PENGGUNAAN TEKNIK MODEL *REFERENCE ADAPTIVE CONTROL* PADA TUNING PENGONTROL PI PADA SISTEM KONTROL KECEPATAN SEPEDA

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI TEKNIK KONTROL

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

MOCHAMMAD HESA IBRAHIM
NIM. 125060307111013

UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2016

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, 28 Juli 2016

Mahasiswa,

MOCHAMMAD HESA IBRAHIM

NIM. 125060307111013

RINGKASAN

Mochammad Hesa Ibrahim, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Juli 2016, *Penggunaan Teknik Model Reference Adaptive Control Pada Tuning Pengontrol PI Pada Sistem Kontrol Kecepatan Sepeda* , Dosen Pembimbing: Dr. Ir. Erni Yudaningtyas, M.T., dan Rahmadwati, S.T., M.T., Ph.D.

Saat ini seperti diketahui bahwa perkembangan transportasi berkembang sangat pesat, diantaranya yaitu sepeda listrik. Sepeda listrik merupakan salah satu transportasi yang tidak menggunakan bahan bakar fosil sebagai tenaga penggeraknya dan ramah lingkungan. Sepeda listrik yang berkembang saat ini menggunakan motor *Brushless Direct Current* (BLDC) sebagai penggerak utamanya. Motor BLDC ini mempunyai kelebihan yaitu memiliki torsi awal yang tinggi, kecepatan tinggi, respon yang cepat, efisiensi perawatan yang rendah, tidak menimbulkan kebisingan, dan juga memungkinkan untuk mendapatkan berbagai kontroler kecepatan.

Pada penelitian ini kontrol kecepatan motor BLDC dengan menggunakan kontroler *Proportional Integral* (PI) dapat menghilangkan *error steady state* pada respon motor, namun berdampak pada kecepatan respon yang lambat dalam mencapai nilai *steady state*. Diantara teknik perancangan kontrol adaptif, yaitu *Model Reference Adaptive Control* (MRAC) yang memiliki ide dasar untuk membuat respon sistem yang dikontrol agar dapat menyerupai perilaku yang sama dengan model referensi. Oleh karena itu, teknik MRAC dapat digunakan sebagai *tuning* kontroler PI pada kontrol kecepatan untuk meningkatkan kecepatan respon dalam mencapai keadaan *steady state*. Respon sistem hasil implementasi dengan *setpoint* 100 rpm, 120 rpm dan 140 rpm memiliki nilai *error steady state* rata-rata berada dibawah toleransi 2%, masing-masing adalah 1%, 0,83%, dan 0,71%. Sedangkan *settling time* masing-masing adalah 6 detik, 5,4 detik dan 6 detik. Ketika sistem diberi gangguan pada *setpoint* 100 rpm, 120 rpm dan 140 rpm, respon akan mengalami perlambatan dan *recovery time* respon kembali pada keadaan *steady state* masing-masing adalah 3,1 detik, 3,8 detik dan 3,1 detik. Ketika gangguan yang diberikan dilepaskan, respon akan mengalami percepatan dan *recovery time* respon kembali pada keadaan *steady state* masing-masing adalah 8,5 detik, 6,8 detik dan 5,3 detik. Ketika sistem diberi gangguan dikendarai pada *setpoint* 100 memiliki nilai *error steady state* sebesar 1,8% dan *settling time* 7,7 detik.

Kata Kunci: Sepeda Listrik, Motor *Brushless Direct Current* (BLDC), Kontrol Kecepatan, Kontroler PI, *Model Reference Adaptive Control*.

SUMMARY

Mochammad Hesa Ibrahim, Department of Electrical Engineering, Faculty of Engineering University of Brawijaya, July 2016, *The use of the technique of Model Reference Adaptive Control On Tuning PI controller on the bike Speed Control System*, Academic Supervisor: Dr. Ir. Erni Yudaningtyas, M.T., and Rahmadwati, S.T., M.T., Ph.D.

This time as it is known that the development of the transportation was growing rapidly such as electric bikes. Electric bike is one of the transportation that does not use of fossil fuels as energy moving centers are and environmentally friendly. Electric bike developed today using motor Brushless Direct Current (BLDC) as the main drive. This BLDC Motor have advantages which have high initial torque, high-speed, Quick responsiveness, efficiency low maintenance, not generate noise and also allows for various speed controller.

This research on BLDC motor speed control using Integral Proportional controller (PI) can eliminate errors steady state on the motor response, but have an impact on the speed of the slow response in value reached steady state. Among adaptive control design technique, namely Model Reference Adaptive Control (MRAC) who have basic ideas to create a controlled system response in order to resemble the same behavior with reference models. Therefore, MRAC technique can be used as tuning PI controller on the speed control to increase the speed of the response to achieve the state of steady state. System response implementation results with setpoint 100 rpm, 120 rpm and 140 rpm have error value steady state the average under the tolerance 2%, each of which is 1%, 0,83%, and 0.71%. While the settling time of each is 6 seconds, 5.4 seconds and 6 seconds. When the system given the disturbance on setpoint 100 rpm, 120 rpm and 140 rpm, The response will be experiencing the deceleration and recovery time the response back to the state of the steady state of each is 3.1 seconds, 3.8 seconds and 3.1 seconds. When disorders that given released, The response will be accelerated and recovery time the response back to the state of the steady state of each is 8.5 seconds, 6.8 seconds and 5.3 seconds. When the system is given at zero setpoint disorders 100 has a value error steady state of 1.8 percent and settling time 7.7 seconds.

Keywords: Electric bicycle, Motor Brushless Direct Current (BLDC), Speed Control, PI Controller, Model Reference Adaptive Control.

PENGANTAR

Bismillahirrohmanirrohim. Alhamdulillah, puji syukur penulis panjatkan kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi yang berjudul "Penggunaan Teknik *Model Reference Adaptive Control* Pada Tuning Pengontrol PI Pada Sistem Kontrol Kecepatan Sepeda" 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 Sidik Harijadi dan Isa Mardiana yang selalu memberikan kasih sayang dan doanya yang tiada akhir. Adik tercinta Sari 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 Dr. Ir. Erni Yudaningtyas, M.T. sebagai dosen pembimbing yang telah banyak memberikan kesempatan, nasehat, pengarahan, motivasi, saran dan masukan yang telah diberikan.
- 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.
- Riska Yulia Windari, terima kasih atas waktu, pengertian, semangat, bantuan dan kesabarannya yang telah diberikan.
- Pranata Laboratorium, Pak Dedy dan Keluarga besar asisten Laboratorium Dasar Elektrik dan Pengukuran, Ferdian, Gadis, Faizal, Fajar, Zakiyah, Bobby, Milky,

Najar, Arif, Citra, Fitrah, Ina, Nola, Dini, Topan, Bima, Rahmat, Taka, Okto, terima kasih telah memberikan banyak bantuan, dukungan dan canda tawa.

- Sahabat-sahabat, Suro, Nora, Zakiya, Faizal, Hanif, Gabriel, Tyo, dan Odi terima kasih telah memberikan banyak bantuan, dukungan dan canda tawa.
- Teman-teman SMA, Asa, Faisal, Teka, Tanti, Eza, Devi, dan Nanang yang telah memberikan semangat dan dukungan.
- 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, 26 Juli 2016

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

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 sepeda listrik. Sepeda listrik merupakan teknologi terbaru pada kendaraan roda dua yang memanfaatkan sumber listrik sebagai bahan bakarnya dan motor listrik sebagai penggerak. Beberapa negara seperti Cina, Jepang dan Belanda sepeda menjadi alat transportasi alternatif untuk bekerja (Hamdi, I.T., 2015).

Penggunaan sepeda listrik adalah salah satu cara untuk menekan jumlah konsumsi pemakaian bahan bakar minyak dan menekan polusi udara. Sepeda listrik tentunya membutuhkan penggerak berupa motor listrik yang sesuai dengan kriteria sepeda listrik yang diinginkan.

Motor *Brushless Direct Current* (BLDC) merupakan penggerak yang banyak digunakan pada sepeda listrik. Agar motor BLDC dapat bekerja, diperlukan adanya medan putar magnet stator. Medan magnet putar stator memerlukan sumber tegangan *Alternative Current* (AC) tiga fasa pada stator motor. Oleh karena itu digunakan inverter 3 fasa untuk mengubah tegangan *Direct Current* (DC) menjadi tegangan *Alternative Current* (AC) 3 fasa. Motor BLDC memiliki efisiensi yang tinggi, respon yang cepat, operasi tidak bersuara, dan rentang kecepatan yang lebih tinggi. Motor BLDC memiliki karakteristik-karakteristik variabel dan telah digunakan secara luas dalam kontrol kecepatan. Motor BLDC dapat menyediakan sebuah torsi awal yang tinggi dan juga memungkinkan untuk mendapatkan berbagai kecepatan (Irham, 2015). Kontrol kecepatan motor BLDC ini dilakukan agar menjaga unjuk kerja motor BLDC yang tinggi. Kontrol ini bertujuan agar *overshoot* dan *error steady state (offset)* sekecil mungkin pada kondisi perubahan setpoint dan berbeban (Kristiyono R., 2014).

Kontrol kecepatan motor BLDC dengan menggunakan kontroler PI dapat menghilangkan *error steady state* pada respon motor (Firdaus M., 2011), namun berdampak pada kecepatan respon *settling time* yang lambat dalam mencapai nilai *steady*

state (Shyam A., 2013). Sehingga diharapkan dengan penambahan model referensi, respon yang lambat tersebut dapat dihilangkan.

Dalam kehidupan sehari-hari, adaptasi adalah mengubah perilaku untuk menyesuaikan dengan lingkungan baru. Secara intuisi, sebuah kontroler adaptif adalah sebuah kontroler yang mampu memodifikasi perilaku responnya untuk mengubah dinamika proses dan karakter gangguan (*disturbance*) (Astrom, 1995).

Berbagai macam teknik perancangan sistem yang dapat diterapkan untuk mendesain suatu sistem, antara lain teknik perancangan dengan kontrol adaptif. Hal ini terutama untuk melakukan hal-hal sebagai berikut: mengeliminasi gangguan dari luar (*disturbance*), mengeliminasi gangguan dari dalam (perubahan parameter sistem atau pada sistem yang bekerja di luar daerah linier), mengatasi keterbatasan perancangan klasik yang umumnya sukar direalisasikan (Astrom, 1995).

Terdapat banyak teknik perancangan sistem kontrol dengan kontrol adaptif, beberapa diataranya adalah *Model Reference Adaptive Control* (MRAC). MRAC atau juga diketahui sebagai *Model Reference Adaptive System* (MRAS) memiliki ide dasar untuk membuat keluaran sistem yang dikontrol agar dapat menyerupai perilaku yang sama dengan model referensi yang diberikan (Ali dkk, 2012).

Pada skripsi ini akan membahas pengembangan alat dan perancangan sistem kontrol dengan menggunakan teknik MRAC pada *tuning* kontroler PI sebagai kontrol kecepatan motor BLDC. Dalam penggunaan kontrol adaptif pada skripsi ini, diharapakan sistem dapat memiliki tingkah laku yang sama dengan model yang diberikan.

1.2 Rumusan Masalah

Mengacu pada permasalahan yang telah diuraikan pada latar belakang, maka rumusan masalah dapat ditekankan pada point berikut:

1. Bagaimana perancangan algoritma sistem kontrol kecepatan pada sepeda listrik dengan menggunakan penggerak motor *Brushless* DC (BLDC) sebagai peggunaan teknik *Model Reference Adaptive Control* (MRAC) pada *tuning* pengendali PI dan menentukan fungsi alih sistem.
2. Bagaimana respon sistem jika diberikan perubahan nilai *setpoint* dan gangguan.

1.3 Batasan Masalah

Karena luasnya objek kajian, maka perlu dilakukan pembatasan masalah agar pembahasan lebih terfokus pada rumusan masalah. Adapun batasan masalah pada skripsi ini antara lain:

1. Motor yang digunakan adalah motor *Brushless* DC 350 Watt dengan catu 36 V, arus 10 A, dan torsi ±18Nm.
2. Mikrokontroler yang digunakan adalah Arduino Mega 2560.
3. Pembahasan ditekankan pada penggunaan teknik *Model Reference Adaptive Control* (MRAC) pada kontroler PI.
4. Perhitungan koefisien gesek dalam gangguan sistem diabaikan.

1.4 Tujuan

Tujuan skripsi ini adalah terwujudnya *tuning* kontroler PI dengan teknik *Model Reference Adaptive Control* (MRAC) pada sistem kontrol kecepatan sepeda.

1. Mengetahui fungsi alih sistem sehingga dapat dirancang algoritma sistem kontrol kecepatan pada sepeda listrik dengan menggunakan penggerak motor *Brushless* DC (BLDC) sebagai peggunaan teknik *Model Reference Adaptive Control* (MRAC) pada *tuning* pengendali PI
2. Mengetahui performansi respon sistem ketika diberikan perubahan nilai setpoint dan diberi gangguan.

1.5 Manfaat

Manfaat dari penelitian ini adalah dapat mengontrol kecepatan yang diinginkan pada sepeda listrik dengan penggerak motor listrik (motor *Brushless Direct Current* (BLDC)).

BAB II
TINJAUAN PUSTAKA

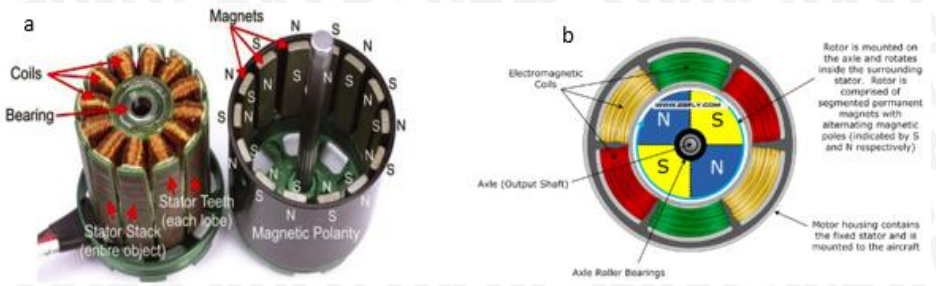
Bab ini menguraikan teori-teori pendukung skripsi, yang terdiri atas :

2.1 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 (Dharmawan A.,2009).

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 pembacaan kecepatan putar motor. 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 atau 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 atau 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 2.1



Gambar 2.1 Bentuk rotor dan stator motor BLDC *(a)outrunner (b)inrunner.*

Sumber: Hamdi I.T.,2015

2.1.1 Cara Kerja Motor *Brushless* DC (BLDC)

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

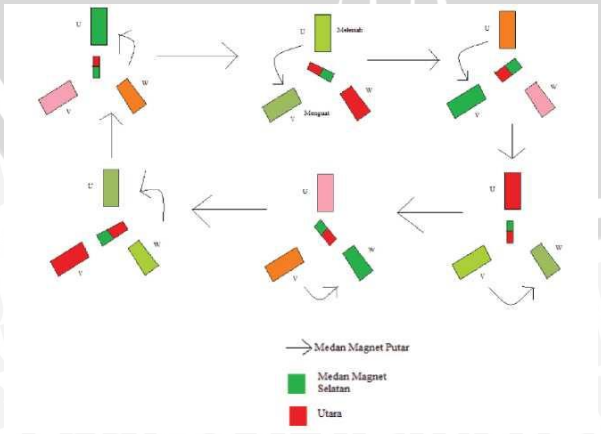
$$B = \frac{\mu i N}{2l} \qquad 2-1$$

Di mana N merupakan jumlah lilitan, i merupakan arus, l merupakan panjang lilitan dan μ merupakan permeabilitas bahan.

Karena arus yang diberikan berupa arus AC 3 phasa 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.

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

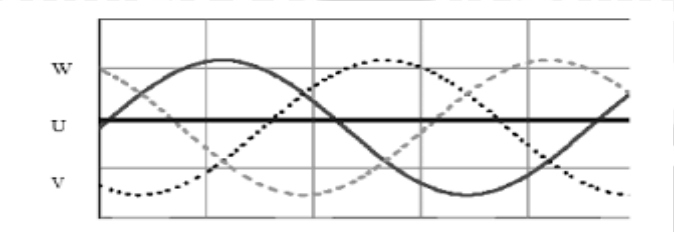
Dengan :
Ns = putaran stator
f = frekuensi
p = jumlah kutub



Gambar 2.2 Medan putar magnet stator

Sumber: Abe dharmawan,2009-6

Berdasarkan Gambar 2.2, 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. Medan putar stator dan rotor ditunjukkan dalam Gambar 2.3.



Gambar 2.3 Analisa Medan Putar stator dan rotor.

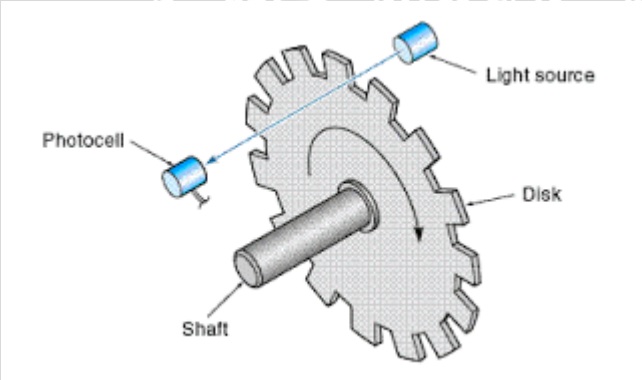
Sumber: Abe dharmawan,2009-6

Berdasarkam Gambar 2.3, 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.

2.2 Sensor Kecepatan (*Rotary Encoder*)

Rotary encoder adalah perangkat elektromekanik yang dapat memonitor gerakan dan posisi. Rotary encoder umumnya menggunakan sensor optik untuk menghasilkan serial pulsa yang dapat diartikan menjadi gerakan, posisi, dan arah. Sehingga posisi sudut suatu poros benda berputar dapat diolah menjadi informasi berupa kode digital oleh *rotary encoder*.

Rotary encoder tersusun dari suatu piringan tipis yang memiliki lubang-lubang pada bagian lingkaran piringan. LED ditempatkan pada salah satu sisi piringan sehingga cahaya akan menuju ke piringan. Di sisi yang lain suatu *photo-transistor* diletakkan sehingga *photo-transistor* ini dapat mendeteksi cahaya dari LED yang berseberangan. Piringan tipis tadi dikopel dengan poros motor, atau alat berputar lainnya yang ingin kita ketahui posisinya, sehingga ketika motor berputar piringan juga akan ikut berputar. Apabila posisi piringan mengakibatkan cahaya dari LED dapat mencapai photo-transistor melalui lubang-lubang yang ada, maka photo-transistor akan mengalami saturasi dan akan menghasilkan suatu pulsa gelombang persegi. Gambar 2.4 menunjukkan desain sederhana dari rotary encoder. Semakin banyak deretan pulsa yang dihasilkan pada satu putaran akan menentukan akurasi *rotary encoder* tersebut.



Gambar 2.4 Desain umum rotary encoder

Sumber : **Lukman H. (2012)**

2.3 Mikrokontroler Arduino Mega 2560

Mikrokontroller adalah sebuah chip yang berfungsi sebagai pengontrol rangkaian elektronik dan umumnya dapat menyimpan program di dalamnya. Dengan kata lain, mikrokontroller adalah suatu alat elektronika digital yang mempunyai *input* dan *output* serta kontrol dengan program yang bisa ditulis dan dihapus. Cara kerja mikrokontroller

sebenarnya membaca dan menulis data. Mikrokontroller digunakan dalam produk dan alat yang dikendalikan secara otomatis.

Arduino Mega 2560 (Gambar 2.5) adalah papan mikrokontroler berdasarkan ATmega328 (lihat Gambar 2.5). Board ini memiliki 54 pin digital *input/output* (14 pin dapat digunakan sebagai *output* PWM), 16 *input* analog, 16 MHz osilator kristal, USB koneksi, jack listrik, header ICSP, dan tombol reset.

Gambar 2.5 Arduino Mega 2560
Sumber : Arduino.cc (2016)

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 V sampai 20 V. Namun jika menggunakan lebih dari 12 V, regulator tegangan bisa panas dan merusak papan. Kisaran yang disarankan adalah 7 V sampai 12 V.

2.4 *Pulse Width Modulation* (PWM)

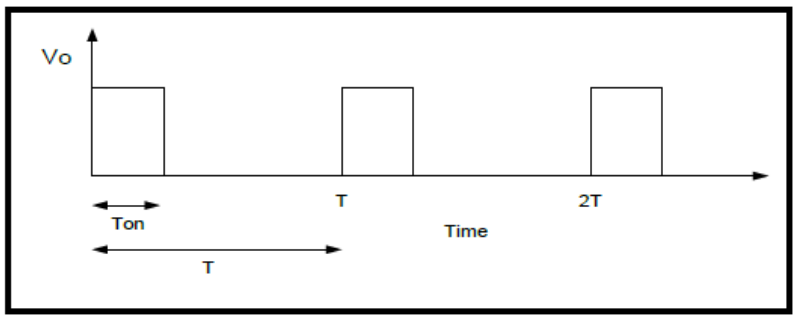
Pulse Width Modulation (PWM) adalah salah satu jenis modulasi. Modulasi PWM dilakukan dengan cara mengubah lebar pulsa dari suatu pulsa data. Total 1 periode (T) pulsa dalam PWM adalah tetap, dan data PWM pada umumnya menggunakan perbandingan pulsa positif terhadap total pulsa. Pada sinyal PWM, frekuensi sinyal konstan sedangkan *duty cycle* bervariasi dari 0%-100%. Dengan mengatur *duty cycle* akan diperoleh *output* yang diinginkan. *Duty cycle* adalah besarnya sinyal kontrol yang diberikan pada plant.

Duty cycle = T_{on}/T x 100% (2-3)

dengan:

T_{on} = periode logika tinggi
T = periode

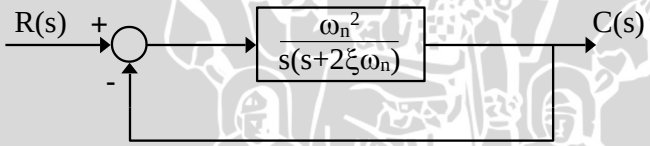
Sinyal PWM secara umum ditunjukkan dalam Gambar 2.6, dimana V_o adalah amplitudo.



Gambar 2. 6 Sinyal PWM
Sumber: Ardyani, F. (2013)

2.5 Output Sistem Orde Dua

Diagram blok sistem orde dua (lihat Gambar 2.7), dengan fungsi alihnya adalah sebagai berikut:



Gambar 2.7 Sistem orde dua

$$\frac{C_{(s)}}{R_{(s)}} = \frac{\frac{{\omega_n}^2}{s(s+2\xi\omega_n)}}{1+\frac{{\omega_n}^2}{s(s+2\xi\omega_n)}}$$

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

$$= \frac{{\omega_n}^2}{s^2+2\xi\omega_n s+{\omega_n}^2} \qquad 2-4$$

Akar-akar penyebut fungsi alih atau persamaan karakteristik adalah

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

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

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

$$s_{12} = -\xi\omega_n \pm j\omega_d \qquad 2-5$$

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 2.8, 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 $(\xi>1)$.

2.5.1 Keadaan Kurang Teredam / Underdamped $(0<\xi<1)$

$$\frac{C_{(s)}}{R_{(s)}} = \frac{{\omega_n}^2}{s^2 + 2\xi\omega_n s + {\omega_n}^2} \qquad 2-6$$

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

$$C_{(s)} = \frac{{\omega_n}^2}{(s^2 + 2\xi\omega_n s + {\omega_n}^2)s} \qquad 2-7$$

Dari Tabel Transformasi Laplace didapatkan

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

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

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

$$c(t) = 1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}} \sin\left[\omega_d t + \arctan\frac{\sqrt{1-\xi^2}}{\xi}\right] \quad t \geq 0 \qquad 2-8$$

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

$$C_{(s)} = \frac{1}{s} - \frac{s + 2\xi\omega_n}{(s^2 + 2\xi\omega_n s + \omega_n{}^2)} \qquad 2-9$$

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

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

$$\mathcal{L}^{-1}\left[\frac{\xi\omega_n}{(s + \xi\omega_n)^2 + \omega_d{}^2}\right] = e^{-\xi\omega_n t}\sin\omega_d t \qquad 2-10$$

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

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

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

$$c(t) = 1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}}\sin\left[\omega_d t + arctan\frac{\sqrt{1-\xi^2}}{\xi}\right] \quad t \geq 0 \qquad 2-11$$

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

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

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

2.5.2 Teredam Kritis / Critically Damped ($\xi = 1$)

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:

$$C_{(s)} = \frac{\omega_n{}^2}{(s + \omega_n)^2 s} \qquad 2-12$$

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

2.5.3 Terlalu Teredam / Overdamped ($\xi > 1$)

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 :

$$C_{(s)} = \frac{{\omega_n}^2}{(s + \xi\omega_n + \omega_n\sqrt{\xi^2 - 1})(s + \xi\omega_n - \omega_n\sqrt{\xi^2 - 1})s} \qquad 2 - 13$$

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

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

$$s_2 = \left(\xi - \sqrt{\xi^2 - 1}\right)\omega_n \qquad 2 - 14$$

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

2.6 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 atau underdamped.

Tanggapan peralihan sistem kontrol terhadap masukan unit step umumnya dikelompokkan sebagai berikut (lihat Gambar 2.8):

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

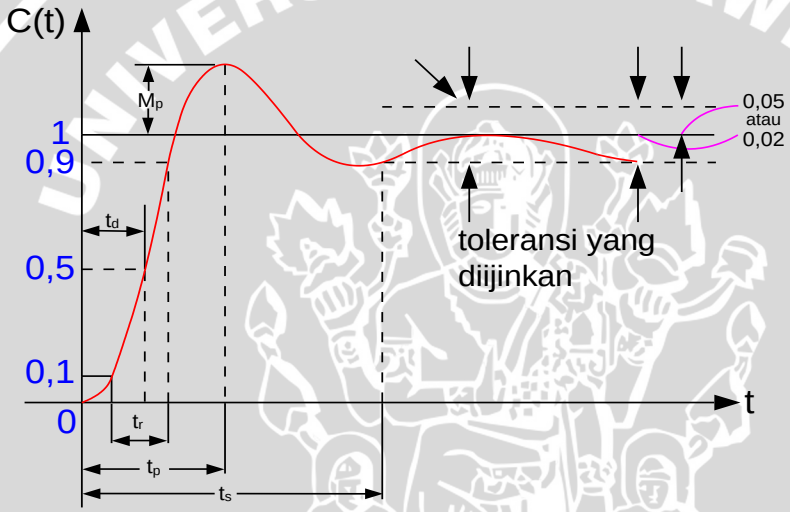
$$Mp = \frac{c(t_p) - c(\infty)}{c(\infty)} \times 100\%$$

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

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

5) *Settling Time* / Waktu Turun, t_s

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



Gambar 2.8 Output Unit Step Sistem Orde Dua

2.7 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., 2010). Fungsi kontroller pada umumnya adalah sebagai berikut:

1) Membandingkan nilai *input* dan *output* sistem secara keseluruhan (*plant*).
2) Menentukan penyimpangan (*error*).
3) Menghasilkan sinyal kontrol (mengurangi penyimpangan (*error*) menjadi nilai nol atau nilai yang kecil).

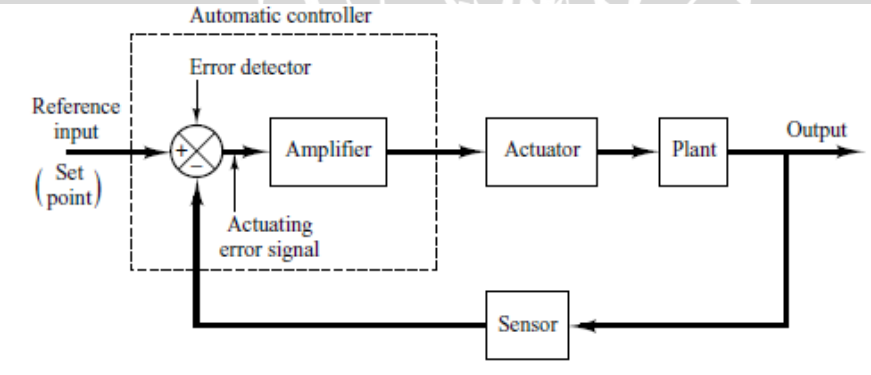
Adapun tujuan kontrol secara khusus adalah sebagai berikut:

1) Meminimalkan *error steady state*.

2) Meminimalkan *settling time*.

3) Mencapai spesifikasi transien yang lain, misalnya meminimalkan *maximum overshoot.*

Sistem loop tertutup (Gambar 2.9) menggunakan sinyal *output* yang diumpanbalikkan terhadap *automatic controller* (kontroller otomatis) (Ogata, K., 2010), yang akan membuat pengubahan terhadap sistem agar *output* sistem seperti yang diinginkan atau sesuai *set point*. Sensor/transduser digunakan sebagai elemen yang langsung mengadakan kontak dengan objek yang diukur. Transduser berfungsi untuk mengubah besaran fisis yang diukur menjadi besaran fisis lainnya, seperti mengubah besaran tekanan, temperatur, aliran, posisi menjadi besaran listrik. *Actuating error signal* merupakan sinyal kesalahan (error) yang merupakan selisih antara sinyal set point dan sinyal output. *Actuator* (aktuator) berfungsi untuk mengontrol aliran energi ke sistem yang dikontrol. Sebagai contoh adalah motor listrik, katub pengontrol, pompa dan sebagainya. *Amplifier* merupakan unit yang dibutuhkan karena daya dari *error detector* tidak cukup kuat untuk menggerakkan elemen *output.* Karena fungsi pengontrolan adalah untuk mengendalikan *output* agar kesalahan (*error*) mendekati nol, maka diperlukan penguat daya (*power amplifier*).



Gambar 2.9 Diagram Blok Sistem dengan Kontroller Otomatis
Sumber: Ogata, K. (2010)

Cara bagaimana kontroller otomatis menghasilkan sinyal kontrol disebut dengan aksi kontrol. Aksi kontrol dasar yang sering digunakan dalam kontroller adalah

1. Kontroler proporsional (P)
2. Kontroler integral (I)
3. Kontroler proporsional integral (PI)

2.7.1 Kontroler Proporsional

Kontroler proporsional adalah sebuah kontroller yang memiliki karakteristik mempercepat *output*. Hubungan antara *output* kontroller u(t) dan sinyal *error* e(t) ditunjukkan dalam persamaan berikut:

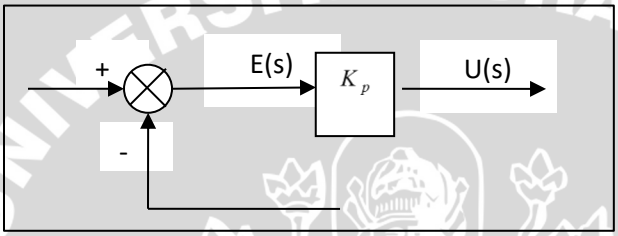
$$u(t) = Kp\, e(t) \qquad (2-15)$$

atau, dalam fungsi alih

$$\frac{U(s)}{E(s)} = Kp$$

dimana K_p adalah penguatan.

Diagram blok kontroler proporsional (P) ditunjukkan dalam Gambar 2.10.



Gambar 2.10 Diagram blok kontroler proposional (P)
Sumber: Ogata, K. (2010)

Apapun wujud mekanisme yang sebenarnya dan apapun bentuk daya penggeraknya, kontroller proporsional pada dasarnya merupakan penguat dengan penguatan yang dapat diatur (Ogata K., 2010).

2.7.2 Kontroler Integral

Kontroler integral (I) memiliki kemampuan untuk mengurangi *offset* yang diakibatkan oleh kontroller proporsional. *Output* kontroller u(t) diubah dengan laju yang sebanding dengan *error* e(t). Persamaan kontroler integral (I) ditunjukkan dalam persamaan berikut (Ogata K., 2010).

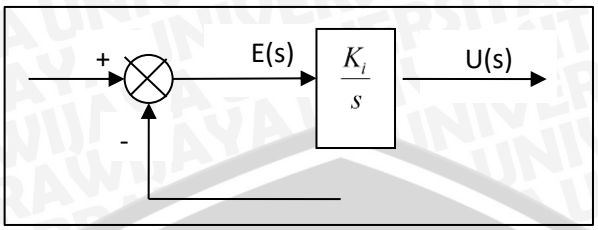
$$\frac{du(t)}{dt} = Ki\, e(t)$$

$$u(t) = Ki \int_0^t e(t)\, dt$$

$$\frac{U(s)}{E(s)} = \frac{K_i}{s} \qquad 2-16$$

yang merupakan fungsi alih kontroller integral (I), dengan K_i adalah konstanta integral yang dapat diubah nilainya. Jika e(t) bernilai nol, maka nilai u(t) tetap konstan. Aksi

kontrol integral biasa disebut dengan kontrol reset. Gambar 2.11 menunjukkan diagram blok kontroller integral (I).



Gambar 2.11 Diagram blok kontroler integral (I)
Sumber: Ogata, K. (2010)

2.7.3 Kontroler Proporsional Integral (PI)

Kontroler proporsional integral (PI) memiliki kemampuan untuk mempercepat *output* dan mengurangi *offset*. Persamaan kontroler Proporsional Integral (PI) adalah

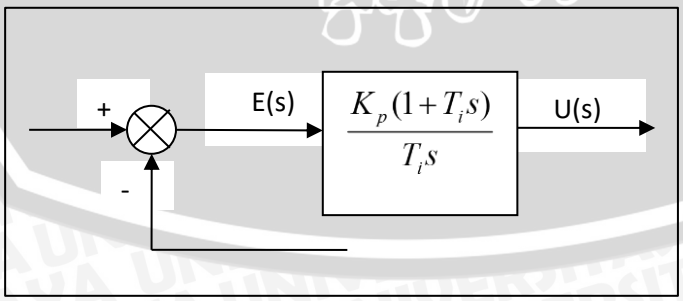
$$u(t) = Kp\, e(t) + \frac{K_p}{T_i}\int_0^t e\,(t)dt \qquad 2-17$$

Adapun fungsi alihnya adalah

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

$$= \frac{K_p(1 + T_i\, s)}{T_i\, s}$$

dengan K_p penguatan proporsional dan T_i disebut waktu integral, yang keduanya dapat ditentukan. Waktu integral mengatur aksi kontrol internal sedangkan perubahan nilai K_p berakibat pada pada bagian aksi kontrol proporsional maupun integral. Gambar 2.12 menunjukkan diagram blok kontroller proporsional integral (PI).

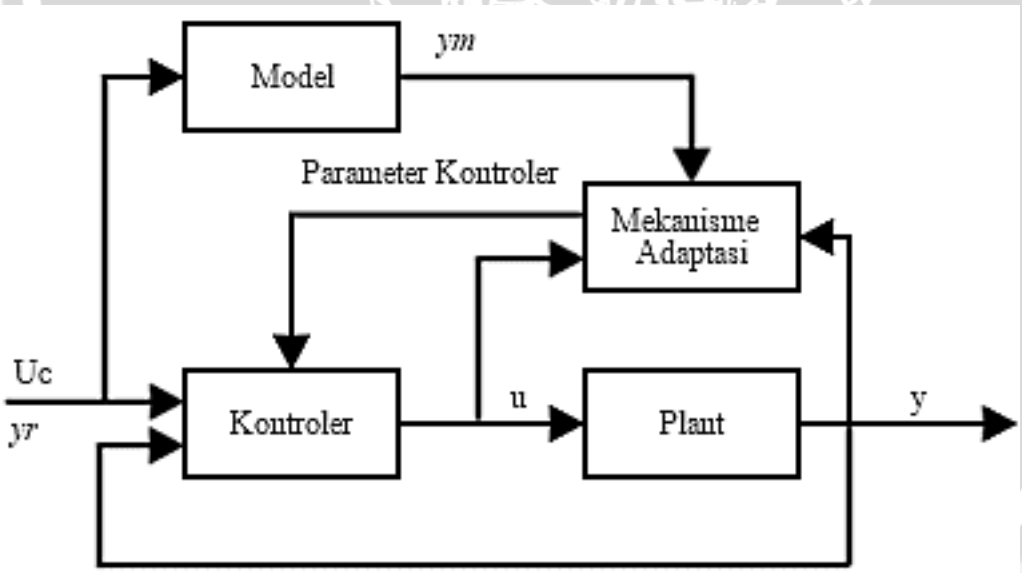


Gambar 2.12 Diagram blok kontroler proporsional integral (PI)
Sumber: Ogata, K. (2010)

2.8 Model Reference Adaptive Control (MRAC)

Dasar pemikiran munculnya *Model Reference Adaptive Control* (MRAC) adalah respon sistem yang sesuai dengan keinginan desain dibuat dalam bentuk model $M_{(s)}$. *Output* dari model ini dibandingkan dengan *output* sistem, selisih *output* model dengan *output* sistem digunakan untuk mengatur strategi kontrol.

Model referensi digunakan untuk menentukan spesifikasi-spesifikasi sistem yang diinginkan. Kontroler digunakan untuk menghasilkan sinyal kontrol u yang selanjutnya digunakan sebagai *input plant*. *Plant* menghasilkan sinyal *output* y dan model referensi menghasilkan sinyal *output* y_m. *Output plant* dibandingkan dengan *output* model referensi, jika terdapat *error* diantara keduanya maka parameter yang ada pada kontroler akan berubah.perubahan parameter kontroler dilakukan melalui hukum adaptasi. Di dalam hukum adaptasi, parameter kontroler ini akan diatur sedemikian rupa sehinga parameter kontroler dapat membentuk sinyal kontrol u yang akan menyebabkan *output plant* sesuai dengan model referensi. Apabila *plant* telah mengikuti referensi, maka nilai parameter kontroler sudah tetap seperti yang diharapkan. Gambar skema *Model Reference Adaptive Control* (MRAC) dalam gambar 2.13



Gambar 2.13 Skema *Model Reference Adaptive Control (MRAC)*

Sumber : Astrom, 1995

Model Reference Adaptive Control mempunyai dua *loop* yaitu *loop* dalam dan *loop* luar. *Loop* dalam terdiri atas *plant* dan kontroler. *Loop* luar bertujuan untuk menyesuaikan nilai parameter yang digunakan pada *loop* dalam. Setelah nilai parameter

kontroler sudah tepat, maka *loop* dalam berjalan seperti sistem pengaturan pada umumnya.

Model Reference Adaptive Control bersifat adaptif karena nilai-nilai parameter kontroler dapat ditentukan secara *on-line*. Nilai-nilai parameter ini akan diperbaiki berdasarkan *error* yang terjadi antara *output* model dengan *output* sistem yang diatur.

2.9 Model Referensi

Model referensi ditentukan melalui derajat relatif sistem. Derajat relatif model referensi sistem harus sama dengan derajat relatif *plant*. Derajat relatif adalah selisih antara derajat polinominal pole dengan derajat polinominal zero. Orde model referensi disesuaikan dengan orde *plant* (Butler H.,1992).

2.10 Aturan Massachusetts Institute of Technology(MIT)

Untuk mengetengahkan aturan MIT, dimisalkan sebuah sistem *loop* tertutup di mana kontrolernya mempunyai sebuah parameter θ yang dapat diatur. Respon *loop* tertutup yang diinginkan ditentukan dengan suatu model yang memiliki *output* y_m. Misalnya e sebagai *error* antara *output* sistem *loop* tertutup y dan *output* model y_m. Satu kemungkinan untuk mengatur parameter dengan meminimalisasi *loss function* (fungsi kerugian, $J(\theta)$).

$$J(\theta) = \frac{1}{2}e^2 \qquad (2-18)$$

Untuk mendapatkan nilai J kecil, merupakan hal yang beralasan untuk mengubah parameter dalam arah negatif gradien dari J, yaitu

$$\frac{d\theta}{dt} = -\gamma\frac{\partial J}{\partial \theta} = -\gamma e\frac{\partial e}{\partial \theta} \qquad (2-19)$$

Turunan parsial $\frac{\partial e}{\partial \theta}$ yang disebut dengan turunan sensitivitas sistem, menunjukkan bagaimana *error* dipengaruhi oleh parameter yang dapat diatur. Jika diasumsikan bahwa parameter berubah lebih lambat daripada variabel yang lain di dalam sistem, sehingga turunan $\frac{\partial e}{\partial \theta}$ dapat dievaluasi di bawah asumsi bahwa θ adalah konstan (Astrom,1995).

Dengan menggunakan aturan MIT, kontroler dalam Gambar 2.13 dapat dirumuskan sebagai berikut :

- Respon kontroler sebagai fungsi parameter kontroler yang berubah sesuai dengan perubahan waktu dikalikan dengan *error* atau selisih antara $y_r(t)$ dan $y(t)$:

$$u(t) = \theta[y_r(t) - y(t)] \qquad (2-20)$$

Dengan

$u(t)$: Respon kontroler.

θ : Parameter kontroler.

$y_r(t)$: *Setpoint*.

$y(t)$: Respon *plant*.

- Perubahan parameter kontroler terhadap waktu dinyatakan dalam model pendekatan gradien, yaitu:

$$\frac{d\theta}{dt} = -\gamma \frac{\partial}{\partial \theta}[y(t) - y_m(t)]^2 \qquad (2-21)$$

atau

$$\frac{d\theta}{dt} = -2\gamma[y(t) - y_m(t)] \frac{\partial}{\partial \theta}[y(t) - y_m(t)] \qquad (2-22)$$

Dengan γ : *gain* adaptasi

2.11 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* :

$$s = \frac{1 - z^{-1}}{T_s} \qquad (2-23)$$

- *Forward difference* :

$$s = \frac{1 - z^{-1}}{T_s z^{-1}} \qquad (2-24)$$

- *Bilinear transform* :

$$s = \frac{2(1 - z^{-1})}{T_s(1 + z^{-1})} \qquad (2 - 25)$$

Dimana T_s adalah periode sampling

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

Metode penelitian pada dasarnya merupakan cara ilmiah untuk mendapatkan informasi dengan tujuan dan manfaat tertentu. Dalam menyelesaikan rumusan masalah dan merealisasikan tujuan penelitian yang terdapat di bab pendahuluan maka diperlukan langkah-langkah untuk menyelesaikan masalah tersebut. Metode penelitian pada skripsi ini meliputi:

1. Perancangan blok diagram sistem
2. Spesifikasi desain
3. Karakterisasi setiap blok

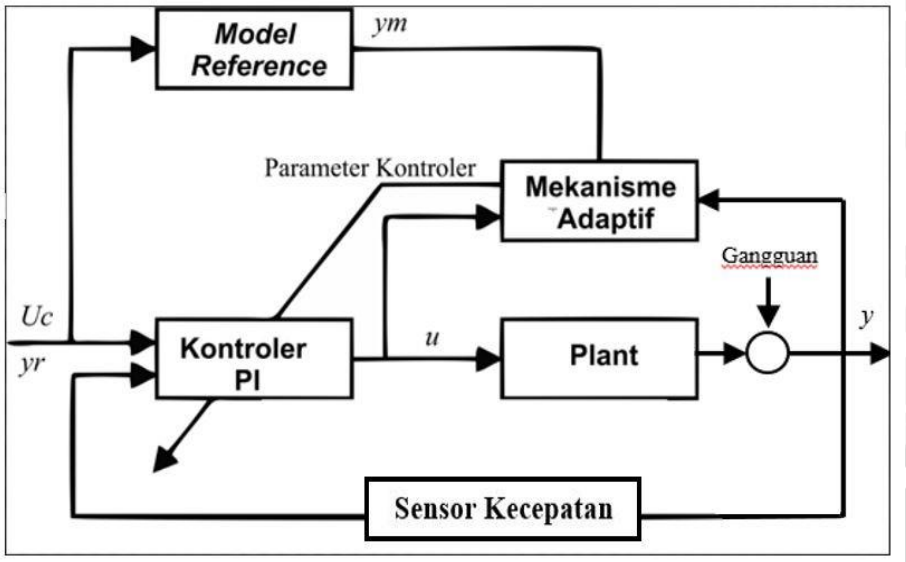
   Karakterisasi setiap blok dilakukan untuk mempermudah analisis sistem. Karakterisasi dibagi menjadi beberapa bagian, yaitu:

   1. Karakterisasi motor *Brushless Direct Current* (BLDC).
   2. Karakterisasi driver motor
   3. Karakterisasi sensor kecepatan (DI-Rotary Encoder).
4. Pembuatan perangkat keras
5. Perancangan algoritma.

   Perancangan algoritma meliputi algoritma teknik *Model Reference Adaptive Control* dan *flowchart* program.

3.1 Perancangan Blok Diagram Sistem

Pada perancangan alat diperlukan perancangan diagram blok sistem yang dapat menjelaskan sistem secara garis besar dan diharapkan alat dapat bekerja sesuai dengan desain yang diinginkan. Blok diagram sistem untuk pengontrolan kecepatan dapat dilihat dalam Gambar 3.1. *Input* sistem adalah kecepatan, kontroler yang digunakan adalah kontroler PI, *plant* adalah sepeda yang menggunakan penggerak motor Brushless Direct Current (BLDC), gangguan berupa rem pada roda sepeda, sensor yang digunakan adalah sensor *rotary encoder* dan *output* berupa kecepatan.



Gambar 3. 1 Blok Diagram Sistem MRAC

Keterangan :

Uc=yr	: *Set point* (nilai masukan sistem yang diinginkan pada keluaran sistem)
ym	: *respon* keluaran *model reference*
y	: *respon* keluaran motor *plant*
u	: *respon* keluaran kontroler PI
Model reference	: Persamaan matematis yang akan menjadi model referensi
Mekanisme Adaptif	: Algoritma kontrol adaptif yang akan menghasilkan parameter kontroler yang baru untuk kontroler PI
Kontroler PI	: Kontroler yang akan menghasilkan sinyal kontrol untuk *plant*
Plant	: Sepeda yang terpasang penggerak Motor BLDC (objek fisik yang akan dikontrol)
Gangguan	: Gangguan yang diberikan ke system (Beban rem)

3.2 Spesifikasi Desain

Desain yang diinginkan pada perancangan pengontrolan kecepatan sepeda listrik dengan penggerak 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. $0 < \xi < 1$

   $0 < \xi < 1$, agar respon melesat mencapai nilai *setpoint* kemudian turun dan berhenti pada nilai *setpoint.*
3. *Settling time* < 15 detik

   Settling time < 15 detik, karena diharapkan penggunaan teknik *Model Reference Adaptive Control* (MRAC) pada tuning kontrol PI mampu mempercepat *settling time* sistem kurang dari 15 detik.
4. *Overshoot* < 10%

   Sistem mempunyai *maximum overshoot* kurang dari 10%.

3.3 Karakterisasi Motor *Brushless* DC (BLDC)

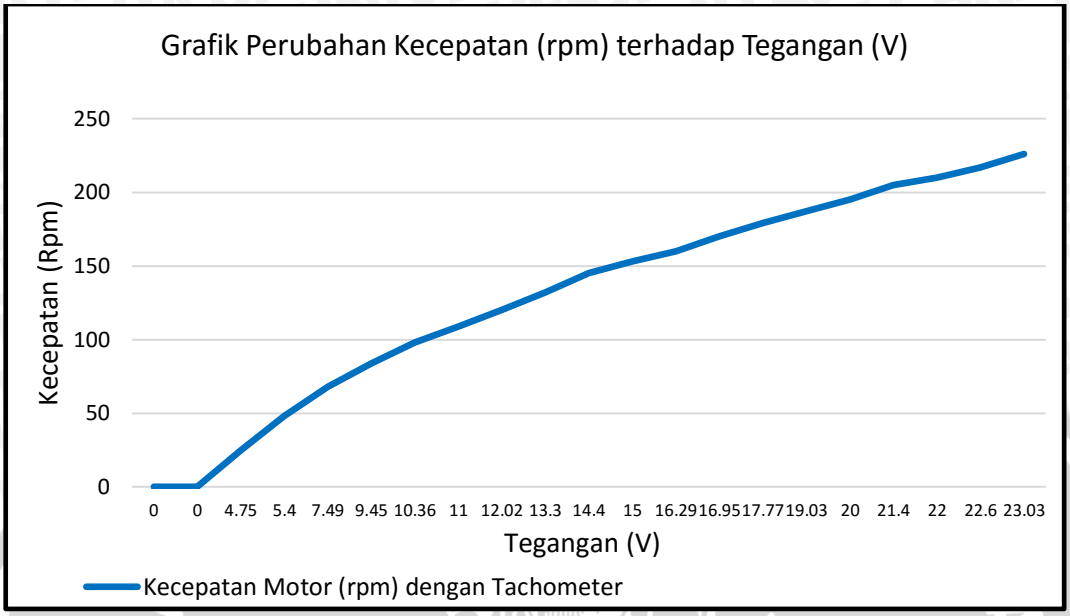
Untuk mengetahui karakteristik kecepatan motor BLDC pada setiap kenaikan tegangan *input* motor, dilakukan pengujian karakterisasi kecepatan motor BLDC. Pengujian dilakukan dengan cara membandingkan tegangan *input* motor terhadap kecepatan motor, menggunakan tachometer digital sebagai pembaca kecepatan motor, dan voltmeter yang dipasang pada *input* motor BLDC sebagai pembaca tegangan *input* motor. Langkah – langkah dalam pengujian karakterisasi motor BLDC yaitu: menghubungkan *output* tegangan baterai dengan *input driver* motor BLDC, lalu mengatur *duty cycle* dari 0% sampai dengan 100% dengan kenaikan 5% setiap pembacaan tegangan input dan kecepatan motor, kemudian hasil dari pengujian tersebut dicatat dalam tabel 3.1

Tabel 3.1 Data pengujian kecepatan motor BLDC (rpm) terhadap tegangan (V) dan *duty cycle* (%)

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

Berdasarkan Tabel 3.1 dapat diketahui bahwa pada *duty cycle* 0% dan 5%, motor BLDC tidak berputar dan tegangan yang terukur sama dengan 0 V. Daerah ini dapat disebut dengan daerah *dead time*. Dengan mengambil data tegangan *input* motor terhadap kecepatan motor

yang terukur, didapatkan kurva kecepatan motor (rpm) terhadap *input* tegangan (V) seperti ditunjukan pada Gambar 3.2



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

3.4 Karakterisasi *Driver Inverter* Tiga Fasa

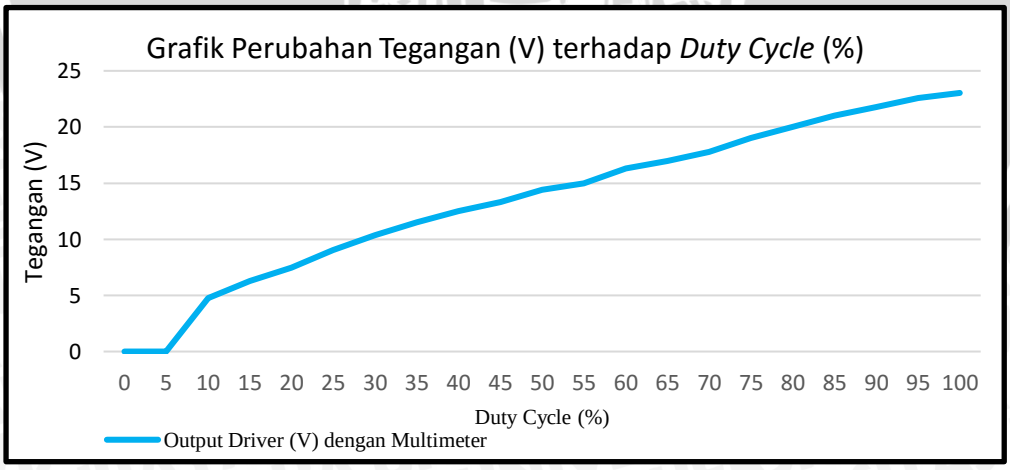
Untuk mengetahui karakteristik, kinerja, dan output rangkaian *driver* motor tiga fasa, dilakukan pengujian karakterisasi *driver* motor tiga fasa. Pengujian dilakukan dengan cara membandingan *input duty cycle* sinyal PWM yang diberikan oleh kontroler Arduino Mega 2560 terhadap tegangan efektif *output driver*, menggunakan voltmeter yang dipasang pada *output* rangkaian *driver* sebagai pembaca tegangan efektif *output driver* motor tiga fasa. Langkah – langkah dalam pengujian karakterisasi *driver* motor tiga fasa yaitu: menghubungkan *output* tegangan baterai dengan *input driver* motor tiga fasa, lalu menghubungkan *input* sinyal kontrol pada *driver* motor tiga fasa dengan pin *output* PWM di Arduino Mega 2560, kemudian mengatur *duty cycle* dari 0% sampai dengan 100% dengan kenaikan 5% setiap pembacaan tegangan efektif *output driver*. Hasil pembacaan yang didapatkan dari pengujian tersebut dicatat 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

Duty Cycle %	Output Driver (V) dengan Multimeter
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 dapat diketahui bahwa pada *duty cycle* 0% dan 5% tegangan efektif *output driver* sama dengan 0 V. Dengan mengambil data *duty cycle* terhadap tegangan efektif *output driver* motor tiga fasa yang terukur, didapatkan kurva tegangan efektif *output 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* (%)

3.5 Karakterisasi Sensor Kecepatan (*Rotary Encoder*)

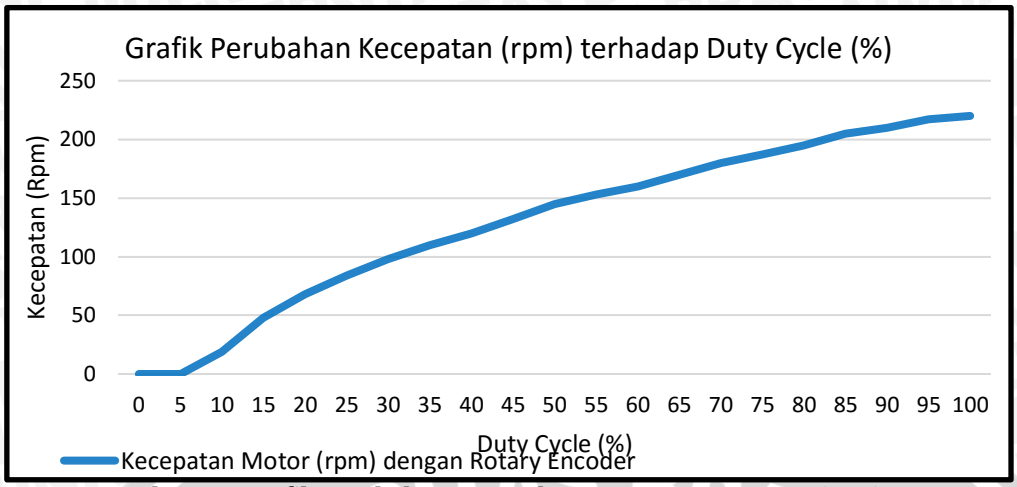
Untuk mengetahui tingkat kelinieran dari *rotary encoder* dalam pembacaan kecepatan motor BLDC, dilakukan pengujian karakterisasi *rotary encoder*. Pengujian dilakukan

dengan cara membandingkan input *duty cycle* sinyal PWM yang diberikan oleh kontroler Arduino Mega 2560 terhadap kecepatan motor BLDC yang dibaca oleh *rotary encoder*. Langkah – langkah dalam pengujian karakterisasi *rotary encoder* yaitu: menghubungkan *output* tegangan baterai dengan *input driver* motor BLDC, lalu menghubungkan pin *ouput rotary encoder* dengan pin interrupt eksternal, dan *input* sinyal kontrol pada *driver* motor tiga fasa dengan pin *output* PWM pada Arduino Mega 2560, kemudian mengatur *duty cycle* dari 0% sampai dengan 100% dengan kenaikan 5% setiap pembacaan kecepatan motor. Hasil pembacaan yang didapatkan dari pengujian tersebut dicatat 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 dapat diketahui bahwa pada *duty cycle* 0% dan 5% kecepatan motor sama dengan 0 rpm. Dengan mengambil data *duty cycle* terhadap kecepatan motor BLDC yang dibaca oleh rotary encoder, akan didapatkan kurva kecepatan motor yang dibaca oleh *rotary encoder* terhadap masukan *duty cycle* sinyal PWM (%) seperti ditunjukkan pada Gambar 3.4.



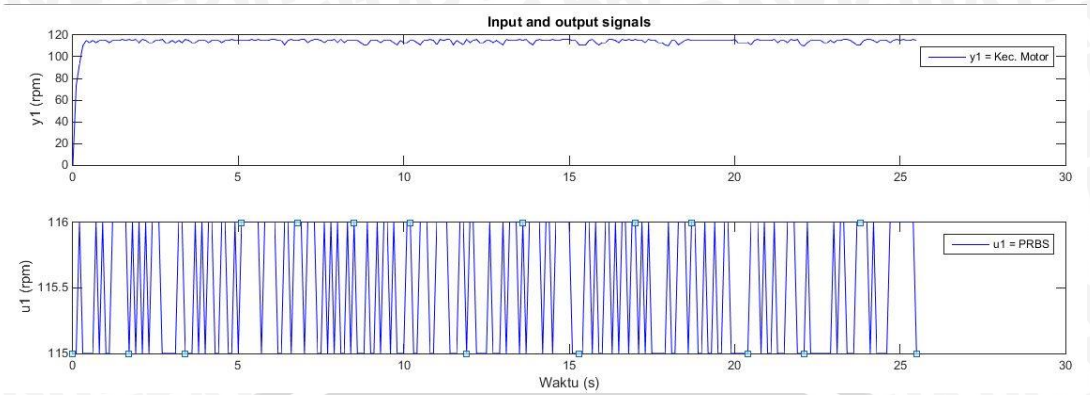
Gambar 3.4 Grafik perubahan respon kecepatan motor BLDC (rpm) terhadap *duty cycle* (%)

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

Pengontrolan kecepatan motor BLDC menggunakan Arduino Mega 2560 sebagai pengolah dan memberikan sinyal kontrol berupa *Pulse Width Modulation* (PWM) agar motor dapat bekerja sesuai dengan spesifikasi yang telah ditentukan. Agar Kontrol PI dapat bekerja secara optimal, diperlukan fungsi alih untuk mendapatkan parameter PI. Motor BLDC yang digunakan pada perancangan ini tidak diketahui fungsi alihnya, sehingga perlu dilakukan pengujian dengan menggunakan *rotary encoder* dan membangkitkan sinyal PRBS sebagai *input*. Pada perancangan ini motor diberi masukan unit step.

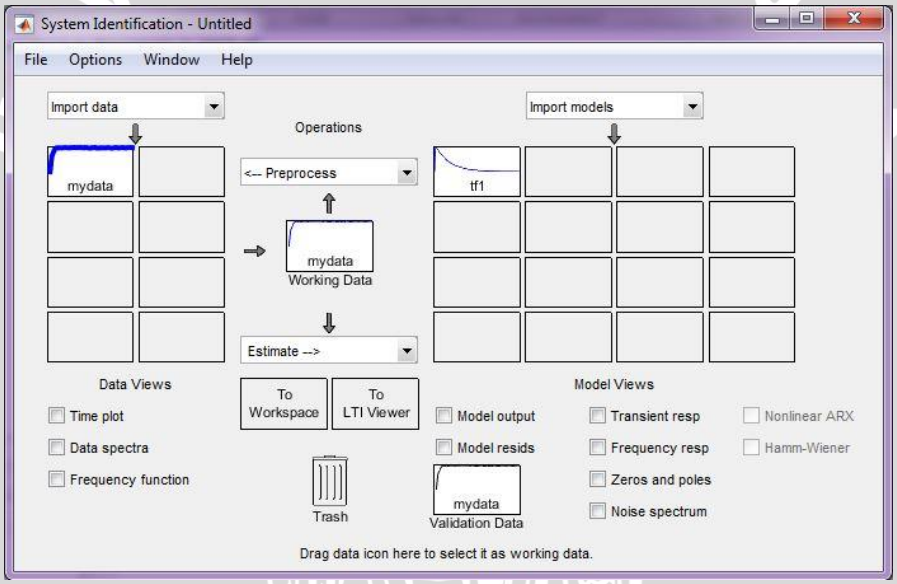
Fungsi alih dari motor didapatkan dari pemodelan dengan cara membangkitkan sinyal *Pseudo Random Binary Sequence* (PRBS). Langkah yang dilakukan untuk menentukan fungsi alih dengan cara membangkitkan sinyal PRBS adalah sebagai berikut:

1. Mencari nilai yang linier dari hasil kecepatan motor terhadap *duty cycle* 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 *input* 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.



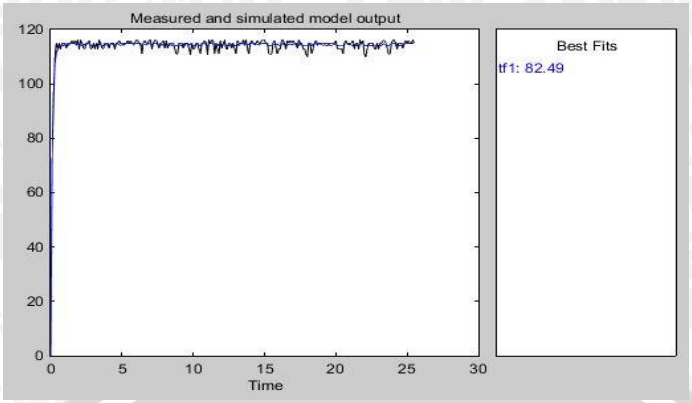
Gambar 3. 5 Output sinyal PRBS

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 *system identification toolbox* pada software MATLAB

6. Setelah melakukan beberapa estimasi model berdasarkan data yang telah di import didapatkan fungsi alih motor dengan *best fits* sebesar 82.49 seperti dalam Gambar 3.7.

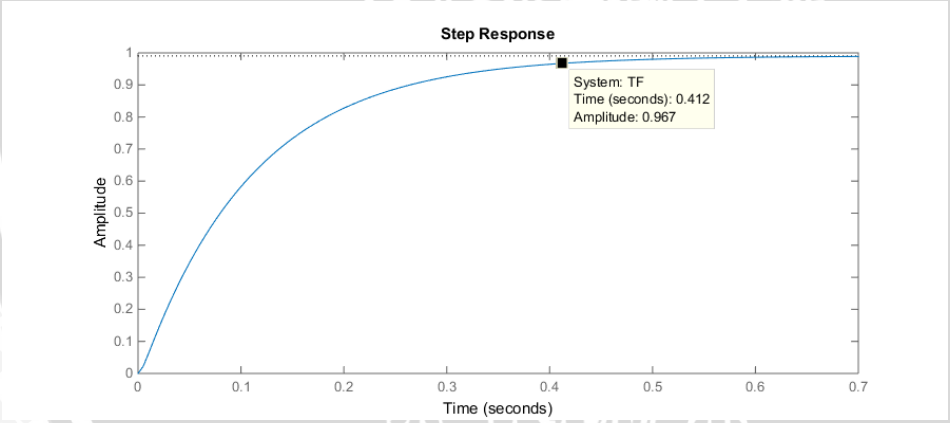


Gambar 3. 7 Hasil estimasi model

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

$$\frac{Y(s)}{U(s)} = \frac{2811}{s^2 + 318.6\,s\, +\, 2838} \qquad (3-1)$$

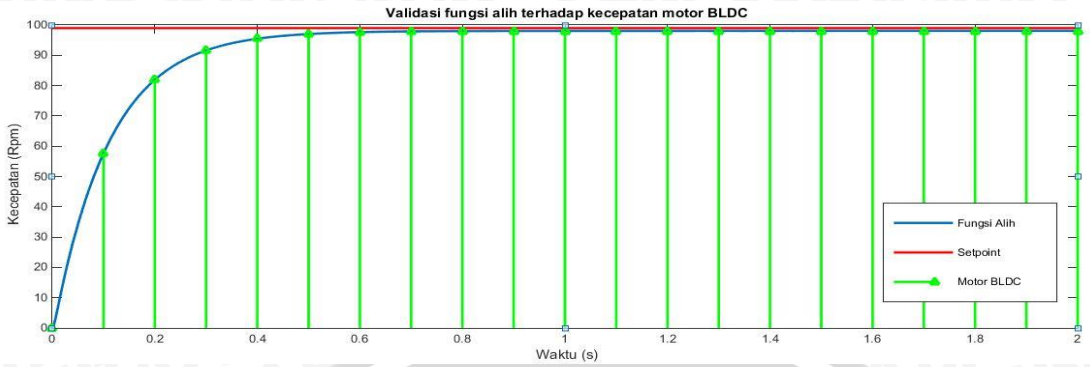
8. Dengan memberikan input *unit step* pada program MATLAB didapatkan output dalam Gambar 3.8, dan didapatkan nilai *settling time* 0.421 s dengan *output* maksimal berada di 0.967



Gambar 3.8 respon fungsi alih motor BLDC dengan input unit step

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

Dalam melakukan validasi fungsi alih motor BLDC dilakukan dengan cara membandingkan respon plant yang didapatkan dari identifikasi fungsi alih dan respon kecepatan motor BLDC yang didapatkan dari pembacaan sensor kecepatan (*rotary encoder*) dengan memberikan masukan pulsa *unit step*. Berikut perbandingan kedua respon yang didapat dengan menggunakan software MATLAB (lihat Gambar 3.9).



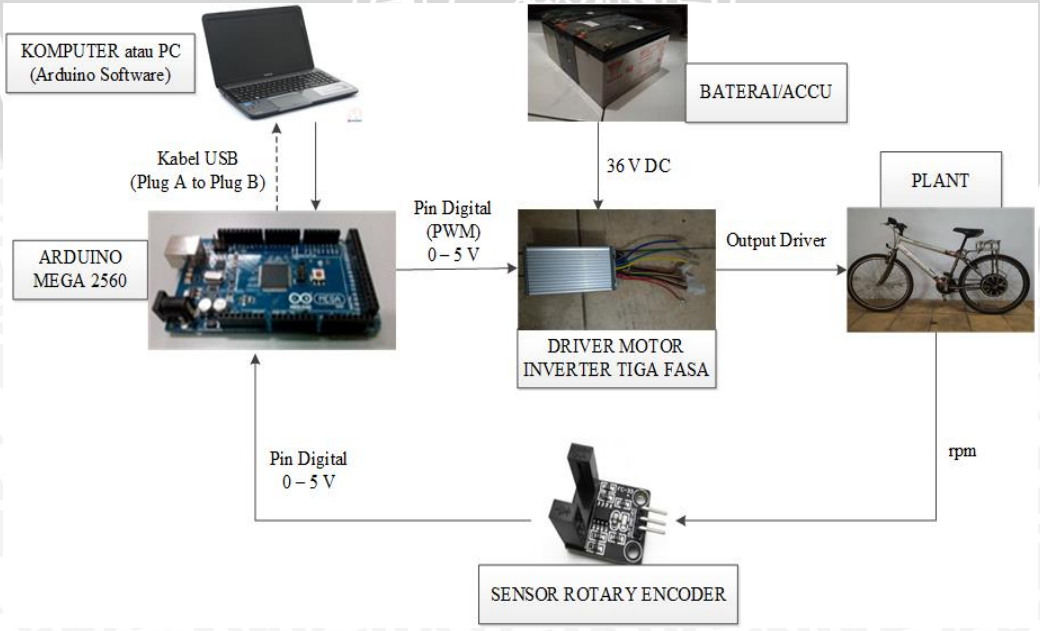
Gambar 3.9 Validasi fungsi alih dengan respon motor BLDC

Dari Gambar 3.9 dapat dilihat bahwa respon plant fungsi alih yang telah didapat dari proses identifikasi hampir menyerupai respon kecepatan motor motor BLDC. Jadi fungsi alih yang telah didapatkan dianggap dapat mewakili identifikasi motor BLDC.

3.8 Pembuatan Perangkat Keras

Pembuatan perangkat keras dilakukan sebagai langkah awal sebelum terbentuknya suatu sistem beserta pemrogramannya. Hal ini dimaksudkan agar pemodelan motor BLDC dengan pengontrol respon kecepatan menggunakan kontrol *Model Reference Adaptive Control* (MRAC) untuk tuning PI dapat berjalan sesuai dengan deskripsi awal yang telah direncanakan.

1. Skema pembuatan perangkat keras (Gambar 3.10)



Gambar 3.10 Skema Perangkat Keras

2. Penentuan modul elektronik yang digunakan meliputi :

- Baterai atau *ACCU* berfungsi sebagai sumber catu daya motor *Brushless Direct Curent* (BLDC) dimana memiliki range 0-36 VDC. (lihat Gambar 3.11).

Gambar 3.11 Baterai *atau Accu*

- Mikrokontroller Arduino Mega 2560 berfungsi sebgai penyimpan algoritma kontroler sistem, pengendali driver motor, dan mengolah data yang dikirim sensor. (lihat Gambar 3.12)

Gambar 3.12 *Arduino Mega 2560*

- Sensor yang digunakan adalah sensor kecepatan (rotary encoder) yang membutuhkan catu daya 5 VDC yang diperoleh dari mikrokontroler (lihat Gambar 3.13) sebagai *feedback*.

Gambar 3.13 Sensor kecepatan (rotary encoder)

- Driver Motor *Bruhless* DC (BLDC) (lihat Gambar 3.14).

Gambar 3.14 *driver inverter* tiga fasa

- Komputer atau PC yang sudah terinstall *software* Arduino dan MATLAB.

Gambar 3.15 Komputer atau PC

- Motor yang digunakan adalah motor *Bruhless* DC (BLDC) dengan catu daya 36 VDC. Motor BLDC berfungsi sebagai penggerak dari sepeda. (lihat Gambar 3.16).

Gambar 3.16 *Motor* BLDC

- Pembuatan sepeda yang telah terpasang motor *Brushless* DC (BLDC) dan telah dikopel oleh sensor kecepatan (*rotary encoder*) dapat dilihat pada Gambar 3.17.

Gambar 3.17 Sistem yang telah dirangkai

3.9 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 atau 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 respon 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 *Model Reference Adaptive Control* (MRAC).
10. Menguji parameter kontroler PI yang didapat dalam simulasi *simulink Matlab*.

11. Membandingkan respon sistem *simulink* dengan respon sistem motor BLDC yang didapat.
12. Mengimplementasikan hasil desain perancanan pada sistem.

3.10 Perancangan Algoritma

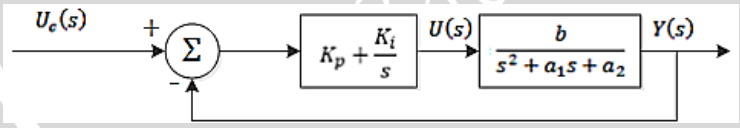
Dalam perancangan perangkat lunak, hal pertama yang dilakukan adalah mengetahui karakteristik motor BLDC. Setelah itu, dilakukan penghitungan untuk menentukan algoritma kontroller dengan *Model Reference Adaptive Control* (*MRAC*). Setelah didapatkan algoritma, kemudian dibuat program untuk mikrokontroller dengan software program Arduino. Perancangan algoritma tersebut melalui beberapa tahap:

1. Perancangan algoritma teknik MRAC pada *tuning* pengendali PI dalam persamaan fungsi s.
2. Konversi persamaan fungsi s ke dalam bentuk diskrit melalui transformasi z yang kemudian dikonversi ke dalam bentuk persamaan beda.
3. Pembuatan simulasi pada software matlab.
4. Penyusunan flowchart yang algoritmanya diterapkan pada mikrokontroller.
5. Implementasi pada alat.

3.11 Desain Kontrol PI Menggunakan Teknik MRAC

Fungsi alih dari motor BLDC merupakan orde dua, maka akan didapatkan bentuk fungsi alih motor BLDC pada Persamaan 3.2.

$$\frac{Y(s)}{U(s)} = \frac{b}{s^2 + a_1 s + a_2} \qquad (3-2)$$



Gambar 3.18 Diagram blok sistem *loop* tertutup

Fungsi alih dari diagram blok sistem *loop* tertutup pada Gambar 3.18 adalah

$$\frac{Y(s)}{U_c(s)} = \frac{bK_p s + bK_i}{s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i} \qquad (3-3)$$

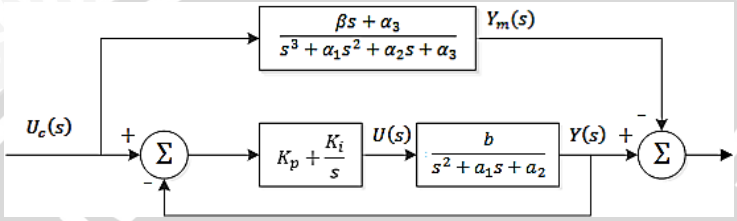
Model referensi ditentukan melalui derajat relatif sistem. Derajat relatif model referensi sistem harus sama dengan derajat relatif *plant*. Derajat relatif adalah selisih antara derajat *polinominal pole* dengan derajat *polinominal zero*. Orde model referensi disesuaikan dengan

orde *plant*, sehingga model referesi dapat diasumsikan dalam bentuk fungsi alih orde 3 pada Persamaan 3.4.

$$\frac{Y_m(s)}{U_c(s)} = \frac{\beta s + \alpha_3}{s^3 + \alpha_1 s^2 + \alpha_2 s + \alpha_3} \qquad (3-4)$$

Dengan: $\beta = bK_p$ $\alpha_1 = a_1$ $\alpha_2 = a_2 + bK_p$ $\alpha_3 = bK_i$

Dalam penggunaan teknik MRAC pada *tuning* kontroler PI pada *plant*, model dan kontroler dapat dibuat dalam bentuk diagram blok pada Gambar 3.19.



Gambar 3.19 Diagram blok sistem dengan model referensi

Perubahan parameter kontroler terhadap waktu dinyatakan dalam model pendekatan gradien dimana error didefinisikan sebagai selisih antara respon plant y dan respon model referensi $y_m (e = y - y_m)$. Sehingga memungkinkan untuk didapatkan parameter kontroler K_p dan K_i menerapkan aturan MIT $\frac{d\theta}{dt} = -\gamma e \frac{\partial e}{\partial \theta}$ dengan θ didapat dari penurunan berikut:

$$\frac{\partial e}{\partial \theta} = \frac{\partial (y - y_m)}{\partial \theta} \qquad (3-5)$$

$$\frac{d\theta}{dt} = s\theta = -\gamma e \frac{\partial e}{\partial \theta} \qquad (3-6)$$

Dari Persamaan 3.5 dan 3.6 didapatkan bahwa

$$\theta = \frac{-\gamma}{s} e \frac{\partial e}{\partial \theta} \qquad (3-7)$$

Dengan mengganti θ dengan K_p dan K_i akan diperoleh Persamaan 3.8 dan 3.9.

$$\frac{dK_p}{dt} = -\gamma_p \left(\frac{\partial J}{\partial K_p}\right) = -\gamma_p \left(\frac{\partial J}{\partial e}\right)\left(\frac{\partial e}{\partial y}\right)\left(\frac{\partial y}{\partial K_p}\right) \qquad (3-8)$$

$$\frac{dK_i}{dt} = -\gamma_i \left(\frac{\partial J}{\partial K_i}\right) = -\gamma_i \left(\frac{\partial J}{\partial e}\right)\left(\frac{\partial e}{\partial y}\right)\left(\frac{\partial y}{\partial K_i}\right) \qquad (3-9)$$

Dimana $e = y - y_m \Rightarrow \frac{\partial e}{\partial y} = 1$; $\frac{\partial J}{\partial y} = e$

Dengan menggunakan hubungan Persamaan 3.8 dan 3.9 didapatkan

$$\frac{dK_p}{dt} = -\gamma_p \left(\frac{\partial J}{\partial K_p}\right) = -\gamma_p e \left(\frac{\partial y}{\partial K_p}\right) \quad (3-10)$$

$$\frac{dK_i}{dt} = -\gamma_i \left(\frac{\partial J}{\partial K_i}\right) = -\gamma_i e \left(\frac{\partial y}{\partial K_i}\right) \quad (3-11)$$

Cara agar mendapatkan $\frac{\partial y}{\partial K_p}$ dan $\frac{\partial y}{\partial K_i}$ dengan cara mendiferensialkan Persamaan 3.3, sehingga

$$\left(s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i\right) = u_c\left(bK_p s + bK_i\right)$$

$$\left(y\frac{\partial}{\partial K_p}\left(s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i\right)\right) + \left(\left(s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i\right)\frac{\partial y}{\partial K_p}\right)$$

$$= \frac{\partial}{\partial K_p} u_c\left(bK_p s + bK_i\right)$$

$$\frac{\partial y}{\partial K_p} = b\frac{s}{s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i}(u_c - y) \quad (3-12)$$

Dengan cara yang sama untuk $\frac{\partial y}{\partial K_i}$

$$y\left(s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i\right) = u_c\left(bK_p s + bK_i\right)$$

$$\left(y\frac{\partial}{\partial K_i}\left(s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i\right)\right) + \left(\left(s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i\right)\frac{\partial y}{\partial K_i}\right)$$

$$= \frac{\partial}{\partial K_i} u_c\left(bK_p s + bK_i\right)$$

$$\frac{\partial y}{\partial K_i} = b\frac{1}{s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i}(u_c - y) \quad (3-13)$$

Dengan mensubstitusi Persamaan 3.12 dan 3.13 ke dalam Persamaan 3.10 dan 3.11 didapatkan Persamaan 3.14 dan 3.15.

$$\frac{dK_p}{dt} = -\gamma_p e\frac{bs}{s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i}(u_c - y) \quad (3-14)$$

$$\frac{dK_i}{dt} = -\gamma_i e\frac{b}{s^3 + a_1 s^2 + (a_2 + bK_p)s + bK_i}(u_c - y) \quad (3-15)$$

Dari Persamaan 3.14 dan 3.15 *gain parameters* tidak dapat ditemukan ketika nilai a_1, a_2 dan b tidak diketahui. Jadi rumus yang diturunkan dari aturan MIT ini belum dapat digunakan. Sebagai gantinya, beberapa pendekatan diperlukan. Jika berfikir sistem

merupakan model yang sempurna, bandingkan hubungan antara *input-output* dari sistem dengan model referensi, pendekatan yang akan digunakan seperti pada persamaan 3.16.

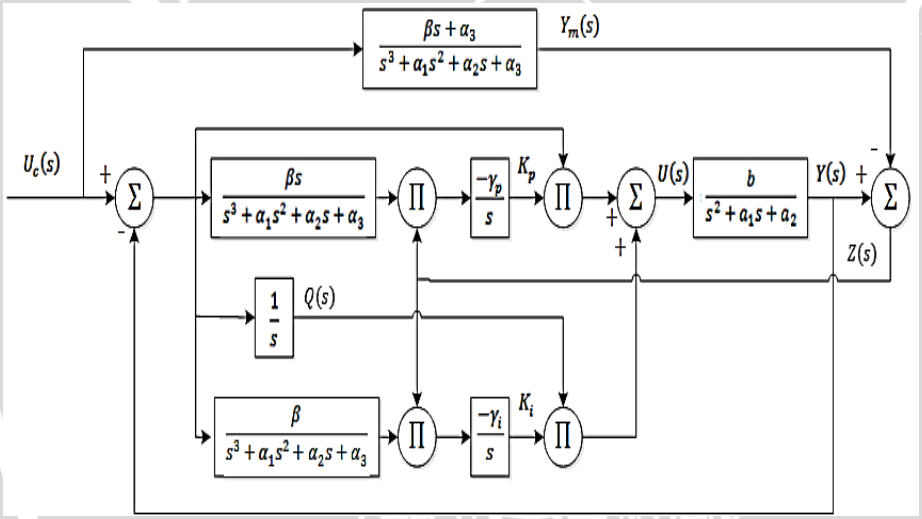
$$\begin{cases} \beta = b \\ \alpha_1 = a_1 \\ \alpha_2 = bK_p + a_2 \\ \alpha_3 = bK_i \end{cases} \quad (3-16)$$

Sehingga persamaan yang baru adalah

$$K_p = \frac{-\gamma_p}{s} e \frac{\beta s}{s^3 + \alpha_1 s^2 + \alpha_2 s + \alpha_3}(u_c - y) \quad (3-17)$$

$$K_i = \frac{-\gamma_i}{s} e \frac{\beta}{s^3 + \alpha_1 s^2 + \alpha_2 s + \alpha_3}(u_c - y) \quad (3-18)$$

Sebuah representasi diagram dari skema adaptif berdasarkan penurunan persamaan yang telah diperoleh dapat dilihat pada Gambar 3.20.



Gambar 3.20 Diagram blok penggunaan teknik MRAC pada *tuning* kontroler PI

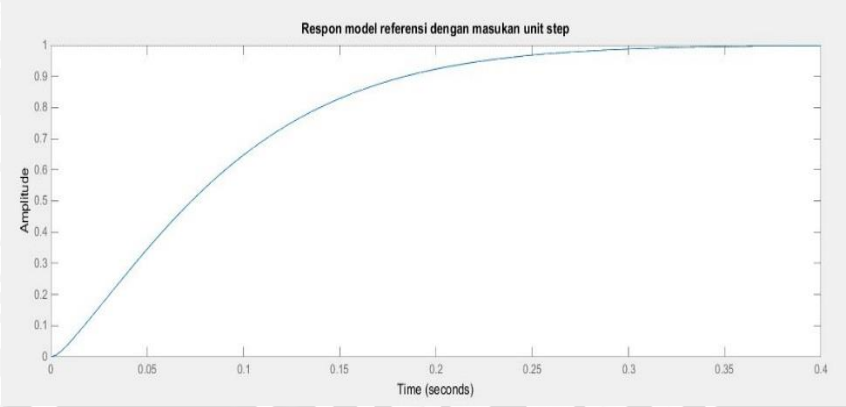
3.12 Penentuan Model Referensi

Penentuan model referensi dilakukan dengan cara *try and error* sehingga didapatkan respon yang stabil yang mengikuti karakteristik dari *plant*. Model referensi yang akan ditentukan tidak memilki *error steady state*, tidak terdapat osilasi dan memilki *settling time* yang cepat. Penentuan model referensi dapat ditentukan dengan cara melihat hasil respon saat simulasi melalui apikasi matlab dan pada saat implementasi sistem.

Fungsi dari model referensi yang digunakan adalah:

$$\frac{Y_m(s)}{U_c(s)} = \frac{307{,}3s + 1291}{s^3 + 71.87s^2 + 583{,}75s + 1291} \quad (3-19)$$

Respon transien model referensi terhadap masukan *unit step* dapat dilihat dalam Gambar 3.21.



Gambar 3.21 Respon model referensi dengan masukan *unit step*

3.13 Penetapan Parameter Kontroler

Penetapan parameter kontroler terhadap waktu dinyatakan dalam pendekatan gradien dimana *error* didefinisikan sebagai selisih antara keluaran *plant* y dan keluaran model referensi $y_m (e = y - y_m)$. Sehingga memungkinkan untuk didapatkan parameter kontroler K_p dan K_i dengan menerapkan aturan MIT. Berdasarkan desain kontroler PI dengan teknik MRAC yang telah dijelaskan sebelumnya, maka akan didapatkan bentuk persamaan 3.20 dan 3.21:

$$K_p = \frac{-\gamma_p}{s} e \frac{307{,}3s}{s^3 + 71{,}87s^2 + 583.75s + 1291}(u_c - y) \qquad (3-20)$$

$$K_i = \frac{-\gamma_i}{s} e \frac{307{,}3}{s^3 + 71{,}87s^2 + 583{,}75s + 1291}(u_c - y) \qquad (3-21)$$

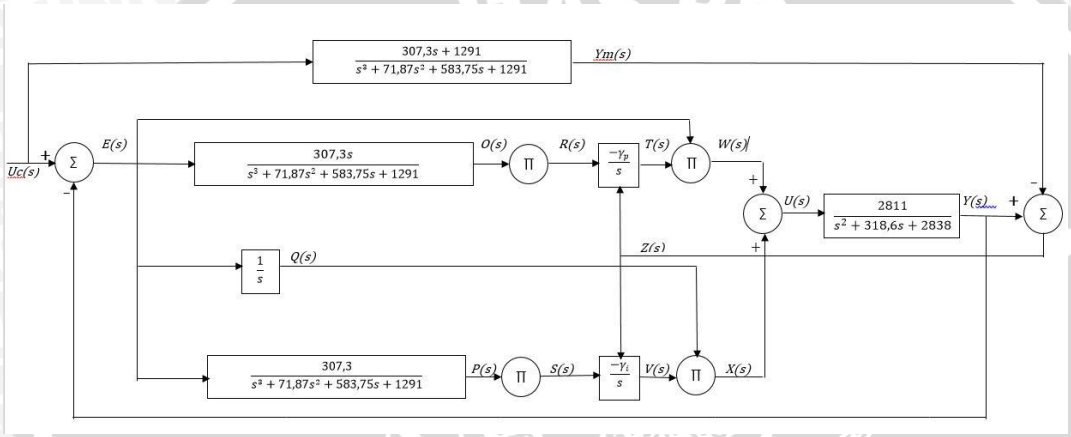
Pada persamaan 3.20 dan 3.21 terdapat dua parameter yang harus ditentukan nilainya terlebih dahulu dalam mendesain sistem yang akan dikontrol, yaitu parameter γ_p dan γ_i. Dimana cara menetukan nilai parameter tersebut dilakukan dengan cara *try and error*.

3.14 Desain Persamaan Beda

Terdapat banyak cara yang dapat digunakan untuk proses diskritisasi, tiga diantaranya yang banyak digunakan dalam bidang kontrol adalah *backward difference*, *forward difference* (Metode Euler), dan *bilinear transformation.* Ketiga metode tersebut hanya merupakan pendekatan, sehingga hasilnya tidak akan persis sama dengan bentuk analog karena dalam bentuk diskrit selalu ada sebagian informasi yang hilang, yaitu informasi yang terjadi antara satu cuplikan dengan cuplikan lainnya walaupun frekuensi cuplik yang

digunakan tinggi. Kedua, karena formula matematika yang digunakan dalam proses diskritisasi juga diturunkan berdasarkan pendekatan agar lebih mudah digunakan. Pada perancangan ini akan digunakan metode *backward difference* karena seluruh daerah stabil dan sebagian daerah tidak stabil di bidang s di-*mapping* ke daerah stabil z, hal inilah yang menyebabkan metode *backward difference* jauh lebih *robust* dibandingkan dengan kedua metode lainnya (Asro, 2009).

Bedasarkan hasil dari penentuan fungsi alih, model referensi, dan parameter kontroler dapat disusun menjadi sebuah sistem yang direpresentasikan dalam diagram blok seperti dalam Gambar 3.22.



Gambar 3.22 Diagram blok sistem dengan desain kontroler PI menggunakan teknik MRAC

Model referensi dari sistem dinyatakan dalam bentuk frekuensi (dalam bentuk s) (lihat Persamaan 3.19). Dengan menggunakan metode *backward difference*, ganti operator s dalam persamaan analog bentuk s dengan persamaan 3.22.

$$s = \frac{1 - z^{-1}}{T_s} \qquad (3-22)$$

Dimana T_s = periode sampling, maka akan didapatkan penurunan persamaan 3.19.

$$Y_m(z) = \left. \frac{307{,}3\left(\frac{1-z^{-1}}{T_s}\right) + 1291}{\left(\frac{1-z^{-1}}{T_s}\right)^3 + 71{,}87\left(\frac{1-z^{-1}}{T_s}\right)^2 + 583{,}75\left(\frac{1-z^{-1}}{T_s}\right) + 1291} U_c(z) \right| \times \frac{T_s}{T_s}$$

$$Y_m(z) = \frac{307{,}3{T_s}^2(1-z^{-1}) + 1291{T_s}^3}{(1-z^{-1})^3 + 71{,}87T_s(1-z^{-1})^2 + 583{,}75{T_s}^2(1-z^{-1}) + 1291{T_s}^3} U_c(z)$$

$$Y_m(z) = \frac{307{,}3{T_s}^2(1-z^{-1}) + 1291{T_s}^3}{(1-3z^{-1}+3z^{-2}-z^{-3}) + 71{,}87T_s(1-2z^{-1}+z^{-2}) + 583{,}75{T_s}^2(1-z^{-1}) + 1291{T_s}^3} U_c(z)$$

$$Y_m(z) = \frac{307{,}3{T_s}^2 - 307{,}3{T_s}^2 z^{-1} + 1291{T_s}^3}{1 - 3z^{-1} + 3z^{-2} - z^{-3} + 71{,}87T_s - 143{,}74T_s z^{-1} + 71{,}87T_s z^{-2} + 58{,}75{T_s}^2 - 58{,}75{T_s}^2 z^{-1} + 1291{T_s}^3} U_c(z)$$

$$Y_m(z) = \frac{\left(1291{T_s}^3 + 307{,}3{T_s}^2\right) - 307{,}3{T_s}^2 z^{-1}}{\left(1291{T_s}^3 + 583{,}75{T_s}^2 + 71{,}87T_s + 1\right) - \left(583{,}75{T_s}^2 + 143{,}74T_s + 3\right)z^{-1} + (71{,}87T_s + 3)z^{-2} - z^{-3}} U_c(z)$$

Kalikan setiap fraksi/komponen dengan $\left(\left(1291{T_s}^3 + 583{,}75{T_s}^2 + 71{,}87T_s + 1\right) - \left(583{,}75{T_s}^2 + 143{,}74T_s + 3\right)z^{-1} + (71{,}87T_s + 3)z^{-2} - z^{-3}\right)$ sehingga diperoleh persaamaan 4.23.

$$\begin{aligned} Y_m(z)\Big(&\left(1291{T_s}^3 + 583{,}75{T_s}^2 + 71{,}87T_s + 1\right) - \left(583{,}75{T_s}^2 + 143{,}74T_s + 3\right)z^{-1} \\ &+ (71{,}87T_s + 3)z^{-2} - z^{-3}\Big) \\ &= \Big(\left(1291{T_s}^3 + 307{,}3{T_s}^2\right) \\ &- 307{,}3{T_s}^2 z^{-1}\Big) U_c(z) \qquad (3-23) \end{aligned}$$

Susun kembali persamaan 3.23 berdasarkan jenis variable (Y_m atau U_c) dan berdasarkan jenis operator (z^{-n}), sehingga menjadi Persamaan 3.24.

$$\begin{aligned} Y_m(z)&\left(1291{T_s}^3 + 583{,}75{T_s}^2 + 71{,}87T_s + 1\right) \\ &= \left(583{,}75{T_s}^2 + 143{,}74T_s + 3\right)Y_m(z)z^{-1} - (71{,}87T_s + 3)Y_m(z)z^{-2} \\ &+ Y_m(z)z^{-3} + \left(1291{T_s}^3 + 307{,}3{T_s}^2\right)U_c(z) \\ &- 307{,}3{T_s}^2 z^{-1} U_c(z) \qquad (3-24) \end{aligned}$$

Jika Persamaan 3.24 disederhanakan lagi akan didapatkan persamaan 3.25.

$$Y_m(z) = \frac{A}{G}Y_m(z)z^{-1} - \frac{B}{G}Y_m(z)z^{-2} + \frac{C}{G}Y_m(z)z^{-3} + \frac{D}{G}U_c(z) - \frac{E}{G}z^{-1}U_c(z) \quad (3-25)$$

Dalam bentuk time domain, dengan mengganti operator z^0 menjadi waktu saat ini (k), z^{-1} menjadi waktu sesaat sebelumnya atau satu kali waktu tunda $(k-1)$, dan seterusnya maka akan didapatakan persamaan 3.26

$$y_m(k) = \frac{A}{G}y_m(k-1) - \frac{B}{G}y_m(k-2) + \frac{C}{G}y_m(k-3) + \frac{D}{G}u_c(k) - \frac{E}{G}u_c(k-1) \quad (3-26)$$

Dengan menggunakan cara yang sama, maka akan didapatkan Persamaan 3.27 sampai 3.31.

- Model referensi untuk Kp:

$$o(k) = \frac{A}{G}o(k-1) - \frac{B}{G}o(k-2) + \frac{C}{G}o(k-3) + \frac{E}{G}e(k) - \frac{E}{G}e(k-1) \qquad (3-27)$$

- Model referensi untuk Ki:

$$p(k) = \frac{A}{G}p(k-1) - \frac{B}{G}p(k-2) + \frac{C}{G}p(k-3) + \frac{F}{G}e(k) \qquad (3-28)$$

Dimana:

$A = 583{,}75{T_s}^2 + 143{,}74T_s + 3$

$B = 71{,}87T_s + 3$

$C = 1$

$D = 1291{T_s}^3 + 307{,}3{T_s}^2$

$E = 307{,}3{T_s}^2$

$F = 307{,}3{T_s}^3$

$G = 1291{T_s}^3 + 583{,}75{T_s}^2 + 71{,}87T_s$

- Integrator:

$$q(k) = q(k-1) + T_s e(k) \qquad (3-29)$$

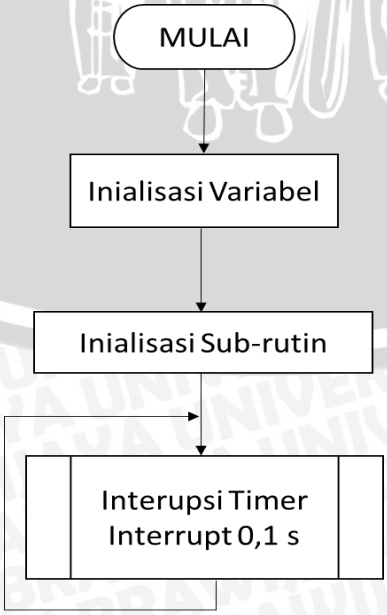
- Parameter kontroler:

$$t(k) = t(k-1) - \gamma_p T_s r(k); \quad r(k) = o(k) * z(k); \quad z(k) = y_m(k) - y(k) \qquad (3-30)$$

$$v(k) = v(k-1) - \gamma_i T_s s(k); \quad s(k) = p(k) * z(k) \qquad (3-31)$$

3.15 *Flowchart* Program Utama

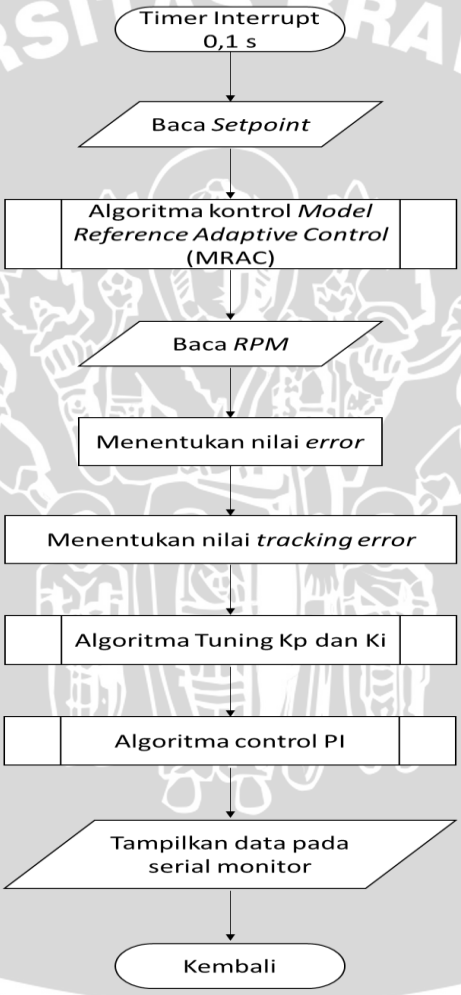
Progam diawali dengan inialisasi masing – masing variabel dan sub-rutin untuk proses control. Pada progam utama terdapat sub-rutin timer interrupt yang dieksekusi setiap 0,1 detik. Flowchart dari program utama dapat dilihat dalam Gambar 3.23.



Gambar 3.23 *Flowchart* program utama

3.15.1 *Flowchart* Sub–Rutin Timer Interrupt

Progam dieksekusi setiap 0,1 detik sekali dan diawali dengan pembacaan setoint. Setelah itu algoritma kontrol MRAC dijalankan. Kemudian, dilakukan pembacaan kecepatan plant yang menghasilkan error ketika dikurangkan terhadap setpoint. Selain error, ditentukan tracking error yaitu selisih antara respon plant dan repon model referensi. Error dan tracking error yang telah didapat digunakan sebagai masukan algoritma tuning Kp dan Ki pada kontroler PI. Nilai kontroler PI dikeluarkan sebagai variabel yang digunakan untuk mengatur kecepatan motor. Flowchart dari sub-rutin *timer interrupt* dapat dilihat dalam Gambar 3.24.

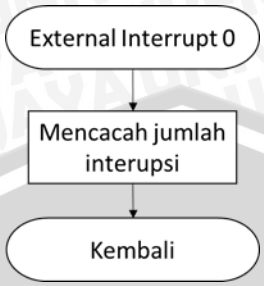


Gambar 3.24 *Flowchart* sub-rutin *timer interrupt* 0,1 s

3.15.2 *Flowchart* Sub–Rutin Timer External Interrupt

Program dieksekusi ketika pin external interrupt 0 membaca tepi turun dari sinyal pulsa yang dikirim oleh *rotary encoder*, dan berfungsi untuk mencacah jumlah

interupsi yang terjadi untuk menentukan kecepatan putar motor BLDC. *Flowchart* dari sub-rutin external interrupt dapat dilihat dalam gambar 3.25.



Gambar 3.25 *Flowchart* sub-rutin *external interrupt*

BAB IV
HASIL DAN PEMBAHASAN

Hasil dan pembahasan dilakukan dengan melakukan simulasi dan analisis sistem. Simulasi dan analisis sistem dilakukan untuk menguji apakah parameter yang sudah didapatkan pada simulasi dapat berfungsi pada alat dengan baik sesuai dengan *input* yang diinginkan serta mengetahui hasil responnya. Simulasi dilakukan untuk mendapatkan letak kesalahan dan mempermudah analisis pada sistem.

Simulasi dibagi menjadi beberapa bagian, yaitu:

1. Simulasi penentuan parameter kontrol γ_p (Gamma-p) dan γ_i (gamma-i).

Analisis sistem dibagi menjadi 2 bagian, yaitu:

1. Analisis sistem berdasarkan simulasi.
2. Analisis sistem berdasarkan implementasi.

4.1 Simulasi Penggunaan Teknik MRAC pada Kontrol PI

Tahapan-tahapan simulasi yang dilakukan adalah

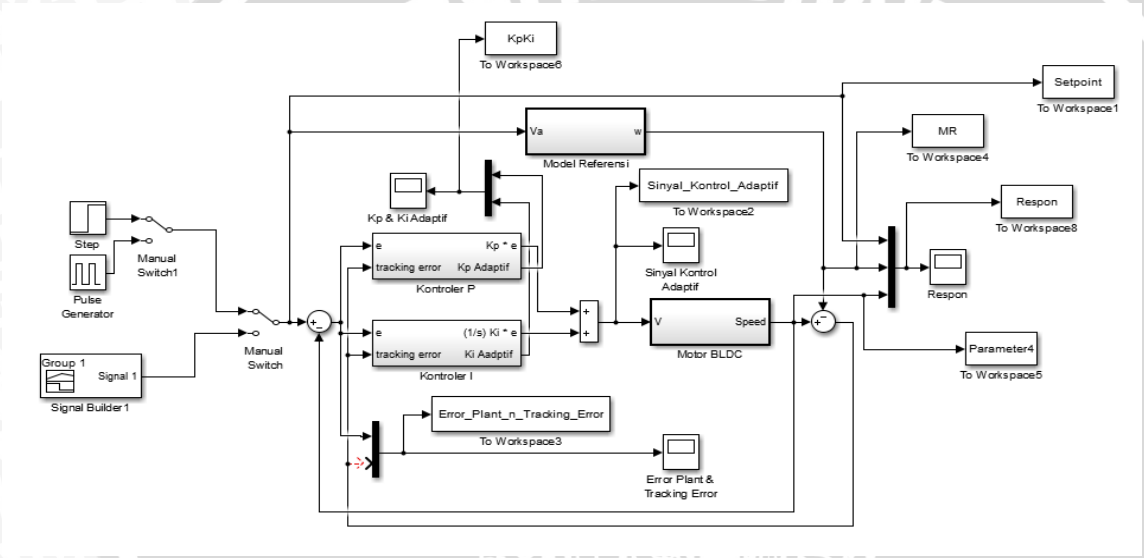
1. Simulasi sistem dengan beberapa nilai parameter kontroler γ_p dan γ_i.
2. Menentukan nilai parameter kontroler γ_p dan γ_i dilakukan untuk mendapatkan respon keluaran *plant* yang dapat menjejaki respon model referensi. Penentuan parameter kontroler γ_p dan γ_i dilakukan dengan cara memberikan nilai awal parameter pada nilai tertentu sebelum simulasi dilakukan.

Langkah-langkah yang dilakukan :

1. Menentukan nilai parameter kontroler γ_p dan γ_i dimulai dari nilai yang terkecil dan amati keluaran sistem yang meliputi respon sinyal masukan, model referensi, respon model, sinyal kontrol dan parameter kontroler. Dari simulasi ini dapat diketahui bagaimana pengaruh perubahan parameter kontroler γ_p dan γ_i dalam sistem.
2. Melakukan simulasi dengan beberapa nilai *setpoint*. Sistem yang digunakan adalah sistem yang tebaik yang didapatkan dari simulasi yang telah dilakukan. Dari simulasi ini akan dapat diamati bagaimana sistem beradaptasi untuk menjejaki model referensi.

4.1.1 Simulasi Hasil Penentuan Nilai Parameter Kontroler

Simulasi ini dilakukan pada sistem dengan sinyal masukan berupa unit step dengan menggunakan diagram blok simulink pada Matlab (lihat Gambar 4.1). Penentuan γ_p dan γ_i dilakukan dengan cara *try and error* agar mendapatkan hasil repon yang diinginkan oleh sistem. γ_p dan γ_i ditentukan dengan cara melihat hasil respon dari hasil simulasi aplikasi matlab dan pada saat implementsi sistem. Dari hasil simulasi pada aplikasi matlab ini diharapkan dapat memperoleh nilai parameter kontroler γ_p dan γ_i yang sesuai.

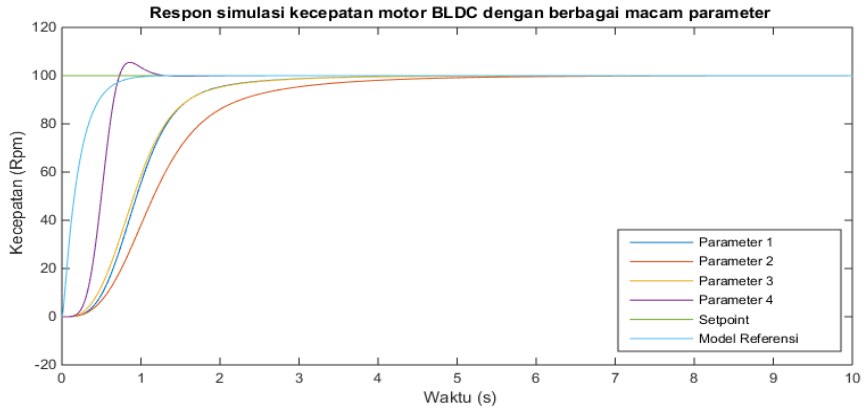


Gambar 4.1 Diagram blok simulink pada Matlab

Terdapat 4 nilai parameter kontroler yang disimulasikan yaitu terdapat pada Tabel 4.1. Respon sinyal masukan, model referensi dan keluaran sistem dapat dilihat dalam Gambar 4.2.

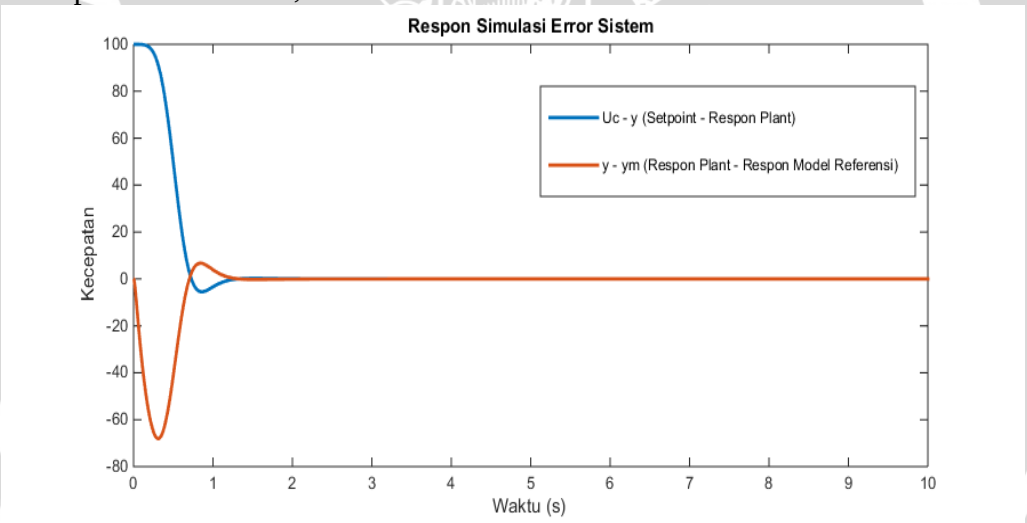
Tabel 4.1 Nilai parameter kontrol

Parameter ke-	γ_p	γ_i
1	0,00005	0,00009
2	0,00005	0.00005
3	0,0001	0,0001
4	0,0001	0,0009

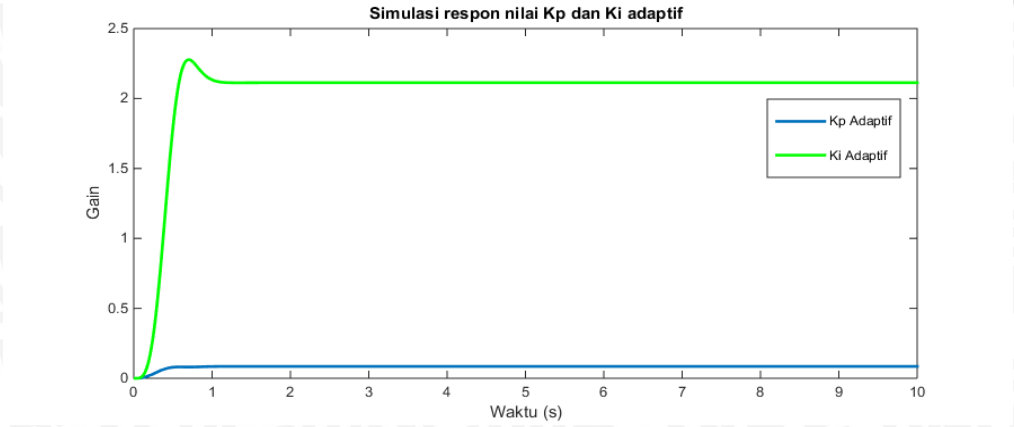


Gambar 4.2 Respon simulasi sistem dengan variasi nilai parameter γ_p dan γ_i

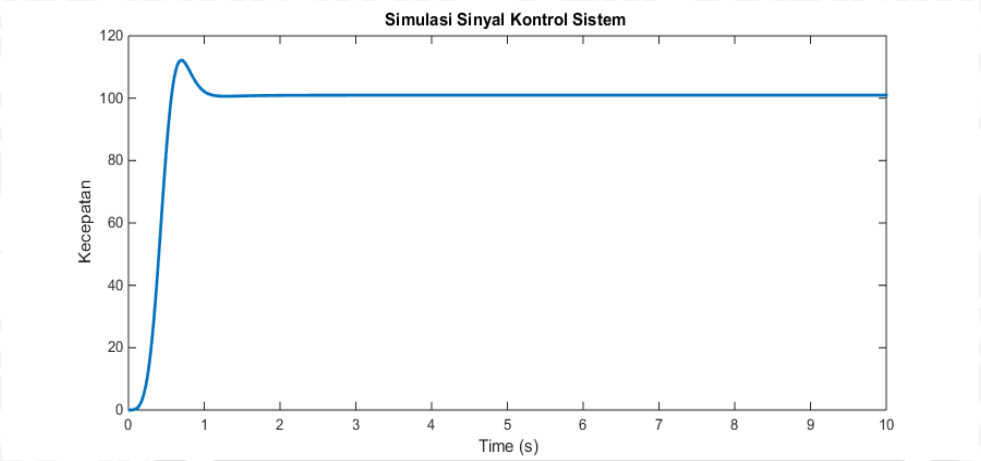
Berdasarkan hasil 4 parameter yang disimulasikan, nilai parameter γ_p dan γ_i ditentukan dengan melihat respon yang memiliki *settling time* dan *maximum overshoot* yang paling kecil, yaitu parameter ke-4. Hasil simulasi respon dari *error* perubahan parameter kontroler dan sinyal kontrol dengan parameter ke-4 dapat dilihat pada Gambar 4.3, 4.4 dan 4.5.



Gambar 4.3 Respon simulasi *error* sistem



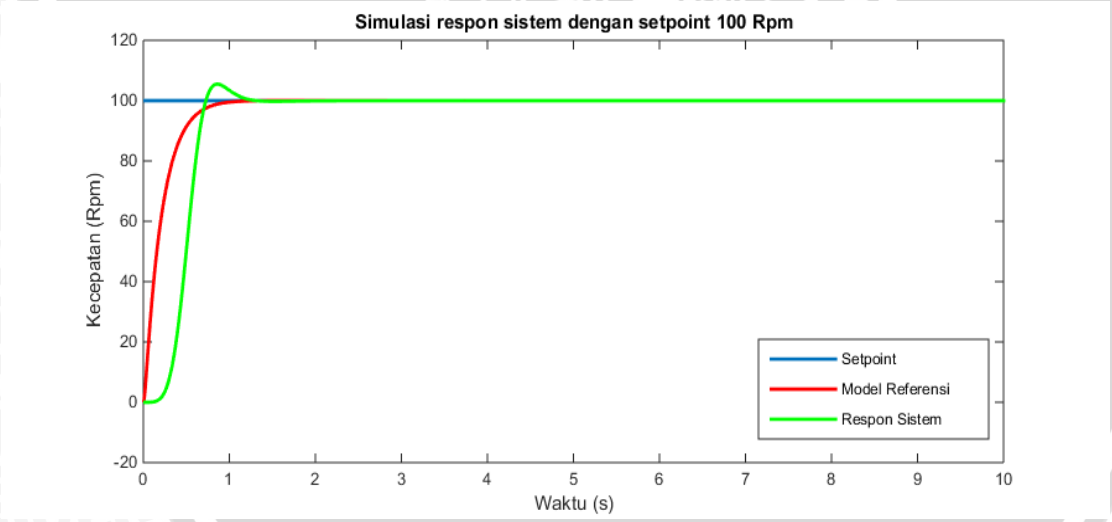
Gambar 4.4 Respon simulasi perubahan Kp dan Ki



Gambar 4.5 Respon simulasi sinyal control

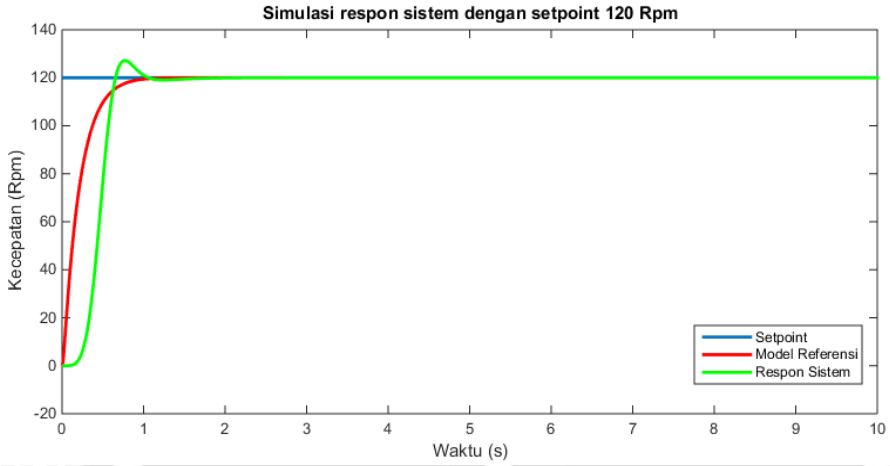
4.1.2 Simulasi Pengujian Sistem

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 4.6, Gambar 4.7, dan Gambar 4.8.



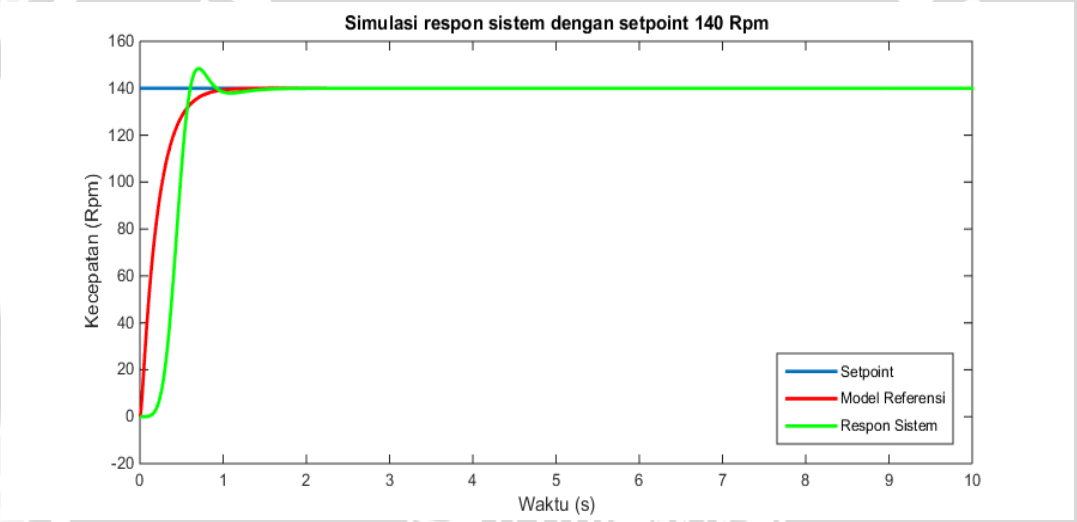
Gambar 4.6 Simulasi respon sistem dengan setpoint 100 Rpm

Dari respon yang ditunjukan dalam Gambar 4.6 dapat disimpulkan bahwa sistem tidak terdapat *error steady state*, *overshoot* sebesar 5%, dan *settling time* sebesar 1,1 detik.



Gambar 4.7 Simulasi respon sistem dengan setpoint 120 Rpm

Dari respon yang ditunjukan dalam Gambar 4.7 dapat disimpulkan bahwa sistem tidak terdapat *error steady state*, *overshoot* sebesar 5%, dan *settling time* sebesar 1 detik.



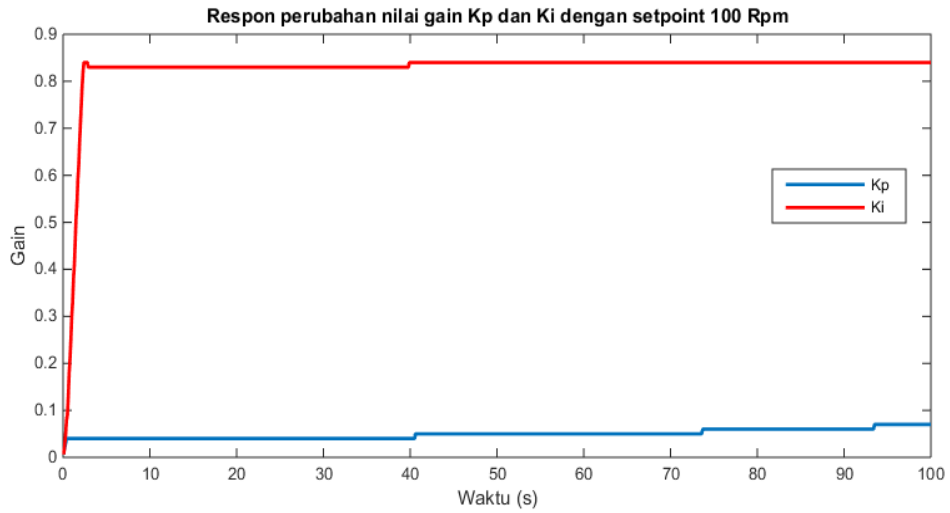
Gambar 4.8 Simulasi respon sistem dengan setpoint 140 Rpm

Dari respon yang ditunjukan dalam Gambar 4.8 dapat disimpulkan bahwa sistem tidak terdapat *error steady state*, *overshoot* sebesar 5%, dan *settling time* sebesar 0,9 detik.

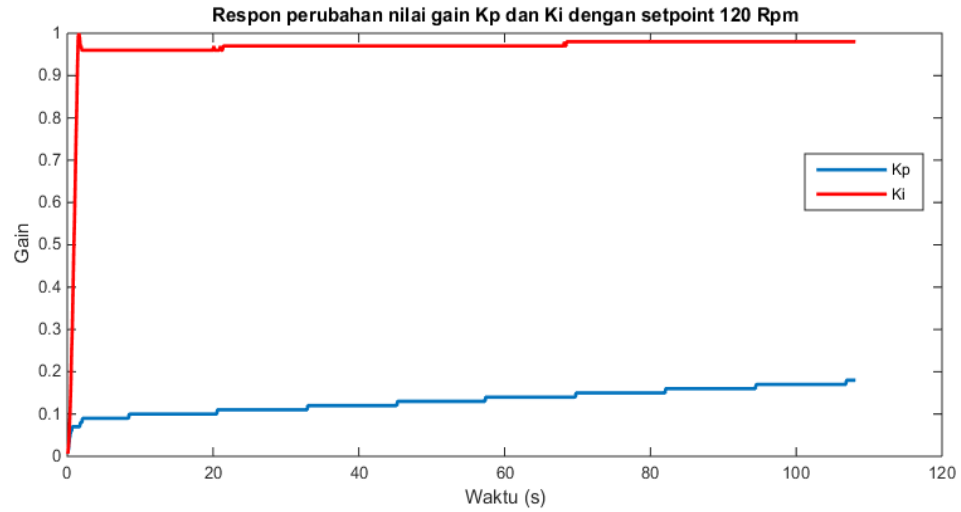
4.2 Implementasi

4.2.1 Hasil Implementasi dengan Beberapa Nilai *Setpoint*

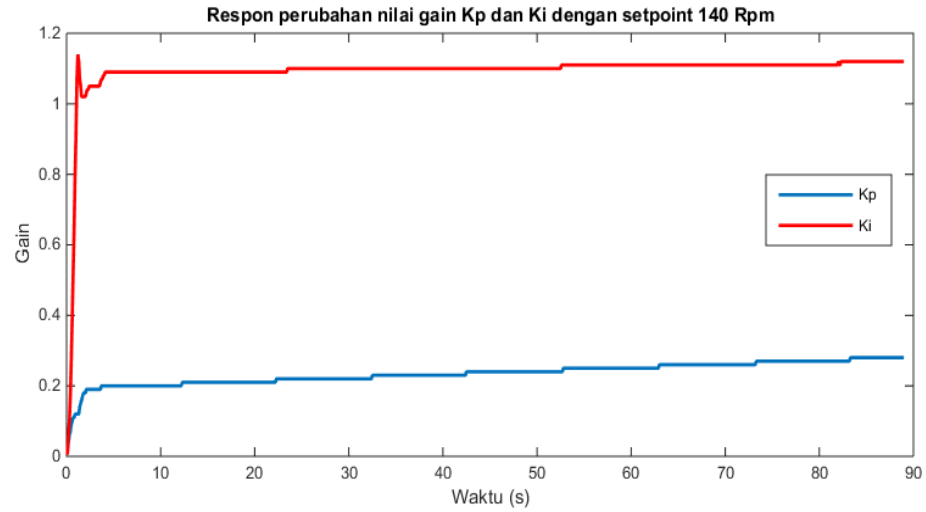
Implementasi dari sistem menggunakan *sampling* setiap 0,1 detik dan 3 nilai *setpoint*, yaitu 100, 120 dan 140 rpm. Data yang diperoleh adalah perubahan nilai Kp dan Ki pada setiap nilai *setpoint* dan data nilai respon kecepatan dari motor BLDC pada setiap *setpoint* dengan hasil perubahan Kp dan Ki. Berikut data yang diperoleh dalam bentuk grafik respon pada Gambar 4.9, 4.10 dan 4.11.



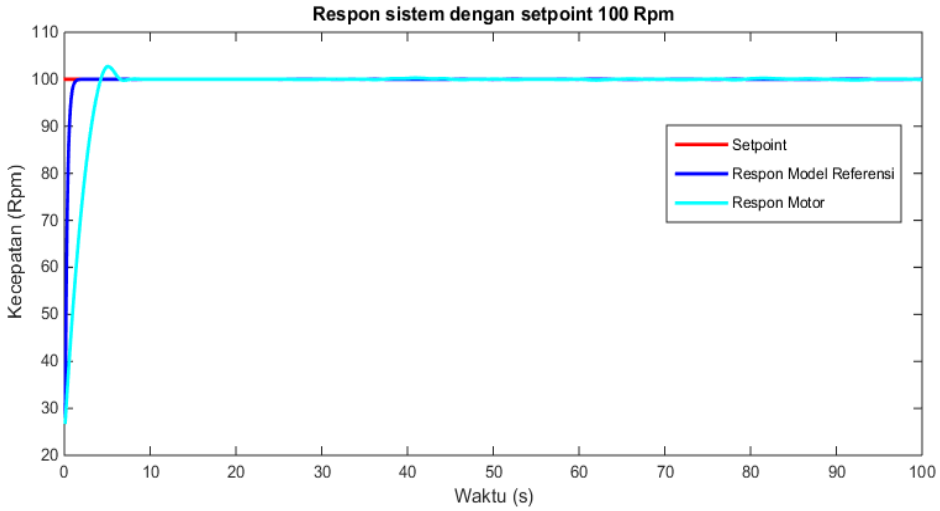
Gambar 4.9 Respon perubahan nilai Kp dan Ki *setpoint* 100 rpm



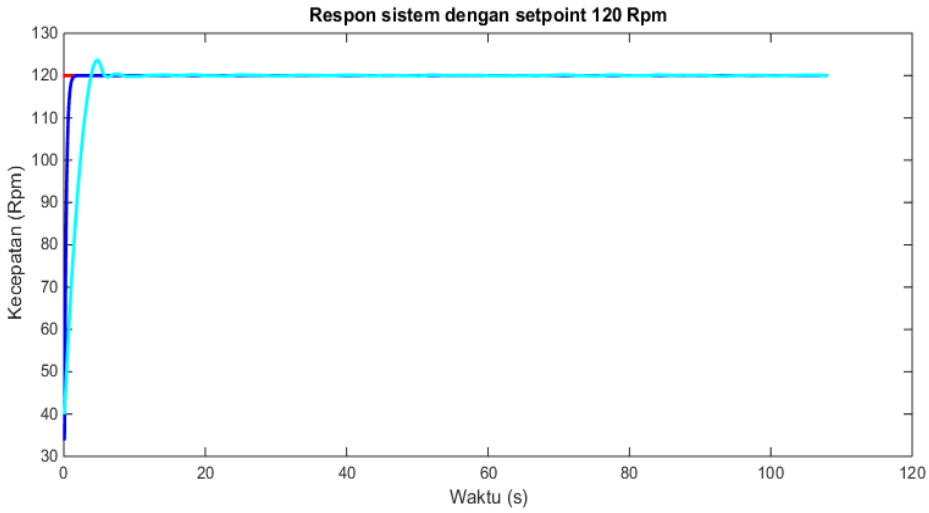
Gambar 4.10 Respon perubahan nilai Kp dan Ki *setpoint* 120 rpm



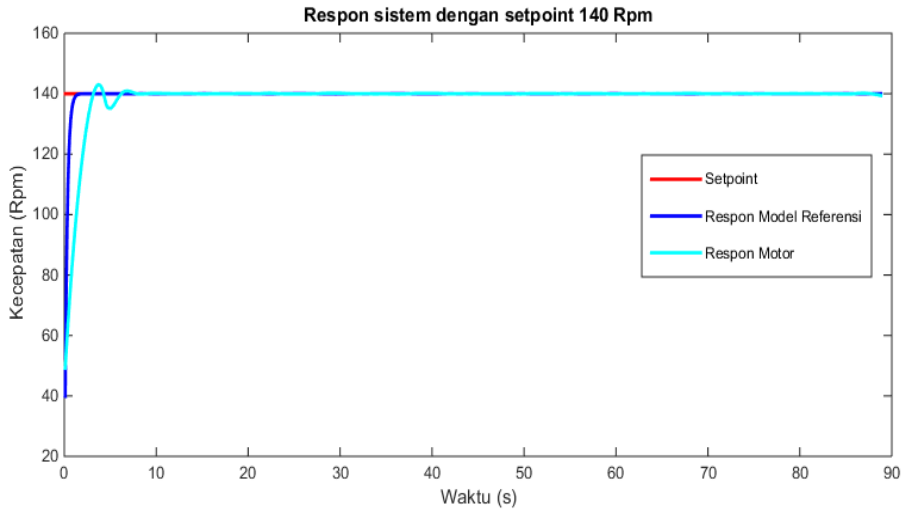
Gambar 4.11 Respon perubahan nilai Kp dan Ki *setpoint* 140 rpm



Gambar 4.12 Respon sistem dengan *setpoint* 100 rpm



Gambar 4.13 Respon sistem dengan *setpoint* 120 rpm



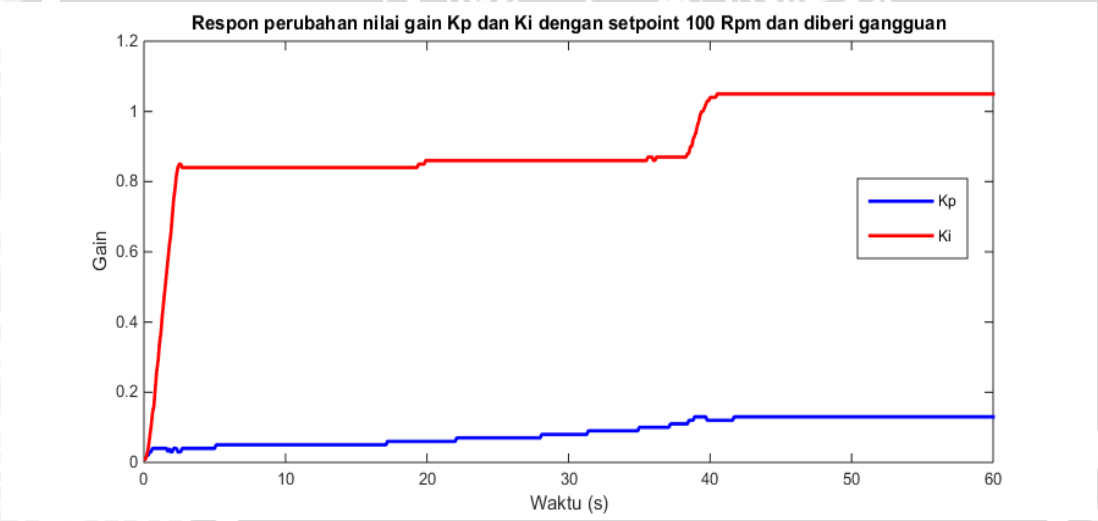
Gambar 4.14 Respon sistem dengan *setpoint* 140 rpm

Dari ketiga grafik (Gambar 4.9, 4.10 dan 4.11), nilai perubahan Kp dan Ki saat motor mulai dinyalakan mengalami kenaikan nilai yang tinggi. Sedangkan ketika motor sudah mencapai keadaan *steady state*, kenaikan nilai Kp dan Ki secara perlahan.

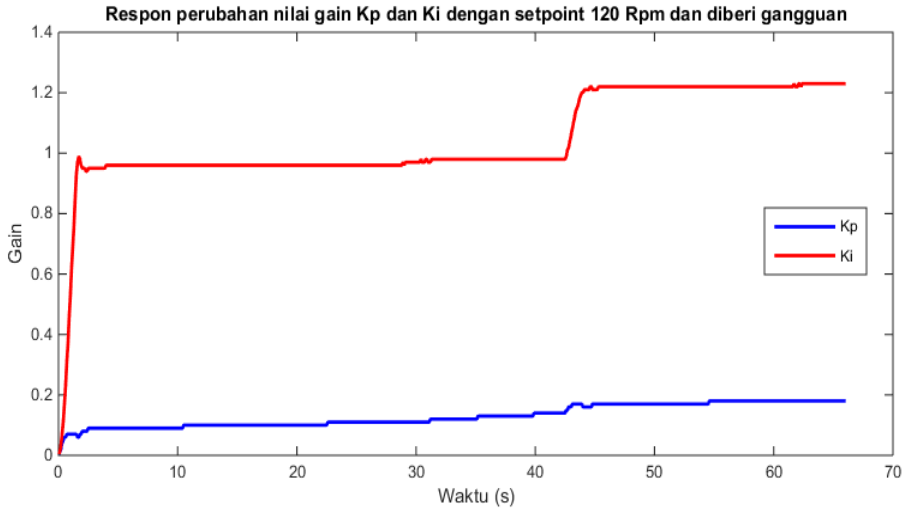
Pada respon sistem (Gambar 4.12, 4.13 dan 4.14) dengan nilai *setpoint* 100 rpm memilki nilai *maximum overshoot* sebesar 3%, nilai *setling time* adalah 6 detik dengan nilai *error steady state* rata-rata berada dibawah toleransi 2% yaitu 1%. Respon sistem dengan nilai *setpoint* 120 rpm memiliki nilai maximum *overshoot* sebesar 2,5%, nilai *setling time* adalah 5,4 detik dengan nilai *error steady state* rata-rata berada dibawah toleransi 2% yaitu 0,83%. Respon sistem dengan nilai *setpoint* 140 rpm memiliki nilai maximum *overshoot* sebesar 2,14%, nilai *setling time* adalah 6 detik dengan nilai *error steady state* rata-rata berada dibawah toleransi 2% yaitu 0,71%.

4.2.2 Hasil Implementasi dengan Diberikan Gangguan

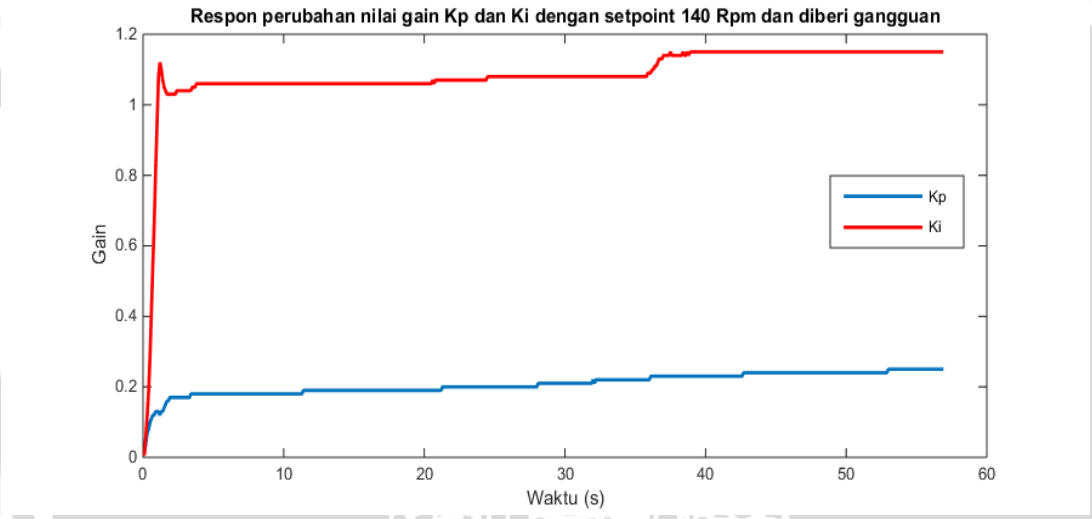
Pada implementasi sistem dengan memberikan gangguan pada saat sistem sedang berada pada keadaan *steady state*, maka akan didapatkan grafik respon Kp dan Ki dan respon sistem seperti pada Gambar 4.15, 4.16 dan 4.17.



Gambar 4.15 Respon perubahan nilai Kp dan Ki *setpoint* 100 rpm dan diberi gangguan

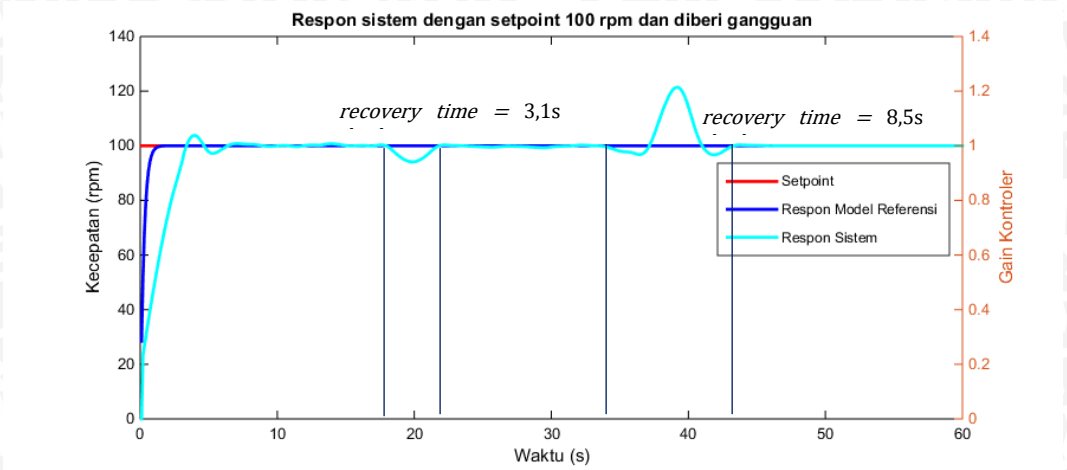


Gambar 4. 16 Respon perubahan nilai Kp dan Ki *setpoint* 120 rpm dan diberi gangguan

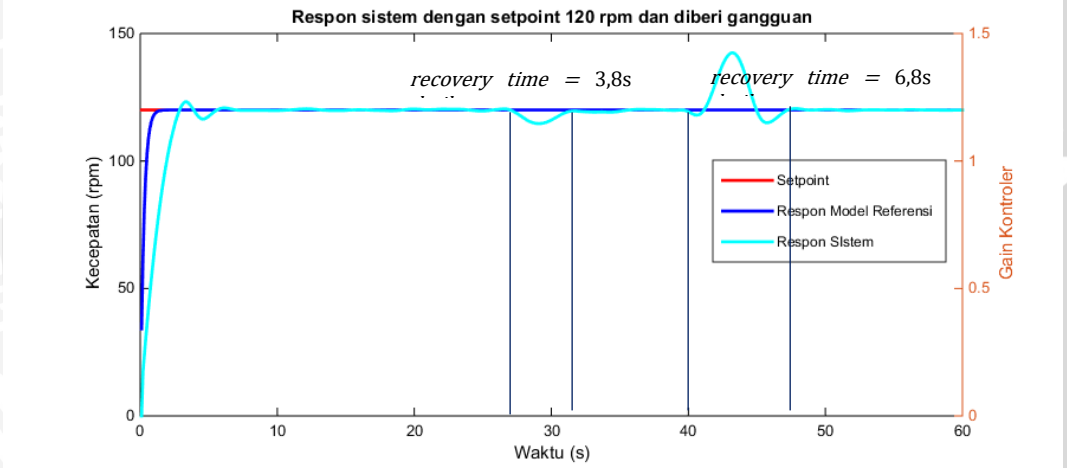


Gambar 4. 17 Respon perubahan nilai Kp dan Ki *setpoint* 140 rpm dan diberi gangguan

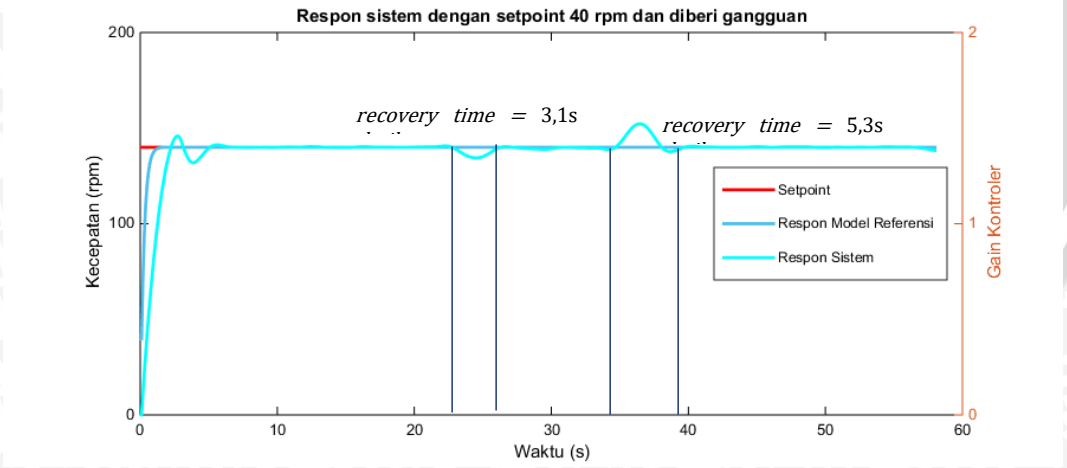
Pada respon sistem (Gambar 4.18, 4.19 dan 4.20) dengan diberikan gangguan pada *setpoint* 100 rpm, respon akan mengalami perlambatan dan *recovery time* respon adalah 3,1 detik. Ketika gangguan yang diberikan dilepaskan, respon akan mengalami percepatan dan *recovery time* respon adalah 8,5 detik. Pada *setpoint* 120 rpm, respon akan mengalami perlambatan dan *recovery time* respon adalah 3,8 detik. Ketika gangguan yang diberikan dilepaskan, respon akan mengalami percepatan dan *recovery time* respon adalah 6,8 detik. Pada *setpoint* 140 rpm, respon akan mengalami perlambatan dan *recovery time* respon adalah 3,1 detik. Ketika gangguan yang diberikan dilepaskan, respon akan mengalami percepatan dan *recovery time* respon adalah 5,3 detik.



Gambar 4. 18 Respon sistem dengan *setpoint* 100 rpm dan diberi gangguan

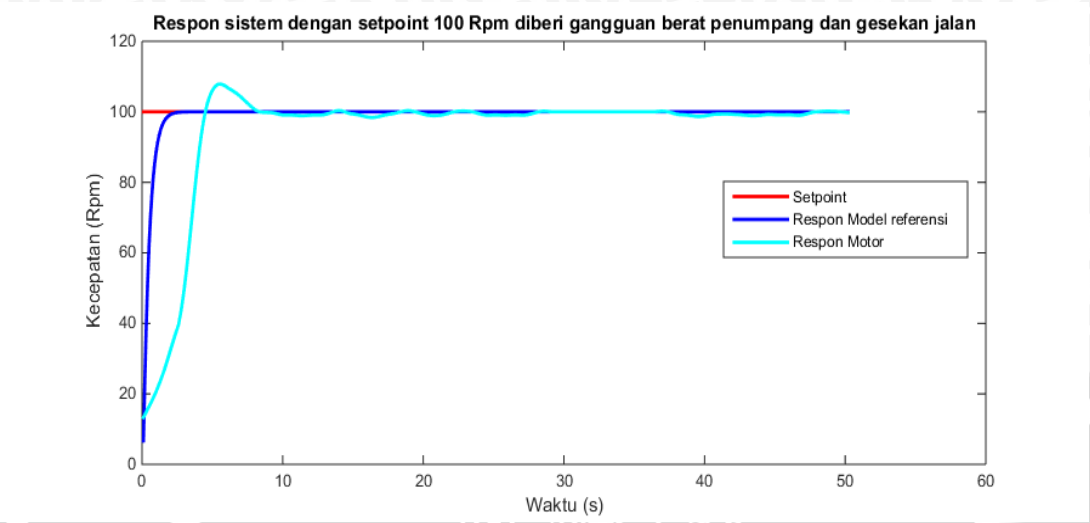


Gambar 4. 19 Respon sistem dengan *setpoint* 120 rpm dan diberi gangguan



Gambar 4. 20 Respon sistem dengan *setpoint* 140 rpm dan diberi gangguan

Pada respon sistem dengan diberikan gangguan berat badan penumpang dan gesekan jalan pada *setpoint* 100 rpm dapat dilihat dalam Gambar 4.21. Respon memilki nilai *maximum overshoot* sebesar 8%, nilai *setling time* adalah 7,7 detik dengan nilai *error steady state* rata-rata berada dibawah toleransi 2% yaitu 1,8%.



Gambar 4. 21 Respon sistem dengan diberikan gangguan jalan dengan *setpoint* 100 rpm

BAB V
KESIMPULAN DAN SARAN

5.1 KESIMPULAN

Kesimpulan dari perancangan dan analisis adalah sebagai berikut:

1. Didapatkan hasil identifikasi *plant* dan parameter kontroler sebagai berikut:
   a. Berdasarkan hasil identifikasi *plant* menggunakan sinyal uji PRBS dan sintaks ident pada *software* MATLAB, fungsi alih motor BLDC memilki *best fit* sebesar 82.49 dan telah divalidasi.
   b. Berdasarkan hasil simulasi didapatkan nilai parameter kontrol yang memenuhi kestabilan respon sistem adalah $\gamma_p = 0{,}0001$ dan $\gamma_i = 0{,}0009$.
2. Didapatkan beberapa hasil implementasi sistem sebagai berikut:
   a. Berdasarkan hasil implementasi tanpa gangguan, respon sistem dengan nilai *setpoint* 100 rpm, 120 rpm dan 140 rpm memiliki nilai *error steady state* rata-rata berada dibawah toleransi 2% dengan nilai masing-masing adalah 1%, 0,83%, dan 0,71%. Sedangkan rata-rata *settling time* pada setiap *setpoint* adalah 6 detik.
   b. Berdasarkan hasil implementasi saat sistem diberi gangguan, pada *setpoint* 100 rpm, 120 rpm dan 140 rpm, respon akan mengalami perlambatan dan *recovery time* respon kembali pada keadaan *steady state* dengan waktu masing-masing adalah 3,1 detik, 3,8 detik dan 3,1 detik. Ketika gangguan yang diberikan dilepaskan, respon akan mengalami percepatan dan *recovery time* respon masing-masing adalah 8,5 detik, 6,8 detik dan 5,3 detik.
   c. Berdasarkan hasil implementasi saat sistem dikendarai dengan *setpoint* 100 rpm didapatkan respon memilki nilai *maximum overshoot* sebesar 8%, nilai *setling time* adalah 7,7 detik dengan nilai *error steady state* 1,8%.

5.2 SARAN

Saran yang dapat diberikan untuk penelitian selanjutnya adalah dapat mengimplementasikan ke dalam kendaraan listrik yang lebih besar dengan motor *Brushless Direct Current* (BLDC) sebagai penggerak sistem, dan menggunakan rangkaian elektronik yang lebih baik untuk efisiensi dan performansi sistem yang lebih baik.

DAFTAR PUSTAKA

Abidin M.F.Z., 2011, *A Comparative Study of PI, Fuzzy and Hybrid PI Fuzzy Controller for Speed Control of Brushless DC Motor Drive,* Malaysia, IEEE, *2011 International Conference on Computer Applications and Industrial Electronics (ICCAIE 2011*).

Ali, A. T.Eisa, B.,Omar, B. 2012. *Adaptive PID Controller for DC Motor Speed Control.* International Journal of Engineering Inventions 1(5), 26-30.

Astrom, K.J. dan B. Wittenmark. 1995. *Adaptive Control*. Addison-Wesley Publishing Company, Inc. USA.

Butler, H. 1992. Model Reference Adaptive Systems, From Theory to Practice. UK: Prentice-Hall, Inc.

Dharmawan, A. 2009. Pengendalian Motor Brushless DC Dengan Metode PWM Sinusoidal Menggunakan ATMega16. *Skripsi*. Tidak dipublikasikan. Depok: Universitas Indonesia.

Hakim, L. 2015. *Implementasi perhitungan robot dengan FPGA menggunakan rotary encoder*. Surabaya: Skripsi Teknik Elektro Institut Teknologi Sepuluh November Surabaya.

Hamdi, I.T. 2015. *Rancang Bangun Three Phase Six Step Pwm Inverter Sebagai Pedal Assisted System (PAS) Sepeda Listrik*. *Skripsi*. Tidak dipublikasikan. Malang: Universitas Brawijaya.

Kristiyono, R. 2015. *Sistem Kendali Kecepatan Motor BLDC menggunakan Algoritma Hybrid PID Fuzzy*. *University Research Colloquium* 2015.

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

Pirabakaran, K. dan V. M. Becerra. 2001. *Automatic Tuning of PID Controllers Using Model Reference Adaptive Control Techniques.* IEEE: *Industrial Electronics Society.*

Rozi, F. 2015. *Tuning kontrol PI dengan menggunakan teknik Model Reference Adaptive Control (MRAC) pada sistem kontrol kecepatan motor DC*. *Skripsi.* Tidak dipublikasikan. Malang: Universitas Brawijaya.

Shyam A., 2013, *A Comparative Study on the Speed Response of BLDC Motor Using Conventional PI Controller, Antiwindup PI Controller and Fuzzy Controller*, India, IEEE, 2013 International Conference on Control Communication and Computing (ICCC).

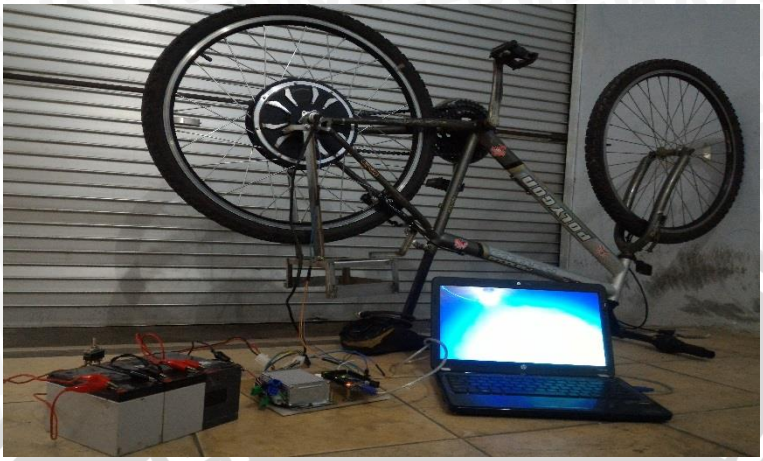
Wain, Y. Suban. https://asro.wordpress.com/2009/01/16/diskritisasi.(diakses pada 25 Juli 2016).

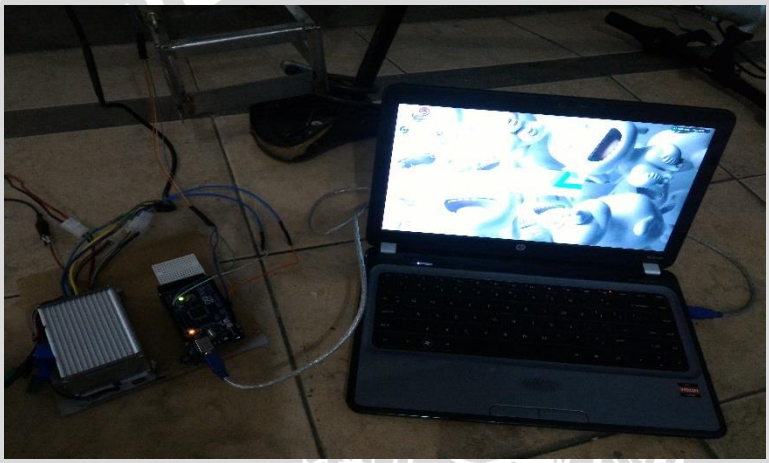
Xiong, Ai dan Yogkun Fan. 2007. *Application of a PID Controller using MRAC Techniques for Control of the DC Electromotor Drive*. IEEE: *International Conference on Mechatronis and Automation.*

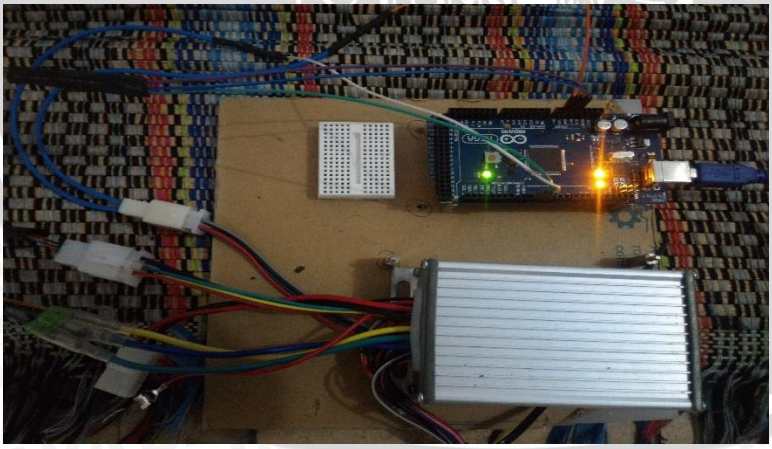
LAMPIRAN I

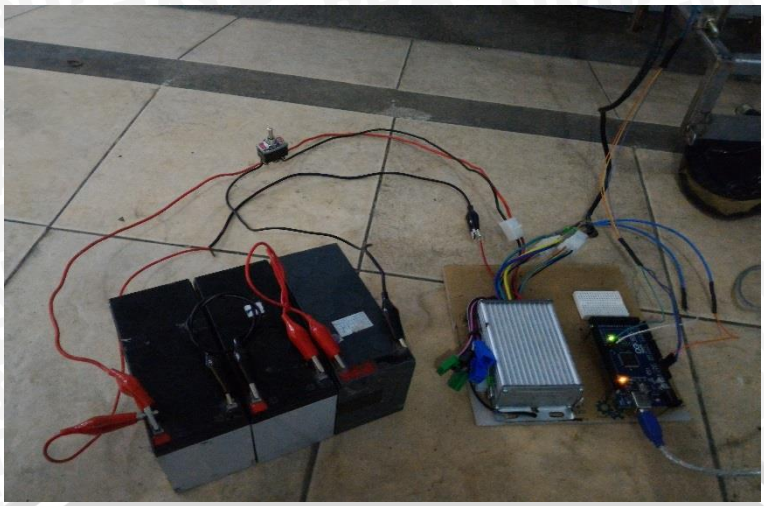
FOTO ALAT

FOTO ALAT










Mikrokontroler Arduino Mega 2560
Baterai
driver motor tiga fasa
motor BLDC
rotary encoder

LAMPIRAN II

LISTING PROGRAM

Listing program M-File simulasi

```
clear all
clc
%%%%%PENENTUAN MODEL REFERENSI%%%%%
betha       = 307.3;
alpha_1     = 71.87;
alpha_2     = 583.7;
alpha_3     = 1291;
%%%%%PENENTUAN PARAMETER GAMMA%%%%%
gamma_p = 0.0001; gamma_i = 0.0009;
```

Listing program Sitem kontrol adaptif dengan MRAC

```
/*************************
Baca Putaran ==> PIN2
PWM ==> PIN3
-*************************/
#define pwm 3


int set_point;              // Set point
float y;                    // Nilai kecepatan Motor BLDC
float error;                // set_point - y
float last1_error;          // error sebelumnya
int last1_set_point;        // Set point sebelumnya
float ym;                   // Nilai keluaran Model referensi
float tracking_error;       // y - ym
float last1_ym;             // Nilai keluaran Model referensi sebelumnya
float last2_ym;             // Nilai keluaran Model referensi sebelumnya sebelumnya
float last3_ym;             // Nilai keluaran Model referensi sebelumnya sebelumnya sebelumnya
float model_p;              // Nilai keluaran Model di P
float last1_model_p;        // Nilai keluaran Model di P sebelumnya
float last2_model_p;        // Nilai keluaran Model di P sebelumnya sebelumnya
float last3_model_p;        // Nilai keluaran Model di P sebelumnya sebelumnya sbelumnya
float model_i;              // Nilai keluaran Model di I
float last1_model_i;        // Nilai keluaran Model di I sebelumnya
float last2_model_i;        // Nilai keluaran Model di I sebelumnya sebelumnya
float last3_model_i;        // Nilai keluaran Model di I sebelumnya sebelumnya sbelumnya
volatile int pulsa;         // pulsa rotary encoder
float gamma_p;              // parameter kendali P
float gamma_i;              // parameter kendali I
float Ts;                   // Time Sampling Model Referensi
float A, B, C, D, E, F;     // Konstanta Model
float G, H, I, J, K;        // Konstanta Model p
float L, M, N, O, P;        // Konstanta Model I
float pwmMotor;             // PWM Motor
float T, U;                 // Batas atas dan bawah sinyal kontrol
float Prop;                 // Var sementara P
float Intg;                 // Var sementara I
float last1_I;              // Var sementara I sebelumnya
double MV;                  // Nilai keuaran PI
double last1_MV;            // Nilai keuaran PI sebelumnya
float Kp;                   // Nilai konstanta Kp
float last1_Kp;             // Nilai konstanta Kp sebelumnya
float Ki;                   // Nilai konstanta Ki
float last1_Ki;             // Nilai konstanta Ki sebelumnya
float integrator;           // Nilai integrator Kontroler I
float last1_integrator;     // Nilai integrator Kontroler I sebelumnya
float R, S;                 // Hasil perkalian model dengan tracking error
int as,ap;
```

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

 Ts = 0.1;
 gamma_p = 0.0001;
 gamma_i = 0.0009;

 //Konstanta Model


 A = 3+(143.74*Ts)+(583.75*Ts*Ts);
 B = 3+(71.87*Ts);
 C = 1;
 D = (307.3*Ts*Ts)+(1291*Ts*Ts*Ts);
 E = (307.3*Ts*Ts);
 F = (307.3*Ts*Ts*Ts);
 G = 1+(71.87*Ts)+(583.75*Ts*Ts)+(1291*Ts*Ts*Ts);


 last1_ym = 0;  last2_ym = 0;        last3_ym = 0;  last1_set_point = 0;
 last1_error = 0; last1_integrator = 0;
 last1_model_p = 0;  last2_model_p = 0;  last3_model_p = 0;
 last1_model_i = 0;  last2_model_i = 0;  last3_model_i = 0;
 last1_Kp = 0;  last1_Ki = 0;

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

 OCR1A   = 6250;          // compare match register 16MHz/256/50Hz/50ms
 TCCR1B |= (1 << WGM12);   // CTC mode
 TCCR1B |= (1 << CS12);    // 256 prescaler
 TIMSK1 |= (1 << OCIE1A);  // enable timer compare interrupt


 TCCR3B = TCCR3B & B11111000 | B00000001; //32k|  B00000010;  4k|
 attachInterrupt(0, hitung_pulsa, FALLING);
 interrupts();             // enable all interrupts
 Serial.begin(9600);
}


ISR(TIMER1_COMPA_vect)          // timer compare interrupt service routine
{
 set_point=100;
 MRAC();

  y = (pulsa*600)/36;
  pulsa = 0;
  error =  set_point - y;
  tracking_error = y - ym;
  Tuning_Kp();
  Integrator();
  Tuning_Ki();
  Kontroler();


  analogWrite(pwm,pwmMotor);
  Serial.print(Kp);
  Serial.print("\t");
  Serial.print(Ki);
  Serial.print("\t");
  Serial.print(set_point);
```

```
  Serial.print("\t");
  Serial.print(ym);
  Serial.print("\t");
  Serial.print(y);
  Serial.print("\t");
  Serial.print("\n");
}

void MRAC()
{
 ym = ((A/G)*last1_ym) - ((B/G)*last2_ym) + ((C/G)*last3_ym) + ((D/G)*set_point)- ((E/G)*last1_set_point);
 last1_set_point = set_point;
 last3_ym = last2_ym;
 last2_ym = last1_ym;
 last1_ym = ym;
}

void Tuning_Kp()
{
 model_p = ((A/G)*last1_model_p) - ((B/G)*last2_model_p) + ((C/G)*last3_model_p) + ((E/G)*error) -
((E/G)*last1_error);
 last3_model_p = last2_model_p;
 last2_model_p = last1_model_p;
 last1_model_p = model_p;
 last1_error = error;
 R = model_p*tracking_error;
 Kp = last1_Kp + ((-gamma_p)*Ts*R);
 last1_Kp = Kp;
}

void Tuning_Ki()
{
 model_i = ((A/G)*last1_model_i) - ((B/G)*last2_model_i) + (((C/G)*last3_model_i) + ((F/G)*error);
 last3_model_i = last2_model_i;
 last2_model_i = last1_model_i;
 last1_model_i = model_i;
 S = model_i*tracking_error;
 Ki = last1_Ki + ((-gamma_i)*Ts*S);
 last1_Ki = Ki;
}

void Integrator()
{
 integrator = last1_integrator + (Ts * error);
 last1_integrator = integrator;
}
void Kontroler()
{
 Prop = Kp * error;
 Intg = Ki * integrator;
 MV = Prop + Intg;

 if (MV<80)pwmMotor = 80;
 else if (MV>160)pwmMotor = 160;

 else pwmMotor = MV;

}


void loop()
{
```

```
}

void hitung_pulsa()
{
  pulsa++;
}
```

LAMPIRAN III

DIAGRAM BLOK

Diagram Blok SIMULINK Subsistem Kontroler P

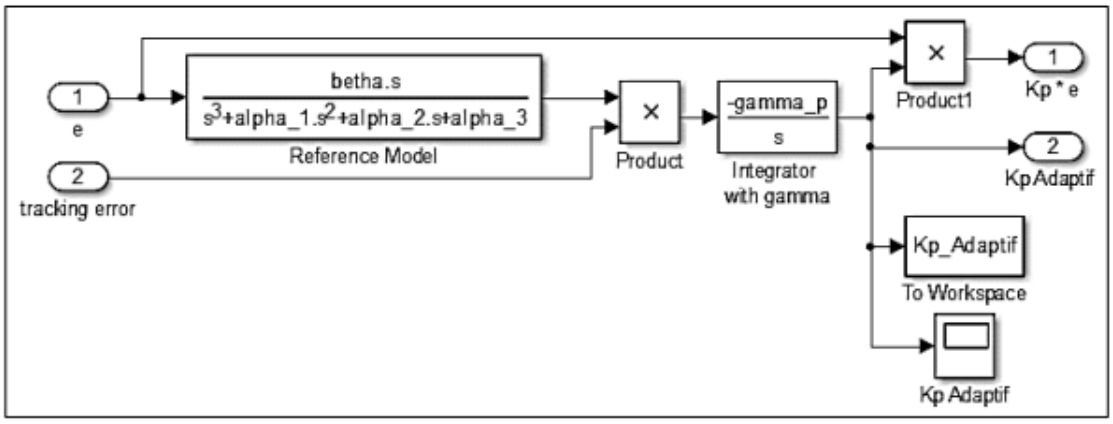


Diagram Blok SIMULINK Subsistem Kontroler I

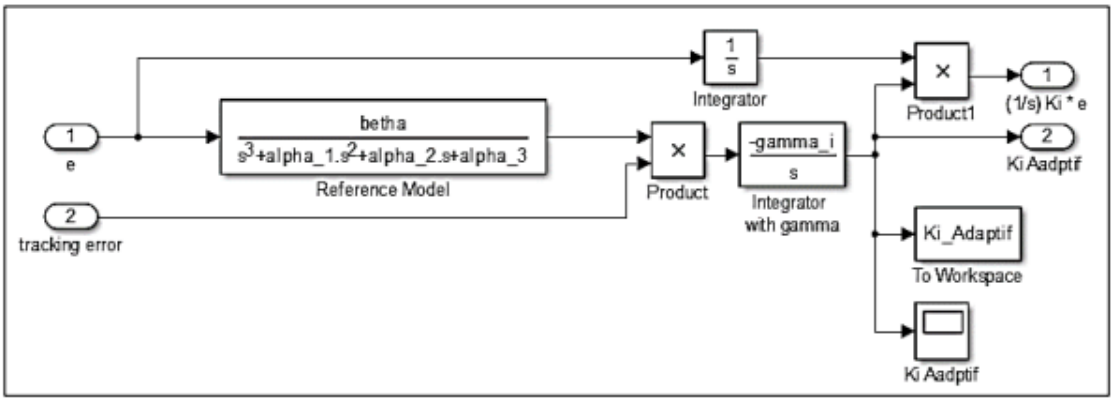
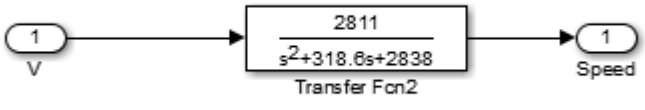


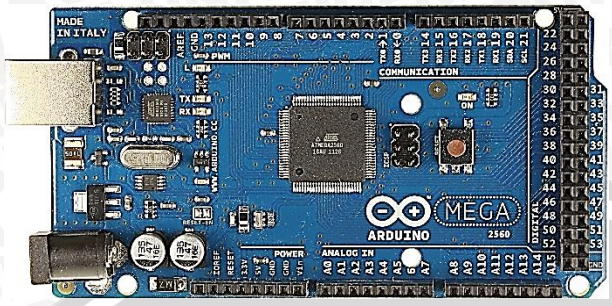
Diagram Blok SIMULINK Subsistem Motor BLDC



LAMPIRAN IV

DATASHEET

Arduino Mega 2560



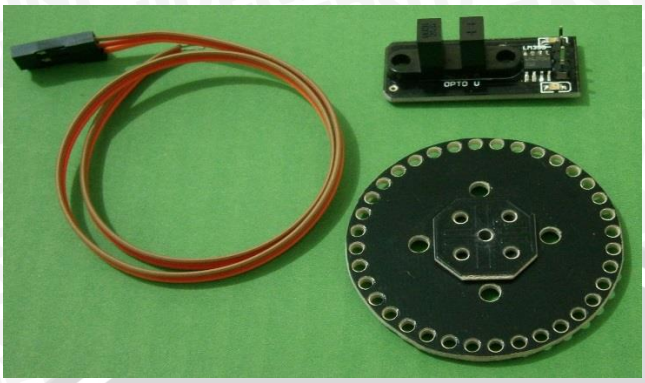
Technical specs

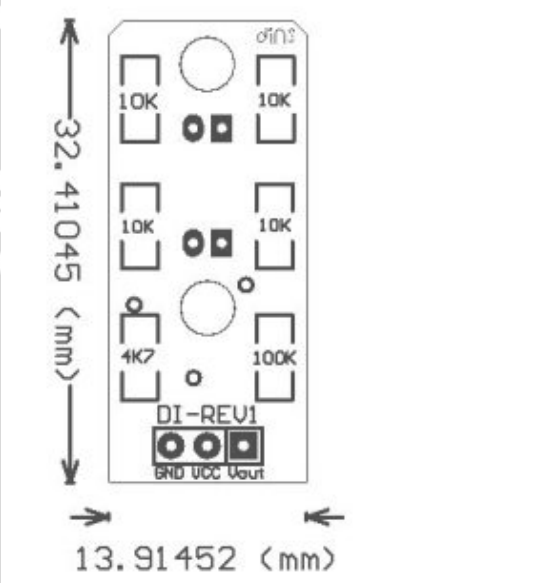
Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	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:

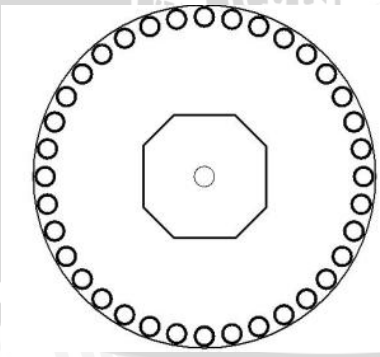
- *Serial*: 0 (RX) dan 1 (TX); *Serial* 1: 19 (RX) dan 18 (TX); *Serial* 2: 17 (RX) dan 16 (TX); *Serial* 3: 15 (RX) dan 14 (TX). Digunakan untuk menerima (RX) dan mengirimkan (TX) TTL data serial. Pin ini dihubungkan ke pin yang berkaitan dengan chip Serial ATmega8U2 USB-to-TTL.
- *Eksternal interrupts*: 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





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