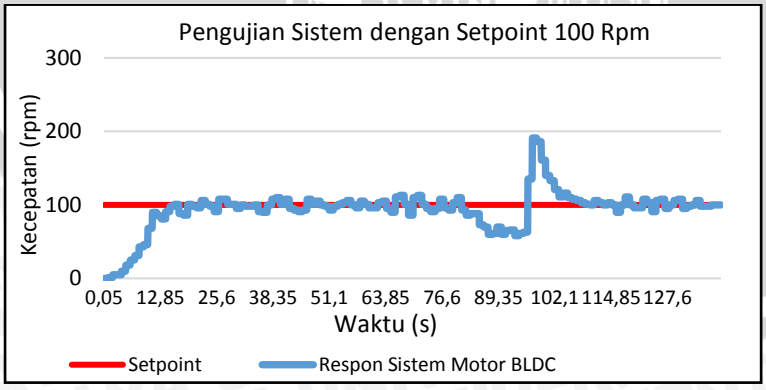


Gambar 4.6 *Output* motor BLDC dengan *setpoint* 140 Rpm

Dari *output* sistem yang ditunjukkan dalam Gambar 4.6 dapat disimpulkan bahwa sistem terdapat *error steady state* sebesar 2.14 %, *settling time* sebesar 20 s, dan tidak terdapat *overshoot.* Sehingga *output* sistem pada alat tidak sama dengan *output* simulasi *Matlab*.

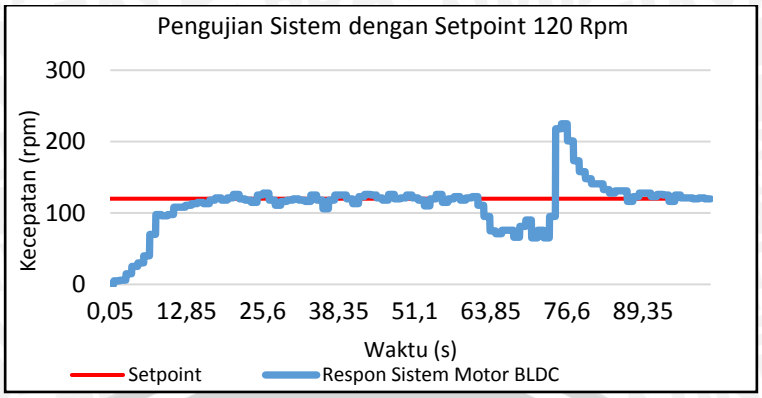
4.2.2 Pengujian Sistem pada Alat dengan Diberi Gangguan

Pengujian ini dilakukan untuk mengetahui apakah *output* dapat kembali dalam keadaan *steady state* ketika diberi gangguan dengan menggunakan kontroler PI. Pada pengujian ini *setpoint* yang diberikan sebesar 100 rpm, 120 rpm, dan 140 rpm dan gangguan yang diberikan berupa gesekan dari rem. Hasil pengujian ditunjukkan dalam Gambar 4.7, Gambar 4.8, dan Gambar 4.9.



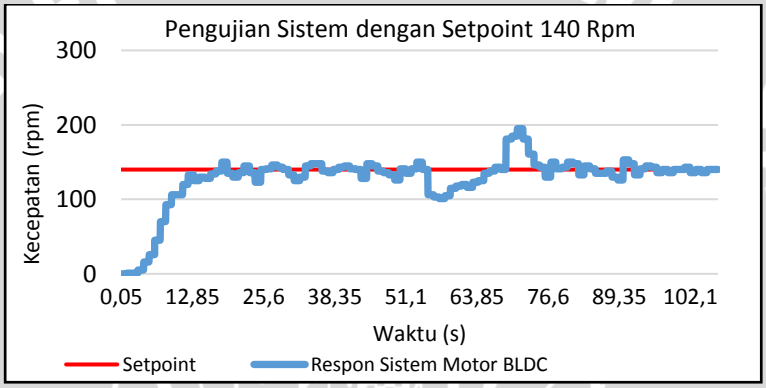
Gambar 4.7 *Output* motor BLDC dengan *setpoint* 100 Rpm dengan gangguan

Dari Gambar 4.7 *output* sistem dapat disimpulkan bahwa sistem terdapat osilasi dan *output* sistem kembali keadaan *steady state* dengan *recovery time* sebesar 38 s.



Gambar 4.8 *Output* motor BLDC dengan *setpoint* 120 Rpm dengan gangguan

Dari Gambar 4.8 *output* sistem dapat disimpulkan bahwa sistem terdapat osilasi dan *output* sistem kembali keadaan *steady state* dengan *recovery time* sebesar 39 s.



Gambar 4.9 *Output* motor BLDC dengan *setpoint* 140 Rpm dengan gangguan

Dari Gambar 4.9 *output* sistem dapat disimpulkan bahwa sistem terdapat osilasi dan *output* sistem kembali keadaan *steady state* dengan *recovery time* sebesar 40 s..

BAB V
KESIMPULAN DAN SARAN

5.1 KESIMPULAN

Kesimpulan dari perancangan dan analisis adalah sebagai berikut:

1. Berdasarkan data *output* sistem yang diperoleh dari pengujian dengan menggunakan sinyal *Pseudo-Random Binary Sequence* (PRBS) didapat nilai fungsi alih $\frac{843.5}{s^2+99.27s+851.1}$ dengan nilai *best fit* sebesar 81.78.
2. Pada hasil pengujian yang didapatkan :
   a. Hasil *output* sistem yang diperoleh dari pengujian dengan menggunakan metode *root locus* didapat nilai parameter kontroler PI dengan penguatan sebesar K_p = 2.7301 dan K_i= 8.
   b. Hasil simulasi dengan *Matlab*, output sistem pada *setpoint* 100 rpm, 120 rpm, dan 140 rpm tidak terdapat *error steady state*, tidak ada *overshoot*, dan *settling time* sebesar 2 detik, 1.5 detik, dan 1.3 detik. Sedangkan hasil implementasi pada alat tanpa diberi gangguan, *output* sistem pada motor BLDC dengan nilai *setpoint* 100 rpm terdapat *error steady state* 5%, *settling time* sebesar 27 detik dan *overshoot* 7%. Pada *setpoint* 120 rpm terdapat *error steady state* 4.17%, *settling time* sebesar 24 detik dan tidak terdapat *overshoot*. Dan pada *setpoint* 140 rpm terdapat *error steady state* 2.14%, *settling time* sebesar 20 detik dan tidak terdapat *overshoot*. Sehingga *output* sistem pada alat tidak sama dengan *output* sistem pada simulasi Matlab.
   c. Hasil implementasi pada alat dengan gangguan, pada *setpoint* 100 rpm, 120 rpm, dan 140 rpm. *Output* mengalami perlambatan dan percepatan. Dan *output* sistem motor kembali keadaan *steady state* dengan *recovery time* sebesar 38 s, 39 s, dan 40 s.

5.2 SARAN

Saran yang dapat diberikan untuk penelitian selanjutnya adalah pada pengujian berbeban proses pengereman dilakukan pada saat sistem pertama kali berjalan, mengimplementasi motor *Brushless* DC sebagai aktuator pada mobil listrik, dan menggunakan sensor kecepatan

yang lebih teliti sehingga data pengujian yang didapatkan mendekati dengan nilai sebenarnya.

DAFTAR PUSTAKA

Adva, Jamaludin. 2015. *Motor Brushless Direct Current.* Bandung.

Amalia, Z. 2016. *Desain Kontroler Struktur Output Feedback Control Dengan Pole Placement Pada Pengontrolan Tegangan Output Generator DC.* Skripsi. Teknik Eleketro. Universitas Brawijaya. Malang.

Artanto, D. 2009. *Merakit PLC dengan Mikrokontroler.* Jakarta : Elex Media Komputindo.

Arduino. *Arduino Mega 2560.* https://www.arduino.cc/en/Main/ArduinoBoardMega2560. (diakses 20 juni 2016)

Bogipower. *Inverter Tiga Fasa.* http://www.bogipower.com/. (diakses 5 Maret 2016).

Hamdi, Irham Tantowi. 2015. *Rancang Bangun Three Phase Six Step PWM Inverter sebagai Pedal Assited System (PAS) Sepeda Listrik.* Malang : Skripsi Teknik Elektro Universitas Brawijaya Malang.

Juniar, Ahmad. 2011. *Sepeda Listrik.* http://ahmadjuniar.blogspot.co.id/2011/06/sepeda-listrik.html. (diakses 20 Mei 2016)

Landau, I. D., Zioto G. 2006. *Digital Control System. London* : Springer-Verlag.

Mupasanta, S. 2016. *Pengontrolan Kadar Keasaman (pH) dan Aliran Air pada Sistem Hidroponik Stroberi menggunakan Kontroler PID.* Skripsi. Teknik Eleketro. Universitas Brawijaya. Malang

Ogata, K. 1997. *Teknik Kontrol Automatik. Terjemahan* : Edi Laksono Ir. Jakarta : Penerbit Erlangga.

Ogata, k. 2010. *Modern Control Engineering.* Pearson Education, Inc,, publishing as Prentice Hall, One Lake Strees, Upper Saddle River, New Jersey 07458. Fifth edition.

Prabarianto, Bayu. 2015. *Metode Root Locus Untuk Mencari Parameter PID Pada Pengontrolan Kecepatn Motor DC D-6759 Menggunakan Arduino Mega 2560.* Malang : Skripsi Teknik Elektro Universitas Brawijaya Malang.

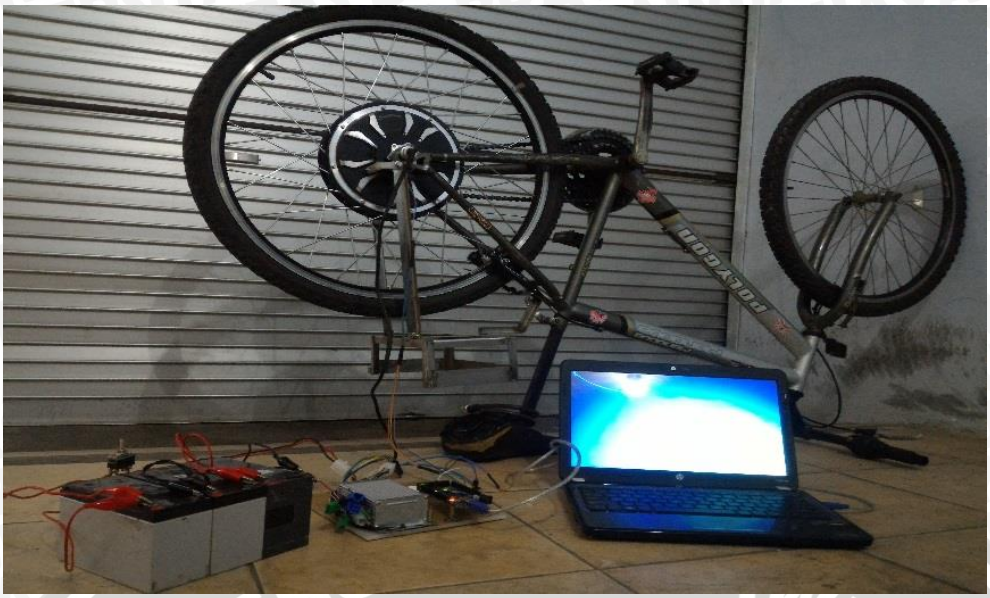
Rozi, Fakhrur. 2015. *Sistem Kontrol Pengendali PI pada Kecepatan Motor Berbasis Arduino Mega 2560.* Malang : Skripsi Teknik Elektro Universitas Brawijaya Malang.

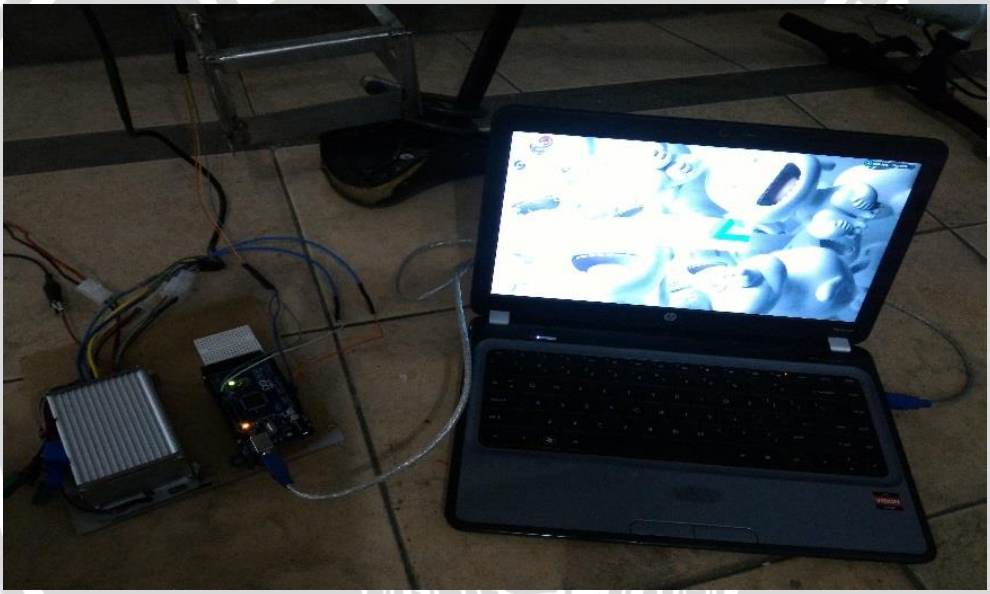
Rizki, Setiawan. 2012. *Kontrol Kecepatan Motor DC Dengan Metode PID Menggunakan Visual Basic6.0 Dan Mikrokontroler Atmega 16.* Skripsi. Teknik Eleketro. Universitas Brawijaya

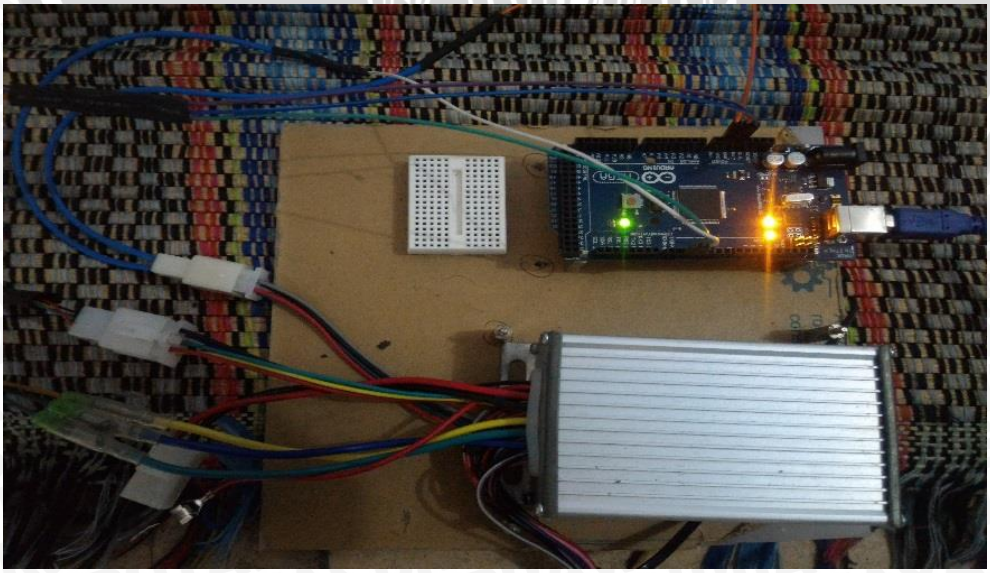
Amruchly, Ade. 2014. *Metode Root Locus Untuk Mencari Parameter PID Dalam Pengendalian Posisi Stamping Rod Berbasis Pneumatic Menggunakan Arduino Uno.* Skripsi. Teknik Elektro. Universitas Brawijaya.

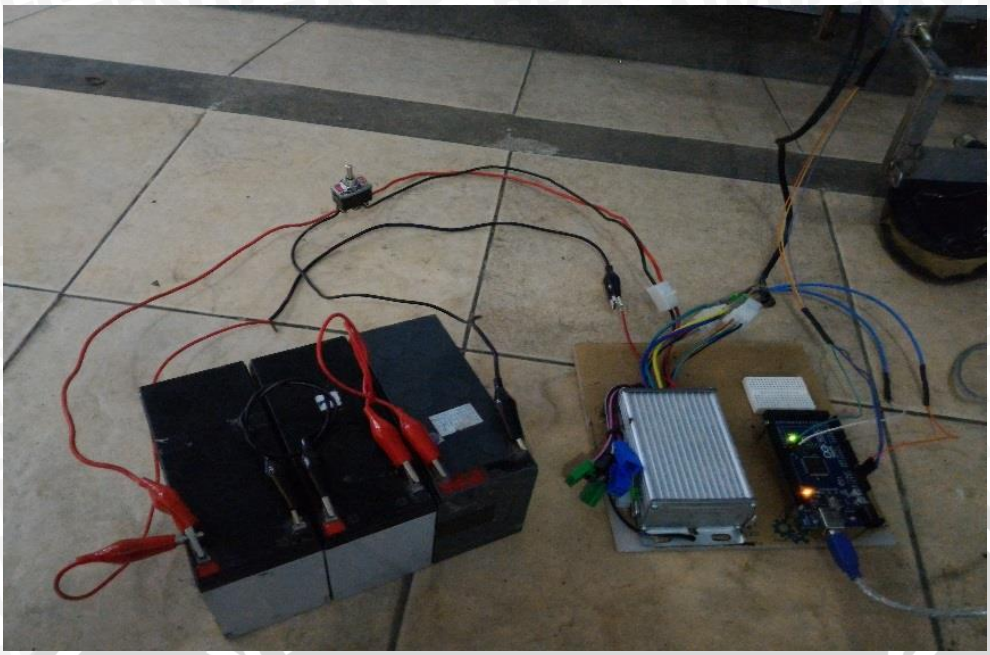
LAMPIRAN I

FOTO ALAT









LAMPIRAN II

LISTING PROGRAM

Listing Program Matlab Mencari S

```
num=[843.5];
den=[1 99.27 851.1];
TF=tf(num,den)

TF =
          843.5
  ---------------------
  s^2 + 99.27 s + 851.1

Continuous-time transfer function.

rlocus(TF)
```

Listing Program Matlab Mencari Rise Time,Settling Time, Peak Time dan Overshoot

```
num=[843.5];
den=[1 99.27 851.1];
TF=tf(num,den)

sys = TF

sys =
                843.5
 ---------------------
       s^2 + 99.27 s + 851.1

Continuous-time transfer function.

S = stepinfo(sys,'RiseTimeLimits',[0.05,0.95])

S =
      RiseTime: 0.3147
  SettlingTime: 0.4245
  SettlingMin: 0.9434
  SettlingMax: 0.9903
  Overshoot: 0
  Undershoot: 0
  Peak: 0.9903
      PeakTime: 0.7608
```

Listing Program Matlab Mencari Parameter Kontroler PI dengan Metode Root Locus

```
s1= -21.9
KI=[1 2 3 5 8]
plant_num=[0 0 843.5];
plant_den=[1 99.27 851.1];

s1mag = abs(s1)
beta = angle(s1)
plant_a1 = polyval(plant_num,s1)/polyval(plant_den,s1);
plants1mag = abs(plant_a1)
psi = angle(plant_a1)
t=0:1:20:300;

for k =1:5

   KP=-sin(beta+psi)/(plants1mag*sin(beta))-2*KI(k)*cos(beta)/s1mag
   nilai_KI = KI(k)

Gcnum = [KP KI (k)];
   Gcden = [1 0];

   Tnum = conv(plant_num,Gcnum);
   Tden = conv(plant_den,Gcden)+conv(plant_num,Gcnum);

   r = roots(Tden)

   step (Tnum,Tden,t)
   hold on
end
hold off
figure, rlocus(Tnum,Tden)
```

Listring Program Keseluruhan dengan Arduino Mega 2560

```
/*********************
 SEMANGAT SKRIPSI 2016
**********************/

#define pwm 3

int set_point;          // Set point
float y;                // Nilai kecepatan Motor DC
float error;            // set_point - y
float last1_error;      // error sebelumnya
volatile int pulsa;     // pulsa rotary encoder
float Ts;               // Time Sampling
float pwmMotor;          // PWM Motor
float Prop;             // Sinyal kontrol Kontroler P
float Intg;             // Sinyal kontrol Kontroler I
float last1_Intg;       // Sinyal kontrol Kontroler I sebelumnya
double MV;              // Sinyal Kontrol Kontroler PI
float Kp;               // Nilai konstanta Kp
float Ki;               // Nilai konstanta Ki
int as,ap;

void setup()
{
 pinMode(pwm,OUTPUT);

 //set_point
 Ts = 0.01;
 Kp = 2.7301;
 Ki = 8;

 last1_error = 0; last1_Intg = 0;

 noInterrupts();
 TCCR1A = 0;
 TCCR1B = 0;
 TCNT1  = 0;

 OCR1A   = 3125;             // compare match register 16MHz/256/20Hz/50ms
 TCCR1B |= (1 << WGM12);     // CTC mode
 TCCR1B |= (1 << CS12);      // 256 prescaler
 TIMSK1 |= (1 << OCIE1A);    // enable timer compare interrupt
```

```
  attachInterrupt(0, hitung_pulsa, FALLING);
  interrupts();                    // enable all interrupts
  Serial.begin(9600);
}
// Timer Controller and Serial Monitor
ISR(TIMER1_COMPA_vect)          // timer compare interrupt service routine
{
   set_point=100;                  // 100 rpm, 120rpm, dan 140 rpm
   if(as==20)
   {
    as=0;
    y = (pulsa*60)/36;
    pulsa = 0;
    }
  as++;
  error =  set_point - y;
  Kontroler();

  Serial.print(set_point);
  Serial.print("\t");
  Serial.print(y);
  Serial.print("\t");
  Serial.print("\n");
}
void Kontroler()
{
  Prop = Kp * error;
  Intg = last1_Intg + (Ki * Ts * error);
  MV = Prop + Intg;

  last1_Intg = Intg;
  last1_error = error;

  if (MV<0)MV = 0;
  pwmMotor = MV*0.046;
}
  void loop()
{
  analogWrite(pwm,pwmMotor);
}
  void hitung_pulsa()
{
   pulsa++;
}
```

LAMPIRAN III

BLOK DIAGRAM

Diagram Blok SIMULINK Sistem tanpa Kontroler

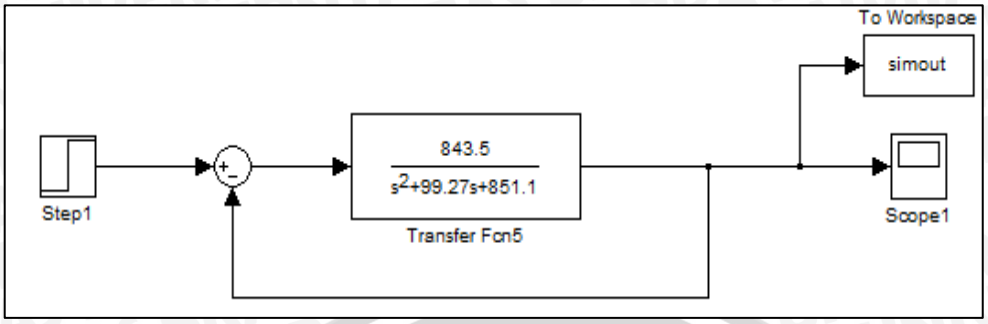
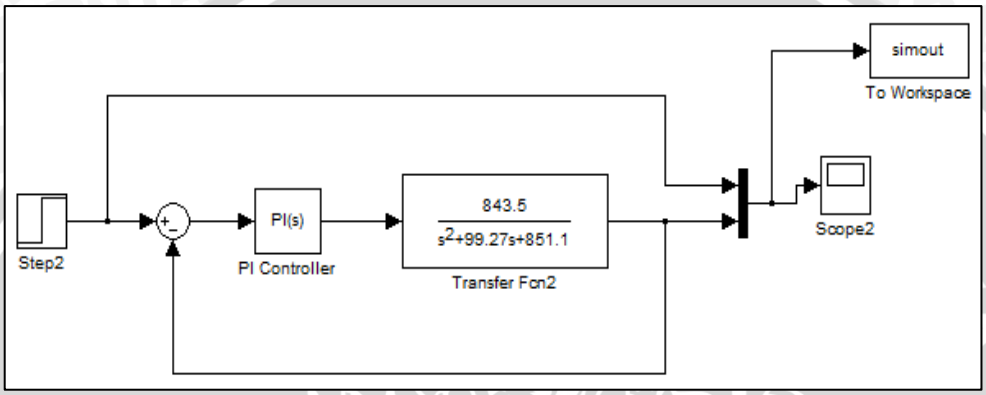


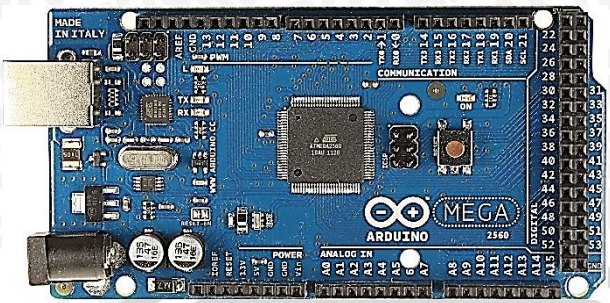
Diagram Blok SIMULINK Sistem dengan Kontroler PI



LAMPIRAN IV

DATA SHEET

Arduino Mega 2560



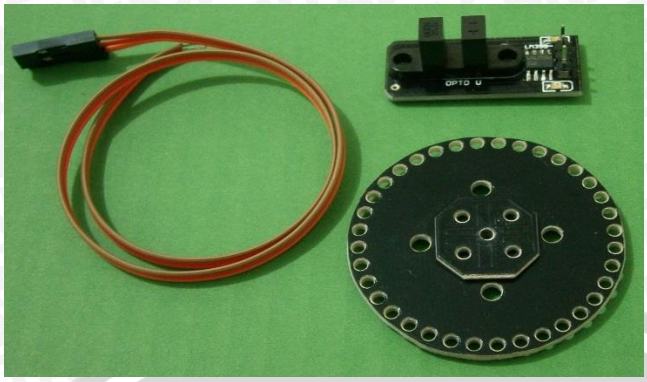
Technical specs

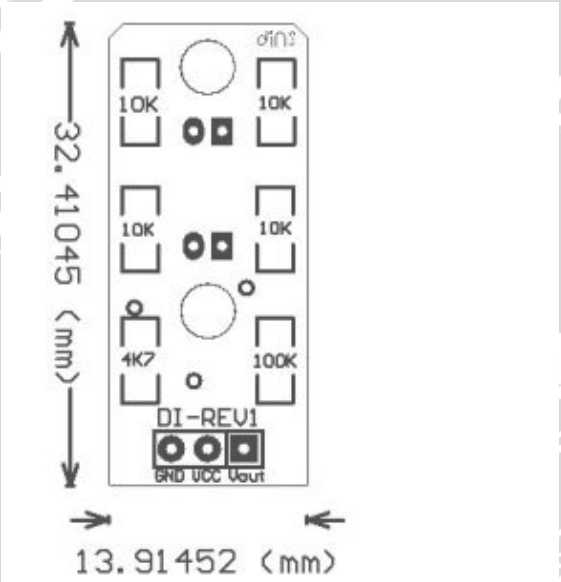
Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Length	101.52 mm
Width	53.3 mm
Weight	37 g

Beberapa pin memiliki fungsi khusus:

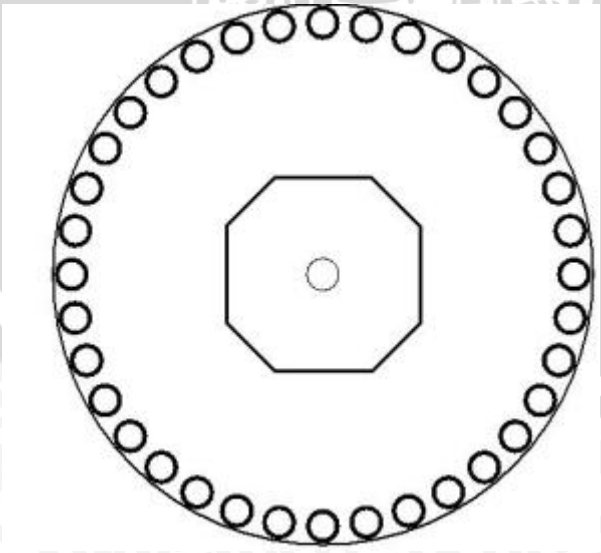
- *Serial*: 0 (RX) dan 1 (TX); *Serial* 1: 19 (RX) dan 18 (TX); *Serial* 2: 17 (RX) dan 16 (TX); *Serial* 3: 15 (RX) dan 14 (TX). Digunakan untuk menerima (RX) dan mengirimkan (TX) TTL data serial. Pin ini dihubungkan ke pin yang berkaitan dengan chip Serial ATmega8U2 USB-to-TTL.
- *Eksternal interrupts*: 2 (*interrupt* 0), 3 (*interrupt* 1), 18 (*interrupt* 5), 19 (*interrupt* 4), 20 (*interrupt* 3), dan 2 (*interrupt* 2). Pin ini dapat dikonfigurasi untuk memicu *interrupt* pada nilai yang rendah, dengan batasan tepi naik atau turun, atau perubahan nilai.
- PWM: 0 - 13. Menyediakan output PWM 8-bit dengan fungsi *analogWrite ()*.
- *SPI*: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). Pin ini mendukung komunikasi SPI menggunakan *SPI library*.
- *LED*: 13. Ada *built-in* LED terhubung ke pin digital 13. Ketika pin bernilai nilai *high* LED menyala dan ketika pin bernilai *low* LED mati.
- I^2C: 20 (SDA) dan 21 (SCL). Dukungan I^2C (TWI) komunikasi menggunakan *wire*.

Sensor Rotary Encoder DRN136





Rangkaian Sensor Optocoupler DRN136



Piringan Derajat DRN136

Karakteristik

- Dua bagian utama:
  1. Rangkaian Sensor Optocoupler yang menggunakan sensor optocoupler tipe celah (*slot*) sebagai sensor pembaca perubahan posisi lubang Piringan-Derajat.
  2. Piringan-derajat dengan 36 lubang pada kelilingnya dengan sudut antar lubang yang berdampingan terhadap titik tengahnya adalah 10°.

- Dimensi:
  - Rangkaian Sensor: 13,91mm(X) x 32,41(Y) x 1,9mm(Z)
  - Piringan-derajat: 42,64mm(Ø) x 1,9mm(Z)

- Tegangan-tegangan operasi:
  - Sumber (VCC): 3,5 – 5,5V
  - Logika output '0': 0 – 0,5V
  - Logika output '1': 3 – 5V (VCC – 0,5V)

- Logika output:
  - 0: Saat celah sensor terhalang
  - 1: Saat celah sensor tanpa-halangan

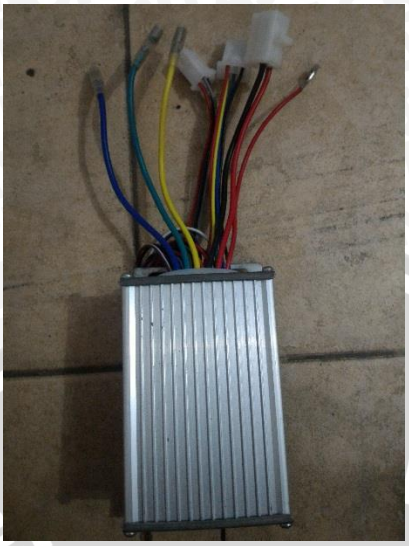
- Kecepatan baca sensor:
  - Kondisi logika *toggle* (0/1): 1500Hz
  - Rotasi dengan 36 lubang: 2500RPM

- Keterangan Fungsi Pin Rangkaian Sensor:

Tabel 1. Fungsi Pin Rangkaian Sensor DRN136.

GND	Sumber tegangan bawah / negatif / *ground*
VCC	Sumber tegangan atas / positif.
V_{out}	Data keluaran rangkaian sensor

Inverter Tiga Fasa



SPESIFIKASI:

Power = 350W – 400W
Tegangan = 36V - 48V Auto Voltage
Under Voltage = 31,6V untuk 36V. 41V untuk 48V
Jumlah FET = 6 MOSFET
Amper = 17A
Input trotle = 1,2V – 4,2V
Derajat motor = Auto detect 60 dan 120 derajat.

Fitur =

- Switch maju-mundur
- Brake signal
- Pedal Assist Sensor untuk sepeda listrik
- signal speedometer
- Cruise controll
- Auto self study, (normal foward dan normal backward)

KETERANGAN SOCKET:

1. Socket isi 3 besar. Socket power
   kabel merah besar terhubung ke + baterai
   kabel hitam besar terhubung ke negatif
   kabel orange kecil ke kunci kontak lalu ke + Baterai

2. Kabel 3 phase Motor: kuning, hijau, biru, (kabel besar skun lonjong)
   ketiga kabel masing-masing terhubung ke motor.
   *ketiga kabel ini tidak boleh terjadi konslet/Short.

3. kabel hall: (socket 6 isi 5)
   merah (+5V), hitam (-), kuning, biru, hijau
   dihubungkan dengan 5 kabel hall yang dari motor

4. throtle/ handle gas: Socket isi 3
   merah (+5V), hitam(-), hijau (signal data)
   *pemasangan yang terbalik + dan - menyebabkan kontroller mematikan diri.

Socket yang lain hanyalah fitur tambahan, tidak digunakan tidak masalah:

- Ø kabel putih sepasang = kalibrasi arah putaran
- Ø socket isi 2 hitam dan biru = cruise controll
- Ø socket isi 2 hitam dan putih = switch rem
- Ø socket isi 2 hitam dan abu-abu = switch maju-mundur
- Ø coklat tunggal = signal speedometer
- Ø 3 kabel pin kecil = Pedal Assist Sensor