TUGAS AKHIR - TE 141599

PERANCANGAN DAN IMPLEMENTASI SISTEM PENGATURAN KECEPATAN MOTOR ARUS SEARAH TANPA SIKAT MENGGUNAKAN METODE PID-*ROBUST*

Ahmad Fachrudin Istiananda
NRP 2212100181

Dosen Pembimbing
Ir. Rusdhianto Effendie A.K., M.T.
Andri Ashfahani, S.T., M.T., M.Sc.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT - TE 141599

SPEED CONTROLLER DESIGN AND IMPLEMENTATION OF BRUSHLESS DIRECT CURRENT MOTOR USING PID-ROBUST METHOD

Ahmad Fachrudin Istiananda
NRP 2212100181

Supervisor
Ir. Rusdhianto Effendie A.K., M.T.
Andri Ashfahani, S.T., M.T., M.Sc.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Industrial Technology
Sepuluh Nopember Insitute of Technology
Surabaya 2016

PERANCANGAN DAN IMPLIMENTASI SISTEM PENGATURAN KECEPATAN MOTOR ARUS SEARAH TANPA SIKAT MENGGUNAKAN METODE PID-*ROBUST*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan Untuk
Memperoleh Gelar Sarjana Teknik Elektro
Pada
Bidang Studi Teknik Sistem Pengaturan
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

LEMBAR PENGESAHAN

Menyetujui

Dosen Pembimbing I,	Dosen Pembimbing II,
Ir. Rusdhianto Effendie A.K., M.T.	**Andri Ashfahani, S.T., M.T., M.Sc.**
NIP. 19570424 198502 1 001	**NIP. 2200 20140 5 003**

SURABAYA
JUNI, 2016

PERANCANGAN DAN IMPLEMENTASI SISTEM PENGATURAN KECEPATAN MOTOR ARUS SEARAH TANPA SIKAT MENGGUNAKAN METODE PID-*ROBUST*

Ahmad Fachrudin Istiananda
2212100181

Dosen Pembimbing I : Ir. Rusdhianto Effendie A.K., M.T.
Dosen Pembimbing II : Andri Ashfahani, S.T., M.T., M.Sc.

ABSTRAK

Energi listrik adalah energi utama yang dibutuhkan bagi peralatan listrik atau energi yang tersimpan dalam arus listrik. Salah satu macam motor listrik adalah motor arus searah tanpa sikat atau yang lebih dikenal *Brushless Direct Current Motor* (BLDCM). BLDCM merupakan suatu jenis motor sinkron yang artinya medan magnet yang dihasilkan oleh stator dan medan magnet yang dihasilkan rotor berputar di frekuensi yang sama. Dalam BLDCM, salah satu hal yang sangat penting untuk dikontrol adalah kecepatan. Permasalahan yang diteliti dan dianalisis pada Tugas Akhir Perancangan dan Implementasi Sistem Pengaturan Kecepatan Motor Arus Searah Tanpa Sikat Menggunakan Metode PID-*Robust* adalah respon kecepatan BLDCM terhadap pembebanan yang berubah-ubah. Pengaturan kecepatan BLDCM dirancang menggunakan Kontroler Proposional, Integral dan Derivatif (PID). Kontroler PID-*Robust* digunakan untuk membantu BLDC pada kondisi pembebanan untuk mempertahankan *setpoint*. Nilai parameter Kp, Ki, dan Kd sebesar 529,9, 0,00000373, dan 0,0624 didapat dari perhitungan *robust* performansi H_∞ dengan metode LMI (*Linear Matrix Inequality*). Setelah dilakukan simulasi, didapatkan bahwa respon *plant* dengan kontroler PID-*Robust* dapat mengikuti model referensi yang diinginkan dengan nilai *rise time* 7,7 untuk beban minimal, 3,75 untuk beban nominal, dan 5,9 untuk beban maksimal serta kuat/*robust* terhadap gangguan/*disturbance.*

Kata Kunci : *Motor, Brushless DC, PID, Robust, Performansi* H_∞

Halaman ini sengaja dikosongkan

SPEED CONTROLLER DESIGN AND IMPLEMENTATION OF BRUSHLESS DIRECT CURRENT MOTOR USING PID-ROBUST METHOD

Ahmad Fachrudin Istiananda
2212100181

Supervisor I : Ir. Rusdhianto Effendie A.K., M.T.
Supervisor II : Andri Ashfahani, S.T., M.T., M.Sc.

ABSTRACT

Electrical energy is primary energy needed to electrical equipment or stored energy in an electric current .One kind of an electric motor is motor direct current without brush or more commonly known brushless direct current motor (BLDCM). BLDCM is another kind of motor synchronous which means magnetic fields resulting by stator and magnetic fields resulting the rotor revolves in the same frequency. The problems investigated and analyzed on this final project of Speed Controller Design and Implementation of Brushless Direct Current Motor Using PID-Robust Method is response to speed BLDCM imposition a changeable. Arrangement speed BLDCM designed use controller Proportional, Integral and Derivatives (PID). Controller PID-Robust used to aid BLDC on condition imposition to maintain setpoint. Value parameter Kp, Ki, and Kd for 529,9, 0,00000373, and 0,0624 obtained from calculation Robust H_∞ performance; with the methods LMI (Linear Matrix Inequality) After simulation, the response plant with controller PID-Robust can follow model reference that needed to value rise time 7,7 to load at minimal, 3,75 to load nominal, and 5,9 to load maximum and strong/robust to disturbance.

Keywords **:** *Motor, Brushless DC, PID, Robust, H_∞Performance*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR NOTASI DAN SIMBOL

$\mathbf{x}$	Vektor *state* x
x_n	*State* ke-*n*
$\dot{x}_n$	Turunan pertama *state* ke-*n*
$\boldsymbol{\mathcal{B}}$	Semesta bilangan bulat
$\boldsymbol{\mathcal{R}}$	Semesta bilangan riil
$\in$	Anggota himpunan
$\subseteq$	Subhimpunan
$\square$	Akhir dari pembuktian
$\Leftrightarrow$	Ekuivalen
$\mathbf{I}$	Matriks identitas
$\mathbf{A}^T$	Transpos Matriks **A**
$\mathbf{A}^{-1}$	Invers matriks **A**
$\det(A)$	Determinan matriks **A**
$\mathbf{A} > 0$	Matriks **A** definit positif
$\mathbf{A} < 0$	Matriks **A** definit negatif
$diag(a_1, \dots, a_n)$	Matriks diagonal dengan a_i merupakan elemen diagonal ke-*i*
j	Bilangan imajiner
γ	Tingkat pelemahan
∞	*Infinity*
$\|\cdot\|_\infty$	∞-*norm*
$\|\cdot\|_2$	L_2-*norm*
sup	Nilai *supremum*
$\sum(\cdot)$	Penjumlahan dari nilai deret
$\prod(\cdot)$	Perkalian dari nilai deret
$\frac{d}{dx}(\cdot)$	Turunan pertama terhadap x
$\frac{\partial}{\partial \mathrm{x}}(\cdot)$	Turunan parsial pertama terhadap x

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Motor listrik digunakan untuk konversi energi listrik menjadi energi mekanis. Motor *Brushed* DC dan BLDC merupakan macam motor listrik. Motor *Brushed* DC dan BLDC masing-masing memiliki kelebihan dan kekurangan. Motor *Brushed* DC memiliki *brushes* sedangkan motor BLDC tidak memiliki *brushes*, dimana *brushes* dapat rusak apabila digunakan secara terus-menerus. Motor *brushes* DC memiliki *electrical noise* lebih besar daripada motor BLDC karena *brushes* pada motor *Brushed* DC memutus dan menghubungkan koneksi aliran arus maka menimbulkan *electrical noise*. Motor *Brushed* DC memiliki sistem pendinginan motor lebih buruk daripada motor BLDC karena posisi elektromagnet berada di rotor maka pendinginan motor lebih sulit. Secara umum, motor BLDC memiliki keuntungan lebih daripada motor *Brushed* DC.[1]

Kontroler PID (*Proportional-Integral-Derivative*) konvensional banyak dijumpai dalam industri karena kesederhanaannya, fungsinya yang jelas, dan kemudahan dalam penerapannya. Telah banyak paper yang membahas tentang kontroler PID dalam penggunaannya sekarang dan inovasi untuk masa depan. Proses mendesain kontroler PID untuk industri otomasi bisa sangat sulit dibuat jika terdapat beberapa keluaran yang ingin dicapai dan saling bertentangan. Hal ini memunculkan pengembangan beberapa metode *tunning* yang tujuannya adalah untuk menemukan inovasi terbaru. Saat ini banyak sistem yang menggunakan kontroler konvensional seperti kontroler PID. Kontroler PID bekerja sangat baik namun hanya dalam keadaan parameter dan beban sistem yang spesifik. Namun, perubahan dari parameter dan beban sistem menyebabkan kinerja sistem *loop* tertutup memburuk, menyebabkan overshoot lebih besar, *rise time* dan *settling time* lebih lama, dan bisa juga membuat sistem tidak stabil. Oleh karena itu dibutuhkan tipe kontroler yang bisa mengatasi nonlinearitas dan bisa beradaptasi dengan perubahan kondisi sistem tanpa harus diprogram ulang dari operator.[2]

Kompensator *robust* akan menjadi metode pendukung untuk kontroler PID dalam penelitian tugas akhir ini dimana kompensator akan didesain untuk mengurangi efek pembebanan pada saat BLDC mencapai suatu kecepatan. Lebih detailnya kompensator *robust* akan mengurangi

sinyal eror yang terjadi agar BLDC dapat mempertahankan kecepatan putar pada kecepatan yang diinginkan.[2]

1.2 Perumusan Masalah

BLDC motor (*Brushless* DC) merupakan motor arus searah tanpa sikat merupakan pengembangan dari motor arus searah namun tanpa sikat sehingga tidak perlu melakukan penggantian sikat. BLDC motor baru-baru ini banyak digunakan dalam dunia industri. Terdapat berbagai pengaturan motor arus searah tanpa sikat diantaranya, pengaturan posisi dan pengaturan kecepatan motor. Terdapat berbagai macam metode kontrol yang dapat digunakan dalam mengatur kecepatan maupun posisi motor.

Motor BLDC akan mengalami perubahan kecepatan apabila motor tersebut diberi beban. Performa BLDC pada saat mempertahankan kecepatan pada kondisi pembebanan masih harus ditingkatkan.Oleh karena itu, dibutuhkan suatu kompensator pada sistem kontrol motor BLDC sehingga mampu mengurangi efek dari pembebanan. Dimana pada saat akselerasi diberikan efek pembebanan maka motor BLDC akan mampu membantu kinerja dari motor BLDC mampu mempertahankan kecepatan yang diinginkan. Dalam tugas akhir ini membahas mengenai kontrol kecepatan pada BLDC motor dengan menggunakan metode PID-*Robust.*

1.3 Batasan Masalah

Permasalahan pada tugas akhir ini dibatasi oleh beberapa hal antara lain:

a. Pembebanan yang dilakukan yaitu beban minimal (16 V), beban nominal (20 V), serta beban maksimal (24 V).
b. Perancangan dan implementasi kontroler PID-*Robust* untuk pengaturan kecepatan motor arus searah tanpa sikat.
c. Nilai parameter Kp, Ki, dan Kd didapatkan dari model referensi yang diinginkan.

1.4 Tujuan Penelitian

Tujuan dari pelaksanaan tugas akhir ini adalah:

a. Mengindentifikasi kinematika dan dinamika BLDC motor sehingga didapatkan *model* matematika sistem
b. Merancang kontroler PID-*Robust*.

c. Mencari nilai parameter Kp, Ki, dan Kd kontroler PID berdasarkan model referensi melalui program LMI yang diinginkan.
d. Menganalisa respon simulasi BLDC motor dengan mensimulasikannya sehingga diketahui tingkat keakuratan dari kontroler

Hasil yang diperoleh dari pelaksanaan tugas akhir ini diharapkan dapat memberikan manfaat dan kontribusi bagi dunia pendidikan, industri dan masyarakat agar dapat dijadikan referensi bagi peneliti lainnya dan sebagai ilmu pengetahuan.

1.5 Sistematika Penulisan

Buku tugas akhir ini terdiri dari lima Bab dan disusun menurut sistematika penulisan berikut ini

BAB I: PENDAHULUAN

Bab ini berisi tentang latar belakang, rumusan masalah, tujuan, batasan masalah, sistematika penulisan, dan relevansi.

BAB 2: DASAR TEORI

Bab ini berisi tentang teori yang menunjang penelitian, berupa teori tentang BLDC dan komponennya, serta metode yang digunakan untuk pengaturan kecepatan motor BLDC.

BAB 3: PERANCANGAN SISTEM DAN KONTROLER

Bab ini berisi tentang perancangan perangkat keras, perangkat lunak, dan perancangan kontroler.

BAB 4: PENGUJIAN DAN ANALISA

Bab ini berisi tentang hasil simulasi kontroler dan analisanya. Selain itu berisi tentang hasil implementasi kontroler pada Simulator BLDC beserta analisa hasil implementasi.

BAB 5: KESIMPULAN DAN SARAN

Berisi kesimpulan dan saran yang dapat dijadikan pertimbangan pengembangan berdasar hasil pengerjaan tugas akhir ini.

1.6 Relevansi

Hasil dari tugas akhir ini diharapkan dapat dijadikan acuan dalam pembuatan sistem kontrol kecepatan BLDC motor. Dapat dijadikan referensi penelitian lebih lanjut mengenai kontrol kecepatan BLDC motor.

BAB 2
TINJAUAN PUSTAKA

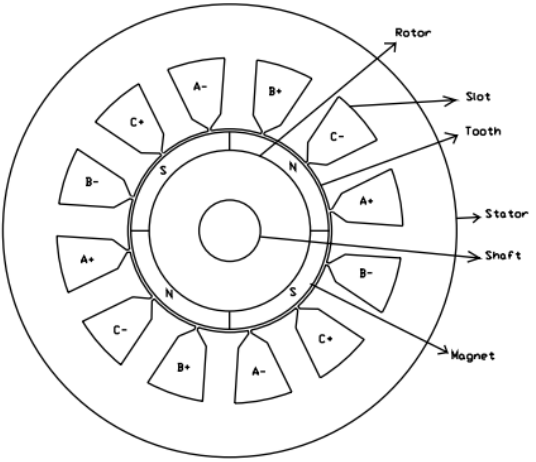
2.1 Motor *Brushless* DC [3]

Motor Brushless DC (BLDC) adalah motor singkron dengan magnet permanen pada rotor dan kumparan jangkar pada stator. Keuntungan paling jelas dari konfigurasi tanpa sikat adalah penghilangan sikat, yang menghilangkan pemeliharaan sikat dan bunga api. Penggunaan kumparan jangkar pada stator membantu konduksi panas pada kumparan. Karena tidak ada kumparan pada rotor, kerugian listrik dalam rotor menjadi minimal. Motor BLDC lebih disukai dibandingkan dengan motor induksi untuk daerah daya sebagian-sebagian. Motor BLDC memiliki efisiensi yang dan faktor daya yang lebih baik dan oleh karena itu, daya keluaran lebih besar untuk kerangka yang sama, karena penguatan medan dipengaruhi oleh magnet permanen dan tidak harus disuplai oleh arus jangkar. Keuntungan-keuntungan yang didapat dari motor BLDC disertai dengan peningkatan kompleksitas di kontroler elektronik dan perlunya penginderaan posisi poros. Eksitasi magnet permanen (PM) lebih layak untuk motor kecil, biasanya di bawah 20 kW. Untuk motor yang lebih besar, biaya dan berat magnet menjadi terlalu tinggi, dan itu akan membuat lebih masuk akal untuk memilih eksitasi dengan elektromagnet. Namun, dengan pengembangan bahan magnet permanen medan tinggi, motor PM dengan kapasitas beberapa megawatt telah dibangun.

2.1.1 Konstruksi

2.1.1.1 *Magnet Permanen*

Pada motor BLDC, magnet permanen dipasang pada rotor. Magnet dibangun dalam bentuk melengkung dan ditempel pada permukaan rotor dengan kutub pole yang berlawanann memiliki polaritas magnet yang berbeda seperti ditunjukkan pada Gambar 2.1.



Gambar 1.1 Motor BLDC 3-fasa dengan 4 pole rotor dan 12 slot stator.

2.1.1.2 *Gulungan Stator*

Motor BLDC sering dianggap memiliki tiga fasa, tetapi hal ini tidak selalu terjadi. Motor kecil untuk aplikasi ringan seperti kipas pendingin lebih murah pembiayaannya jika dibangun dengan satu atau dua fasa. Di sisi lain, pada aplikasi berat dengan rating megawatt, akan lebih baik jika menggunakan motor dengan jumlah fasa yang banyak. Motor dengan 15 fasa telah dibangun untuk propulsi kapal.

Jumlah stator dipilih tergantung pada kutub-kutub rotor, jumlah fasa, dan konfigurasi gulungan. Secara umum, desain slot/kutub fraksional banyak dipilih untuk meminimalkan torsi cogging.

2.1.1.3 *Karasterisik Motor*

Bentuk gelombang *air-gap-flux-density* pada dasarnya adalah sebuah gelombang persegi, tapi fringing menyebabkan bentuk gelombangnya menjadi sedikit melengkung. Saat rotor berputar, bentuk gelombang tegangan yang terinduksi pada tiap fasa terhadap waktunya merupakan replika dari bentuk gelombang *air-gap-flux-density* terhadap posisi rotor. *Fringing* menyebabkan bentuk gelombang *back-emf* berbentuk trapezoidal. Bentuk gelombang trapezoidal inilah yang membedakan motor BLDC dengan *permanent magnet synchronous motor* (PMSM), yang memiliki bentuk gelombang *back-emf* berupa

sinusoidal. Dengan memberikan arus *rectangular* pada tiap fasa yang *back-emf*-nya berada pada keadaan nilai penuh, dimungkinkan untuk mendapatkan torsi motor BLDC yang hampir konstan.

Tegangan *back-emf* dan gelombang arus fasa 120^0 ideal untuk motor BLDC tiga fasa ditunjukkan pada Gambar 2.2. Switch inverter yang aktif di tiap interval 60^0 ditunjukkan pada Gambar 2.3 dan Gambar 2.4.

Amplitudo dari tegangan *back-emf* sebanding dengan kecepatan rotor dengan persamaan

$$E = k\phi\omega_m \tag{2.1}$$

Dimana k adalah konstanta yang tergantung pada jumlah lilitan di tiap fasa, ϕ adalah fluks magnet permanen, dan ω_m adalah kecepatan mekanis.

Selama tiap interval 120^0, daya sesaat yang dikonversi dari elektris ke mekanis adalah

$$P_o = \omega_m T_e = 2EI \tag{2.2}$$

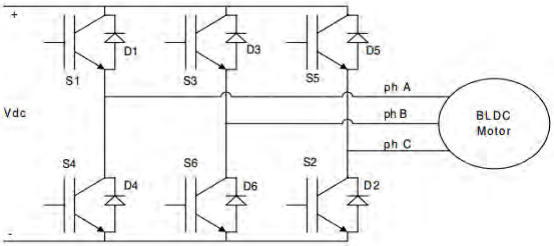
dimana T_e adalah torsi keluaran dan I adalah amplitudo arus fasa. Dari Persamaan (2.1) dan (2.2), persamaan torsi keluaran dapat ditulis

$$T_e = 2k\phi I = k_t I \tag{2.3}$$

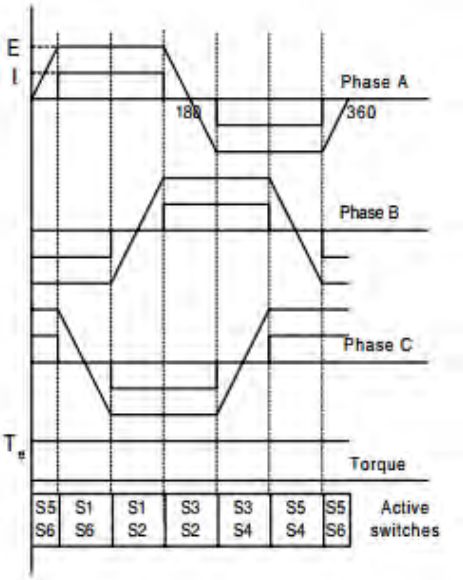
dimana k_t adalah konstanta torsi

Prinsip kerja Motor BLDC sebenarnya sama dengan motor listrik DC konvensional. Perbedaan hanya terletak pada penggunaan *brush* (sikat). Pada motor DC konvensional, sikat dan komutator mekanik digunakan dalam proses komutasi. Sedangkan motor BLDC sudah menggunakan teknologi elektronik dalam proses komutasinya, yaitu *sensor* Hall dan kontroler. [15]

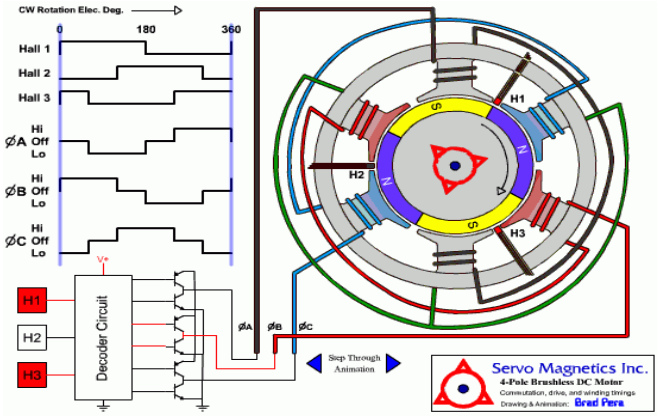
Dan untuk lebih jelasnya, skema kerja dari motor BLDC dapat dilihat pada Gambar 2.2



Gambar 1.2 Skematik dari inverter berbasis IGBT untuk motor BLDC fasa



Gambar 1.3 Gelombang back-emf dan arus fasa untuk motor BLDC tiga fasa dengan arus bipolar 120^0



Gambar 1.4 Skema Kerja *Brushless* DC Motor [15]

2.1.2 Motor BLDC Daikin

Motor yang digunakan pada tugas akhir ini merupakan motor yang diambil dari sebuah *air-conditioner* buatan Daikin. Bentuk fisik motor tersebut dapat dilihat pada Gambar 2.5. Motor ini memiliki built-in driver dengan 5 buah kabel input-output. Masing-masing kabel memiliki warna isolasi berbeda dengan fungsi yang dapat dilihat pada Tabel 2.1. Motor ini memiliki 4 buah pasangan kutub.

Tabel 1.1 Daftar Warna Kabel Input-Output Driver Motor dan Fungsinya

Warna	Keterangan
Jingga	Merupakan kabel input sinyal *control*. Input kabel ini merupakan tegangan dengan rentang 0-5 Volt. Kecepatan putar motor akan berubah sesuai dengan tegangan yang diberikan pada kabel ini.
Putih	Merupakan kabel output sinyal informasi kecepatan motor. Kabel ini men-*generate* sinyal kotak dengan frekuensi sesuai dengan kecepatan putar motor. Hubungan antara frekuensi dan kecepatan motor adalah $\omega = 15 \times f$ Dimana ω adalah kecepatan putar motor dalam rpm dan f adalah frekuensi sinyal dalam hertz.
Merah	Merupakan kabel *power* yang men-*supply* daya ke motor dan *driver*.
Biru	Merupakan kabel *common* dari *driver* dan motor.
Coklat	*Motor adjustment pin*. *Low* untuk kecepatan rendah, high untuk kecepatan tinggi. Pada penelitian ini digunakan masukan *low*.

Gambar 1.5 Penampang Fisik Motor BLDC

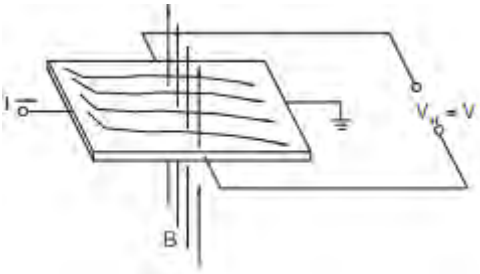
2.2 *Hall Effect Sensor* [4]

2.2.1 Teori *Hall Effect*

Jika sebuah konduktor berarus diletakkan di dalam medan magnet, akan muncul tegangan yang tegak lurus dengan arus dan medan magnet. Prinsip ini disebut dengan efek Hall.

Jika medan magnet tegak lurus dengan arus seperti ditunjukkan pada Gambar 2.6, akan muncul gaya Lorentz yang bekerja pada elektron-elektron konduktor. Gaya ini akan mengganggu distribusi arus, menyebabkan perbedaan potensial (tegangan). Tegangan ini adalah tegangan Hall (V_H). Hubungan dari besar tegangan Hall dengan arus dan medan magnet ditunjukkan pada Persamaan (2.4).

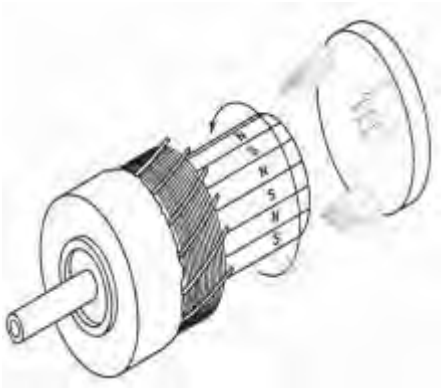
$$V_H \propto I \times B \tag{2.4}$$



Gambar 1.6 Prinsip efek hall

2.2.2 Sensor Hall untuk motor BLDC

Pada motor BLDC, komutasi dilakukan secara elektris. Gambar 2.7 menggambarkan bagaimana komutasi dapat dilakukan menggunakan 3 buah sensor. Sensor mendeteksi posisi angular poros dan menyampaikan informasi tersebut ke sebuah rangkaian logika. Rangkaian logika tersebut kemudian mengolah informasi posisi rotor untuk mengatur switch dari rangkaian driver motor.

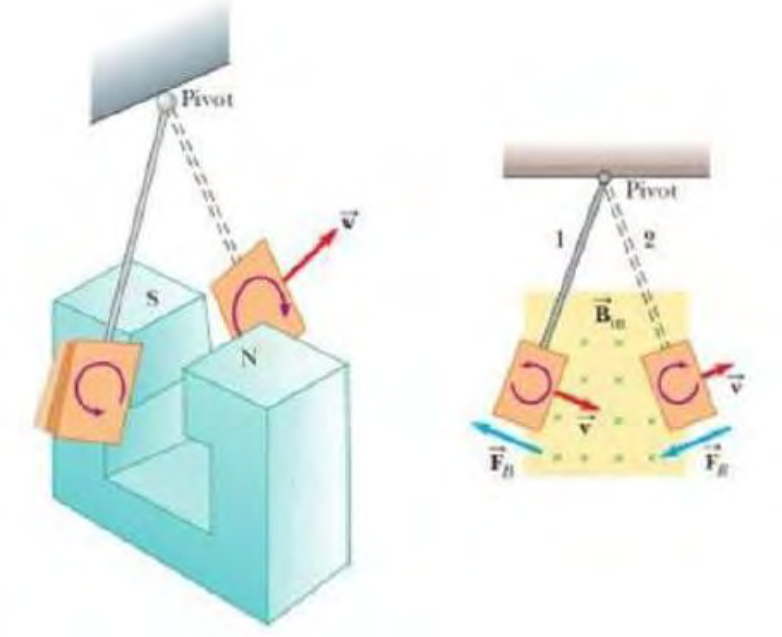


Gambar 1.7 Peletakkan sensor posisi motor BLDC.

2.3 Rem Elektromagnetik

Sistem pengereman ini menggunakan gaya elektromagnetik yang timbul dari suatu magnet permanen tetap atau kumparan yang diberikan arus listrik untuk memperlambat suatu gerakan. Konstruksi dasarnya berupa suatu piringan logam non-feromagnetik yang terpasang pada suatu

poros yang berputar. Piringan logam tersebut diapit oleh kumparan yang dialiri arus listrik hingga menimbulkan medan magnet yang kutubnya saling berlawanan. Logam piringan tersebut akan memotong medan magnet yang ditimbulkan oleh kumparan tersebut sehingga menimbulkan *eddy current* atau arus *eddy*.



Gambar 1.8 Prinsip Arus *Eddy* Pada Logam yang Bergerak [5]

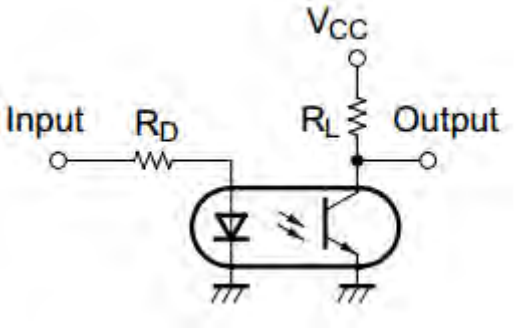
Arus *eddy* merupakan arus listrik yang timbul bilamana suatu piringan logam berada di sekitar medan magnet yang garis-garis gayanya sedang berubah-ubah. Arus *eddy* ini mempunyai medan magnet yang arahnya berlawanan dengan arah gerak piringan logam. Akibatnya laju piringan logam akan tertahan akibat dari adanya arus *eddy* ini.

Rem elektromagnetik biasa diletakkan dekat dengan bagian yang bergerak. Rem ini bekerja pada kondisi yang dingin dan memenuhi persyaratan energi pengereman kecepatan tinggi karena tanpa adanya gesekan. Selain pada kendaraan listrik, aplikasi rem elektromagnetik juga dapat ditemukan pada sistem pengereman kereta api.

2.4 Rangkaian *Optocoupler*

Rangkaian *optocoupler* digunakan untuk memisahkan perangkat arduino dengan *plant* sehingga perangkat arduino dapat terhindar dari kerusakan akibat arus berlebih dari *power supply* jika terjadi kecelakaan.

Rangkaian isolator yang digunakan terdiri dari *optocoupler* PIC817 dengan rangkaian yang dapat dilihat pada Gambar 2.9

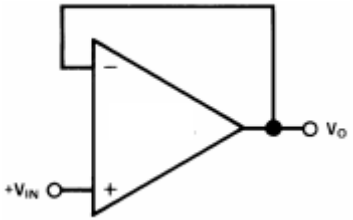


Gambar 1.9 Rangkaian *Optocoupler* [6]

2.4.1.1 *Rangkaian Voltage Follower*

Rangkaian voltage follower berguna agar tidak terjadi pembagian arus yang berakibat pada drop tegangan. Drop tegangan akan menyebabkan kesalahan pada pembacaan sinyal.

Rangkaian voltage follower yang digunakan terdiri dari sebuah op-amp dengan rangkaian yang dapat dilihat pada Gambar 2.10



Gambar 1.10 Rangkaian *Voltage Follower*

2.5 Arduino

2.5.1 Pengertian Arduino [12]

Arduino adalah pengendali mikro *single-board* yang bersifat *open-source*, diturunkan dari *wiring platform*, dirancang untuk memudahkan penggunaan elektronik dalam berbagai bidang. *Hardware*nya memiliki prosesor Atmel AVR dan *software*nya memiliki bahasa pemrograman sendiri. Saat ini Arduino sangat populer di seluruh dunia. Banyak pemula yang belajar mengenal robotika dan elektronika lewat Arduino karena mudah dipelajari. Bahasa yang dipakai dalam Arduino bukan *assembler* yang relatif sulit, tetapi bahasa C yang disederhanakan dengan bantuan pustaka-pustaka (*libraries*) Arduino. Arduino juga menyederhanakan proses bekerja dengan mikrokontroler. Salah satu produk Arduino yang digunakan dalam Perancangan Tugas Akhir ini merupakan Arduino Uno.

Arduino Uno adalah *board* mikrokontroler berbasis Atmega328. Alat ini mempunyai 14 *pin input*/*output* digital dan 6 diantaranya dapat digunakan sebagai *output* PWM, 6 *input analog*, resonator keramik 19MHz, koneksi USB, soket listrik, ICSP *header,* dan tombol *reset*. Dengan kabel USB alat ini mudah dihubungkan ke komputer dan dapat diaktifkan dengan baterai atau adaptor AC ke DC. Tabel 2.2 menjelaskan mengenai spesifikasi Arduino Uno:

Tabel 1.2 Spesifikasi Arduino Uno

Mikrokontroler	Atmega328
Tegangan operasi	5V
Tegangan *input* (direkomendasikan)	7-12V
Tegangan *input* (batasan)	6-20V
***Pin input*/*output* digital**	14 (6 diantaranya *output* PWM)
Pin input analog	6
Arus DC tiap *pin* I/O	40mA
Arus DC untuk *pin* 3,3 V	50mA

2.5.2 *Input*/*Output* Arduino [12]

Input/*output* digital atau digital *pin* adalah *pin-pin* untuk menghubungkan Arduino dengan komponen atau rangkaian digital. Komponen lain yang menghasilkan *output digital* atau menerima *input digital* bisa disambungkan ke *pin-pin* ini.

2.5.3 *Pin-pin* Catu Daya [12]

Pin-pin catu daya adalah *pin* yang memberikan tegangan untuk komponen atau rangkaian yang dihubungkan dengan Arduino. Pada bagian catu daya ini terdapat *pin Vin* dan *reset*.

2.5.4 Soket Baterai [12]

Soket baterai digunakan untuk menyuplai Arduino dengan tegangan dari baterai 9V pada saat Arduino sedang tidak disambungkan ke komputer. Bentuk Arduino Uno sendiri dapat dilihat pada Gambar 2.11

Gambar 1.11 Arduino Uno [12]

2.6 MATLAB [13]

MATLAB merupakan paket program dengan bahasa pemrograman yang tinggi untuk mengembangkan algoritma, visualisasi data, dan komputasi numerik. Program MATLAB ini dapat digunakan untuk menyelesaikan masalah komputasi dengan lebih cepat dibandingkan dengan bahasa pemrograman tradisional, seperti C, C++, dan Fortran. MATLAB digunakan untuk banyak aplikasi seperti *signal and image processing*, desain kontrol, pengujian dan pengukuran, per*mode*lan, dan analisis.

Simulink merupakan bagian dari MATLAB untuk memodelkan, mensimulasikan, dan menganalisa sistem dinamik. *Simulink* dapat membentuk *mode*l dari awal atau memodifikasi *mode*l yang sudah ada sesuai dengan apa yang diinginkan. Selain itu *simulink* juga mendukung sistem *linier* dan *non-linier*, pemodelan waktu kontinyu atau diskrit, atau gabungan. *Simulink* ini dapat digunakan sebagai media untuk menyelesaikan masalah dalam industri nyata meliputi kedirgantaraan dan pertahanan, otomotif, komunikasi, elektronik dan pemrosesan sinyal,

Salah satu modul dalam *Simulink* yang dapat digunakan untuk komunikasi perangkat keras adalah *Instrument Control Toolbox*. Modul ini merupakan kunpulan fungsi *m-file* yang dibangun pada lingkungan komputasi teknis MATLAB. *Toolbox* ini menyediakan kerangka kerja untuk komunikasi instrumen yang mendukung GPIB *interface,* standar VISA, TCP/IP, dan protokol UDP. *Toolbox* ini memperluas fitur dasar *serial port* yang ada dalam MATLAB. Selain itu *toolbox* ini berfugsi untuk komunikasi data antara *workspace* MATLAB dan peralatan lainnya. Data tersebut dapat berbentuk biner atau *text*.

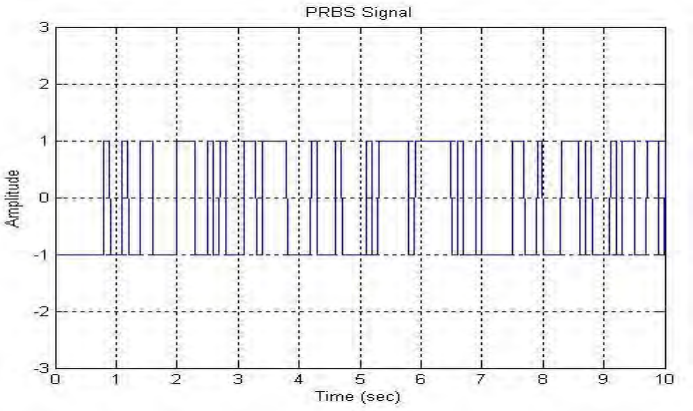
Komunikasi *serial* merupakan protokol dasar tingkat rendah untuk komunikasi antara dua peralatan atau lebih. Pada umumnya satu komputer dengan *mode*m, *printer*, mikrokontroler, atau peralatan lainnya. *Serial port* mengirim dan menerima informasi *bytes* dengan hubungan seri. *Bytes* tersebut dikirimkan menggunakan format biner atau karakter ASCII (*American Standard Code for Information Interchange*). Dalam komunikasi serial MATLAB, agar data ASCII dapat diproses *real time*, maka digunakan ASCII *encode* dan *decode* yang terdapat pada *xPC Target Library for RS232.* ASCII *encode* merupakan blok dalam *simulink* yang digunakan untuk mengubah data *bytes* menjadi karakter ASCII. Sedangkan ASCII *decode* merupakan blok *Simulink* yang digunakan untuk mengubah karakter ASCII menjadi data *bytes* yang kemudian dapat dikonversi sesuai kebutuhan.

2.7 Identifikasi Sistem [14]

Identifikasi merupakan suatu metode untuk mendapatkan parameter-parameter dari suatu sistem berdasarkan data hasil pengukuran masukan atau keluaran dari suatu *plant*. Parameter–parameter yang diperoleh digunakan untuk mendapatkan model dari suatu sistem atau *plant*.

Identifikasi dapat dilakukan pada sistem *close loop* maupun *open loop*. Identifikasi *close loop* biasa digunakan pada sistem yang kurang stabil. Sedangkan identifikasi open *loop* lebih sederhana, namun sulit diterapkan untuk sistem–sistem yang memiliki ketidakpastian respon.

Metode identifikasi statis dilakukan dengan menggunakan pendekatan grafis, di mana sinyal uji diberikan pada sistem untuk mengetahui respon *open loop* sistem. Dari respon sistem, dapat diketahui karakteristik-karakteristik dari sistem. Sedangkan identifikasi dinamis, sinyal uji yang digunakan berupa sinyal acak atau sinyal semi acak yang biasa disebut dengan *Pseudo Random Binary Sequence* (PRBS). Pendekatan berbasis waktu dan berbasis frekuensi merupakan identifikasi statis, yaitu identifikasi *plant* dengan memberikan sinyal masukan ke *plant* dengan nilai tertentu dan tetap. Gambar 2.12 merupakan tampilan dari sinyal PRBS.

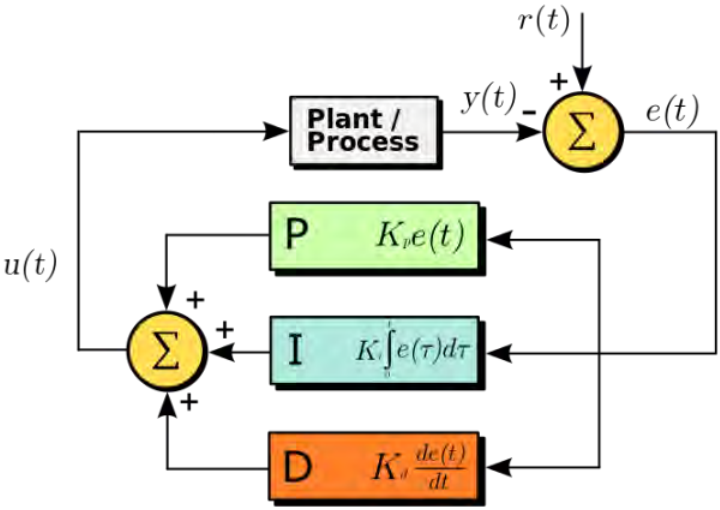


Gambar 1.12 Tampilan Sinyal PRBS [14]

Terdapat beberapa perbedaan mendasar antara identifikasi statis dan dinamis. Diantaranya adalah perbedaan antara sinyal uji dan metode pemodelan yang digunakan. Jika identifikasi statis menggunakan *input setpoint* yang konstan terhadap waktu, maka identifikasi dinamis menggunakan sinyal acak atau *random* sebagai sinyal ujinya. Sinyal ini memiliki frekuensi yang berubah-ubah, sehingga memungkinkan karakteristik sistem dapat diketahui secara lebih teliti.
Sinyal tersebut dinamakan sinyal *Pseudo-Random Binary Sequence* (PRBS). Sinyal PRBS seperti pada Gambar 2.12 mirip dengan bilangan acak secara nyata, tapi juga dapat disebut semu atau *pseudo* karena bersifat deterministik.

2.8 Kontroler PID [8]

PID (*Proportional–Integral–Derivative control*ler) merupakan kontroler untuk menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya umpan balik pada sistem tesebut. Pengontrol PID adalah pengontrol konvensional yang banyak dipakai dalam dunia industri. Blok diagram dari kontroler PID dapat dilihat pada Gambar 2.13 di bawah ini:



Gambar 1.13 Blok Diagram Kontroler PID [8]

Adapun persamaan Kontroler PID adalah:

$$MV(t) = K_P\left(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}\right) \quad (2.5)$$

$MV(t)$: *Manipulated Variable*
Kp : Konstanta Proporsional
Ti : *Time* Integral
Td : *Time* Derivatif
$e(t)$: Eror

Komponen kontrol PID ini terdiri dari tiga jenis yaitu proporsional, integratif dan derivatif. Ketiganya dapat dipakai bersamaan maupun sendiri-sendiri tergantung dari respon yang kita inginkan terhadap suatu *plant*.

2.8.1 Kontroler Proporsional

Kontroler proporsional jika $G(s) = Kp$, dengan Kp adalah konstanta proporsional. Kp berlaku sebagai *gain* (penguat) saja tanpa memberikan efek dinamik kepada kinerja kontroler. Penggunaan kontroler proporsional memiliki berbagai keterbatasan karena sifat kontrol yang tidak dinamik ini. Walaupun demikian dalam aplikasi-aplikasi dasar yang sederhana kontrol P ini mampu untuk memperbaiki respon transien khususnya *rise time* dan *settling time*. Pengontrol proporsional memiliki keluaran yang sebanding/proporsional dengan besarnya sinyal kesalahan (selisih antara besaran yang diinginkan dengan harga aktualnya).

Ciri-ciri pengontrol proporsional:

1. Jika nilai Kp kecil, pengontrol proporsional hanya mampu melakukan koreksi kesalahan yang kecil, sehingga akan menghasilkan respon sistem yang lambat (menambah *rise time*).
2. Jika nilai Kp dinaikkan, respon/tanggapan sistem akan semakin cepat mencapai keadaan mantapnya (mengurangi *rise time*).
3. Namun jika nilai Kp diperbesar sehingga mencapai harga yang berlebihan, akan mengakibatkan sistem bekerja tidak stabil atau respon sistem akan berosilasi.
4. Nilai Kp dapat di*set* sedemikian sehingga mengurangi *steady state* eror, tetapi tidak menghilangkannya.

2.8.2 Kontroler Integratif

Kontroler integral berfungsi untuk menghasilkan respon sistem yang memiliki kesalahan keadaan mantap nol (*Error Steady State* = 0). Jika sebuah pengontrol tidak memiliki unsur integrator, pengontrol proporsional tidak mampu menjamin keluaran sistem dengan kesalahan keadaan mantapnya nol.

Ketika $e(T)$ mendekati konstan (bukan nol) maka u(t) akan menjadi sangat besar sehingga diharapkan dapat memperbaiki eror. Jika $e(T)$ mendekati nol maka efek kontrol I ini semakin kecil. Kontroler integral dapat memperbaiki sekaligus menghilangkan respon *steady-state*, namun

pemilihan Ki yang tidak tepat dapat menyebabkan respon transien yang tinggi sehingga dapat menyebabkan ketidakstabilan sistem. Pemilihan Ki yang sangat tinggi justru dapat menyebabkan *output* berosilasi karena menambah orde *system*

Keluaran pengontrol ini merupakan hasil penjumlahan yang terus menerus dari perubahan masukannya. Jika sinyal kesalahan tidak mengalami perubahan, maka keluaran akan menjaga keadaan seperti sebelum terjadinya perubahan masukan. Sinyal keluaran pengontrol integral merupakan luas bidang yang dibentuk oleh kurva kesalahan/ eror.

Ciri-ciri pengontrol integral:

1. Keluaran pengontrol integral membutuhkan selang waktu tertentu, sehingga pengontrol integral cenderung memperlambat respon.
2. Ketika sinyal kesalahan berharga nol, keluaran pengontrol bertahan pada nilai sebelumnya.
3. Jika sinyal kesalahan tidak berharga nol, keluaran menunjukkan kenaikan atau penurunan yang dipengaruhi oleh besarnya sinyal kesalahan dan nilai Ki.
4. Konstanta integral Ki yang berharga besar mempercepat hilangnya offset. Tetapi semakin besar nilai konstanta Ki mengakibatkan peningkatan osilasi dari sinyal keluaran pengontrol.

2.8.3 Kontroler Derivatif

Keluaran pengontrol diferensial memiliki sifat seperti halnya suatu operasi derivatif. Perubahan yang mendadak pada masukan pengontrol akan mengakibatkan perubahan yang sangat besar dan cepat. Ketika masukannya tidak mengalami perubahan, keluaran pengontrol juga tidak mengalami perubahan, sedangkan apabila sinyal masukan berubah mendadak dan menaik (berbentuk fungsi *step*), keluaran menghasilkan sinyal berbentuk impuls. Jika sinyal masukan berubah naik secara perlahan (fungsi *ramp*), keluarannya justru merupakan fungsi *step* yang besar magnitudonya sangat dipengaruhi oleh kecepatan naik dari fungsi *ramp* dan konstanta Kd.

Sinyal kontrol u yang dihasilkan oleh kontrol D dapat dinyatakan sebagai $G(s) = s.Kd$. Dari persamaan di atas, nampak bahwa sifat dari kontroler derivatif ini dalam konteks "kecepatan" atau *rate* dari eror. Dengan sifat ini ia dapat digunakan untuk memperbaiki respon transien dengan memprediksi eror yang akan terjadi. Kontroler derivatif hanya

berubah saat ada perubahan eror sehingga saat eror statis kontrol ini tidak akan bereaksi, hal ini pula yang menyebabkan kontroler derivatif tidak dapat dipakai sendiri.

Ciri-ciri pengontrol derivatif:

1. Pengontrol tidak dapat menghasilkan keluaran jika tidak ada perubahan pada masukannya (berupa perubahan sinyal kesalahan)
2. Jika sinyal kesalahan berubah terhadap waktu, maka keluaran yang dihasilkan pengontrol tergantung pada nilai Kd dan laju perubahan sinyal kesalahan.
3. Pengontrol diferensial mempunyai suatu karakter untuk mendahului, sehingga pengontrol ini dapat menghasilkan koreksi yang signifikan sebelum kesalahan pembangkit menjadi sangat besar. Jadi pengontrol diferensial dapat mengantisipasi kesalahan pembangkit, memberikan aksi yang bersifat korektif dan cenderung meningkatkan stabilitas sistem.
4. Dengan meningkatkan nilai Kd, dapat meningkatkan stabilitas sistem dan mengurangi *overshoot*.

Berdasarkan karakteristik pengontrol ini, pengontrol diferensial umumnya dipakai untuk mempercepat respon awal suatu sistem, tetapi tidak memperkecil kesalahan pada keadaan tunaknya. Efek dari setiap pengontrol Proporsional, Integral dan Derivatif pada sistem lup tertutup disimpulkan pada Tabel 2.3 berikut ini:

Tabel 1.3 Karakteristik Kontroler Proporsional, Integral, dan Derivatif

Respon *Close-Loop*	***Rise Time***	***Overshoot***	***Setting Time***	***Steady State Errror***
Proporsional	Turun	Naik	Perubahan Kecil	Turun
Integral	Turun	Naik	Naik	Hilang
Derivatif	Perubahan Kecil	Turun	Turun	Perubahan Kecil

2.9 Analisa Kestabilan Lyapunov

Ada beberapa cara yang dapat digunakan untuk menguji kestabilan suatu sistem. Salah satu metode yang banyak digunakan adalah analisa kestabilan Lyapunov. Menurut Lyapunov, kestabilan sistem dapat ditentukan dari energi yang disimpan oleh sistem tersebut. Sistem dikatakan stabil jika energi yang disimpan makin lama makin kecil. Hal ini memiliki artian ketika energi yang disimpan berkurang seiring dengan bertambahnya waktu, maka energi tersebut akan mencapai nilai minimalnya pada titik ekuilibrium sistem tersebut. Untuk mengetahui energi yang disimpan dari suatu sistem, Lyapunov memperkenalkan fungsi Lyapunov yang merupakan fungsi energi buatan.

Fungsi Lyapunov merupakan fungsi energi buatan yang bergantung pada vektor *state* sistem ($x = [x = x_1 \quad x_2 \quad \cdots \quad x_n]^T$) yang dapat dinyatakan dengan $V(x)$. Dalam metode kedua Lyapunov, perilaku $V(x)$ dan turunan parsial pertamanya $dV(x)/dt$ dapat memberikan informasi mengenai kestabilan sistem tanpa harus mendapatkan solusi persamaan diferensial sistem. Selain itu, analisa kestabilan Lyapunov dapat digunakan untuk sistem linear maupun nonlinear.

Teorema kestabilan Lyapunov dapat dijelaskan sebagai berikut [8]:

Teorema 1 :

Jika terdapat fungsi Lyapunov $V(x)$ definit positif, dengan x adalah vektor *state* sistem ($x \in R^n$), maka vektor *state* x akan memenuhi

$V(x) = C \;\; ; \;\; C > 0$

dengan C adalah tetapan positif.

Jika terdapat tetapan positif sehingga $0 < C1 < C2$, maka $V(x) = C1$ lebih kecil daripada $V(x) = C2$. Jika titik ekuilibrium sistem berada pada *origin* ($x_e = 0$) maka fungsi Lyapunov pada titik ekulibrium adalah $V(xe) = V(0) = 0$.

Teorema 2 :

Jika terdapat sistem dengan persamaan

$$\dot{x} = f\,(x, u) \tag{2.6}$$

dengan $f\,(0,0) = 0$ untuk semua t,

maka sistem (2.6) akan stabil asimtotik pada titik ekuilibrium di *origin* jika terdapat fungsi skalar $V(x)$ yang kontinyu dan memiliki turunan parsial pertama yang memenuhi kondisi:

1. $V(x)$ definit positif
2. $\frac{dV(x)}{dt}$ definit negatif (2.7)

Dari definisi (2.7) dapat dilihat bahwa fungsi $V(x) > 0$ memiliki laju perubahan $dV(x)/dt$ yang negatif. Laju perubahan yang negatif menyebabkan fungsi $V(x)$ berangsur-angsur mengecil sehingga ketika *t* mendekati tak hingga, fungsi $V(x)$ bernilai 0. Fungsi $V(x)$ yang bernilai 0 menunjukkan bahwa vektor *state* sistem bernilai 0 dan *state* sistem bergerak menuju *origin.* Sehingga sistem dikatakan stabil karena *state* sistem bergerak menuju titik ekuilibriumnya di *origin*.

Analisa kestabilan Lyapunov dapat diterapkan pada sistem lup terbuka maupun lup tertutup. Jika terdapat persamaan sistem lup terbuka,

$$\dot{x}(t) = Ax(t) \tag{2.8}$$

dan dipilih kandidat fungsi Lyapunov, yaitu

$$V\big(x(t)\big) = x(t)^T Px(t) \tag{2.9}$$

Dengan P adalah matriks simetris, maka sistem (2.8) dikatakan stabil jika $V(x(t))$ definit positif dan $dV(x(t))/dt$ definit negatif. Untuk menjamin agar $V(x(t))$ definit positif untuk semua x, maka matriks P harus definit positif. $dV(x(t))/dt$ dapat dihitung dari (2.9) sebagai berikut:

$$\dot{V}\big(x(t)\big) = \dot{x}(t)^T Px(t) + x(t)^T P\dot{x}(t) \tag{2.10}$$

Substitusi (2.8) ke (2.10) akan didapat

$$\dot{V}\big(x(t)\big) = [Ax(t)]^T Px(t) + x(t)^T P[Ax(t)]$$
$$\dot{V}\big(x(t)\big) = x(t)^T A^T Px(t) + x(t)^T PAx(t)$$
$$\dot{V}(x(t)) = x(t)^T [A^T P + PA]x(t)$$
$$\dot{V}\big(x(t)\big) = -x(t)^T Qx(t)$$

dengan

$$Q = -[A^T P + PA] \tag{2.11}$$

Untuk menjamin agar $dV(x(t))/dt$ definit negatif untuk semua x, maka matriks Q harus definit positif. Sehingga untuk menjamin kestabilan sistem (2.8) harus terdapat matriks P yang memenuhi:

1. $P > 0$
2. $A^T P + PA < 0$ (2.12)

Untuk menjamin kestabilan sistem lup dapat dipilih kandidat fungsi Lyapunov sesuai dengan Persamaan (2.9). Agar $V(x(t))$ definit positif untuk semua x, maka matriks P harus definit positif. $dV(x(t))/dt$ dapat dihitung dari (2.29) sehingga didapat Persamaan (2.10).

Substitusi (2.25) ke (2.30) akan didapat

$$\dot{V}(x(t)) = [(A_i - B_i K_j)x(t)]^T Px(t) + x(t)^T P[(A_i - B_i K_j)x(t)]$$
$$\dot{V}(x(t)) = x(t)^T [A_i{}^T P + PA_i - K_j{}^T B_i{}^T P - PB_i K_j]x(t)$$
$$\dot{V}(x(t)) = -x(t)^T Q x(t)^T \quad (2.13)$$

dengan

$$Q = -[A_i{}^T P + PA_i - K_j{}^T B_i{}^T P - PB_i K_j] \quad (2.14)$$

Untuk menjamin agar $dV(x(t))/dt$ definit negatif untuk semua x, maka matriks Q pada (2.13) harus definit positif. Sehingga untuk menjamin kestabilan sistem harus terdapat matriks P yang memenuhi:

1. $P > 0$
2. $A_i{}^T P + PA_i - K_j{}^T B_i{}^T P - PB_i K_j < 0 \quad (2.15)$

2.10 Performansi H_∞

Tujuan utama dari perancangan sistem kontrol adalah mampu menstabilkan sistem dengan performansi tertentu. Dalam sistem nyata, ketidakpastian parameter maupun gangguan dari luar dapat mengganggu kestabilan sistem, sehingga kontroler yang dirancang harus *robust* terhadap ketidakpastian serta gangguan tersebut. Sistem dikatakan memiliki performansi *robust* apabila sistem mampu menjaga kestabilannya meskipun terdapat ketidakpastian parameter serta gangguan dari luar. Dalam Tugas Akhir ini kontroler dirancang agar memiliki performansi H_∞.

2.10.1 Kontrol *Robust* H_∞

Salah satu metode kontrol *robust* yang sering digunakan adalah kontrol *robust* H_∞. Pada H_∞, kontroler didesain untuk meminimalkan pengaruh terburuk keluaran performansi ketika terdapat gangguan yang diberikan pada sistem. Pengaruh terburuk gangguan terhadap keluaran performansi dapat diketahui dengan menghitung ∞–*norm* dari fungsi alih sistem. ∞–*norm* dari suatu fungsi alih didefinisikan sebagai

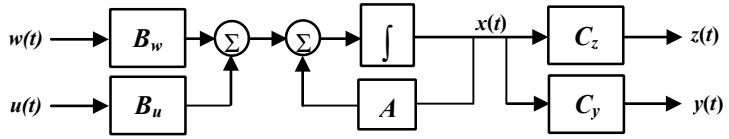
$$\|G(s)\|_\infty = \sup_\omega |G(j\omega)| \quad (2.16)$$

Dari Persamaan (2.16), ∞–*norm* dari suatu fungsi alih adalah nilai maksimal dari plot *magnitude* Bode. Jika $G(s)$ menyatakan fungsi alih keluaran performansi terhadap gangguan, dan ∞–*norm* dari $G(s)$ bernilai kecil, maka dapat dikatakan bahwa gangguan yang diberikan teredam sesuai besarnya ∞–*norm*.

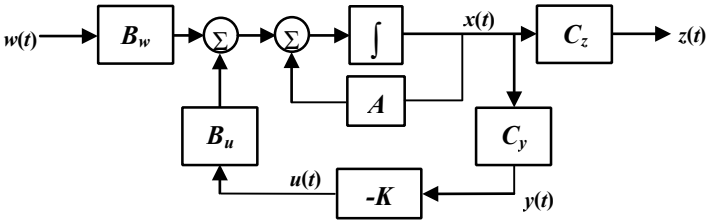
Gambar 2.14 menunjukkan diagram blok sistem dengan dua masukan dan dua keluaran. $w(t)$ adalah gangguan dari luar, $u(t)$ adalah sinyal kontrol, $z(t)$ adalah keluaran performansi, dan $y(t)$ adalah pengukuran untuk kontroler. Dalam bentuk *state space*, sistem pada Gambar 2.14 dapat ditulis sebagai berikut:

$$\dot{x}(t) = Ax(t) + B_w W(t) + B_u u(t)$$
$$z(t) = C_z x(t)$$
$$y(t) = C_y x(t) \tag{2.17}$$

Pada Persamaan (2.17), jika sinyal kontrol yang digunakan adalah $u(t) = -Ky(t)$, maka sistem (2.17) dapat disederhanakan menjadi sistem dengan satu masukan yaitu $w(t)$ dan satu keluaran yaitu $z(t)$ dengan bentuk *state space*,



Gambar 1.14 Diagram Blok Sistem (2.17)



Gambar 1.15 Diagram Blok Sistem (2.18)

$$\dot{x}(t) = \overline{A}x(t) + \overline{B}w(t)$$
$$z(t) = \overline{C}x(t) \tag{2.18}$$

dan dengan

$$\overline{A} = (A - Bu\,KC_y)\,,\ \overline{B} = Bw\,,\ \text{ dan } \overline{C} = Cz \tag{2.19}$$

Gambar 2.15 menunjukkan diagram blok sistem sesuai dengan Persamaan (2.18).

Persamaan (2.18) menunjukkan bentuk *state space* sistem (2.17) dengan $u(t) = -Ky(t)$. Jika $T_{zw}(s)$ menunjukkan fungsi alih $Z(s)$ terhadap $W(s)$, maka $T_{zw}(s)$ dapat dihitung sebagai berikut:

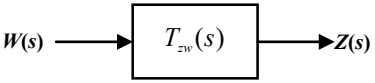
$$T_{zw}(s) = \overline{C}(sI - \overline{A})^{-1}\overline{B}$$

$$T_{zw}(s) = C_z(sI - A + B_u K C_y)^{-1}B_w \qquad (2.20)$$

Bentuk sistem pada Gambar 2.15 dapat digambarkan dengan diagram blok seperti ditunjukkan pada Gambar 2.16. Hubungan antara masukan $W(s)$ dan keluaran $Z(s)$ dapat dinyatakan dengan

$$\frac{Z(s)}{W(s)} = T_{zw}(s)$$

Dalam desain kontrol *robust* H_∞, dicari nilai *gain state-feedback K* yang menyebabkan pengaruh terburuk gangguan $w(t)$ terhadap keluaran $z(t)$ atau dengan kata lain ∞–*norm* dari fungsi alih $T_{zw}(s)$ kurang dari tingkat pelemahan tertentu. Jika L_2-*norm* dari sinyal $w(t)$ dinyatakan dengan,



Gambar 1.16 Diagram Blok Sistem $\boldsymbol{T_{zw}(s)}$

$$\|w(t)\|_2 = \sqrt{\int_0^\infty w(t)^T w(t) dt}$$

maka L_2-*Gain* atau perbandingan L_2-*norm* $z(t)$ terhadap L_2-*norm* $w(t)$ adalah

$$\frac{\|z(t)\|_2}{\|w(t)\|_2} = \frac{\sqrt{\int_0^\infty z(t)^T z(t) dt}}{\sqrt{\int_0^\infty w(t)^T w(t) dt}} \qquad (2.21)$$

Pada [10] didefinisikan bahwa ∞–*norm* dari fungsi alih $T_{zw}(s)$ dapat dihitung dari nilai maksimal L_2-*Gain* $z(t)$ terhadap $w(t)$. Sesuai definisi tersebut, maka ∞–*norm* dari fungsi alih $T_{zw}(s)$ dapat dihitung sesuai pada Persamaan (2.37).

$$\| T_{zw}(s)\|_\infty = \sup_\omega |T_{zw}(j\omega)| = \sup_{\|w(t)\|\neq 0} \frac{\|z(t)\|_2}{\|w(t)\|_2} \quad (2.22)$$

Jika tingkat pelemahan maksimal yang diinginkan adalah kurang dari γ, maka performansi desain yang digunakan dalam kontrol *robust* H_∞ dapat dinyatakan seperti pada Pertidaksamaan (2.23).

$$\| T_{zw}(s)\|_\infty = \gamma^* < \gamma$$
$$\sup_{\|w\|_2\neq 0} \frac{\|z(t)\|_2}{\|w(t)\|_2} = \gamma^* < \gamma \quad (2.23)$$

2.11 *Linear Matrix Inequalities* (LMI)

Kestabilan suatu sistem dapat diketahui jika terdapat fungsi Lyapunov yang memenuhi definisi (2.7). Jika fungsi Lyapunov didefinisikan seperti pada (2.9), maka sistem (2.8) dikatakan stabil jika terdapat matriks simetris *P* yang memenuhi:

1. $P > 0$
2. $A^T P + PA < 0$ (2.24)

Untuk sistem yang sederhana, matriks *P* dapat dicari melalui perhitungan analitik. Namun untuk sistem orde tinggi, penggunaan perhitungan analitik tidak mudah dilakukan. Sehingga untuk menyelesaikan persoalan tersebut, dapat digunakan perhitungan numerik menggunakan metode optimasi konveks yang disebut persoalan *Linear Matrix Inequalities* (LMI) [11].

LMI memiliki bentuk umum sebagai berikut:

$$F(x) = F_0 + \sum_{i=1}^{m} x_i F_i > 0 \quad (2.25)$$

dengan $x\epsilon R^m$ adalah variabel dan $F_i = F_i^T \epsilon R^{nxn}$, $i = 0,\ldots,m$, adalah matriks simetris yang diketahui. Tanda pertidaksamaan pada (2.25) menunjukkan bahwa $F(x)$ adalah definit positif. LMI (2.25) merupakan fungsi kendala dari *x*, sehingga tujuan optimasi dari LMI (2.25) adalah mencari nilai *x* sedemikian hingga $F(x) > 0$.

Jika terdapat beberapa LMI, $F^{(1)}(x) < 0$, … , $F^{(p)}(x) < 0$, maka LMI tersebut dapat direpresentasikan sebagai

$$\overline{F}(x) = \begin{bmatrix} F^{(1)}(x) & 0 & \dots & 0 \\ 0 & F^{(2)}(x) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & F^{(p)}(x) \end{bmatrix} < 0 \Leftrightarrow \begin{matrix} F^{(1)}(x) \\ F^{(2)}(x) \\ \vdots \\ F^{(p)}(x) \end{matrix} \tag{2.26}$$

LMI (2.26) menunjukkan bahwa beberapa LMI dapat disederhanakan menjadi satu LMI saja, sehingga penyelesaian optimasi untuk beberapa LMI dapat disederhanakan menjadi optimasi LMI $\overline{F}(x) < 0$.

Pertidaksamaan (2.26) merupakan contoh LMI karena setiap kendala pertidaksamaan merupakan kombinasi linear dari variabel *x*. Jika terdapat kendala dalam bentuk pertidaksamaan nonlinear, maka kendala tersebut dapat diubah menjadi LMI dengan menggunakan *Schur Complement* [11]. Sebagai contoh, jika terdapat LMI

$$L(x) = \begin{bmatrix} E(x) & F(x) \\ G(x) & H(x) \end{bmatrix} < 0 \tag{2.27}$$

dengan $E(x) = E(x)^T$, $H(x) = H(x)^T$, dan $G(x) = F(x)^T$, maka LMI (2.27) merupakan penyederhanaan dari fungsi kendala pertidaksamaan nonlinear, yaitu

$$L(x) < 0 \Leftrightarrow \begin{cases} E(x) < 0 \\ H(x) - G(x)E(x)^{-1}F(x) < 0 \end{cases}$$

atau

$$L(x) < 0 \Leftrightarrow \begin{cases} H(x) < 0 \\ E(x) - F(x)H(x)^{-1}G(x) < 0 \end{cases} \tag{2.28}$$

Pada LMI (2.25), variabel yang digunakan merupakan vektor $x \epsilon R^m$. Namun variabel yang digunakan bisa juga berupa matriks simetris seperti pada pertidaksamaan Lyapunov. Syarat kestabilan Lyapunov pada (2.24) dapat dinyatakan dengan LMI

$$F^{(1)} < 0 \quad \text{dan} \quad F^{(2)} < 0 \tag{2.29}$$

dan dengan

$$F^{(1)} = -P \quad \text{dan} \quad F^2 = A^T P + PA \tag{2.30}$$

Penggunaan Pertidaksamaan (2.26) dapat menyederhanakan LMI (2.29) menjadi

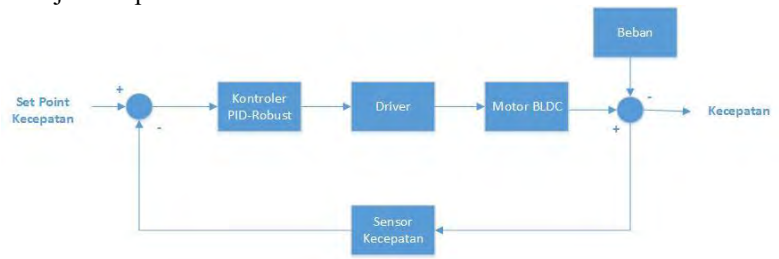
$$\overline{F} = \begin{bmatrix} F^{(1)} & 0 \\ 0 & F^{(2)} \end{bmatrix} = \begin{bmatrix} -P & 0 \\ 0 & A^T P + PA \end{bmatrix} < 0 \tag{2.31}$$

BAB 3
PERANCANGAN SISTEM

3.1. Gambaran Umum Sistem

Dalam tugas akhir ini penulis merancang sistem pengaturan kecepatan motor BLDC seperti yang ditunjukkan pada Gambar 3.1. Sistem tersebut merupakan sistem kontrol negatif *feedback* yang tersusun atas komponen berikut motor BLDC sebagai *plant* yang akan dikontrol kecepatannya, beban mekanik berupa rem elektromagnetik yang diberikan pada motor untuk memberikan efek pembebanan pada motor BLDC selain itu terdapat *driver* motor BLDC yang digunakan sebagai aktuator yang menjembatani antara kontroler dengan *plant*. Sinyal kontrol dari kontroler berupa sinyal PWM yang digunakan sebagai masukan *driver* untuk mengontrol kecepatan motor BLDC, kontroler yang digunakan adalah kontroler PID bekerja melalui komputer dengan menggunakan *software* MATLAB dan menggunakan Arduino sebagai *interface* antara komputer dengan *driver* motor BLDC. Dalam sistem pengaturan kecepatan motor BLDC ini juga terdapat dua sensor yaitu sensor kecepatan yang digunakan sebagai negatif *feedback* untuk melihat kecepatan motor secara aktual.

Blok diagram sistem yang digunakan pada Tugas Akhir ini ditunjukkan pada Gambar 3.1



Gambar 3.1 Blok Diagram Sistem Pengaturan Kecepatan Motor *Brushless DC* (BLDC).

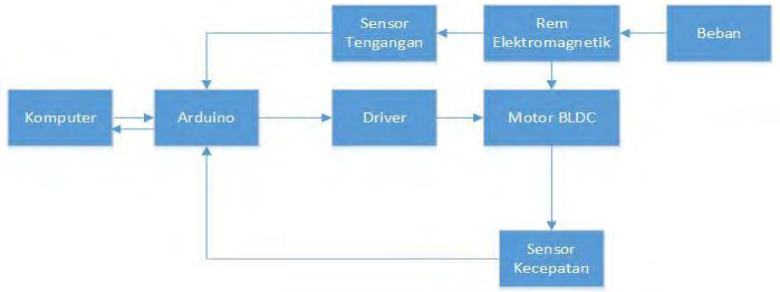
Selain perangkat keras berupa motor BLDC dan rem elektromagnetik, ditambahkan pula beberapa komponen pendukung, seperti *driver* untuk motor BLDC dan *driver* untuk rem elektromagnetik. Mikrokontroler Arduino juga akan dipakai pada sistem ini sebagai

perantara antara *sensor* dan *driver* dengan komputer. Serta rangkaian pengatur PWM (*Pulse Width Modulation)* yang digunakan untuk mengatur besarnya sinyal kontrol yang diberikan pada rem untuk membantu besarnya pembebanan rem magnetik.

Metode yang digunakan dalam pengaturan kecepatan motor BLDC dalam Tugas Akhir ini menggunakan metode kontrol PID-*Robust* dengan tuning paramter penyesuaian beban dengan metode *Robust* H∞ LMI *(Linear Matrix Inequality)*. Penggunaan metode ini diharapkan mampu meningkatkan performa motor untuk menghasilkan respon performansi.

3.2. Perancangan Perangkat Keras

Pada tahap perancangan simulator BLDC, perangkat keras dibagi menjadi 2 golongan yaitu perangkat mekanik dan perangkat elektronik. Komponen yang termasuk perangkat mekanik antara lain motor BLDC, kopel karet, dan rem elektromagnetik. Komponen yang termasuk perangkat elektronik antara lain, *driver* motor BLDC, *supply* DC untuk rem elektromagnetik, sensor tegangan rem elektromagnetik, Arduino, dan komputer. Secara umum susunan perangkat dalam simulator BLDC seperti terlihat pada Gambar 3.2.



Gambar 3.2 Konfigurasi Perangkat Keras Simulator BLDC.

Arduino bertugas untuk menerima data dari sensor kecepatan dan sensor tegangan kemudian mengirimkan data tersebut kedalam komputer lalu Arduino akan mengeluarkan *output* berupa sinyal PWM ke *driver* motor BLDC. Terdapat pula rangkaian sensor yang berfungsi mengukur *input* tegangan yang masuk ke dalam rem elektromagnetik dan sensor kecepatan motor BLDC yang menggunakan teknik *sensorless*.

3.2.1. Perancangan Mekanik

Pada *plant* sistem pengaturan kecepatan motor BLDC terdapat beberapa komponen utama yang digunakan antara lain motor BLDC sebagai objek yang dikontrol. Selain itu, terdapat rem elektromagnetik yang digunakan untuk memberikan efek pembebanan pada motor BLDC. Untuk lebih jelasnya, penjelasan mengenai konstruksi motor BLDC dan rem elektromagnetik dapat dilihat pada Subbab di bawah ini.

3.2.1.1. ***Motor Brushless DC***

Motor *Brushless* DC (BLDC) yang digunakan merupakan motor BLDC yang digunakan pada AC *inventer* Daikin. Motor BLDC jenis ini merupakan tipe dari motor BLDC yang menggunakan teknik *sensorless* yang memiliki 5 kabel untuk *supply* motor, *supply driver* motor, sinyal kontrol, sensor kecepatan dan *ground*. Bentuk fisik motor BLDC yang digunakan pada *plant* ini dapat dilihat pada Gambar 3.3. Sedangkan spesifikasi motor BLDC dapat dilihat pada Tabel 3.1 di bawah ini.

Tabel 3.1 Spesifikasi Motor BLDC Daikin D43F.

Parameter		**Nilai**
Tipe		Daikin D43F
Berat Motor		1200 gram
Tegangan Kerja		307 VDC
Kecepatan Motor	Beban Minimal	2410 rpm
	Beban Nominal	1502 rpm
	Beban Maksimal	1433 rpm

Gambar 3.3 Motor *Brushless* DC Tipe *Inrunning* dengan Tipe Daikin D43F.

3.2.1.2. ***Rem Elektromagnetik***

Rem elektromagnetik pada *plant* ini terpasang pada poros motor BLDC yang berguna sebagai pembebanan pada motor. Medan elektromagnetik dari rem ini dihasilkan oleh delapan kumparan yang dihubungkan secara seri dan diberikan masukan arus DC. Daya DC yang masuk ke dalam rem elektromagnetik ini diberikan oleh *power supply* yang memiliki rentang tegangan antara 16-24V, resistansi rem elektromagnetik adalah tetap sehingga pengaturan besar magnetisasi yang dihasilkan dapat diatur melalui tegangan *supply* yang diberikan karena tegangan proporsional terhadap arus yang mengalir pada rem elektromagnetik tersebut.

Rem elektromagnetik ini terbuat dari 8 kumparan terdiri dari 4 kumparan di setiap sisinya yang disusun secara seri. Diantara celah kedua sisi kumparan dipasang piringan aluminium. Piringan aluminium tersebut kemudian dipasang ke dalam sebuah *shaft* yang selanjutnya dikopel dengan motor BLDC. Spesifikasi dari rem elektromagnetik adalah sebagai berikut:

a. Jumlah kumparan : 8 (masing-masing 400 lilitan)
b. Resistansi kumparan : 12 Ω
c. Diameter kumparan : 3 cm
d. Diameter kawat : 0,6 mm
e. Tegangan kerja : 0-30 V
f. Arus maksimal : 2 A

3.2.1.3. ***Kopel***

Pada konstruksi simulator BLDC, *shaft* motor BLDC dipasang seporos dengan rem elektromagnetik. Oleh karena itu konstruksi penahan *shaft* pada rem elektromagnetik harus benar-benar lurus dengan shaft motor BLDC. Jika kedua *shaft* tidak lurus maka dapat menyebabkan kerusakan pada *bearing*. Bahkan *shaft* bisa bengkok jika kedua *shaft* diputar pada keadaan tidak lurus namun untuk menjaga kedua *shaft* ini agar tetap seporos sangat sulit. Hal ini dikarenakan saat motor berputar, akan timbul getaran pada motor sehingga menyebabkan kedua *shaft* tidak lurus. Untuk menghindari hal ini perlu dipasang kopel yang menghubungkan *shaft* motor BLDC dengan *shaft* rem elektromagnetik. Pemasangan kopel sendiri perlu diperhatikan dengan baik agar kopel terpasang dengan benar supaya *shaft* tidak bergetar saat motor berputar.

3.2.2. Perancangan Elektronik

Perancangan elektronik terdiri dari beberapa bagian, antara lain rangkaian sensor kecepatan motor BLDC, rangkaian sensor tegangan rem elektromagnetik, dan rangkaian *driver* motor BLDC selain itu akan dibahas juga mengenai Arduino yang berfungsi sebagai *interface.*

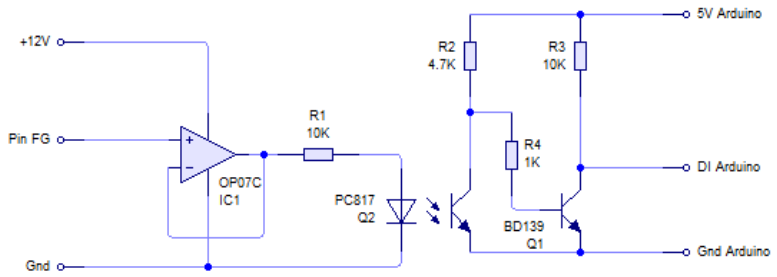
3.2.1.1. ***Rangkaian Sensor Kecepatan Motor BLDC***

Rangkaian sensor kecepatan BLDC digunakan untuk membaca kecepatan motor BLDC. Pada motor BLDC yang digunakan pengukuran kecepatan menggunakan teknik *sensorless* yaitu dengan membaca ggl balik pada setiap belitan fasa untuk mendeteksi posisi *rotor*. Pada pin *frequency generator* (FG), rangkaian *driver* menghasilkan sinyal kotak dengan *duty cycle* tetap 50% dan frekuensi sinyal yang berubah-ubah proporsional terhadap kecepatan motor BLDC. Untuk membaca sinyal kotak pada pin FG maka digunakan Arduino dengan rangkaian *signal conditioning* seperti ditunjukkan pada Gambar 3.4.

Perhitungan kecepatan motor dapat dilakukan dengan menghitung frekuensi gelombang kotak yang dihasilkan pin FG. Untuk mendapatkan kecepatan motor dapat digunakan Persamaan (3.1) berikut.

$$N = \frac{60FG}{P} \tag{3.1}$$

N : Kecepatan motor (rpm)
FG : *Frequency generator* (Hz)
P : Jumlah pasang kutub



Gambar 3.4 Rangkaian Sensor Kecepatan Motor BLDC.

Rangkaian pembaca sensor kecepatan ini terdiri dari *optocoupler* untuk isolasi antara pin FG dengan pin di Arduino dan *op amp* yang digunakan sebagai *buffer* untuk mengurangi pelemahan sinyal dari pin FG.

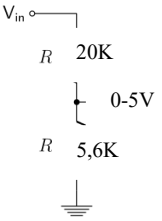
3.2.1.2. ***Mikrokontroler Arduino***

Pada simulator BLDC ini, Mikrokontroler Arduino yang digunakan adalah tipe UNO R3. Arduino digunakan untuk mengakuisisi data kecepatan motor BLDC dan *supply* tegangan rem elektromagnetik. Tegangan yang masuk ke Arduino akan diubah menjadi data digital dengan resolusi 10 bit dan akan diolah oleh program kontroler pada komputer. Kemudian kontroler tersebut akan menghasilkan sinyal kontrol berbentuk pwm yang dikeluarkan melalui pin *output* digital pwm pada Arduino untuk menjadi sinyal kontrol pada *plant* motor BLDC. Pengiriman data antara komputer dan Arduino ini dilakukan melalui *port* komunikasi *serial*. Berikut ini pemilihan pin *input*/*output* pada Arduino,

a. Pin A0 : *input* analog dari rangkaian pembagi tegangan.
b. Pin 7 : *output* PWM untuk *driver* motor BLDC.
c. Pin 6 : *input digital* sensor kecepatan.
d. Pin +5V
e. Pin *Ground*

3.2.1.3. ***Rangkaian Pembagi Tegangan***

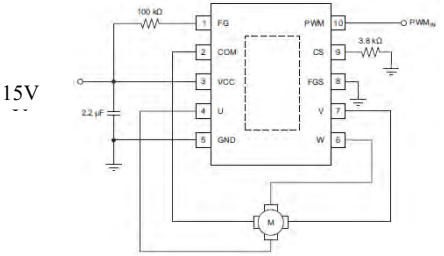
Rangkaian pembagi tegangan pada Gambar 3.5 digunakan untuk membaca tegangan yang diberikan pada rem magnetik. Dalam pembebanan yang diberikan memiliki rentang tegangan dari 0-24V, tegangan ini nantiny[illegible] 0-24V [illegible]iubah kedalam rentang tegangan 0-5V oleh rangkaian pemba[illegible]ngan supaya dapat dibaca dengan mikrokontroler Arduino.



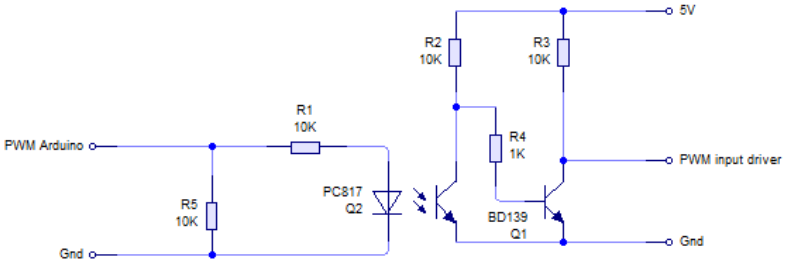
Gambar 3.5 Rangkaian Pembagi Tegangan.

3.2.1.4. ***Driver Motor BLDC*** **[9]**

Pada motor BLDC, *driver* berfungsi mengubah sinyal kontrol dari kontroler yang tegangan analog (PWM) menjadi urutan komutasi kumparan pada motor BLDC. Bentuk fisik dari rangkaian *driver* motor BLDC dapat ditunjukkan pada Gambar 3.6. Untuk dapat mengatur kecepatan motor BLDC dapat dilakukan dengan dua cara yaitu memberi sinyal PWM pada pin *input* PWM atau mengatur nilai tegangan pada pin *input* Vcc. Pada tugas akhir kali ini dipilih pengaturan kecepatan dengan memberikan sinyal PWM. Sinyal PWM ini akan diberikan oleh Arduino dengan rangkaian isolasi seperti pada Gambar 3.7.



Gambar 3.6 Rangkaian *Driver* Motor BLDC.



Gambar 3.7 Rangkaian Isolasi Arduino dengan *Driver* BLDC.

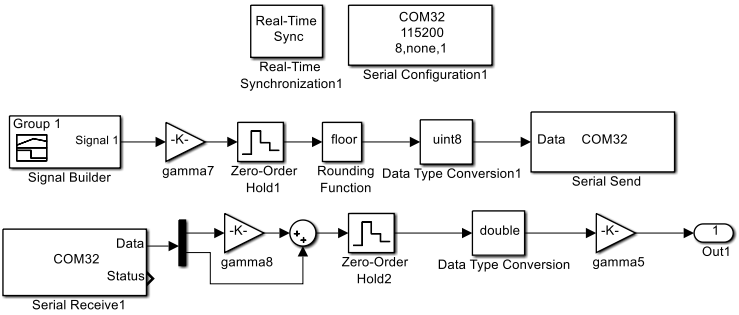
Driver ini menggunakan teknik *sensorless* dalam menggerakkan motor BLDC jadi tidak menggunakan sensor *hall effect* maupun sensor kecepatan tetapi dengan mendeteksi sinyal ggl balik. Dengan mengamati sinyal ggl balik yang timbul pada kumparan *stator* maka posisi dan kecepatan motor dapat diperkirakan.

3.3. Perancangan Perangkat Lunak

Dalam sistem pengaturan motor BLDC, beberapa perangkat lunak harus digunakan untuk proses pengambilan data, perancangan kontroler, dan pengiriman data. Perangkat lunak yang digunakan yaitu MATLAB dan *software* Arduino.

3.2.1. Perangkat Lunak MATLAB

Perangkat lunak ini digunakan pada proses pengambilan data dan pengiriman data. Dengan menggunakan komunikasi *serial* dengan Arduino, Simulink MATLAB dapat mengolah data dari blok *serial receive* dan mengirimkannya kembali melalui blok *serial send.* Contoh diagram blok Simulink MATLAB untuk pengambilan data *open loop* ditunjukkan pada gambar berikut.



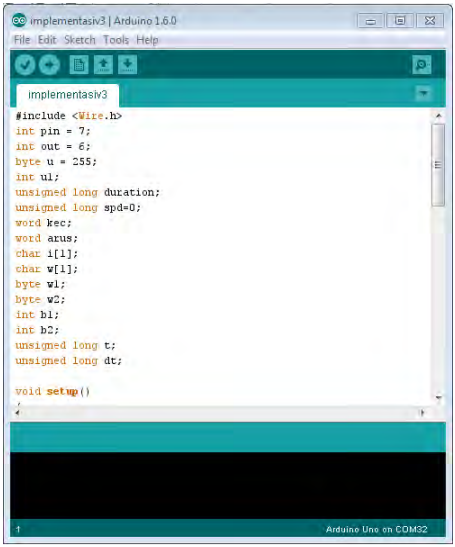
Gambar 3.8 Blok Simulink *Open Loop.*

3.2.2. Perangkat Lunak Arduino

Mikrokontroler Arduino dalam sistem pengaturan motor BLDC ini digunakan sebagai *interface* antara sensor, dan *output* ke motor BLDC dengan Simulink MATLAB. Artinya, Arduino ini berfungsi sebagai media pengirim data yang menghubungkan antara sensor dan *plant* dengan Simulink MATLAB. Data yang diterima oleh *serial receive* pada Simulink MATLAB merupakan data *integer* 8 bit. Dari data integer yang dikirim ini terdiri dari 2 variabel yaitu nilai kecepatan pada sensor yang memiliki satuan rpm dan data pada rem magnetik. Dan Arduino akan menerima data sinyal kontrol yang dikirim dari MATLAB melalui *serial send* dengan range sinyal kontrol 0 sampai 255 yang menunjukkan *duty cycle* sekitar 0% sampai 100%.

Selanjutnya data dari kecepatan motor BLDC yang berupa sinyal kotak dengan *duty cycle* 50% dan memiliki frekuensi yang berbeda – beda akan dibaca dengan menggunakan fungsi PulseIn(). Fungsi ini akan mengembalikan sebuah nilai pada sebuah variabel yang telah di deskripsikan. Pertama-tama fungsi ini akan membaca nilai *high* atau *low* pada sebuah pin, pada sistem ini dipakai pin 8 dan yang dibaca pada pin merupakan *pulse high* lalu apabila pin 7 membaca nilai *high*, *timer* akan berjalan dan akan menunggu hingga pin 7 membaca nilai *low*. Apabila pin 7 membaca nilai *low* maka *timer* akan berhenti dan variabel yang telah disebutkan akan mendapatkan nilai waktu lamanya pulsa berada pada keadaan *high*. Disinilah data kecepatan motor bisa didapatkan, dari waktu tersebut didapatkan nilai kecepatan motor dalam satuan rpm. Dan yang terakhir data besar tegangan yang disuplai pada rem magnetik. Arduino dapat bekerja berdasarkan pada *script* yang telah dirancang

dengan menggunakan Arduino IDE. Program yang telah dibuat bisa di *upload* pada Arduino dengan menggunakan Arduino IDE. Berikut tampilan dari Arduino IDE.



Gambar 3.9 Tampilan IDE Arduino.

3.4. Identifikasi dan Pemodelan Sistem

Pada pemodelan sistem di tugas akhir kali ini digunakan identifikasi dinamis. Metode dinamis ini dilakukan dengan cara memberikan *input* berapa sinyal *Pseudorandom Binary Sequence* (PRBS). Setelah itu *output* kecepatan akan diidentifikasi dengan menggunakan metode ARX yang ada pada *toolbox* MATLAB. Setelah hal ini dilakukan didapatkanlah model matematika dari *plant.*

3.2.1. Metode dan Pembebanan Sistem

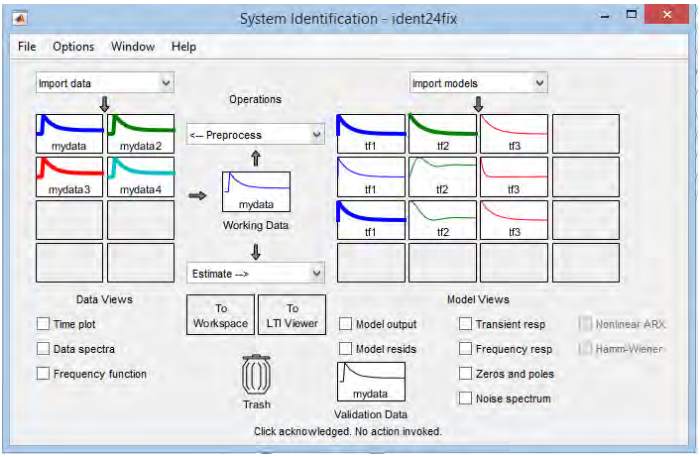
Respons *plant* dapat dilihat dari beberapa cara. Salah satu cara yaitu dengan memberikan pembebanan. Pada *plant* motor BLDC pembebanan dilakukan dengan menggunakan rem elektromagnetik. Pembebanan dilakukan untuk mendapatkan model sistem dengan beban yang ditentukan. Pada sistem ini rem magnetik di berikan tegangan *supply* yang bervariasi mulai dari *input* 16V, 20V, dan 24V. Beban-beban tersebut disebut dengan beban minimal, nominal, dan maksimal.

3.2.2. Metode Identifikasi dan Pemodelan

Identifikasi pada motor BLDC dilakukan dengan menggunakan identifikasi statis. Hal ini dilakukan dengan cara memberikan *input* dengan menggunakan sinyal *step* ke *plant*. Pada tugas akhir ini sinyal *step* yang digunakan sebesar 0,9608.

Setelah didapatkan data dari *input* dan *output* motor maka tahap selanjutnya adalah melakukan identifikasi dari data yang telah didapatkan. Metode ini dilakukan dengan *Toolbox Identification System* yang ada pada program MATLAB. Berikut hasil fungsi alih yang didapatkan setelah dilakukan identifikasi menggunakan metode ARX.

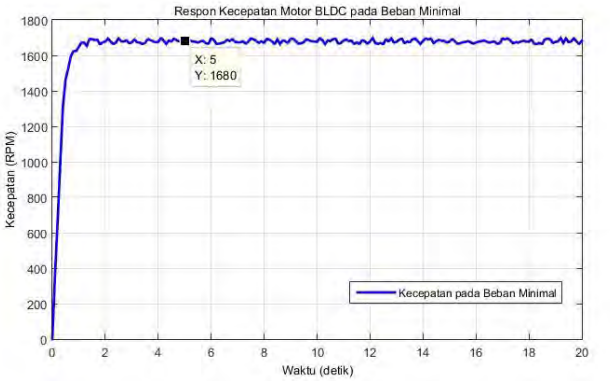
Model matematika *plant* didapatkan melalui *toolbox* ini dengan cara meng*input*kan data *input-output*, melakukan *preprocessing,* dan memilih bentuk model yang diinginkan.. Dalam proses identifikasi ini, data *input-output* yang ada dimasukkan ke dalam aplikasi dengan memilih "*time domain data"* saat menekan akan timbul tulisan "*import data"*. Setelah itu dilakukan *preprocessing*. Data yang telah melalui tahap *preprocess* kemudian dipilih sebagai *working data* sekaligus *validation data* dengan cara *drag-and-drop* kotak yang berisi data tersebut ke kotak *working data* dan *validation data*. Kemudian dilakukan identifikasi model dengan memilih "*transfer function*" *set*elah meng-klik *popout* bertuliskan "*estimate*". Tampilan dari *system identification toolbox* pada Matlab dapat dilihat pada Gambar 3.10.



Gambar 3.10 Tampilan *System Identification Toolbox*

3.2.1.1. ***Beban Minimal***

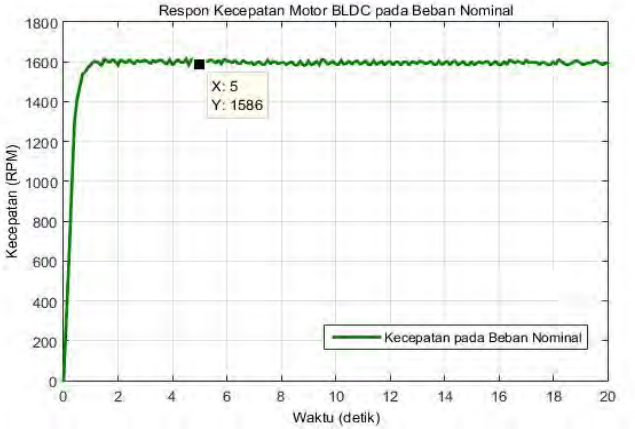
Pada kondisi beban minimal motor dipasang seporos dengan rem elektromagnetik dengan suplai tegangan 16 VDC. Respon *open loop* motor BLDC pada beban minimal ditunjukkan pada Gambar 3.11.



Gambar 3.11 Respon Kecepatan Motor BLDC pada Beban Minimal

3.2.1.2. ***Beban Nominal***

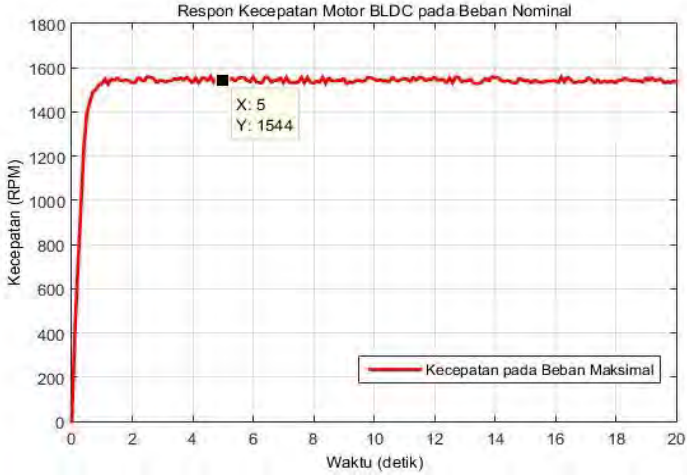
Pada kondisi beban minimal motor dipasang seporos dengan rem elektromagnetik dengan suplai tegangan 20 VDC. Respon *open loop* motor BLDC pada beban minimal ditunjukkan pada Gambar 3.11.



Gambar 3.12 Respon Kecepatan Motor BLDC pada Beban Nominal

3.2.1.3. ***Beban Maksimal***

Pada kondisi beban minimal motor dipasang seporos dengan rem elektromagnetik dengan suplai tegangan 24 VDC. Respon *open loop* motor BLDC pada beban minimal ditunjukkan pada Gambar 3.13.



Gambar 3.13 Respon Kecepatan Motor BLDC pada Beban Maksimal

Tabel 3.2 Fungsi Alih *Plant* Motor BLDC.

Beban	**Tegangan *input* beban**	**Fungsi Alih**	**Fit to Estimation**	**MSE**
Minimal	16 Volt	$\frac{428{,}8s + 1279}{s^2 + 2{,}351s + 0{,}7318}$	78,9%	376,6
Nominal	20 Volt	$\frac{536{,}2s + 1969}{s^2 + 3{,}461s + 1{,}185}$	78,12%	346,9
Maksimal	24 Volt	$\frac{460{,}3s + 1396}{s^2 + 2{,}47s + 0{,}8689}$	79,55%	271

3.5. Perancangan Kontroler berbasis PID

Setelah fungsi alih dari sistem telah didapatkan, langkah selanjutnya adalah merancang kontroler untuk pengaturan kecepatan motor. Kontroler digunakan untuk membuat *plant* mengikuti model referensi yang diinginkan sekalipun motor BLDC diberi beban. Pertama-tama, dirumuskan dahulu *gain state-feedback* yang mampu menjamin kestabilan sistem lup tertutup berdasarkan kriteria Lyapunov. Selain mampu menjamin kestabilan sistem, performansi H_∞ juga disertakan dalam perumusan pertidaksamaan Lyapunov yang kemudian akan diubah ke dalam bentuk LMI. Batasan pada sinyal kontrol serta keluaran yang dibatasi juga dirumuskan dalam bentuk LMI. Dari LMI yang didapat, nilai *gain state-feedback* dihitung menggunakan LMI *Toolbox* dari MATLAB sehingga kestabilan sistem terjamin serta spesifikasi desain terpenuhi.

3.2.1. Perancangan Kontroler PID

Sebelum merancang kontroler PID-*Robust,* diperlukan model referensi yang diinginkan. Model yang diinginkan memiliki *rise time* sebesar 6 detik, dengan *time* constant (Tc) 2 detik dan *pole* λ_2 = 1/Tc = 0,5. Nilai *pole* dominan ditentukan λ_1 sebesar 5x *pole* λ_2 yaitu sebesar 2,5. Sehingga fungsi *transfer* model menjadi [17]:

$$\frac{1{,}25}{s^2 + 3s + 1{,}25}$$

Nilai dari parameter Kp, Ki, dan Kd didapatkan dari persamaan di bawah ini:

$$\begin{aligned} K_p &= \lambda_1 + \lambda_2 = 2{,}5 + 0{,}5 = 3 \\ K_i &= \lambda_1 \lambda_2 = (2{,}5)(0{,}5) = 1{,}25 \\ K_d &= 1 \end{aligned} \tag{3.1}$$

Perancangan kontroler didesain dan diimplementasikan untuk mempercepat *risetime* dan mempertahankan kecepatan motor BLDC pada saat diberi beban tertentu. Pada hasil identifikasi dengan toolbox *transfer function* yang terdapat pada MATLAB didapat fungsi alih dari *plant*:

$$G(s) = \frac{\Omega(s)}{Ea(s)} = \frac{K}{s^2 + As + B} \tag{3.2}$$

Di mana :
K = 1969
A = 3,461

B = 1,185

3.2.2. Perancangan Model *State Space*

Kontroler PID-*Robust* merupakan kontroler berbasis model. Artinya, diperlukan sebuah pemodelan fungsi alih yang baik agar kontroler yang telah didesain dapat bekerja secara optimal. Fungsi alih yang digunakan merupakan fungsi alih motor BLDC pada pembebanan maksimal. Fungsi alih ini tidak bisa langsung digunakan untuk merancang kontroler PID-*Robust*, melainkan harus diubah terlebih dahulu ke dalam bentuk *state-space*.

Pada pembebanan maksimal, representasi fungsi alih yang bernilai RMSE paling kecil adalah sebagai berikut:

$$G(s) = \frac{\Omega(s)}{Ea(s)} = \frac{K}{s^2 + As + B} \tag{3.3}$$

$$\frac{\Omega(s)}{Ea(s)} = \frac{K}{s^2 + as + b} \tag{3.4}$$

Persamaan beda dari fungsi alih ini menjadi model matematika input Ea output Ω

$$\ddot{\Omega} + a\dot{\Omega} + b\Omega = KEa$$
$$\ddot{\Omega} = -a\dot{\Omega} - b\Omega + KEa \tag{3.5}$$

Misal state $x_1 = \dot{\Omega}$; $x_2 = \Omega$; dan $x_3 = \int \Omega$

$$\dot{x}_1 = \ddot{\Omega} = -a\dot{\Omega} - b\Omega + kEa \tag{3.6}$$

$$\dot{x}_2 = \dot{\Omega} = x_1 \tag{3.7}$$

$$\dot{x}_3 = \Omega = x_2 \tag{3.8}$$

Bentuk *state-space* yang digunakan pada Tugas Akhir ini adalah bentuk *controllable canonical form* [8]. Bentuk *state-space* untuk orde 2 tersebut mempunyai struktur sebagai berikut:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -a & -b & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} K \\ 0 \\ 0 \end{bmatrix} Ea$$

$$y = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{3.9}$$

Berdasarkan bentuk model *state-space* dari Persamaan (3.1) dapat kita simpulkan bahwa gain dari K_1, K_2 dan K_3 sebagai berikut:

$$K = [K_1 \quad K_2 \quad K_3] \tag{3.10}$$

dimana

$$\begin{aligned} K_1 &= K_d \\ K_2 &= K_p \\ K_3 &= K_i \end{aligned} \tag{3.11}$$

Persamaan (3.11) dan (3.12) merupakan gain yang didapat dari parameter oleh persamaan diferensial untuk *Robustness* dengan performansi H_∞ menggunakan penyelesaian dengan metode LMI *(Linear Matrix Inequalities)*, sinyal kontrol PID-*Robust* yang nantinya akan didesain pada diagram blok program kontroler dengan menggunakan MATLAB [16].

3.2.3. Analisa Kestabilan Lyapunov

Model motor dapat di ketahui dengan cara analisa persamaan beda yang digunakan secara keseluruhan dapat dinyatakan seperti pada Persamaan (3.4), yaitu:

$$\begin{aligned} \dot{x}(t) &= \sum_{i=1}^{2}\sum_{i=1}^{2} \alpha_i\,(x_2(t))\alpha_j(x_2(t))[(Ai - B_{u,i}K_j)x(t)B_{w,i}w(t)] \\ z_1(t) &= \sum_{i=1}^{2} \alpha_i(x_2(t))[C_{z1,i}x(t)] \\ z_2(t) &= \sum_{i=1}^{2} \alpha_i(x_2(t))[C_{z1,i}x(t)] \end{aligned} \tag{3.12}$$

dengan

$$\alpha_i(x_2(t)) = \frac{\mu_i(x_2(t))}{\sum_{i=1}^{2}\mu_i(x_2(t))} \text{ dan } \mu_i(x_2(t)) = M_i(x_2(t))$$

Sesuai dengan sifat pembobot pada (2.17), maka (3.19) dapat disederhanakan menjadi

$$\begin{aligned} &\dot{x}(t) = (Ai - B_{u,i}K_j)x(t)B_{w,i}w(t) \\ &z_1(t) = C_{z1,i}x(t) \\ &z_2(t) = C_{z1,i}x(t) \\ &i = 1,2 \quad j = 1,2 \end{aligned} \tag{3.13}$$

Jika dipilih kandidat fungsi Lyapunov seperti pada Persamaan (2.29), yaitu

$$V\big(x(t)\big) = x(t)^T\, Px(t) \tag{3.14}$$

dengan P adalah matriks simetris, maka turunan dari (3.21) adalah

$$\dot{V}\big(x(t)\big) = \dot{x}(t)^T\, Px(t) + x(t)^T\, P\dot{x}(t) \tag{3.15}$$

Substitusi (3.20) ke (3.22) akan didapat

$$\dot{V}\big(x(t)\big) = [\dot{x}(t)\big(A_i - B_{u,i}K_j\big)x(t)B_{w,i}w(t)]^T\, Px(t) + x(t)^T\, P[\dot{x}(t)\big(A_i - B_{u,i}K_j\big)x(t)B_{w,i}w(t)]$$

$$\dot{V}\big(x(t)\big) = \begin{bmatrix} x(t) \\ w(t) \end{bmatrix}^T \begin{bmatrix} A_i{}^T P + PA_i - K_j{}^T B_{u,i}{}^T P - PB_{u,i}K_j & PB_{w,i} \\ B_{w,i}{}^T P & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix} \tag{3.16}$$

Jadi agar sistem (3.20) terjamin kestabilannya, harus terdapat matriks simetris P yang memenuhi LMI:

1. $P > 0$ (3.17)
2. $\begin{bmatrix} A_i{}^T P + PA_i - K_j{}^T B_{u,i}{}^T P - PB_{u,i}K_j & PB_{w,i} \\ B_{w,i}{}^T P & 0 \end{bmatrix} < 0$ (3.18)

3.2.4. Performansi H_∞

Berikut ini akan dibahas mengenai perumusan performansi H_∞ yang akan menjamin kestabilan sistem (3.13) dengan tingkat pelemahan gangguan $w(t)$ terhadap keluaran performansi $z(t)$ maksimal kurang dari γ.

Sesuai dengan Pertidaksamaan (2.38), kriteria performansi H_∞ yang digunakan adalah

$$\sup_{\|w(t)\|\neq 0} \frac{\|z(t)\|_2}{\|w(t)\|_2} = \gamma^* < \gamma$$

$$\frac{\|z(t)\|_2}{\|w(t)\|_2} < \gamma \tag{3.19}$$

Jika Pertidaksamaan (3.19) menunjukkan L_2-*Gain* sistem (3.13), dan terdapat fungsi Lyapunov $V\big(x(t)\big) = x(t)^T Px(t)$, $P > 0$, dan $\gamma \geq 0$ sehingga untuk semua t,

$$\dot{V}(x(t)) + z_1(t)^T z_1(t) - \gamma^2 w(t)^T w(t) < 0 \quad ; \quad \forall t \geq 0 \qquad (3.20)$$

maka L_2-*Gain* sistem (3.13) memiliki nilai kurang dari γ [11].

Untuk membuktikan Pertidaksamaan (3.20), integralkan (3.20) dari 0 sampai T dengan kondisi awal $x(0) = 0$

$$\int_0^T dV\big(x(t)\big) + \int_0^T z_1(t)^T z_1(t) dt - \gamma^2 \int_0^T w(t)^T w(t) dt < 0$$

$$V\big(x(T)\big) - V\big(x(0)\big) + \int_0^T z_1(t)^T z_1(t) dt - \gamma^2 \int_0^T w(t)^T w(t) dt < 0$$

$$V\big(x(T)\big) + \int_0^T z_1(t)^T z_1(t) dt - \gamma^2 \int_0^T w(t)^T w(t) dt < 0$$

Karena $V\big(x(T)\big) \geq 0$, maka secara implisit didapat

$$z - \gamma^2 \int_0^T w(t)^T w(t) dt < 0$$

$$\sqrt{\int_0^T z_1(t)^T z_1(t) dt} < \gamma \sqrt{\int_0^T w(t)^T w(t) dt}$$

$$\frac{\sqrt{\int_0^T z_1(t)^T z_1(t) dt}}{\sqrt{\int_0^T w(t)^T w(t) dt}} < \gamma$$

$$\frac{\|z(t)\|_2}{\|w(t)\|_2} < \gamma \qquad (3.21)$$

Dapat dilihat bahwa bagian terakhir dalam pembuktian Pertidaksamaan (3.20) didapat dari (2.21). Jadi, jika terdapat fungsi Lyapunov yang memenuhi (3.20), maka performansi H_∞ pada (3.24) akan terpenuhi.

Dari Pertidaksamaan (3.20) komponen $\dot{V}(x(t))$ memiliki bentuk seperti pada Persamaan (3.16), sedangkan komponen $z_1(t)^T z_1(t)$ dapat dijabarkan dari (3.13) sebagai berikut:

$$z_1(t)^T z_1(t) = [C_{z1,i}\, x(t)]^T [C_{z1,i}\, x(t)]$$

$$z_1(t)^T z_1(t) = x(t)^T C_{z1,i}\, x(t)^T C_{z1,i}\, x(t)^T$$

$$z_1(t)^T z_1(t) = \begin{bmatrix} x(t) \\ w(t) \end{bmatrix}^T \begin{bmatrix} {C_{z1,i}}^T C_{z1,i} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix} \quad (3.22)$$

Selanjutnya, komponen $\gamma^2 w(t)^T\, w(t)$ dapat dijabarkan sebagai berikut:

$$\gamma^2 w(t)^T\, w(t) = [w(t)]^T [\gamma^2 I][w(t)]$$

$$\gamma^2 w(t)^T\, w(t) = \begin{bmatrix} x(t) \\ w(t) \end{bmatrix}^T \begin{bmatrix} 0 & 0 \\ 0 & \gamma^2 I \end{bmatrix} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix}^T \quad (3.23)$$

Substitusi (3.16), (3.22), dan (3.23) pada Pertidaksamaan (3.20) akan didapat

$$\dot{V}(x(t) + z_1(t)^T z_1(t) - \gamma^2 w(t)^T w(t) < 0$$

$$\begin{bmatrix} x(t) \\ w(t) \end{bmatrix}^T \begin{bmatrix} {A_i}^T P + P A_i - {K_j}^T {B_{u,i}}^T P - P B_{u,i} K_j & P B_{w,i} \\ {B_{w,i}}^T P & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix} +$$

$$\begin{bmatrix} x(t) \\ w(t) \end{bmatrix}^T \begin{bmatrix} {C_{z1,i}}^T C_{z1,i} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix} - \begin{bmatrix} x(t) \\ w(t) \end{bmatrix}^T \begin{bmatrix} 0 & 0 \\ 0 & \gamma^2 I \end{bmatrix} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix}^T < 0$$

$$\begin{bmatrix} x(t) \\ w(t) \end{bmatrix}^T \left\{ \begin{bmatrix} {A_i}^T P + P A_i - {K_j}^T {B_{u,i}}^T P - P B_{u,i} K_j & P B_{w,i} \\ {B_{w,i}}^T P & \gamma^2 I \end{bmatrix} + \right.$$

$$\left. \begin{bmatrix} {C_{z1,i}}^T \\ 0 \end{bmatrix} [C_{z1,i} \quad 0] \right\} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix} < 0 \quad (3.24)$$

Dari Pertidaksamaan (3.24), akan didapat LMI:

$$\begin{bmatrix} {A_i}^T P + P A_i - {K_j}^T {B_{u,i}}^T P - P B_{u,i} K_j & P B_{w,i} \\ {B_{w,i}}^T P & \gamma^2 I \end{bmatrix} +$$

$$\begin{bmatrix} {C_{z1,i}}^T \\ 0 \end{bmatrix} [I] [C_{z1,i} \quad 0] < 0 \quad (3.25)$$

Penerapan *Schur Complement* pada LMI (3.25) dan *pre-multiplying* dan *post-multiplying* LMI (3.25) dengan matriks P^{-1} akan menghasilkan LMI (3.26).

$$\begin{bmatrix} {A_i}^T Q + Q A_i - {K_j}^T {B_{u,i}}^T P - P B_{u,i} K_j & B_{w,i} & Q {C_{z1,i}}^T \\ {B_{w,i}}^T & -\gamma^2 I & 0 \\ C_{z1,i} Q & 0 & -I \end{bmatrix} \quad (3.26)$$

dengan $Q = P^{-1}$ dan $Y_j = K_j P^{-1}$

Jika terdapat matriks simetris $Q > 0$ yang memenuhi LMI (3.26), maka sistem (3.13) dikatakan stabil dengan tingkat pelemahan gangguan $w(t)$ terhadap keluaran $z_1(t)$ kurang dari γ. Penurunan LMI (3.26) dapat dilihat pada Lampiran A.1.

3.6. Perhitungan *Gain State-Feedback*

Pada Subbab 3.5 telah dibahas penurunan LMI beserta batasannya sehingga sistem (3.13) akan memenuhi spesifikasi desain sebagai berikut:

1. Sistem stabil asimtotik.
2. Sistem memenuhi performansi H_∞ dengan tingkat pelemahan gangguan $w(t)$ terhadap keluaran performansi $z_1(t)$ kurang dari γ.

Untuk memenuhi spesifikasi desain tersebut, maka harus terdapat matriks simteris Q yang memenuhi LMI (3.27)[7], yaitu

$$\Theta < 0$$

dan

$$\Theta_{ii} + \frac{1}{2}\left(\Theta_{ij} + \Theta_{ji}\right) < 0 \qquad ; \qquad 1 \le i \ne j \le 2 \tag{3.27}$$

dengan

$$\Theta_{ii} = \begin{bmatrix} A_i{}^T Q + Q A_i - K_j{}^T B_{u,i}{}^T P - P B_{u,i} K_j & B_{w,i} & Q C_{z1,i}{}^T \\ B_{w,i}{}^T & -\gamma^2 I & 0 \\ C_{z1,i} Q & 0 & -I \end{bmatrix}$$

$$\varepsilon_Q = \{x | x^T Q x \le 0\}$$

$$Q = P^{-1} \quad \text{dan} \quad Y_j = K_j P^{-1}$$

Jika LMI (3.27) adalah *feasible*, maka nilai *gain state-feedback* adalah

$$K_1 = Y_1 P \quad \text{dan} \quad K_2 = Y_2 P \tag{3.28}$$

Dalam Tugas Akhir ini digunakan parameter-parameter sebagai berikut:

1. A dan $B_{u,1}$ adalah matriks *state-space* pada Motor BLDC menggunakan persamaan beda pada Persamaan (3.9).
2. $B_{w,1} = B_{u,1}$ dan $B_{w,2} = B_{u,2}$.
3. $C =$ [0 1 0], yang menyatakan bahwa performansi keluaran yang menyatakan bahwa keluaran yang dibatasi hanya x_2

Untuk menyelesaikan LMI (3.27), dapat digunakan LMI *Toolbox* dari MATLAB. Algoritma yang digunakan untuk menyelesaikan LMI (3.42) dapat dilihat pada Lampiran B.1. Nilai parameter γ divariasi antara 1 dan 0 dan didapat hasil terbaik yaitu ketika $\gamma = 0.8$. Adapun hasil matriks P, Y yang didapat adalah

$$P = \begin{bmatrix} 7{,}69 * 10^{-9} & 9{,}57 * 10^{-7} & 6{,}74 * 10^{-15} \\ 9{,}57 * 10^{-7} & 0{,}0080 & 5{,}63 * 10^{-11} \\ 6{,}74 * 10^{-15} & 5{,}63 * 10^{-11} & 7{,}45 * 10^{-9} \end{bmatrix} \quad (3.28)$$

$$Y = 10^5 * [-1{,}5197 \quad 0{,}6629 \quad 0{,}000000163] \quad (3.29)$$

Dari Persamaan (3.43) dapat dihitung *gain state-feedback K* untuk parameter gain dari K_p, K_i dan K_d pada PID-*Robust*, yaitu

$$K = [0{,}0624 \quad 529{,}9 \quad 0{,}00000373] \quad (3.30)$$

Substitusi (3.28) ke (3.13) akan didapat *eigenvalue* sistem (3.13) sebagai berikut:

$$\det(\lambda \mathbf{I} - \boldsymbol{A} + \boldsymbol{BK}) = 0$$

$$\lambda = 10^3 * \{-0{,}0631 + j1{,}01; -0{,}0631 + j1{,}101; -7{,}04 * 10^{-12}\} \quad (3.31)$$

Persamaan (3.31) menunjukkan bahwa *eigenvalue* untuk kedua sistem hasil linearisasi memiliki bagian real negatif. Hal ini menunjukkan bahwa nilai *K* pada (3.30) akan menjamin kestabilan sistem (3.13).

Fungsi alih $Z(s)$ terhadap $W(s)$, yaitu $T_{z_1w}(s)$ untuk kedua sistem hasil linearisasi dapat dihitung dari (2.20) dengan C_y adalah matriks identitas sebagai berikut:

$$\boldsymbol{T}_{z_1w}(s) = C_z(s\mathbf{I} - \boldsymbol{A} + \boldsymbol{BK})^{-1}\boldsymbol{B}$$

$$\boldsymbol{T}_{z_1w}(s) = \frac{122{,}8656s^2 + 1043400s + 0{,}0073}{s^3 + 126{,}3266s^2 + 1043401{,}185s + 0{,}0073} \quad (3.31)$$

BAB 4
PENGUJIAN DAN ANALISA

Pada tahapan ini dilakukan beberapa jenis pengujian yaitu pengujian *sensor*, pengujian *open loop* dari motor BLDC, lalu pengujian kontroler yang disimulasikan pada hasil identifikasi model, lalu yang terakhir merupakan pengujian implementasi kontroler pada *plant* motor BLDC.

4.1. Pengujian *Sensor* Kecepatan Motor BLDC

Pengujian *sensor* kecepatan motor dilakukan dengan cara menghitung frekuensi dari sinyal kotak *driver* motor menggunakan fungsi *pulsein* Arduino. Kecepatan motor dihitung dengan hubungan $\omega = 15 \times f$. *Output* dari *sensor* dibandingkan dengan pengukuran menggunakan *laser tachometer*. Hasil pembacaan *sensor* kecepatan memiliki kesalahan *output* maksimal sebesar 1,5%. Tabel 4.1 merupakan hasil pembacaan kecepatan motor.

Tabel 0.1 *Output* Pembacaan Kecepatan Motor.

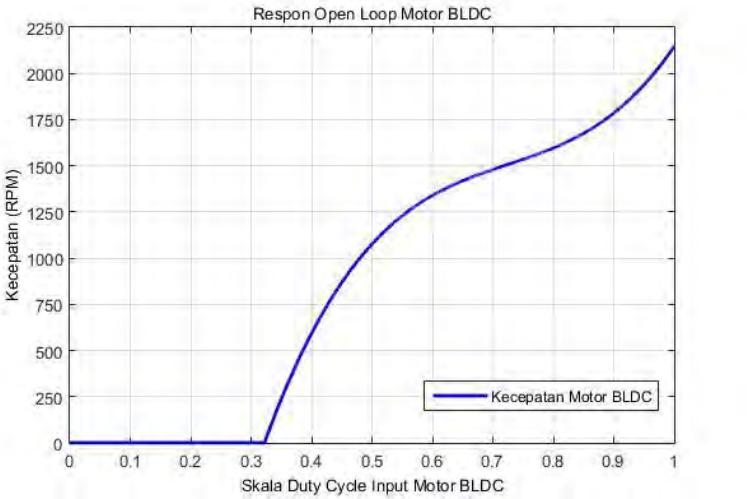
Hasil Pembacaan		Selisih	% error
Arduino	***Tachogenerator***		
972,36	984,6	12,24	1,258793
1022,1	1035,41	13,31	1,302221
1201,8	1218,04	16,24	1,351306
1363,4	1382,67	19,27	1,413378
1429,5	1448,82	19,32	1,351522
1703,6	1726,00	22,40	1,314863
2253	2290,45	37,45	1,670000
2397	2441,21	44,21	1,840000

Dari hasil perbandingan pembacaan Arduino dan *tachogenerator* pada Tabel 4.1, selisih antara keduanya tidak terlalu besar. Eror pembacaan hanya berkisar 1,2 – 1,8%. Sehingga dapat ditarik kesimpulan bahwa metode pengukuran kecepatan motor menggunakan frekuensi pulsa dari *output driver* motor sudah benar. Semakin besar kecepatan motor maka persentase eror dari pembacaan juga semakin besar.

4.2. Pengujian Open Loop Kecepatan Motor

Selanjutnya dilakukan pengujian pada respon kecepatan motor BLDC yang dipergunakan. Pengujian ini dilakukan untuk mengetahui grafik hubungan antara *input-output plant*. Pengujian dilakukan untuk melihat hubungan antara *input* yang berupa *duty cycle* dan *output* berupa kecepatan dari motor BLDC. *Driver* BLDC diberikan masukan berupa sinyal *step* dan kemudian diukur kecepatannya. Respon hasil pengujian *open loop* kecepatan motor dapat dilihat pada Gambar 4.1

Pada kondisi awal, terjadi lonjakan kecepatan hingga sekitar 2216 rpm. Hal ini karena mekanisme *starting driver* motor. Saat motor mulai berjalan dari keadaan berhenti, *driver* secara otomatis memberikan masukan tegangan PWM 5V sampai ggl balik di setiap fasa terdeteksi, setelah itu proses komutasi baru berjalan normal.



Gambar 0.1 Hubungan Tegangan *Input* dan *Output* Kecepatan Motor.

Dari grafik di atas dapat dilihat bahwa hubungan *input* dan *output* pada motor BLDC linier ketika kecepatan di atas 1000-1500 rpm. Pada kecepatan di bawah 1000 rpm, kecepatan motor tidak dapat terukur dengan baik. Hal ini menunjukkan adanya kekurangan dari *sensor* kecepatan motor sehingga rentang kerja dari *plant* harus lebih dari 1000 rpm. Sehingga pada tugas akhir ini, dipilih rentang kerja motor antara 1000-1500 rpm.

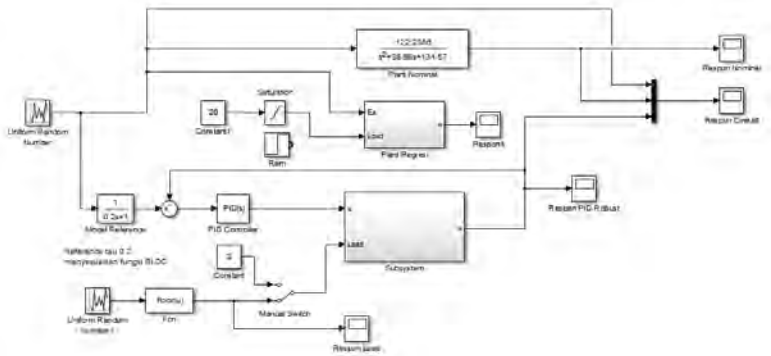
4.3. Pengujian Simulasi Kontroler

Sistem yang telah dirancang pada Bab III disimulasikan terlebih dahulu sebelum diimplementasikan, simulasi menggunakan *software* Matlab R2015a untuk mengetahui respon sinyal kontrol dengan menggunakan beban maupun tanpa beban.

4.3.1 Blok Diagram *Simulink*

Pada blok diagram simulasi, kontroler berupa *toolbox* PID yang dimana parameter dari gain kontroler di dapat dari penyelesaian perhitungan kestabilan *Lyapunov* dengan metode LMI, input dari subsystem plant berupa *state-space* adalah sinyal keluaran dari kontroler merupakan penjumlahan antara sinyal kontrol ekivalen dan sinyal kontrol PID dan beban yang dipasang sesuai keinginan atau dengan *random variable*/nilai acak sehingga merepresentasikan keadaan *real-system* yg dimana beban tidak dapat diprediksi. *Input* pada sinyal beban pada *subsystem* yang berupa eror/beban dan *output* dari kontroler, untuk memilih parameter pembebanan yang diinginkan sesuai dengan fungsi transfer dari sistem, misalkan pembebanan terpilih yakni minimal, nomial, dan maksimal pada 16 V, 20 V, dan 24 V.

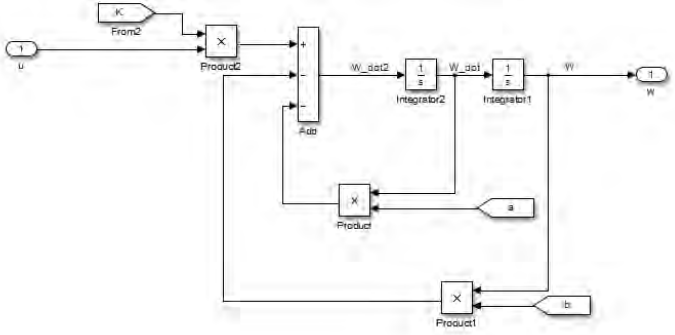
Keluaran dari *plant* berupa *output* y yang menjadi sinyal keluaran terkontrol dari sistem. Model referensi yang diharapkan mendapatkan masukan berupa sinyal *step*. Keluaran dari *plant* diharapkan dapat mengikuti keluaran dari model referensi yang sudah di desain.
Simulasi kontroler ditampilkan pada Gambar 4.2



Gambar 0.2 Simulasi Kontroler PID-*Robust*

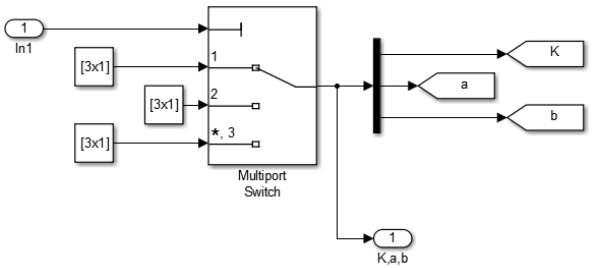
Gambar 4.3 merupakan *subsystem* dari plant yang berbentuk *state-space* agar bisa di selesaikan dengan perhitungan LMI, pada *mathscript*

di program MATLAB 2015a, sinyal kontrol di mana nilai dari parameter didapatkan dari parameter *plant* yang ditulis pada Bab 3. Adapun parameter yang digunakan dalam merancang sinyal kontrol pada subsystem ini yaitu nilai K, A, B pada setiap fungsi transfer masing-masing beban.



Gambar 0.3 *Subsystem State Space Plant*

Gambar 4.4 merupakan *subsystem* dari parameter sistem yang berfungsi untuk memilih fungsi transfer sesuai dengan kondisi beban saat ini. Pemilihan beban sebagai parameter *plant* sesuai yang ditulis pada Bab 3. Kondisi ini dapat dibuat dalam bentuk persamaan regresi sehingga dapat mengikuti segala beban yang ada sesuai dengan keadaan real-system. Parameter dibawah ditentukan bahwa pada blok diagram switch pada MATLAB pemilihan switch 1 adalah pembebanan minimal, switch 2 adalah pembebanan nominal, dan pembebanan 3 adalah maksimal.



Gambar 0.4 *Subsystem* Parameter Sistem

4.3.2 Hasil dan Analisa Simulasi

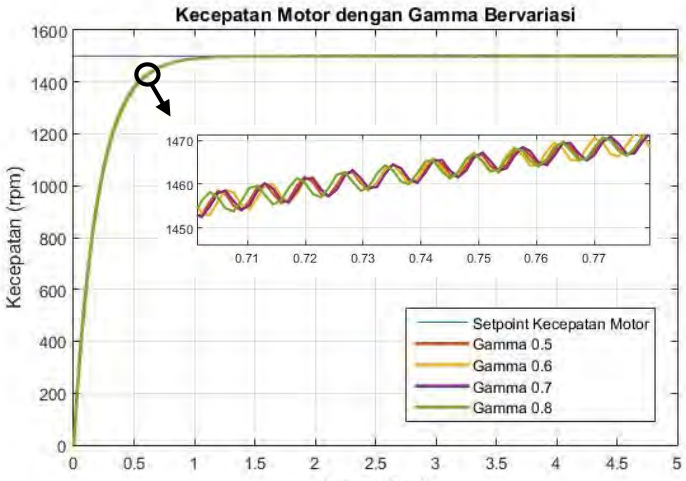
Pada Subbab ini akan ditunjukkan hasil simulasi untuk kondisi awal yang berbeda serta gangguan pada sistem.

4.3.2.1 *Hasil Simulasi dengan Berbagai Kondisi Awal*

Pada simulasi ini nilai *gain state-feedback* yang digunakan sesuai dengan Persamaan (3.28) dibandingkan dengan nilai gamma atau tingkat pelemahan (γ) yang bervariasi menjadi 0,8;0,7;0,6 dan 0,5. Perbedaan pada tingkat pelemahan γ akan mempengaruhi nilai dari matriks P sehingga nilai K, dan ∞–*norm* yang didapat juga berbeda pula. Nilai *gain state-feedback* K serta ∞–*norm* untuk nilai γ yang berbeda dapat dilihat pada Tabel 4.1,Tabel 4.2 dan Gambar 4.5 menunjukkan hasil simulasi sistem dengan gangguan $w(t)$ dan nilai γ yang berbeda.

Tabel 0.2 *Output* Pembacaan Kecepatan Motor.

γ	K	$T_z w(s) = \gamma^*$
0,80	[0,0624 529,9456 0,00000373]	0,3321
0,70	[0,0631 549,8129 0,00000385]	0,3090
0,60	[0,0647 608,0361 0,00000407]	0,3745
0,50	[0,0620 550,7199 0,00000426]	0,3837



Gambar 0.5 Respon *Plant* Motor pada Simulasi dengan Berbagai Nilai γ

Tabel 4.2 Penyimpangan Kecepatan Motor dan Nilai IAE dengan Berbagai Nilai γ

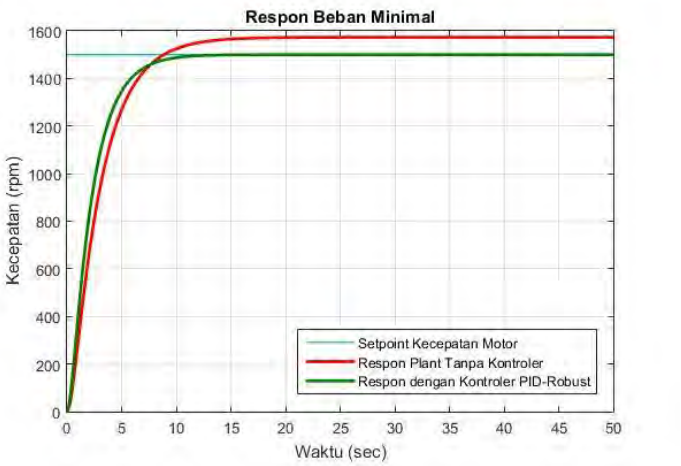
γ	Penyimpangan x (rpm)	IAE x
0,80	±0,0273	0,2743
0,70	±0,0248	0,3081
0,60	±0,0328	0,3607
0,50	±0,0370	0,4044

Dari Gambar 4.5 dapat dilihat bahwa perbedaan nilai γ akan mempengaruhi besar simpangan posisi kereta. Selain itu, penyimpangan posisi kereta terkecil terjadi ketika γ = 0,80. Hal ini bersesuaian dengan Tabel 4.1 yang menunjukkan tingkat pelemahan terkecil adalah ketika γ = 0,80, yaitu γ^* = 0,3090. Untuk mengetahui jumlah *error* pada posisi kereta saat gangguan diberikan pada sistem, nilai *Integral Absolute Error* (IAE) dihitung mulai detik ke-0.5. Besar penyimpangan posisi kereta dan nilai IAE untuk masing-masing nilai γ dapat dilihat pada Tabel 4.2.

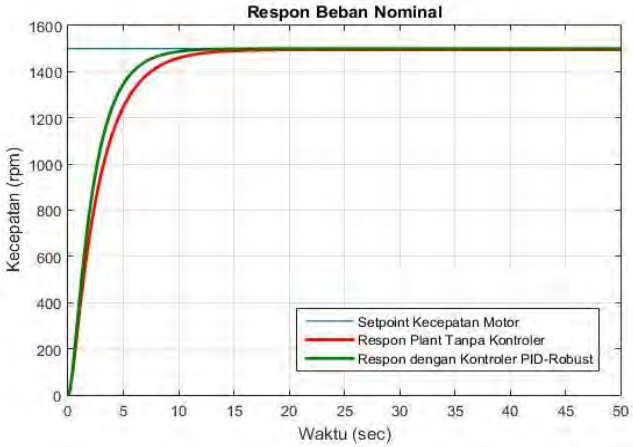
Pengujian dilakukan dengan memberikan sinyal *input* y_R berupa sinyal *step*. Sebelum dilakukan pengujian, dimasukkan nilai parameter yang didapatkan dari model matematis *plant* serta model referensi. Simulasi yang dilakukan dengan membandingkan respon model referensi dengan respon *plant* ketika diberikan beban minimal, beban nominal, serta beban maksimal.

4.3.2.2 *Hasil Simulasi dengan Gangguan pada Sistem*

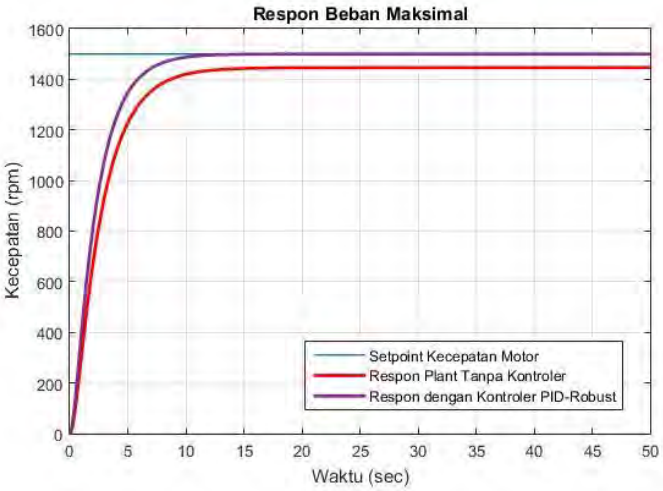
Gambar 4.6, 4.7, dan 4.8 menunjukkan perbandingan antara respon *plant* dengan kontroler, respon *plant* tanpa kontroler, serta respon model referensi ketika diberikan beban minimal, nominal, serta maksimal. Terlihat bahwa respon p emodelan *plant* dengan menggunakan PID-*Robust* berimpit dengan respon dari model yang diinginkan. Menggunakan skala pelemahan atau gamma yakni γ=0.8. Berikut adalah data respon yang diambil dengan *software* MATLAB 2015a *tools Simulink.* Spesifikasi desain yang diinginkan pada permasalahan disain kontroler motor adalah memiliki *Overshoot(%Mp)* 0% dan *Error Steady State(Ess)* 0%.



Gambar 0.6 Respon *Plant* dengan Beban Minimal



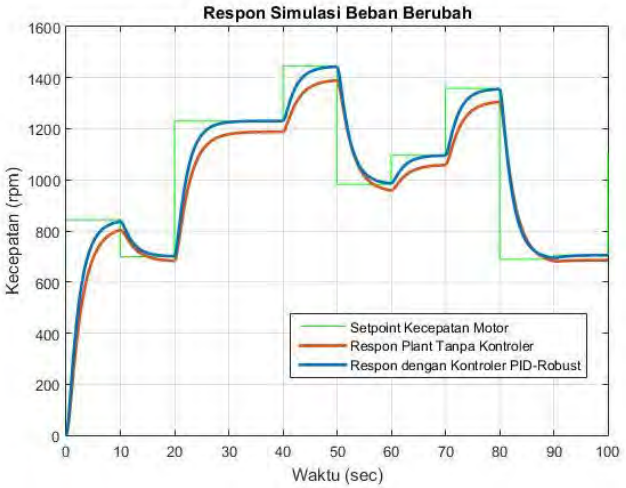
Gambar 0.7 Respon *Plant* dengan Beban Minimal



Gambar 0.8 Respon *Plant* dengan Beban Maksimal

Dari ketiga Gambar 4.6 Gambar 4.7 dan Gambar 4.8 terlihat bahwa pemodelan *plant* membutuhkan waktu *steady state* yang lebih lama dari respon *plant* yang sebenarnya. Hal ini dikarenakan pemodelan *plant* mengikuti model yang diinginkan. Waktu untuk mencapai *steady state* pada respon asli *plant* sebesar 14,2 detik sedangkan saat diberi kontroler PID-*Robust* untuk mengikuti *steady state* pada model yang diinginkan sebesar 10,4 detik. Kecepatan dari motor dalam bentuk persentase sehingga perlu dikalikan dengan kecepatan maksimal sebesar 1535 rpm.

Pada Gambar 4.9 dibawah adalah dimana respon input beban berubah untuk membuktikan bahwa kontroler PID-*Robust* dapat bertahan pada nilai *setpoint* ketika diberi *disturbance random* dengan input yang menunjukan *real system* bahwa beban tidak dapat terprediksi. Input *disturbance random* berubah sesuai 10 detik, ketika respon kontroler sudah mencapai *steady state* sesuai pada model yang diinginkan dengan nilai *settling* sebesar 10,2 detik namun respon asli *plant* belum mencapai *steady state*, begitupun pada perubahan setiap beban (minimal/nominal/maksimal).



Gambar 0.9 Respon *Plant* dengan Beban Campuran

Indeks performansi dari respon *step* dapat dilihat pada Tabel 4.3.

Tabel 0.3 Indeks Performansi *Plant* saat Simulasi

Indeks Performansi	**PID-*Robust***		
	Beban Minimal	**Beban Nominal**	**Beban Maksimal**
T_R (detik)	5,053	5,05	5,049
T_S (detik)	10,431	10,443	10,439
RMSE	0	0	0
Overshoot(%Mp)	0	0	0
Error Steady State(Ess)	0	0	0

4.4. Implementasi Sistem

Pada tahapan ini kontroler yang telah dibuat dicoba dijalankan pada *plant real* motor BLDC. Pengujian ini dilakukan dengan memberikan respon *step* pada saat *plant* diberi beban minimal, nominal, dan maksimal.

4.4.1 Realisasi *Plant*

Berikut ini merupakan *plant* dari sistem pengaturan motor BLDC yang telah di realisasikan. *Plant* terbagi menjadi 2 sistem, yaitu sistem yang bekerja menjalankan motor, lalu sistem yang menjalankan rem. Dari

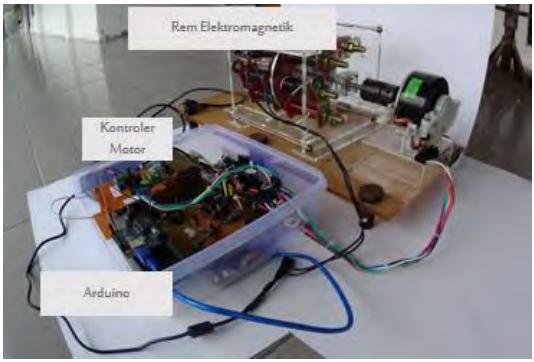
kedua sistem ini untuk bisa terhubung dengan kontroler harus memiliki *interface. Interface* yang kami gunakan adalah Arduino Uno.

Board bawaan Daikin yang ada di dalam *air conditioner* merupakan *board* yang digunakan untuk mengatur kecepatan motor BLDC. Selain itu juga memiliki fungsi sebagai *power supply driver* motor yang berada di dalam motor. Pada perancangan *plant* motor BLDC, *board* ini hanya digunakan sebagai *power supply driver* motor, sedangkan pengaturan kecepatannya melalui kontroler yang sudah dibuat di Matlab.

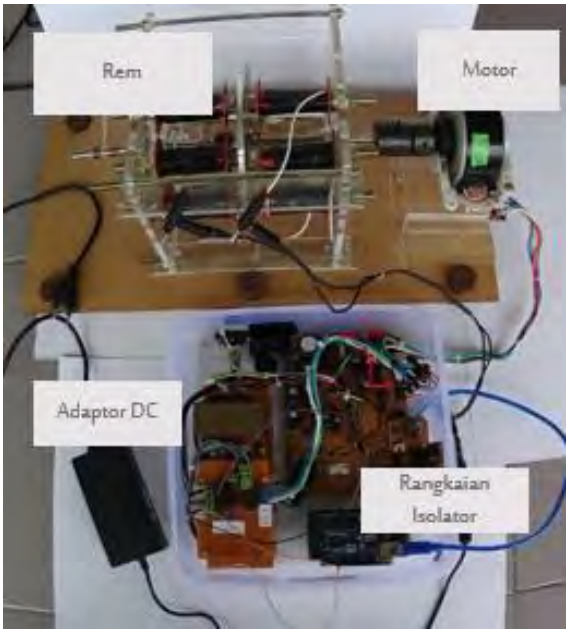
Arduino Uno digunakan sebagai *interface* yang menghubungkan antara *plant* dengan laptop. Arduino menghitung kecepatan motor degan cara mengukur periode pulsa yang dikirim oleh *driver* motor lalu mengirim nilai kecepatan ke laptop. Arduino juga berfungsi untuk menerima masukan data dari laptop untuk mengubah nilai PWM dari *pin output* yang akan digunakan untuk sinyal kontrol yang mengatur kecepatan motor. Daya untuk rem dapat diubah-ubah dengan mengubah nilai *power supply* yang digunakan untuk men*supply* rem elektromagnetik.

Gambar 4.11 merupakan tampilan fisik dari *plant* motor BLDC yang diambil dari *plant* yang telah direalisasikan. Pada *plant* ini sebagian besar bahan dasar yang dipakai untuk dudukan terbuat dari akrilik. Pada *plant* ini terdapat dua buah PCB yang diantaranya rangkaian *board* Daikin dan rangkaian isolasi. Setelah itu terdapat 8 buah lilitan kawat yang berfungsi sebagai rem magnetik. Selain itu terdapat 1 buah piringan yang satu terbuat dari alumunium yang berfungsi sebagai beban pada poros.

Kontruksi rem magnetik adalah baut yang memiliki diameter 1 cm. lalu baut tersebut dililit sebanyak 400 lilitan agar didpatkan medan magnet yang cukup kuat untuk membebani motor BLDC. Pada Gambar 4.10 dan Gambar 4.11 tersebut juga terlihat ada piringan diantara dua baut yang terbuat dari alumunium setebal 5 mm. diameter dari piringan ini adalah 15 cm. piringan terpasang pada sebuah poros yang terpasang dengan kopel yang terhubung dengan motor. Poros ini juga ikut menyebabkan piringan berputar. Kopel pada poros yang terpasang dengan *shaft* pada piringan motor dikaitkan dengan rem elektromagnetik untuk metode pembebanan pada motor BLDC.



Gambar 0.10 *Plant* Motor BLDC (1)



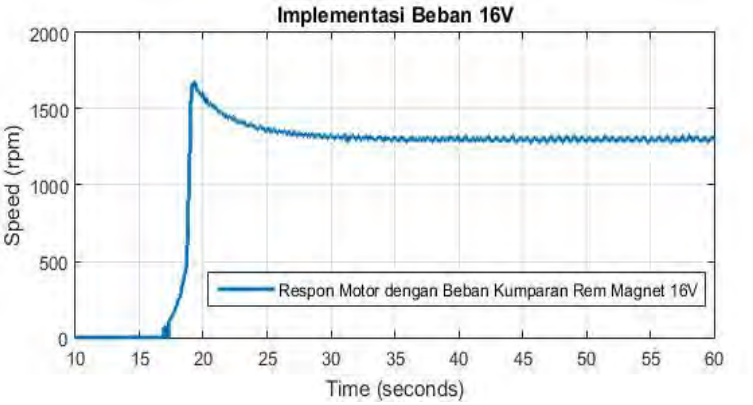
Gambar 0.11 *Plant* Motor BLDC (2)

4.4.2 Implementasi Kontroler PID-*Robust* pada *Plant*

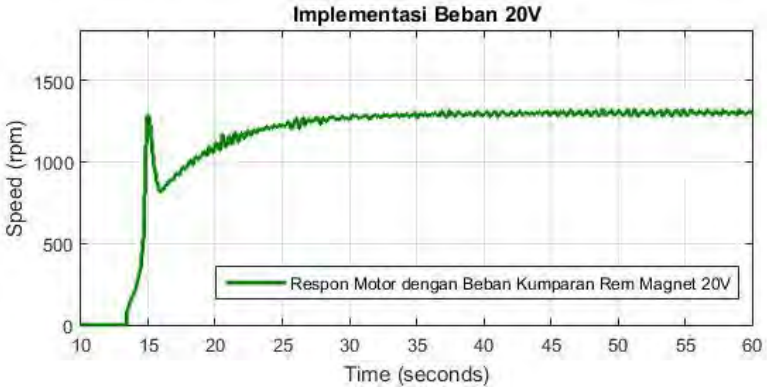
Dalam tahap implementasi ini pertama-tama menentukan parameter yang digunakan pada kontroler PID-*Robust.* Setelah menentukan parameter, selanjutnya menjalankan sistem dengan menggunakan kontroler PID-*Robust* dan telah mendapatkan nilai *gain* Proporsional, Integral dan Derivatif. Implementasi yang dilakukan dengan memberikan beban minimal, nominal, dan maksimal pada saat motor dijalankan.

Pada Implementasi diberikan nilai *setpoint* berupa sinyal *step* yang bernilai 1000 rpm. Pengujian dilakukan pada setiap beban dengan *setpoint* yang sama. Setelah itu respon dari motor dapat dianalisa dan dicari nilai *settling time, overshoot,* dan *error steady state*.

Gambar 4.12, 4.13, 4.14, merupakan respon *output plant* dengan PID-*Robust* pada kondisi beban minimal, nominal, dan maksimal. Dari gambar dapat ditarik kesimpulan bahwa pada saat diberi beban minimal, nominal, maupun maksimal, motor dapat mengikuti respon model referensi yang diinginkan.



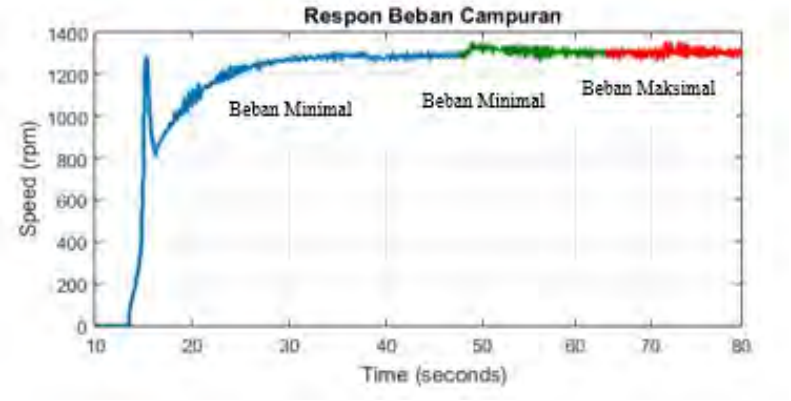
Gambar 0.12 *Step* Respon Sistem saat Beban Minimal



Gambar 0.13 *Step* Respon Sistem saat Beban Nominal

Gambar 0.14 *Step* Respon Sistem saat Beban Maksimal

Gambar 4.15 merupakan hasil respon sistem ketika motor diberi beban minimal, lalu setelah *steady* beban diubah menjadi beban nominal, dan setelah *steady* diubah menjadi beban maksimal.



Gambar 0.15 *Step* Respon Sistem saat Beban Campuran

Dari Gambar 4.15 dapat dilihat bahwa saat diberi 16 V kondisi sudah *steady* dengan waktu 30 detik. Lalu beban diubah menjadi 20 V di mana terjadi penurunan kecepatan, lalu kecepatan akan menjadi *steady*. Setelah kondisi *steady*, maka beban diubah lagi menjadi 24 V sehingga terjadi penurunan kecepatan, hingga kecepatan akan menjadi *steady* lagi. Tabel 4.4 merupakan indeks performansi *plant* pada saat implementasi dilakukan.

Tabel 0.4 Indeks Performansi *Plant* saat Implementasi

Indeks Performansi	**PID-*Robust***		
	Beban Minimal	**Beban Nominal**	**Beban Maksimal**
T_R (detik)	7,7	3,75	5,9
T_s (detik)	7,8	3,87	6,01
RMSE	0	0	0
Overshoot(%Mp)	0	0	0
Error Steady State(Ess)	0	0	0

LAMPIRAN A

A.1 Penurunan LMI untuk Performansi H_∞

Sesuai dengan teori *Schur Complement* pada Subbab 2.11, LMI (3.25) dapat disederhanakan dengan memisalkan:

$$E = \begin{bmatrix} A_i{}^T P + P A_i - K_j{}^T B_{u,i}{}^T P - P B_{u,i} K_j & P B_{w,i} \\ B_{w,i}{}^T P & \gamma^2 I \end{bmatrix}$$

$$G = F^T = [C_{z1,i} \quad 0]$$

$$H = -I \rightarrow H^{-1} = -I$$

Sehingga didapat bentuk sederhana dari LMI (3.29), yaitu

$$E - F H^{-1} G < 0 \tag{A.1}$$

LMI (A.1) memiliki bentuk seperti LMI (2.28), sehingga LMI (A.1) ekuivalen dengan

$$L < 0 \Leftrightarrow \begin{cases} H < 0 \\ E - F H^{-1} G < 0 \end{cases}$$

$$L = \begin{bmatrix} E & F \\ G & H \end{bmatrix} < 0$$

$$L = \begin{bmatrix} A_i{}^T P + P A_i - K_j{}^T B_{u,i}{}^T P - P B_{u,i} K_j & P B_{w,i} & C_{z1,i}{}^T \\ B_{w,i}{}^T P & \gamma^2 I & 0 \\ C_{z1,i} & 0 & -I \end{bmatrix} \tag{A.2}$$

Pada LMI (A.2) terdapat dua variabel LMI yang tidak diketahui dalam satu suku, yaitu matriks *P* dan matriks *K* pada suku $-K_j{}^T B_{u,i}{}^T P$ dan $-P B_{u,i} K_j$. Metode penyelesaian LMI tidak dapat dilakukan jika terdapat dua variabel LMI dalam satu suku. Untuk menyelesaikan persoalan tersebut, maka dilakukan manipulasi variabel dengan cara *premultiplying* dan *post-multiplying* LMI (A.1) dengan matriks P^{-1} sebagai berikut:

$$P^{-1}[E - F H^{-1} G] P^{-1} < P^{-1} 0 P^{-1}$$

$$P^{-1} E P^{-1} - P^{-1} F H^{-1} G P^{-1} < 0$$

$$\bar{E} - \bar{F} H^{-1} \bar{G} < 0 \tag{A.3}$$

dengan

$$\bar{E} = P^{-1}EP^{-1},\ \bar{F} = P^{-1}F \text{ , dan } \bar{G} = GP^{-1} \tag{A.4}$$

Dari LMI (A.24) dipilih matriks *E*, *F*, *G*, dan *H* yaitu

$$E = {A_i}^T P + PA_i - {K_j}^T {B_{u,i}}^T P - PB_{u,i}K_j$$

$$F = G^T = \begin{bmatrix} PB_{w,i} & {C_{z,i}}^T \end{bmatrix}$$

$$H = \begin{bmatrix} -\gamma^2 I & 0 \\ 0 & -I \end{bmatrix} \tag{A.5}$$

Substitusi (A.5) ke (A.4) akan didapat

$$\bar{E} = P^{-1}{A_i}^T + A_i P^{-1} - P^{-1}{K_j}^T {B_{u,i}}^T - B_{u,i}K_j P^{-1}$$

$$\bar{F} = \begin{bmatrix} B_{w,i} & P^{-1}{C_{z,i}}^T \end{bmatrix}$$

Dengan *Schur Complement*, LMI (A.3) ekuivalen dengan

$$L < 0 \Leftrightarrow \begin{cases} H < 0 \\ \bar{E} - \bar{F}H^{-1}\bar{G} < 0 \end{cases}$$

$$L = \begin{bmatrix} \bar{E} & \bar{F} \\ G & H \end{bmatrix} < 0$$

$$L = \begin{bmatrix} {A_i}^T P + PA_i - {K_j}^T {B_{u,i}}^T P - PB_{u,i}K_j & PB_{w,i} & P^{-1}{C_{z1,i}}^T \\ {B_{w,i}}^T & \gamma^2 I & 0 \\ C_{z1,i}P^{-1} & 0 & -I \end{bmatrix} \tag{A.6}$$

Substitusi $Q = P^{-1}$dan $Y_j = K_j P^{-1}$ pada LMI (A.4) akan didapat

$$\begin{bmatrix} {A_i}^T Q + QA_i - {K_j}^T {B_{u,i}}^T P - PB_{u,i}K_j & B_{w,i} & Q{C_{z1,i}}^T \\ {B_{w,i}}^T & -\gamma^2 I & 0 \\ C_{z1,i}Q & 0 & -I \end{bmatrix} < 0 \tag{A.7}$$

Sehingga Untuk memenuhi spesifikasi desain tersebut, maka harus terdapat matriks simteris Q yang memenuhi LMI (3.27)[7], yaitu

$$\Theta < 0$$

dan

$$\Theta_{ii} + \frac{1}{2}\left(\Theta_{ij} + \Theta_{ji}\right) < 0 \qquad ; \qquad 1 \le i \ne j \le 2 \tag{A.8}$$

dengan

$$\Theta_{ii} = \begin{bmatrix} A_i{}^T Q + Q A_i - K_j{}^T B_{u,i}{}^T P - P B_{u,i} K_j & B_{w,i} & Q C_{z1,i}{}^T \\ B_{w,i}{}^T & -\gamma^2 I & 0 \\ C_{z1,i} Q & 0 & -I \end{bmatrix}$$

$$\varepsilon_Q = \{x | x^T Q x \leq 0\} \qquad \text{(A.10)}$$

$$Q = P^{-1} \quad \text{dan} \quad Y_j = K_j P^{-1}$$

Halaman ini sengaja dikosongkan

LAMPIRAN B

B.1 Program Arduino

```
int pin = 7;
int out = 6;
byte u = 255;
unsigned long duration;
unsigned long spd=0;
word kec;
byte w1;
byte w2;

void setup(){
  Serial.begin(115200);
  pinMode(pin,INPUT);
  pinMode(out,OUTPUT);
  analogWrite(out,u);}

void loop(){
  if (Serial.available()) {
    u=255-Serial.read();
    analogWrite(out,u);
    baca();}}

void baca(){
  duration = pulseIn(pin, LOW, 30000);
     if (duration == 0 && i<3){
        spd = 0;}
      else{
        spd=(15*1000000/(2*duration));}
  kec=int(spd);
  w1=kec/256;
  w2=kec%256;
  Serial.write(w1);
 Serial.write(w2);}
```

B.2 Program untuk Menghitung *Linear Matrix Inequality*

```
%----------Kontrol LMI Robust--------------------%
%Regulator Kecepatan Motor BLDC
%Menentukan nilai K LMI untuk paramater Kontroler
Kp,Ki,Kd
%-----------------------------------------------%
%-------------Ahmad Fachrudin Istiananda----------%
%-------------------2212100181--------------------%

%Persamaan state plant
A=[-28.9879 -134.5795 0;1 0 0;0 1 0];   %Nominal
B=[122.2386;0;0];

%Spesifikasi desain yang digunakan
Bw=B;
C=[0 1 0];

%Tingkat pelemahan maksimal
gam=0.5;

%Insialisasi LMI
setlmis([]);
Q=lmivar(1,[3 1]);
Y1=lmivar(2,[1 3]);

%LMI
lmiterm([1 1 1 Q],A,1,'s')
% LMI #1: A*Q+Q*A'
lmiterm([1 1 1 Y1],B,-1,'s')
% LMI #1: -Bu*Y1-Y1'*Bu'
lmiterm([1 2 1 0],Bw')
% LMI #1: Bw1'
lmiterm([1 2 2 0],-(gam^2))
% LMI #1: -(gam^2)*I
lmiterm([1 3 1 Q],C,1)
% LMI #1: Cz*Q

lmiterm([1 3 3 0],-1)
% LMI #1: -I

lmiterm([-2 1 1 Q],1,1)
% LMI #2: Syarat P>0
```

```
lmis=getlmis;
[tm,xf]=feasp(lmis);

%Hasil matriks Q, Y1, P, K1
Q=dec2mat(lmis,xf,Q)
Y1=dec2mat(lmis,xf,Y1)
P=inv(Q)
K1=Y1*P
```

BAB 5
PENUTUP

Pada Bab ini akan dibahas mengenai hasil akhir dari pengerjaan Tugas Akhir meliputi kesimpulan dan saran sebagai referensi tambahan agar penelitian yang akan dilakukan setelahnya dapat lebih baik lagi.

5.1. Kesimpulan

Dari analisa yang telah dilakukan terhadap hasil simulasi maka dapat disimpulkan bahwa pengaturan kecepatan *Brushless* Motor DC (BLDC) menggunakan PID-*Robust* berbasis performansi H_∞ menghasilkan respon sistem yang dapat mengikuti respon model referensi yang diharapkan. Besar nilai γ akan mempengaruhi nilai ∞-norm yang akan mempengaruhi besar kecepatan motor ketika diberi *disturbance*. Pada implementasi kontroler, respon plant dapat mengikuti model referensi namun masih ada eror yang terjadi saat peningkatan kecepatan motor.

5.2. Saran

Untuk penelitian selanjutnya disarankan agar membuat *sensor* arus untuk membaca nilai arus pada rem elektromagnetik dikarenakan pada perancangan Tugas Akhir ini dianggap bahwa resistansi dari rem elektromagnetik tetap sehingga nilai arus akan sebanding dengan nilai tegangan.

DAFTAR PUSTAKA

[1] L. K. Baxter, Capacitive Sensors: Design and Applications, John Wiley & Sons, 1996.

[2] Guifang CAI, Kun QIAN, dkk "*Robust PID Controller in Brushless DC Motor Application*", IEEE International Conference on Control and Automation, Guangzhou, CHINA, 2007.

[3] Kazuo Tanaka, Takayuki Ikeda, dan Hua O. Wang, "Robust Stabilization of a Class of Uncertain Nonlinear Systems via Fuzzy Control: Quadratic Stabilizability, H∞ Control Theory, and Linear Matrix Inequalities," *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 1, February 1996.

[4] Hall Effect Sensing and Application, Illinois: Honeywell Inc

[5] W. J. Brin, "Design and Frabrication of an Eddy Current Brake Dynamometer for Efficiency Determination of Electric Wheelchair Motor," Wright State University, Fairbor, 2013.

[6] Kho, Dickson, "*Pengertian Optocoupler dan Prinsip Kerjanya*", <URL: http://teknikelektronika.com/pengertian-*optocoupler*-fungsi-prinsip-kerja-*optocoupler*/>, April, 2014..

[7] H. D. Tuan, P. Apkarian, T. Narikiyo, dan Y. Yamamoto, "Parameterized Linear Matrix Inequality Techniques in Fuzzy Control System Design," *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 2, pp. 324-332, April 2001

[8] Katsuhiko Ogata, *Modern Control Engineering*, 3rd ed. New Jersey: Prentice-Hall, 1997.

[9] E. Cerruto, A. Consoli dkk, "A robust adaptive controller for PM motor drives in robotic application", *IEEE Trans. Ind. Applicat*, 1992

[10] Kemin Zhou, *Essentials of Robust Control*. New Jersey: PrenticeHall, May 1999.

[11] Stephen Boyd, Laurent El Ghaoui, Eric Feron, dan Venkataramanan Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. Philadelphia: Society for Industrial and Applied Mathematics, 1994.

[12] ....., "*Arduino*", <URL: http://arduino.cc/>, Mei, 2016.

[13] Elrosa, Ilmiyah, "Traction Control pada Parallel Hybrid Electric Vehicle dengan Metode Generalized Predictive Control", *Tugas Akhir*, Jurusan Teknik Elektro ITS Surabaya, 2014.

[14] Ljung, Lenart dan Torkel, Glad, "*Modelling of Dynamic Systems*", Prentice – Hall International: New Jersey. 1994.

[15] Husaini, Achmad Nur, "*Prinsip Kerja Motor Brushless DC (BLDC Motor)*", <URL: http://www.insinyoer.com/prinsip-kerja-motor-brushless-dc-bldc-motor/3/>, September, 2015.

[16] Febriarianto, Tito, "*Desain Kontrol Fuzzy Berbasis Performansi H∞ dengan Batasan Input-Output untuk Sistem Pendulum-Kereta"*, Tugas Akhir Jurusan Teknik Elektro ITS Surabaya, 2012.

[17] Putra, Guntur Shadiea, "*Perancangan Kontrol Kecepatan Motor Arus Searah Tanpa Sikat Menggunakan Sliding Mode Berbasis PID"*, Tugas Akhir Jurusan Teknik Elektro ITS Surabaya, 2015.

RIWAYAT HIDUP

Ahmad Fachrudin Istiananda adalah nama lengkap penulis yang dikenal dengan nama panggilan Fachry. Penulis lahir di Ibukota Indonesia, Jakarta pada tanggal 18 November 1994 yang merupakan anak terakhir dari tiga bersaudara pasangan Sri Widyastuti dan Iskandar Hadi. Penulis memulai pendidikannya di Depok. SD, SLTP, dan SMA Islam Nurul Fikri. Setelah lulus dari SMA pada tahun 2012, penulis melanjutkan studi di Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nompeber Surabaya melalui jalur SPMB pada tahun yang sama. Konsentrasi penulis adalah pada bidang studi Teknik Sistem Pengaturan dan selama kuliah, penulis aktif menjadi asisten praktikum Sistem Pengaturan Analog dan menjadi koordinator praktikum Sistem Pengaturan Digital pada praktikum Sistem Pengaturan Digital dan Otomasi Sistem. Selain itu, penulis juga menjadi tim pengembang Robotic pada tim AIRRG (*Automation and Industrial Robot Research Group*). Pada bulan Mei - Juni 2016 penulis mengikuti seminar dan ujian Tugas Akhir sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Teknik Elektro.

e-mail : ahmad.fachrudin12@mhs.ee.its.ac.id
ahmd.fachrudin@gmail.com