SISTEM PENGENDALI KECEPATAN MOTOR DC PADA KONVEYOR BARANG MENGGUNAKAN KONTROLER PI BERBASIS ARDUINO UNO

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI TEKNIK KONTROL

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

RAY SELVY FIRMANSYAH PUTRA
NIM. 125060307111027

UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2016

LEMBAR PENGESAHAN

SISTEM PENGENDALI KECEPATAN MOTOR DC PADA KONVEYOR BARANG MENGGUNAKAN KONTROLER PI BERBASIS ARDUINO UNO

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI TEKNIK KONTROL

Ditujukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

RAY SELVY FIRMANSYAH PUTRA

NIM. 125060307111027

Skripsi ini telah direvisi dan disetujui oleh dosen pembimbing
pada tanggal 08 Juni 2016

Dosen Pembimbing I	**Dosen Pembimbing II**
Ir. Purwanto, M.T.	**Ir. Moch. Rusli, Dipl. Ing.**
NIP. 19540424 198601 1 001	**NIP. 19630104 198701 1 001**

Mengetahui

Ketua Jurusan

M. Aziz Muslim, S.T, M.T., Ph.D.

NIP. 19741203 200012 1 001

JUDUL SKRIPSI:

SISTEM PENGENDALI KECEPATAN MOTOR DC PADA KONVEYOR BARANG MENGGUNAKAN KONTROLER PI BERBASIS ARDUINO UNO.

Nama Mahasiswa : RAY SELVY FIRMANSYAH PUTRA

NIM : 125060307111027

Program Studi : TEKNIK ELEKTRO

Konsentrasi : TEKNIK KONTROL

Komisi Pembimbing :

Ketua : Ir. PURWANTO, M.T. .....

Anggota : Ir. MOCH. RUSLI, Dipl.-Ing. .....

TIM DOSEN PENGUJI :

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

Dosen Penguji 2 : Dr. Ir. ERNI YUDANINGTYAS, M.T.....

Dosen Penguji 3 : GOEGOES DWI N., S.T., M.T. .....

Tanggal Ujian : 03 JUNI 2016

SK Penguji : 691/UN10.6/SK/2016

PERNYATAAN ORISINALITAS SKRIPSI

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

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

Malang, 25 Mei 2016

Mahasiswa,

RAY SELVY FIRMANSYAH PUTRA

NIM. 125060307111027

Teriring Ucapan Terima Kasih kepada:

Ayahanda dan Ibunda tercinta

RINGKASAN

Ray Selvy Firmansyah Putra, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Mei 2016, *Sistem Pengendali Kecepatan Motor DC Pada Konveyor Barang Menggunakan Kontroler PI Berbasis Arduino Uno*, Dosen Pembimbing: Purwanto dan Mochammad Rusli.

Konveyor merupakan suatu sistem mekanik yang mempunyai fungsi memindahkan barang dari satu tempat ke tempat yang lain. Konveyor banyak dipakai di industri untuk transportasi barang yang jumlahnya sangat banyak dan berkelanjutan. Dalam kondisi tertentu, konveyor banyak dipakai karena mempunyai nilai ekonomis dibanding transportasi berat seperti truk dan mobil pengangkut.

Komponen utama pada konveyor salah satunya yaitu motor *direct current* (DC), yang mana berfungsi sebagai penggerak pada konveyor. Oleh sebab itu, motor DC harus dijaga agar dapat mempertahankan kecepatan putar pada konveyor saat terjadi gangguan berupa perubahan berat beban material dengan tetap mempertahankan keluaran sistem sesuai dengan *setpoint* yang diinginkan. Hal ini sangat penting dilakukan agar konveyor dapat berjalan dengan baik dan lancar. Sehingga untuk mengatasi masalah tersebut yaitu melakukan pengendalian kecepatan motor DC pada konveyor barang dengan menggunakan kontroler PI.

Respon motor DC dengan kontroler PI yang memiliki penguatan K_p = 1.6326 dan K_i = 5 yang didapat menggunakan metode *root locus* setelah implementasi dengan *setpoint* 145 rpm dan gangguan berupa timbel dengan berat 0.5 kg, 1 kg, serta 1.5 kg, memiliki *error steady state* berada dibawah toleransi 2% dengan *settling time* masing-masing 12.4 ms, 16.8 ms, dan 21.7 ms. Setelah mengalami perlambatan ketika diberi gangguan, sistem dapat kembali pada keadaan *steady state* sesuai *setpoint* dengan *recovery time* yang sangat cepat. Sehingga dapat diketahui bahwa kontroler bekerja dengan baik.

Kata kunci : konveyor, motor DC, kontroler PI, *root locus*.

SUMMARY

Ray Selvy Firmansyah Putra, Department of Electrical Engineering, Faculty of Engineering University of Brawijaya, Mei 2016, "*Speed Control of DC Motor On Conveyor Using PI Controller Based Arduino Uno*", Academic Supervisor: Purwanto and Moch. Rusli.

Conveyor is a mechanical system that has the function of moving goods from one place to another. Conveyor is widely used in industry to transport vast quantities of goods and sustainable. Under certain conditions, the conveyor is widely used because it has economic value compared to the heavy transport such as trucks and vans.

The main components on a conveyor one of which is motor direct current (DC), which serves as a driver on a conveyor. Therefore, the DC motor must be maintained in order to maintain the rotational speed on a conveyor during disturbances include changes in the weight of material while maintaining the output of the system in accordance with the desired setpoint. It is very important to keep the conveyor can run well and smoothly. So as to overcome the problem is to control the DC motor speed on a conveyor goods by using the PI controller.

Response DC motor with PI controller that has a strengthening Kp = 1.6326 and Ki = 5 were obtained using the method of *root locus* after the implementation of the setpoint 145 rpm and disruption of lead by weight of 0.5 kg, 1 kg and 1.5 kg, has an error steady state under a tolerance of 2% with a settling time of each 12.4 ms, 16.8 ms and 21.7 ms. After experiencing a slowdown when given disorder, the system can return to the steady state in accordance setpoint with a very fast recovery time. So it can be seen that the controller works well.

Keywords : *conveyor, DC motor, PI controller, root locus*

KATA PENGANTAR

Alhamdulillâh, segala puji hanya bagi Allâh Subhanahu Wa Taála, Rabb alam semesta. Dialah Allâh, Tuhan Yang Maha Satu, Yang Maha Pengasih lagi Maha Penyayang. Dialah Sebaik baik Penolong dan Sebaik baik Pelindung. Shalawat dan salâm kepada Nabi Muhammad Rasulullâh Shallallâhu Alaihi Wa Salâm, Sang pembawa kabar gembira dan sebaik baik suri tauladan bagi yang mengharap Rahmat dan Hidayah-Nya.

Sungguh hanya melalui Pertolongan dan Perlindungan Allâh SWT semata sehingga saya dapat menyelesaikan skripsi ini. Dengan seizin Allâh SWT, di kesempatan yang baik ini saya ingin menghaturkan rasa terima kasih dan penghargaan yang sebesar besarnya atas bantuan sehingga terselesainya skripsi ini kepada:

- Keluarga tercinta, kedua orang tua Dasril Abdullah dan Suci Mulyati yang selalu memberikan kasih sayang dan doanya yang tiada akhir dan adik-adikku yang selalu memberikan semangat.
- Budhe Tutiati, mas Gagut, mas Anas, mas Tyar, mas Agung, mas Dhani, serta seluruh keluarga besar atas dukungan dan doanya.
- Bapak M. Aziz Muslim, ST., MT., Ph.D selaku Ketua Jurusan Teknik Elektro Universitas Brawijaya.
- Bapak Ir. Purwanto, MT., selaku KKDK Teknik Kontrol sekaligus sebagai dosen pembimbing I yang telah banyak memberikan kesempatan, nasehat, pengarahan, motivasi, saran dan masukan yang telah diberikan.
- Bapak Ir. Moch. Rusli, Dipl.-Ing., selaku pembimbing II yang telah memberikan banyak waktu dan ilmunya selama bimbingan.
- Bapak Raden Arief Setyawan, ST., MT., selaku dosen pembimbing akademik yang telah meluangkan waktu dalam memberi bimbingan akademik.
- Bapak Ibu dosen, karyawan, staf recording dan RBTE atas segala bantuan dan kemudahan.
- Mbak Eka selaku laboran Lab. Sistem Kontrol atas segala bantuan dan kemudahannya.
- Keluarga besar Lab. Sistem Kontrol dan seluruh laboratorium di lingkungan teknik elektro atas segala bantuannya.
- Teman-teman LSO Forsitek terima kasih atas kerjasamanya.

- Keluarga besar kelas C angkatan 2012 terima kasih atas kebersamaannya dan keceriannya.
- Teman-teman kos MT. Haryono 60B, Ridho, Firman, Eril, Nala, bli Ahmad, Eko, Frengky, Epot, Waduk terima kasih telah berbagi kesenangan, pelajaran hidup, serta canda dan tawa.
- Teman main Vrisco, Malik, Pak tyo, Budi, Ridwan, Mete, Faris, Reza, Jaka, Andhika, Mahes, Danang, Tyo Pur, Graha, Kemas, Yudha, Dito, Dirga atas kebersamaannya.
- Teman-teman komunitas Bar Rampal Malang atas dukungan, doa, kebersamaan, serta keceriannya.
- Bapak Parman dan Ibu Aminah selaku pemilik kos terima kasih atas segala keramahan dan kesabarannya.
- Teman-teman teknik elektro angkatan tahun 2012 serta semua pihak yang telah memberikan bantuan serta dukungan baik secara langsung maupun tidak langsung atas penyusunan skripsi ini.

Sekiranya Allâh SWT mencatat amalan ikhlas kami dan semua pihak yang turut membantu sehingga skripsi ini terselesaikan. Akhirnya, penulis menyadari bahwa skripsi ini masih jauh dari sempurna namun semoga skripsi ini dapat memberikan manfaat bagi kita semua. Aamiin

Malang, 25 Mei 2016

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi dewasa ini adalah sangat pesat. Seiring dengan perkembangan dan kemajuan teknologi, dalam dunia industri segalanya dituntut serba cepat untuk memenuhi kebutuhan konsumen. Hampir semua perusahaan menerapkan otomatisasi pada alat-alatnya untuk menunjang proses produksi yang cepat, efisien, dan tidak membutuhkan banyak tenaga kerja.

Konveyor merupakan alat yang sangat vital dalam dunia industri. konveyor itu sendiri merupakan suatu sistem mekanik yang mempunyai fungsi memindahkan barang dari satu tempat ke tempat yang lain. Konveyor pada industri digunakan untuk transportasi barang yang jumlahnya sangat banyak dan berkelanjutan. Dalam kondisi tertentu, konveyor banyak dipakai karena mempunyai nilai ekonomis dibanding transportasi berat seperti truk dan mobil pengangkut (Ade A., 2014).

Konveyor tersusun dari berbagai macam komponen salah satunya motor *direct current* (DC), yang mana berfungsi sebagai penggerak pada konveyor. Oleh sebab itu, motor DC harus dijaga agar dapat mempertahankan kecepatan putar pada konveyor saat terjadi gangguan berupa perubahan berat barang produksi pada konveyor dengan tetap mempertahankan keluaran sistem sesuai dengan *setpoint*. Hal ini sangat penting dilakukan agar konveyor dapat berjalan dengan baik dan lancar. Sehingga tidak mengganggu kegiatan produksi yang sedang berlangsung.

Solusi yang dapat diterapkan untuk mengatasi masalah tersebut adalah melakukan pengendalian kecepatan motor DC pada konveyor. Dengan mengatur kecepatan motor DC pada konveyor barang tersebut, diharapkan dapat memindahkan barang dengan waktu yang sama meskipun berat barang yang dipindahkan berbeda-beda dengan batasan yang telah ditentukan.

Pengendalian kecepatan motor DC pada konveyor dilakukan dengan menggunakan kontroler PI. Kontroler PI merupakan gabungan dari kontroler proporsional dan kontroler integral. Keuntungan dari kontroler PI yaitu sebuah sistem yang sederhana sehingga lebih cepat dalam mengambil sebuah keputusan. Diharapkan

dengan menggunakan kontroler PI performa sistem yang didapatkan menjadi lebih cepat, menghilangkan *offset* dan menghasilkan perubahan awal yang besar.

Pada skripsi ini akan dibuat suatu desain pengendali yang digunakan sebagai pengendali kecepatan motor DC pada konveyor barang menggunakan kontroler proporsional integral (PI) dan juga dilengkapi dengan metode *root locus* atau letak kedudukan akar untuk menentukan parameternya.

1.2 Rumusan Masalah

Berdasarkan latar belakang dapat disusun rumusan masalah sebagai berikut:

1. Bagaimana perancangan algoritma dan performansi sistem pengendali kecepatan motor DC pada konveyor barang menggunakan kontroler PI?
2. Bagaimana respon sistem tanpa kontroler dan dengan penambahan kontroler jika diberi gangguan?

1.3 Batasan Masalah

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

a) *Plant* yang digunakan adalah sebuah *prototipe* konveyor barang milik Laboratorium Sistem Kontrol Teknik Elektro Universitas Brawijaya yang telah dikopel dengan dengan motor DC.
b) Mikrokontroler yang digunakan adalah Arduino Uno R3.
c) Sensor yang digunakan adalah *rotary encoder* E40S6-360-6-L-5 dengan *range* pulsa 0-360 per rotasi.
d) *Driver* motor yang digunakan adalah H-Bridge Infineon BTS 7960.
e) Konveyor yang digunakan memiliki panjang 80 cm dan lebar 16 cm.
f) Motor DC mempunyai spesifikasi, catu day 12 V – 24 V, arus 15 A dan torsi 70 kg.cm serta telah dilengkapi dengan *gear box*.
g) *Belt* konveyor yang digunakan merupakan kain.
h) Dalam penelitian ini slip yang terjadi pada konveyor tidak diperhitungkan.
i) Gangguan yang diberikan berupa timbel dengan berat timbel sebesar 0.5 kg, 1 kg, dan 1.5 kg.
j) Pembahasan ditekankan pada penggunaan kontroler PI pada sistem, untuk masalah elektrik dan sistematis tidak dibahas secara mendalam.

1.4 Tujuan

Tujuan dari penelitian ini adalah terwujudnya sistem yang dapat mempertahankan kecepatan putar motor DC pada konveyor barang saat terjadi gangguan berupa perubahan berat beban material pada konveyor dengan tetap mempertahankan keluaran sistem sesuai *setpoint*.

1.5 Sistematika Pembahasan

Adapun sistematika dari penulisan skripsi ini adalah:

BAB I PENDAHULUAN

Menjelaskan tentang latar belakang, rumusan masalah, batasan masalah, tujuan, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Menjelaskan dasar teori yang mendukung dalam pembuatan alat dan perancangan algoritma yang terdiri atas mikrokontroler, rotary encoder, motor dc, konveyor, kontroler PI, dan metode *root locus*.

BAB III METODELOGI PENELITIAN

Bab ini menjelaskan bagaimana melakukan pengujian alat-alat yang digunakan untuk membuat sistem pengendali kecepatan motor dc pada konveyor menggunakan kontroler PI dengan metode *root locus* serta penentuan fungsi alih *plant*, *driver*, sensor dan pembuatan algoritma.

BAB IV HASIL DAN PEMBAHASAN

Membahas hasil pengujian sistem, simulasi sistem dan analisis data secara keseluruhan terhadap alat yang telah direalisasikan.

BAB V KESIMPULAN DAN SARAN

Menjelaskan tentang pengambilan kesimpulan sesuai dengan hasil perancangan algoritma dan pengujian alat serta saran yang diperlukan untuk dilakukan pengembangan selanjutnya.

Halaman ini sengaja dikosongkan

BAB II

TINJAUAN PUSTAKA

2.1 Konveyor

Konveyor merupakan suatu sistem mekanik yang mempunyai fungsi memindahkan barang dari satu tempat ke tempat yang lain. Konveyor banyak dipakai di industri untuk transportasi barang yang jumlahnya sangat banyak dan berkelanjutan. Dalam kondisi tertentu, konveyor banyak dipakai karena mempunyai nilai ekonomis dibanding transportasi berat seperti truk dan mobil pengangkut (Ade A., 2104).

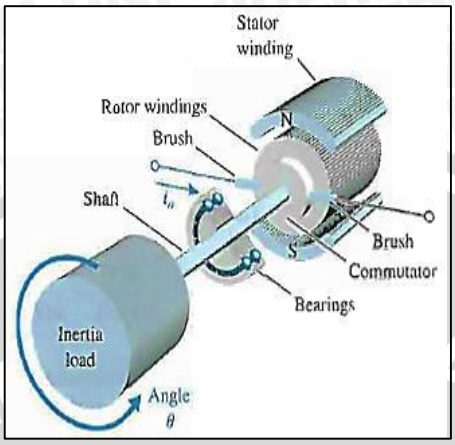
Jenis konveyor membuat penanganan alat berat tersebut / produk lebih mudah dan lebih efektif. Banyak konveyor rol dapat bergerak secepat 75 kaki / menit. Konveyor dapat memobilisasi barang dalam jumlah banyak dan kontinyu dari satu tempat ke tempat lain. Perpindahan tempat tersebut harus mempunyai lokasi yang tetap agar sistem konveyor mempunyai nilai ekonomis (Surirah, 2009). Gambar 2.1 merupakan konveyor barang.

Gambar 2.1 Konveyor Barang
(Sumber: www.midasmakina.com)

2.2 Motor *Direct Current* (DC)

Motor *direct current* (DC) memerlukan sumber tegangan yang searah pada kumparan medan untuk diubah menjadi energi mekanik. Bagian utama motor DC pada Gambar 2.2 adalah stator (bagian yang tidak berputar) dan kumparan jangkar disebut rotor (bagian yang berputar). Tegangan sumber DC dari *power supply* atau baterai menuju

ke lilitan melalui sikat yang menyentuh komutator (dua segmen yang terhubung dengan dua ujung lilitan). Kumparan dalam satu lilitan disebut angker dinamo. Angker dinamo adalah sebutan untuk komponen yang berputar diantara medan magnet.



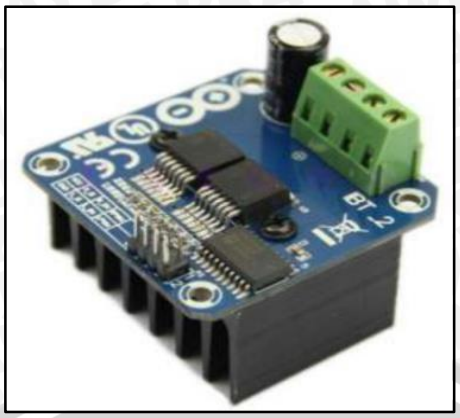
Gambar 2.2 Elemen-Elemen Dasar Motor DC
(Sumber: Dorf and Robert, 2008)

Prinsip kerja motor DC adalah jika sikat arang terhubungkan dengan satu sumber arus serah diluar dengan tegangan V, maka satu arus I masuk ke terminal kumparan rotor dan menghasilkan fluks. Dengan adanya fluks stator dan arus rotor akan menghasilkan satu gaya yang bekerja pada kumparan yang dikenal dengan gaya lorentz.

2.3 Driver Motor

Driver motor berfungsi untuk mengubah sinyal PWM dari mikrokontroler menjadi tegangan. Dalam aplikasinya *driver* motor biasanya tersusun dari rangkaian transistor-transistor yang tersusun sedemikian rupa sehingga mampu mengendalikan arah putar dan kecepatan motor berdasarkan arah *loop* dan tegangan kutub motor.

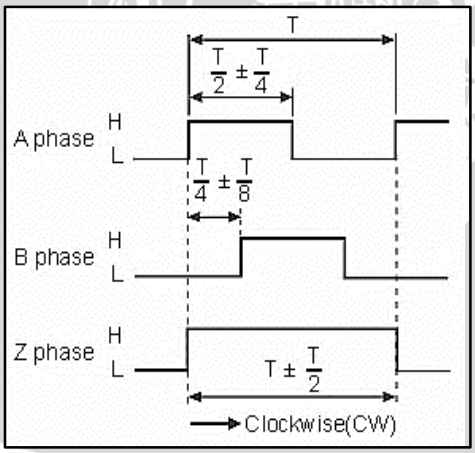
Driver yang akan digunakan adalah *driver* motor BTS 7960 dengan metode *H-bridge*. Rangkaian ini diberi nama *H-bridge* karena bentuk rangkaiannya yang menyerupai huruf H. Motor DC yang digunakan mempunyai rains tegangan 0-24 V sedangkan PWM output Arduino UNO R3 mempunyai rains tegangan 0-5 V sehingga dibutuhkan *driver* motor (Gambar 2.3) untuk memaksimalkan kinerja motor DC. Untuk arusnya sendiri yaitu *high current 30A DC motor driver* (*two high current half bridge Infineon BTS 7960 chip*).



Gambar 2.3 Driver Motor H-Bridge BTS9760
(Sumber: Infineon Technologies Datasheet)

2.4 *Incremental Rotary Encoder*

Incremental rotary encoder merupakan sebuah sensor untuk mengukur kecepatan yang berjenis *increment*. Sensor *rotary encoder* yang digunakan yaitu *rotary encoder* autonics E40 Series. Prinsip kerja *Incremental rotary encoder* adalah mengukur nilai sesaat posisi angular dari sebuah *shaft* yang sedang berotasi dan menghasilkan pulsa-pulsa pada chanel-chanelnya. Pulsa-pulsa yang dihasilkan ini berbentuk gelombang *square*. *Incremental rotary encoder* biasanya memiliki tiga buah sinyal keluaran, sinyal A, sinyal B, dan sinyal Z yang ditunjukkan pada Gambar 2.4 dan *Incremental rotary encoder* yang akan digunakan dapat dilihat pada Gambar 2.5.



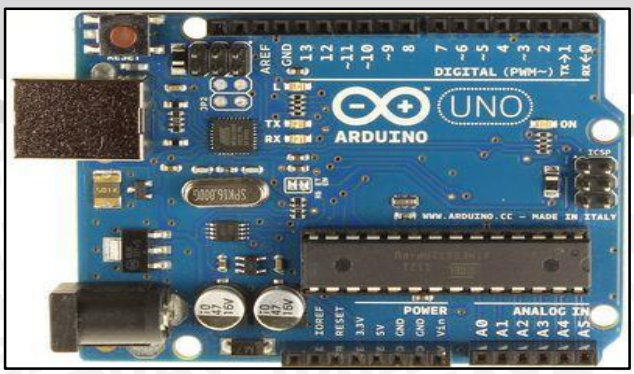
Gambar 2.4 Sinyal keluaran *rotary encoder*
(Sumber: Autonics E40 Series Datasheets)

Untuk kebanyakan peralatan mesin motor atau aplikasi *positioning*, sinyal Z dikenal sebagai indeks sinyal yang memilki peranan penting dalam menentukan *zero position* dengan cara memberikan sebuah pulsa keluaran tunggal per satu revolusi (Morris, S Alan).

Gambar 2.5 Increment Rotary Encoder E40 Series
(Sumber: Autonics E40 Series Datasheets)

2.5 Mikrokontroler

Mikrokontroler adalah suatu kontroler digital yang berbentuk suatu *Integrated Circuit* (IC) dengan kepadatan yang sangat tinggi, dimana semua bagian yang diperlukan untuk suatu kontroler sudah dikemas dalam satu keping (dalam bentuk modul), biasanya terdiri dari *Central Processing Unit* (CPU), *Random Access Memory* (RAM), *Electrically Erasable Programmable Read-Only Memory* (EEPROM), *Input/Output* (I/O), *timer,* dan *interupt controller*. Modul mikrokontroler merupakan sebuah alat dimana terdiri dari mikrokontroler dan *minimum system* sehingga mikrokontroler siap digunakan, salah satu contohnya yaitu modul mikrokontroler Arduino Uno R3 (Gambar 2.6). Arduino atau genuino adalah papan mikrokontroler yang menggunakan IC ATmega328P yang mempunyai 14 digital pin *input* atau *output* (yang mana 6 pin bisa digunakan sebagai PWM *output*), 6 analog *input*, 16 MHz quartz crystal (clock), fasilitas *Universal Serial Bus* (USB) *connection*, sebuah *power jack Direct Current* (DC), sebuah *In-Circuit Serial Programming* (ICSP) dan satu buah tombol reset. Spesifikasi dari miokrokontroler Arduino Uno R3 dapat dilihat pada Table 2.1.



Gambar 2.6 Arduino Uno R3
(Sumber: www.arduino.cc)

Tabel 2.1 Spesifikasi Arduino Uno R3
(Sumber: www.arduino.cc/Main/ArduinoBoardUno)

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

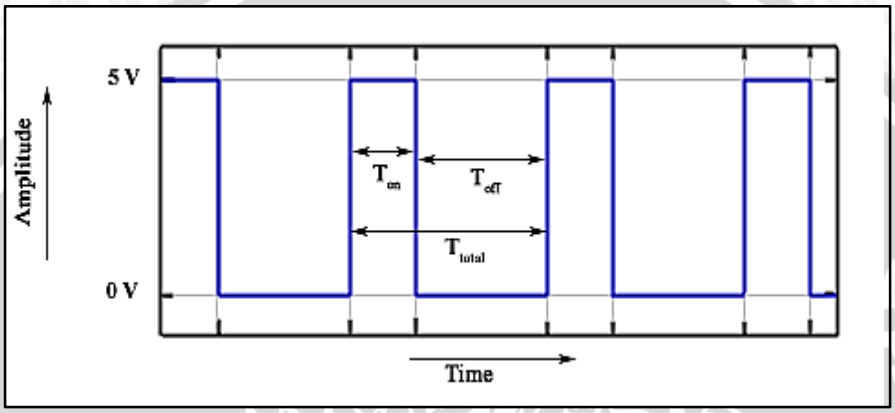
Sumber tengangan Arduino Uno R3 dapat diaktifkan melalui koneksi USB atau dengan daya eksternal. Eksternal (non-USB) daya dapat berasal baik dari adaptor maupun baterai. Adaptor ini dapat dihubungkan dengan menancapkan *plug jack* pusat positif ukuran 2.1 mm konektor power. Ujung kepala dari baterai dapat dimasukkan kedalam *ground* dan vcc pin header dari konektor power. Arduino dapat beroperasi dengan catu daya eksternal 6 V sampai 20 V. Namun jika menggunakan lebih dari 12 V, regulator tegangan bisa panas dan merusak papan. Kisaran yang disarankan adalah 7 V sampai 12 V. Arduino juga menyediakan *Software Serial Library* memungkinkan komputer untuk berkomunikasi secara serial pada salah satu pin digital pada *board* Arduino Uno R3.

2.6 *Pulse Width Modulation*

Pengaturan tegangan sumber biasanya menggunakan metode *Pulse Width Modulation* (PWM). Sinyal *Pulse Width Modulation* (PWM) adalah metode yang dapat digunakan

untuk mengontrol kecepatan motor DC. Dimana kecepatan motor DC tergantung pada besarnya *duty cycle* yang diberikan pada motor DC tersebut.

Pada sinyal PWM, frekuensi sinyal konstan sedangkan *duty cycle* bervariasi dari 0%-100%. *Duty cycle* adalah besarnya sinyal kontrol yang diberikan pada motor. Persamaan untuk perhitungan *duty cycle* ditunjukkan pada Persamaan 2.1 dengan Ton adalah periode logika tinggi, dan T adalah periode keseluruhan. Sinyal PWM secara umum dapat dilihat dalam Gambar 2.7.



Gambar 2.7 Sinyal PWM secara umum
(Sumber: *www.8051projects.net*)

$$Duty\ Cycle\ \frac{T_{on}}{T_{total}} \times 100\% \quad (2.1)$$

Sedangkan frekuensinya dapat ditentukan dengan Persamaan 2.2

$$fOCn\ \frac{fclk\ I/O}{N\ .\ 256} \quad (2.2)$$

Timer atau *counter* yang digunakan pada PWM ini yaitu *timer* atau *counter* 0 (8 bit) dengan metode fast PWM dan *prescaler factor* (N) yaitu 256.

2.7 Kontroler

Kontroler dalam sebuah sistem kontrol mempunyai kontribusi yang besar terhadap perilaku sistem. Hal itu disebabkan oleh tidak dapat dirubahnya komponen penyusun sistem tersebut. Artinya, karakteristik *plant* harus diterima sebagaimana adanya, perubahan perilaku sistem hanya dapat dilakukan melalui penambahan kontroler.

Salah satu fungsi kontroler adalah mengurangi sinyal *error*, sinyal *error* adalah perbedaan nilai *setpoint* dengan nilai *output plant*. *Setpoint* adalah nilai referensi atau nilai yang diinginkan, sedangkan *output plant* adalah nilai aktual yang terukur pada

output plant. Semakin kecil nilai sinyal *error* maka kinerja sistem kontrol dinilai semakin baik.

Prinsip kerja kontroler adalah membandingkan nilai *output plant* dengan nilai *setpoint*, menentukan nilai kesalahan dan akhirnya menghasilkan sinyal kontrol untuk meminimalkan kesalahan (Ogata K,1995).

2.7.1 Kontrol Proporsional (P)

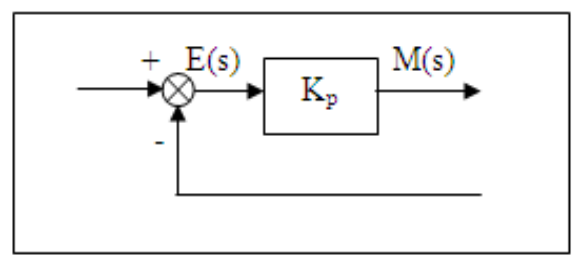
Kontroler proporsional memiliki *output* yang besarnya sebanding dengan besarnya sinyal *error*. *Output* kontroler merupakan perkalian antara penguatan proporsional dengan sinyal *error*, yang dapat dituliskan sebagai berikut:

$m(t)=K_p\, e(t)$ (2.3)

atau, dalam besaran transformasi Laplace,

$\frac{M(s)}{E(s)} = Kp$ (2.4)

Apapun wujud mekanisme yang sebenarnya dan apapun bentuk daya penggeraknya, kontroler proporsional pada dasarnya merupakan penguat dengan penguatan yang dapat diatur (Ogata K.,1997). Diagram blok kontroler proporsional dapat dilihat pada Gambar 2.8.



Gambar 2.8 Diagram Blok Kontroler Proposional
(Sumber: Ogata K., 1997)

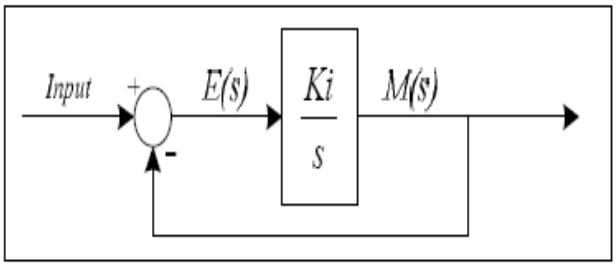
2.7.2Kontroler Integral (I)

Pada kontroler dengan aksi integral, harga keluaran kontroler m(t) diubah dengan laju yang sebanding dengan sinyal kesalahan penggerak *e(t).*

Jadi,

$\frac{dm(t)}{dt} = Kie(t)$ (2.5)

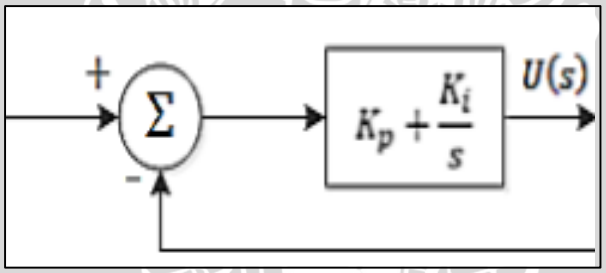
Jika harga e(t) diduakalikan, maka harga m(t) berubah dengan laju perubahan menjadi dua kali semula. Jika kesalahan penggerak nol, maka harga m(t) tetap stasioner. Aksi kontrol integral seringkali disebut kontrol *reset* (Ogata K.,1997).



Gambar 2.9 Diagram Blok Kontroler Integral
(Sumber: Ogata K., 1997)

2.7.3 Kontroler Proporsional Integral (PI)

Kontroler *Proportional Integral* (PI) (Gambar 2.10) memiliki sifat yang tidak mengeluarkan respon sebelum selang waktu tertentu, pada kontrol P suatu *plant* yang fungsi alihnya tidak mempunyai integrator 1/s, terdapat *error steady state* (*offset*) pada masukan tangga atau *unit step*. *Offset* semacam ini dapat dihilangkan jika ditambahkan aksi integral pada kontroler. kontroler I mengakibatkan respon menjadi lambat walaupun dapat menghilangkan *error steady state* (*offset*). Dalam memperbaiki respon yang lambat tersebut, umumnya kontroler I dipasang paralel dengan kontroler P.



Gambar 2.10 Diagram Blok Kontroler Proporsional Integral
(Sumber: Ogata K., 1997)

Bentuk fungsi alih dari kontroler Proporsional Integral (PI) ditunjukkan pada Persamaan 2.6.

$$U(s) = \left[Kp + \frac{Ki}{s}\right] E(s) \quad (2.6)$$

Dengan*U(s)*: Sinyal Kontrol

E(s): Nilai *error* sistem

Kp: *Gain* Kontroler Proporsional

Ki: *Gain* Kontroler Integral

Karena kontroler PI merupakan penggabungan dari dua unit kontroler P dan I, semua kelebuhan serta kekurangan yang ada pada kontroler P dan kontroler I masih tetap ada. Sifat kontroler P yang selalu meninggalkan *error steady state* (*offset*) dapat ditutupi oleh kelebihan kontroler I, sedangkan sifat kontroler I yang lambat dapat ditutupi oleh kontroler P, sehingga kontroler PI menghasilkan respon yang lebih cepat dari kontroler I tapi mampu menghilangkan *error steady state* (*offset*) yang dihasilkan kontroler P.

2.8 Metode *Root Locus*

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

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

2.9 *Pseudo Random Binary Sequence*

Pseudo Random Binary Sequence (PRBS) adalah sinyal kotak yang termodulasi pada lebarnya dan berlangsung secara sekuensial. Sinyal ini biasanya dibangkitkan menggunakan *Linear Feedback Shift Register* (LFSR). Pada LFSR memiliki 2 parameter dasar yang menentukan sifat sekuensial yang dihasilkan, yaitu: panjang dari *shift register* dan susunan umpan balik. PRBS memiliki variasi panjang sekuensialnya tergantung dari panjangnya *shift register* seperti ditunjukkan dalam Tabel 2.2 berikut.

Tabel 2.2 Variasi Panjang Sekuensial PRBS
(Sumber: Landau, 2006)

Panjang Register (N)	Panjang Sekuensial $L=2^N-1$	Posisi Tap Umpan Balik
2	3	1 dan 2
3	7	1 dan 3
4	15	3 dan 4

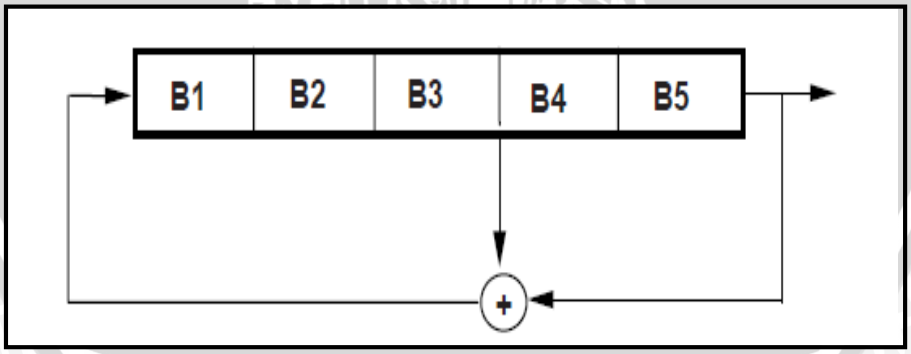
5	31	3 dan 5
6	63	5 dan 6
7	127	4 dan 7
8	255	2, 3, 4, dan 8
9	511	5 dan 9
10	1023	7 dan 10

Panjang dari *shift register* menentukan periode maksimum yang dapat dihasilkan dari sekuensial PRBS yang tidak berulang dan dapat dinyatakan dengan persamaan:

$$L_{PRBS} = 2^n - 1 \qquad (2.7)$$

Dengan n adalah panjang dari register LFSR (jumlah bit). Panjang maksimum dari PRBS disebut M-*sequence*.

Panjang maksimum yang dapat dihasilkan PRBS untuk panjang tertentu dari *shift register* dapat dicapai dengan mempersiapkan konfigurasi dari *feedback*. Pada dasarnya ada 2 kemungkinan untuk realisasi LFSR, yaitu: Fibonacci (*many to one*) dan Galois (*one to many*). Keduanya dapat didasarkan pada gerbang XOR atau XNOR menggunakan bermacam-macam angka dan kombinasi dari *feedback*-taps-keluaran dari *shift register* khusus. Dengan mengubah konfigurasi umpanbalik (jumlah tap dan posisinya) memungkinkan untuk menemukan M-*sequence* yang berbeda untuk panjang tertentu dari *shift register* (lihat Gambar 2.11).



Gambar 2.11 Register Geser 5 Bit dengan Umpan Balik
(Sumber: Landau., 2006)

Prinsip pengujian proses dengan sinyal PRBS adalah membuat perubahan input kecil secara acak untuk membangkitkan gangguan (*perturbation*) yang kontinyu pada variabel *output*. Salah satu keuntungan penggunaan dari pendekatan ini adalah amplitudo perubahan input yang dibutuhkan dapat lebih kecil jika dibandingkan dengan perubahan *step* pada *step testing*. Selain itu, proses pengujian dapat dilakukan tanpa

harus menunggu proses dalam keadaan tunak (*steadystate*). Jika pengujian dengan sinyal PRBS dilakukan, sinyal *input* secara teoritis disebut *white* (tidak berkorelasi) dan akan menghasilkan parameter model estimasi yang lebih baik. Frekuensi dari PRBS dapat dipilih untuk putaran (*flips*) cepat (*fast*) atau lambat (*slow*). Pemilihan frekuensi ini dapat menentukan jenis informasi terbaik yang akan didapat, misalnya untuk *fast* akan memberikan informasi yang akurat mengenai *dead time*, *slow* akan memberikan informasi *steady state gain* yang tepat sedangkan *medium* memberikan informasi *time constant* lebih baik.

2.10 Transmormasi Z

Jika pada sistem analog dikenal dengan transformasi Laplace yang merupakan bentuk umum dari transformasi Fourier, dalam sistem diskrit bentuk umum dari transformasi Fourier adalah transformasi Z. Sebuah transformasi ditetapkan sebagai deret angka yang berurutan, Fungsi $E(z)$ ditetapkan sebagai sebuah *power series* dalam z^{-k} dengan koefisien yang sama untuk nilai dari deret angka $\{e(k)\}$. Transformasi ini disebut sebagai transformasi Z, yang ditunjukkan dalam bentuk Persamaan 2.8.

$$E(z) = \mathcal{Z}[\{e(k)\}] = e(0) + e(1)z{-}1 + e(2)z{-}2 + \cdots \quad (2.8)$$

Dengan,

$$e(k) = Z^{-1}[E(z)] = \frac{1}{2\pi j}\oint_r E(z)z^{k-1}\ dz\ j = \sqrt{-1}$$

Sehingga $\mathcal{Z}(\cdots)$ mengindikasikan operasi transformasi Z dan $Z^{-1}(\cdots)$ mengindikasikan invers dari transormasi z. $E(z)$ dalam Persamaan 2.8 dapat dituliskan dalam notasi yang disederhanakan seperti pada Persamaan 2.9.

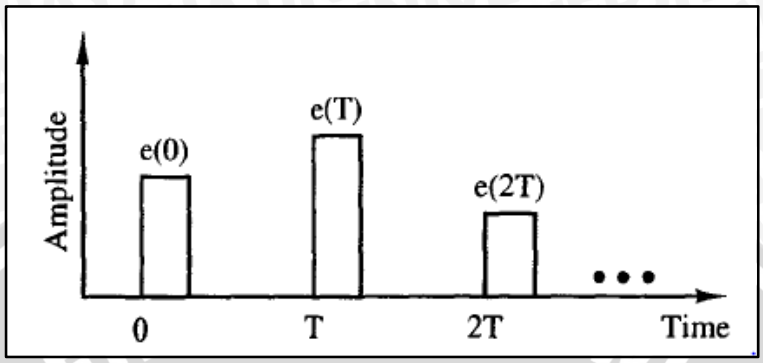
$$E(z) = Z[\{e(k)\}] = \sum_{k=0}^{\infty} e(k)z^{-k} \quad (2.9)$$

Transformasi Z terdefinisikan untuk beberapa deret angka $\{e(k)\}$, dan mungkin digunakan dalam menganalisis dari beberapa jenis sistem yang diuraikan oleh persamaan diferensial *time-invariant* linier.

2.11 Sampling Data

Terdapat sebuah perangkat rekonstruksi data yang dinamakan sampling data, yang termasuk kedalam sistem secara langsung mengikuti pencuplik (*sampler*). Tujuan dari sampling data adalah merekonstruksi sinyal yang tercuplik ke dalam sebuah bentuk yang menyerupai sinyal sebelum dicuplik. Perangkat data rekonstruksi sederhana dan sejauh

ini yang paling umum adalah *Zero Order Hold* (ZOH). Operasi dari sebuah pencuplik/kombinasi ZOH tergambarkan pada sinyal yang ditunjukkan pada Gambar 2.12. ZOH mencuplik sinyal *output* ke nilai yang sama dengan sinyal *input* pada saat proses pencuplikan yang ditunjukkan pada Gambar 2.13



Gambar 2.12 Sinyal tercuplik dalam bentuk pulsa
(Sumber: Phillips, 1995)

Pencuplik dan ZOH dapat direpresentasikan dalam diagram blok yang ditunjukkan pada Gambar 2.12. Sinyal $\bar{e}(t)$ dapat dinyatakan sebagai berikut:

$$\bar{e}(t) = e(0)[u(t) - u(t-T)] + e(T)[u(t-T) - u(t-2T)] + e(2T)[u(t-2T) - u(t-3T)] + \cdots \quad (2.10)$$

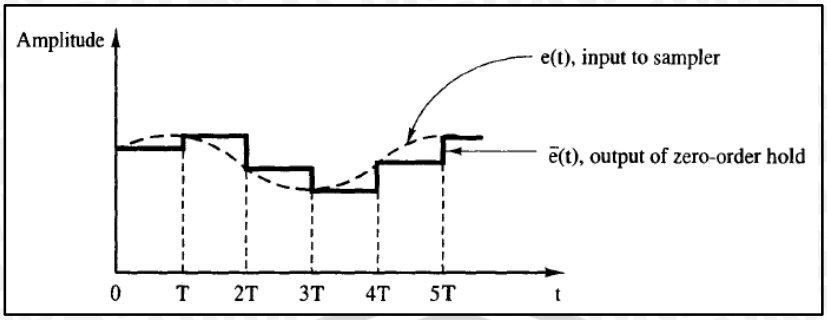
Dengan $u(t)$ adalah fungsi *unit step*. Transformasi laplace dari $\bar{e}(t)$ adalah $\bar{E}(s)$, yang diberikan oleh

$$E(s) = e(0)\left[\frac{1}{s} - \frac{e^{-Ts}}{s}\right] + e(T)\left[\frac{e^{-Ts}}{s} - \frac{e^{-2Ts}}{s}\right] + e(2T)\left[\frac{e^{-2Ts}}{s} - \frac{e^{-3Ts}}{s}\right] + \cdots$$

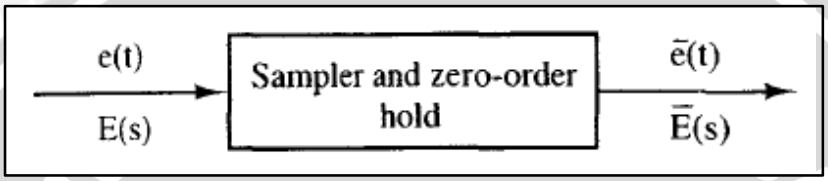
$$= \left[\frac{1-e^{-Ts}}{s}\right][e(0) + e(T)e^{-Ts} + e(2T)e^{-2Ts} + \cdots$$

$$= \left[\sum_{n=0}^{\infty} e(nT) - e^{-nTs}\right]\left[\frac{1-e^{-Ts}}{s}\right] \quad (2.11)$$

Faktor pertama dalam ekspresi terakhir pada Persamaan 2.14 terlihat sebuah sinyal *input* $e(t)$ dan periode sampling T. Faktor kedua terlihat e(t) yang *independent* dan oleh karena itu dapat dianggap menjadi sebuah fungsi alih.



Gambar 2.13 Sinyal *input* dan *output* dari data pencuplik/*data hold*
(Sumber: Phillips, 1995)

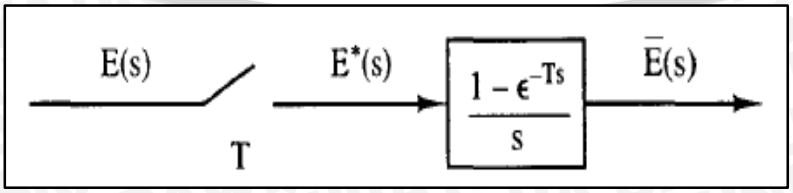


Gambar 2.14 Pencuplik dan *data hold*
(Sumber: Phillips, 1995)

Sehingga pencuplik dan *data hold* dapat direpresentasikan seperti pada Gambar 2.14. fungsi E*(s) disebut dengan transformasi terbintang yang didefinisikan sebagai

$$E^* = [\sum_{n=0}^{\infty} e(nT)e^{-nTs}] \qquad (2.12)$$

Maka Persamaan 2.11 ini terpenuhi dengan representasi pada Gambar 2.14. Operasi yang dilambangkan pada Gambar 2.14 telah terdefinisikan pada Persamaan 2.12 dan disebut dengan pencuplik yang ideal. Operasi yang digambarkan oleh fungsi alih disebut sebagai *data hold*. Hal ini harus ditekankan bahwa $E^*(s)$ tidak tampak dalam bentuk fisik sistem tetapi muncul pada sebuah hasil dari mengfaktorkan Persamaan 2.11. Pencuplik (*switch*) dalam Gambar 2.15 bukanlah sebuah model fisik pencuplik dan blok juga bukan sebuah model fisik dari *data hold*. Bagaimanapun juga, kombinasi tersebut bukan merpakan keakuratan model karakteristik *input-output* dari pencuplik perangkat *data hold* seperti yang ditunjukkan sebelumnya.



Gambar 2.15 Representasi dari pencuplik dan *data hold*
(Sumber: Phillips, 1995)

2.12 Diskritisasi

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

Menurut teori sampling Nyquist dan Shannon mengatakan jika suatu fungsi waktu $e(t)$ tidak mengandung komponen frekuensi yang lebih tinggi dari f_0 hertz, dalam hal ini nilai dai nilai $e(t)$ dapat ditentukan dengan memberikan titik sampling berjarak $1/2f_0$ detik terpisah.

Berikut tahapan diskritisasi yang dapat digunakan:

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

   - *Backward difference* :

     $$s = \left[\frac{1-z^{-1}}{Ts}\right] \qquad (2.13)$$

   - *Forward difference* :

     $$s = \left[\frac{1-z^{-1}}{Tsz^{-1}}\right] \qquad (2.14)$$

   - *Bilioner transform* :

     $$s = \left[\frac{2\,(1-z^{-1}\}}{Ts\,(1+z^{-1})}\right] \qquad (2.15)$$

Dengan Ts adalah waktu cuplik (*time sampling*)

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

Halaman ini sengaja dikosongkan

BAB III
METODE PENELITIAN

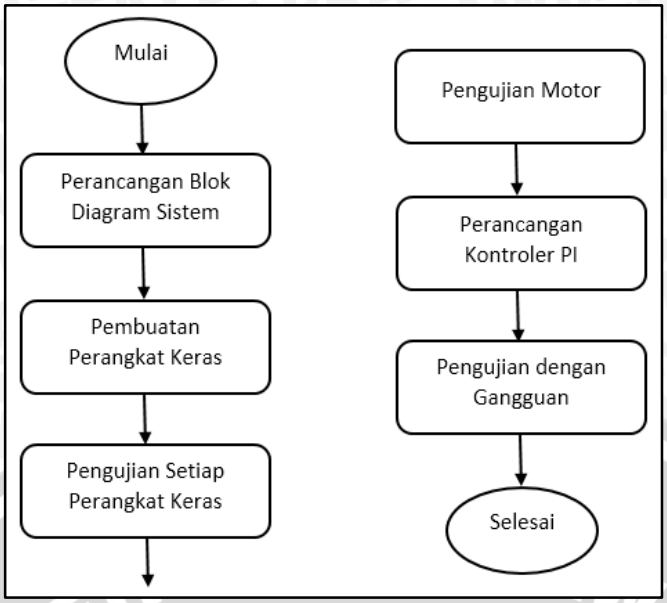
Metode penelitian pada dasarnya merupakan cara ilmiah untuk mendapatkan informasi dengan tujuan dan manfaat tertentu. Untuk menyelesaikan rumusan masalah dan merealisasikan tujuan skripsi yang terdapat di bab sebelumnya bahwa skripsi ini bertujuan untuk merancang sistem pengendali kecepatan motor DC pada konveyor barang menggunakan kontroler PI berbasis Arduino Uno, maka diperlukan langkah-langkah untuk menyelesaikan masalah tersebut. Metode yang digunakan diuraikan dalam kerangka penelitian.

3.1 Kerangka Penelitian

Pada penelitian ini terdapat beberapa tahapan yang akan dilalui untuk mencapai tujuan penelitian, berikut tahapa-tahapannya :

1. Pembuatan diagram blok sistem bertujuan untuk memahami struktur sistem secara keseluruhan.
2. Pembuatan perangkat keras bertujuan mengetahui kebutuhan perangkat keras sistem yang dibutuhkan serta mengetahui cara kerja sistem dari sisi perangkat keras.
3. Pengujian setiap perangkat keras bertujuan mengetahui karakteristik dan fungsi alihnya.
4. Pengujian motor DC bertujuan mengetahui hubungan tagangan masukan dan putaran rotor motor serta mengetahui fungsi alih *plant* yang akan dikontrol.
5. Merancang kontrol PI bertujuan mendapatkan parameter-parameter kontroler yang sesuai untuk mengontrol *plant*.
6. Melakukan pengujian sistem keseluruhan dengan memberikan gangguan perubahan berat beban material yang bertujuan untuk mengetahui respon sistem secara keseluruhan.

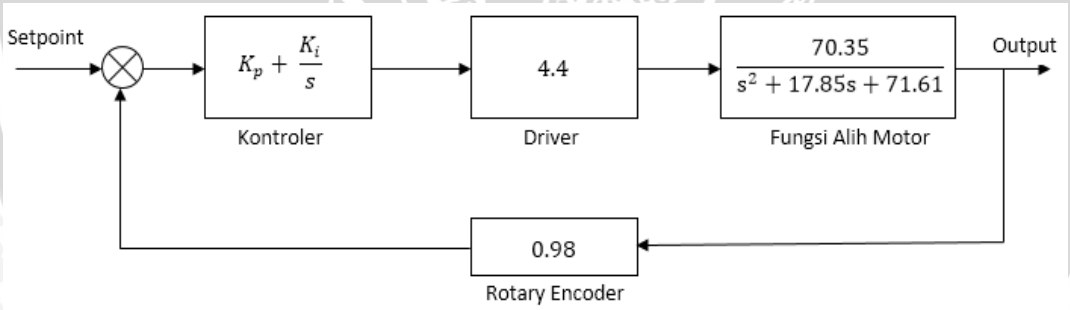
Diagram alir tahapan penelitian diatas dapat dilihat pada Gambar 3.1 dibawah ini.



Gambar 3.1 Kerangka Penelitian

3.2 Perancangan Diagram Blok Sistem

Pada pembuatan alat diperlukan perancangan diagram blok sistem yang dapat menjelaskan sistem secara garis besar dan diharapkan alat dapat bekerja sesuai dengan rencana. Diagram blok sistem dapat dilihat pada Gambar 3.2.



Gambar 3.2 Blok Diagram Sistem

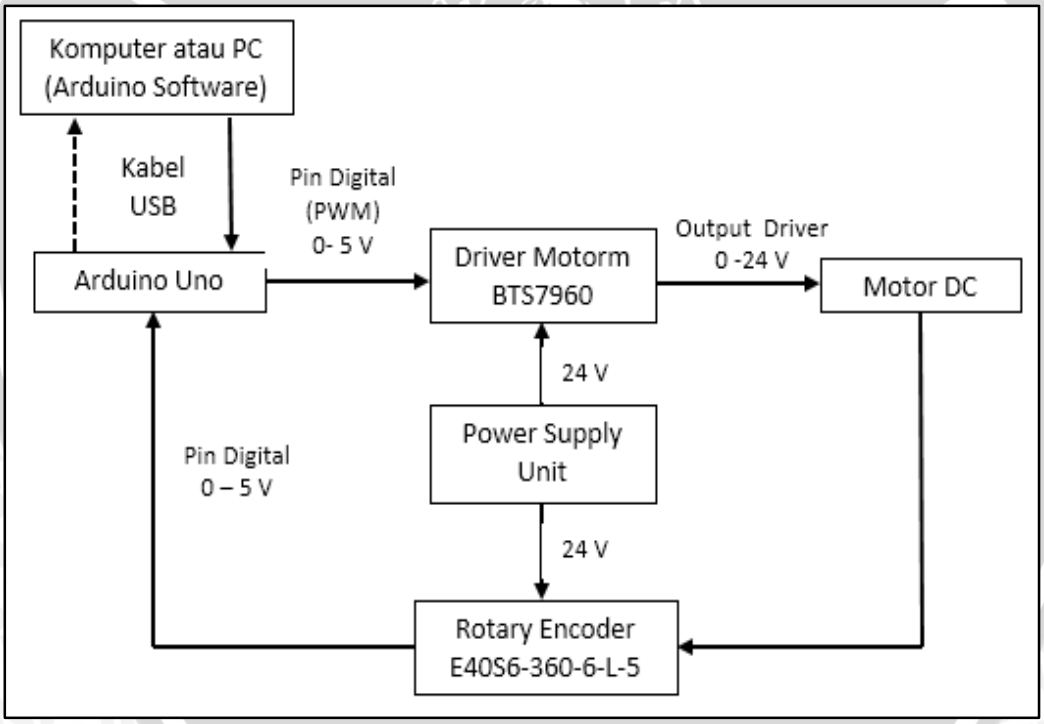
Set point masukan berupa nilai kecepatan yang diberikan oleh *user*. Nilai *setpoint* akan disimpan ke dalam mikrokontroler. Selain *setpoint* terdapat nilai *error*, yaitu deviasi antara pembacaan aktual kecepatan motor dari *rotary encoder* dan nilai *setpoint*. Kedua nilai tersebut akan dikalkulasi dan diolah oleh mikrokontroler. Nilai yang dikalkulasi adalah *manipulated variable*/ sinyal kontrol yang akan digunakan sebagai masukan *driver* motor. *Driver* motor ini akan menggerakan motor DC sesuai dengan sinyal kontrol keluaran mikrokontroler. Kondisi kecepatan sekarang akan dibaca lagi oleh sensor *rotary encoder* sehingga menjadi masukan *error* lagi bagi kontroler.

3.3 Pembuatan Perangkat Keras

Pembuatan perangkat keras dilakukan sebagai langkah awal sebelum terbentuknya suatu sistem beserta pemrogramannya, hal ini dimaksudkan agar sistem pengendali kecepatan motor DC pada konveyor barang menggunakan kontroler PI berbasis Arduino Uno dapat berjalan sesuai deskripsi awal yang telah direncanakan.

Pembuatan perangkat keras yang dilakukan meliputi:

1. Skema perancangan perangkat keras (Gambar 3.3).
2. Penentuan modul elektronik yang digunakan meliputi:
   - *Power Supply Unit* (PSU)
   - Mikrokontroler Arduino Uno R3
   - *Driver* Motor BTS7960
   - Motor DC
   - *Rotary Encoder Autonics* E40S6-360-6-L-5



Gambar 3.3 Skema pembuatan perangkat keras

3.4 Spesifikasi Alat

Spesifikasi alat yang meliputi komponen-komponen pendukung pada skripsi ini adalah sebagai berikut:

1. Komputer atau PC yang sudah terinstall *software* Arduino dan MATLAB 2014a.

2. *Power Supply Unit* (PSU), yang ditunjukkan pada Gambar 3.4 berfungsi sebagai sumber tegangan atau catu daya pada *driver* motor dimana PSU ysng digunakan ini mempunyai tegangan keluaran 0-30 V. Dalam skripsi ini tegangan keluaran PSU diatur pada tegangan 0-24 V.

Gambar 3.4 *Power Supply Unit*

3. Perangkat kontroler yang digunakan adalah Arduino Uno R3. Mikrokontroler Arduino Uno R3 (Gambar 3.5) berfungsi sebagai penyimpan algoritma kontroler sistem, pengendali *driver* motor, dan menerima serta mengolah data yang dikirimkan sensor *rotary encoder* (sensor kecepatan). Mikrokontroler menggunakan catu tegangan sebesar 5 Volt yang didapat dari komputer atau PC.

Gambar 3.5 Mikrokontroler Arduino Uno R3

4. *Driver* motor yang digunakan adalah modul yang menggunakan *Half Bridge Infineon* BTS 7960 *chips* yang ditunjukkan pada Gambar 3.6. *Driver* motor berfungsi sebagai pengali tegangan dimana tegangan masukan dari mikrokontroler Arduino dikali dengan sebuah nilai *gain* atau penguatan tertentu menjadi tegangan keluaran *driver* motor yang digunakan untuk mencatu motor. Catu tegangan *driver* motor didapat dari *power supply unit* (PSU) dengan tegangan maksimal 27 V dan arus maksimalnya sebesar 30 A.

Gambar 3.6 Driver Motor H-Bridge BTS9760

5. Motor DC (Gambar 3.7) yang digunakan memiliki tegangan maksimal sebesar 24 V, dan memiliki arus maksimal sebesar 15 A serta sudah dilengkapi sebuah *gear box*. Motor DC ini berfungsi sebagai aktuator untuk menggerakkan konveyor.

Gambar 3.7 Motor DC

6. Sensor kecepatan yang digunakan adalah *rotary encoder autonics* E40S6-360-6-L-5 yang ditunjukkan pada Gambar 3.8. *Rotary encoder* membutuhkan suplai daya sebesar 5 V yang diperoleh dari mikrokontroler. Keluaran *rotary encoder* berupa pulsa yang didapat dari proses pembacaan kecepatan pada motor DC yang kemudian dikirim ke mikrokontroler untuk diproses selanjutnya oleh Arduino Uno.

Gambar 3.8 Increment Rotary Encoder E40 Series

7. Konveyor barang ditunjukkan pada Gambar 3.9 yang digunakan memiliki panjang 80 cm dan lebar 16 cm. *Belt* konveyor terbuat dari bahan kain dan kerangka konveyor terbuat dari alumunium.

Gambar 3.9 Konveyor Barang

3.5 Prinsip Kerja Sistem

Prinsip kerja sistem adalah sebagai berikut:

1. *Power Supply Unit* (PSU) diberi catu 220 VAC.
2. *Power Supply Unit* (PSU) mengkonversi tegangan dari 220 VAC menjadi 24 VDC sebagai suplai rangkaian *driver* motor.
3. Catu daya Arduino Uno berupa tegangan 5 VDC yang didapat dari kabel USB yang dihubungkan pada komputer.

4. Keluaran *rotary encoder* berupa pulsa dengan tegangan 0 V sampai 5 V sebagai masukan digital pada pin *external interrupt* Arduino Uno R3 yang kemudian di konversi menjadi nilai pembacaan kecepatan motor.
5. Sinyal kontrol dari Arduino Uno masuk ke driver motor BTS 7960 (*H-Bridge*). *Driver* motor berfungsi untuk menguatkan sinyal yang dihasilkan mikrokontroler Arduino Uno dari 0-5 V menjadi 0-24 V.
6. Motor DC yang beputar telah dikopel dengan konveyor dan juga *rotary encoder* Autonics E40S6-360-6-L-5 sebagai pembaca putaran motor DC.
7. Mencari respon kecepatan motor DC terhadap perubahan sinyal *Pulse Width Modulation (*PWM).
8. Mencari fungsi alih motor DC dengan memberikan sinyal *Pseudo Random Binary Sequence* (PRBS).
9. Mensimulasikan metode *root locus* dengan meggunakan *software* MATLAB.
10. Membandingkan respon sistem *simulink* dengan respon sistem motor DC yang didapat.
11. Mengimplementasikan hasil desain perancanan pada sistem.

3.6 Pengujian *Driver* Motor

a. Tujuan

Mengetahui kinerja dan respon rangkaian *driver* motor BTS 7960 dengan membandingkan *output* tegangan efektif *driver* dengan masukan *duty cycle* sinyal PWM yang diberikan oleh Arduino Uno R3.

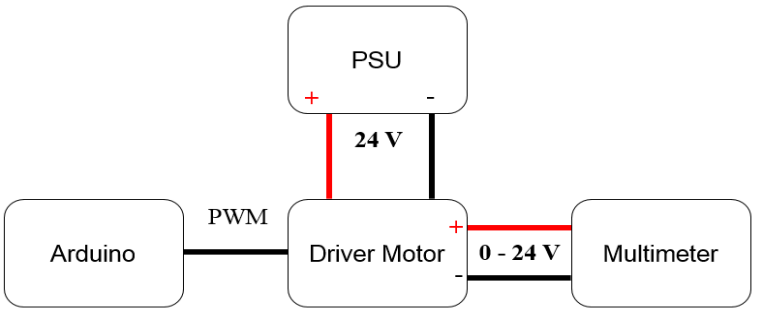
b. Peralatan yang digunakan :

1. Komputer atau PC.
2. *Power Supply Unit* (PSU).
3. *Driver* motor BTS 7960.
4. Multimeter.
5. Arduino Uno R3
6. Kabel penghubung

c. Langkah Pengujian

1. Hubungkan *output* tegangan *Power Supply Unit* (PSU) pada *input* tegangan referensi *driver* motor BTS 7960.
2. Hubungkan *input* tegangan *driver* motor BTS 7960 dengan pin *output* PWM di Arduino Uno R3. Gambar 3.10 adalah ilustrasi rangkaian pengujian *driver* motor.

3. Hubungkan *output* tegangan *driver* motor BTS 7960 dengan multimeter.
4. Atur *duty cycle* sinyal PWM pada Arduino Uno R3 dengan nilai 0%-100%.
5. Amati dan catat hasil pembacaan multimemter disetiap kenaikan 5%.



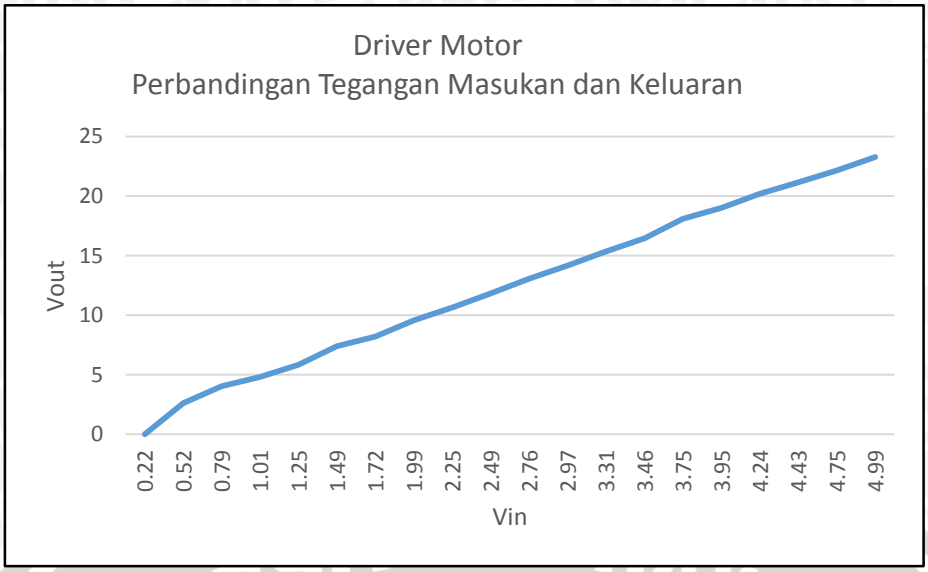
Gambar 3.10 Ilustrasi rangkaian pengujian *driver* motor

d. Hasil Pengujian

Data pengujian *driver* motor ditunjukkan dalam Tabel 3.1 dan Gambar 3.11.

Tabel 3.1 Hasil Pengujian Driver Motor

Duty cycle (%)	PWM	Vin (Volt)	Vout (Volt)
0	0	0	0
5	12.75	0.23	1.2
10	25.5	0.48	2.41
15	38.25	0.73	3.6
20	51	0.98	4.78
25	63.75	1.21	5.87
30	76.5	1.46	7.05
35	89.25	1.72	8.23
40	102	1.97	9.41
45	114.75	2.2	10.5
50	127.5	2.45	11.68
55	140.25	2.7	12.87
60	153	2.95	14.06
65	165.75	3.18	15.16
70	178.5	3.42	16.35
75	191.25	3.68	17.56
80	204	3.93	18.76
85	216.75	4.16	19.87
90	229.5	4.41	21.07
95	242.25	4.67	22.26
100	255	4.92	23.4



Gambar 3.11 Grafik Hasil Pengujian Driver

Kesimpulan dari pengujian diatas didapat *gain* atau fungsi alih driver motor sebesar:

$$m = \frac{Vout_{13} - Vout_{9}}{Vin_{13} - Vin_{9}} = \frac{15.35 - 9.59}{3.31 - 1.99} = 4.4 \qquad (3.1)$$

3.7 Pengujian *Rotary Encoder*

a. Tujuan

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

b. Peralatan yang digunkan :

1. Komputer atau PC.
2. *Power Supply Unit* (PSU).
3. *Rotary Encoder* E40S6-360-6-L-5.
4. *Driver* motor BTS 7960.
5. Motro DC.
6. Arduino Uno R3
7. Kabel Penghubung
8. Tachometer

c. Langkah Pengujian

1. Hubungkan *output* dengan *Power Supply Unit* (PSU) pada *input* tegangan sumber *rotary encoder* E40S6-360-6-L-5 dan *input* tegangan referensi *driver* motor BTS 7960.

2. Hubungkan pin *output rotary encoder* E40S6-360-6-L-5 pada pin *interrupt* eksternal Arduino Uno R3.
3. Atur *duty cycle* sinyal PWM pada Arduino Uno R3 dengan nilai 0%-100%.
4. Untuk pengukuran menggunakan tachometer, hubungkan tachometer pada poros motor serta atur tegangan masukan pada motor dengan nilai 0-24 V.
5. Catat hasil pembacaan *rotary enoder* E40S6-360-6-L-5 dan juga tachometer di setiap kenaikan 1 V.
6. Bandingkan data hasil pengukuran tachometer dengan sensor *rotary encoder*.

d. Hasil Pengujian

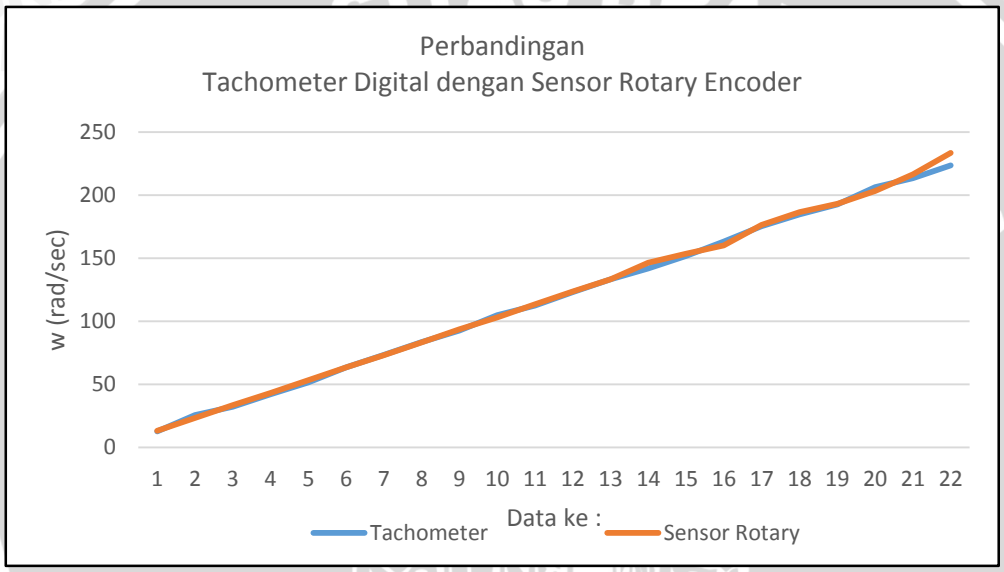
Data hasil pengujian sensor *rotary encoder* E40S6-360-6-L-5 ditunjukkan dalam Tabel 3.2.

Tabel 3.2 Hasil Pengujian Sensor Rotary Encoder

No	Tachometer (rpm)	Rotary Encoder (rpm)
1	12.8	13.2
2	25.7	23.4
3	32.2	33.6
4	42.1	43.2
5	51.6	53.4
6	63.5	63.6
7	73.3	73.2
8	83.5	83.4
9	92.8	93.6
10	104.8	103.2
11	112.4	113.4
12	123.1	123.6
13	133.3	133.2
14	141.9	146.4
15	151.8	153.6
16	163.2	160.2
17	175.4	176.4
18	184.7	186.6
19	192.6	193.2
20	206.3	203.4

21	213.5	216.6
22	223.5	233.4
23	232.6	243.6
24	243.7	246.6

Dari hasil pengujian kecepatan motor dengan menggunakan tachometer dan sensor *retary encoder* yang ditunjukkan pada Tabel 3.2, dapat disimpulkan bahwa pembacaan keduanya relatif sama, hanya terdapat sedikit selisih atau *error* pembacaan antara tachometer dengan sensor *rotary encoder*. Untuk memperjelas, dapat lihat pada Gambar 3.12.



Gamabr 3.12 Grafik Pengujian Sensor Kecepatan

Kesimpulan dari pengujian diatas diperoleh *gain* atau fungsi alih sensor kecepatan sebesar:

$$m = \frac{Vsensor_{13} - Vsensor_5}{Vtacho_{13} - Vtacho_5} = \frac{133.2-53.4}{133.3-51.6} = 0.98 \qquad (3.2)$$

3.8 Penentuan Steady State Gain Motor

Steady state gain motor adalah variable yang menggambarkan hubungan tegangan masuk motor dengan putaran motor. *Steady state gain* motor berguna menentukan tegangan yang dibutuhkan untuk nilai putaran suatu motor.

a. Tujuan

Mendapatkan *steady state gain* motor.

b. Peralatan yang digunakan :

1. Komputer atau PC.
2. *Power Supply Unit* (PSU).
3. Arduino Uno R3.
4. Motor DC.
5. *Driver* motor BTS 7960.
6. *Rotary Encoder* E40S6-360-6-L-5
7. Konveyor Barang.
8. Voltmeter
9. Kabel penghubung.

c. Langkah Pengujian

1. Hubungkan *output* tegangan *Power Supply Unit* (PSU) dengan motor DC yang telah dikopel dengan konveyor barang.
2. Ubah- ubah tegangan keluaran PSU dan ukur menggunakan voltmeter.
3. Catat perubahan kecepatan setiap terjadi perubahan teganggan masukan motor.

d. Hasil Pengujian

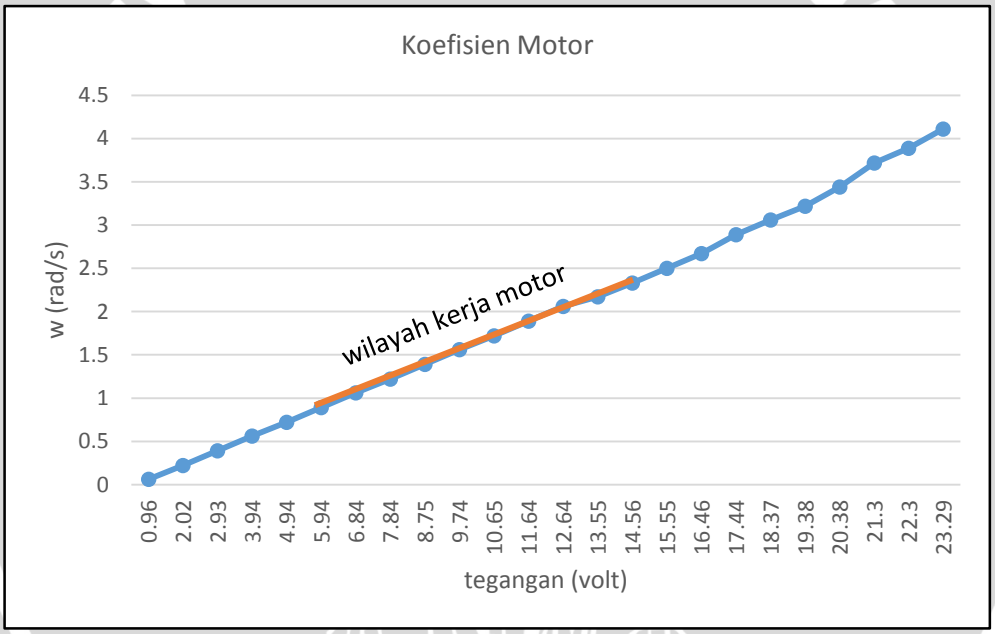
Berikut ini hasil pengujian untuk menemukan koefisien motor yang ditampilkan pada Tabel 3.3 dan Grafik 3.13.

Tabel 3.3 Hasil Data Motor

No	Vin	w (rad/s)
1	1.2	0.11
2	2.41	0.33
3	3.6	0.5
4	4.78	0.72
5	5.87	0.89
6	7.05	1.06
7	8.23	1.28
8	9.41	1.44
9	10.5	1.67
10	11.68	1.83
11	12.87	2.06
12	14.06	2.28
13	15.16	2.44

14	16.35	2.67
15	17.56	2.83
16	18.76	3.06
17	19.87	3.22
18	21.07	3.44
19	22.26	3.61
20	23.4	3.83

Dari hasil pengujian *steady state gain* motor yang ditunjukkan pada Tabel 3.3 didapat batas bawah dan batas atas wilayah kerja motor yang digunakan untuk mendapatkan fungsi alih *plant*. Wilayah kerja motor dapat dilihat pada Gambar 3.13.



Gambar 3.13 Grafik Hasil Data Motor

Kesimpulan dari hasil pengujian di atas diperoleh koefisien motor dengan menghitung gradien grafik di atas sebagai berikut:

$$m = \frac{X_2 - X_1}{Y_2 - Y_1} = \frac{17.56 - 9.41}{2.83 - 1.44} = 5.86 \qquad (3.3)$$

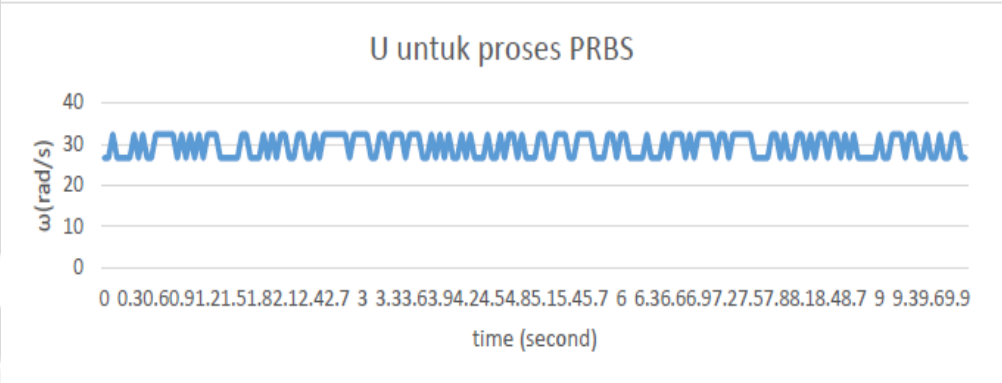
3.9 Penentuan Fungsi Alih Motor DC dan Validasi Respon

Dalam pengendalian kecepatan konveyor dibutuhkan sistem yang mampu mengontrol motor DC dikarenakan motor tersebut termasuk bagian dari plant. Pengontrolan kecepatan motor DC menggunakan Arduino Uno R3 sebagai pengolah dan

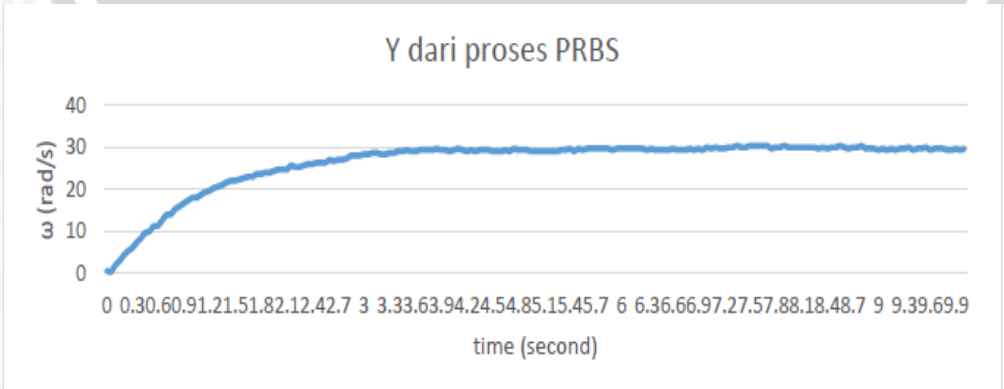
memberikan data berupa *Pulse Width Modulation* (PWM) agar motor bergerak. Motor DC yang digunakan pada perancangan ini tidak diketahui karakteristiknya, sehingga yang perlu dilakukan adalah melakukan pengujian dengan menggunakan *rotary encoder*. Karakteristik motor DC pada perancangan ini didapatkan dengan cara memberikan masukan *unit step*.

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

1. Mencari nilai yang linear dari hasil pengujian kecepatan motor DC terhadap *duty cycle* PWM.
2. Memasukkan nilai batas atas dan batas bawah berdasarkan nilai yang linear untuk membangkitkan sinyal PRBS (Gambar 3.14)
3. Sinyal PRBS yang telah dibangkitkan kemudian digunakan sebagai masukan motor DC.
4. Setelah didapatkan data sinyal PRBS dan data kecepatan motor DC (Gambar 3.15), selanjutnya adalah melakukan identifikasi dengan menggunakan *software* MATLAB.

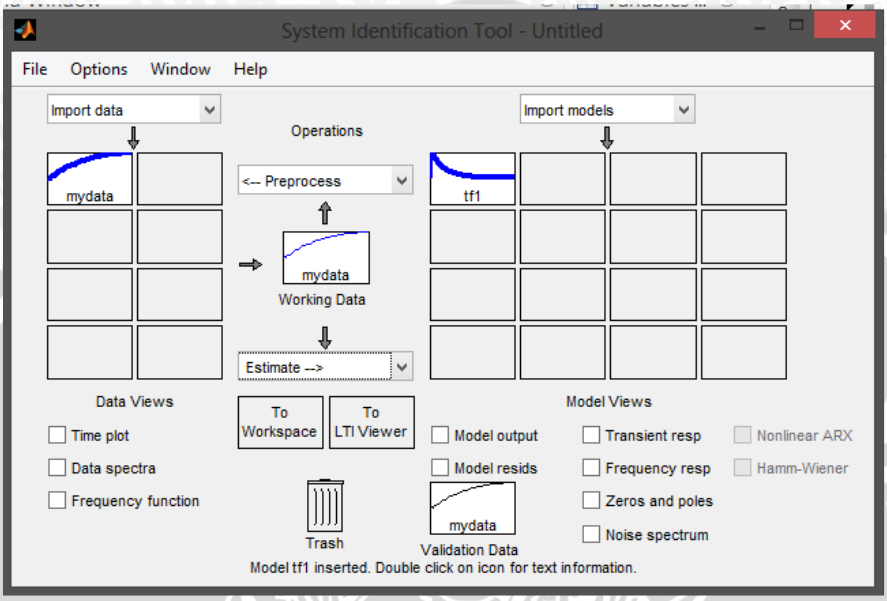


Gambar 3.14 *White Noise* Sebagai Input dalam Proses PRBS

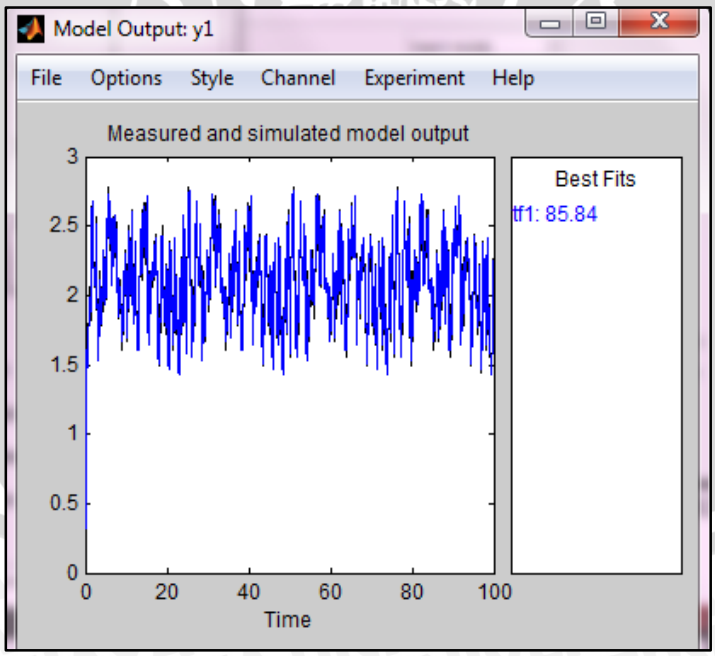


Gambar 3.15 Respon Proses PRBS

5. Dengan menggunakan sintaks ident pada *command window* pada MATLAB R2014a, data sinyal PRBS dan data kecepatan motor yang telah disimpan kemudian di *import* pada blok *System Identification Toolbox* (Gambar 3.16). Setelah melakukan beberapa estimasi model berdasarkan data yang telah di *import* didapatkan fungsi alih dari motor dengan *best fit* sebesar 85.84% (Gambar 3.17).



Gambar 3.16 Tampilan Aplikasi Ident di Software Matlab R2014b

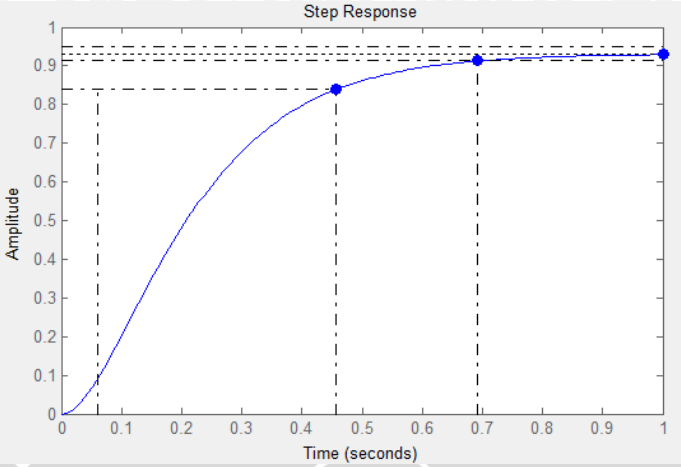


Gambar 3.17 Tampilan *Best Fit*

6. Dari hasil identifikasi, fungsi alih yang didapat adalah :

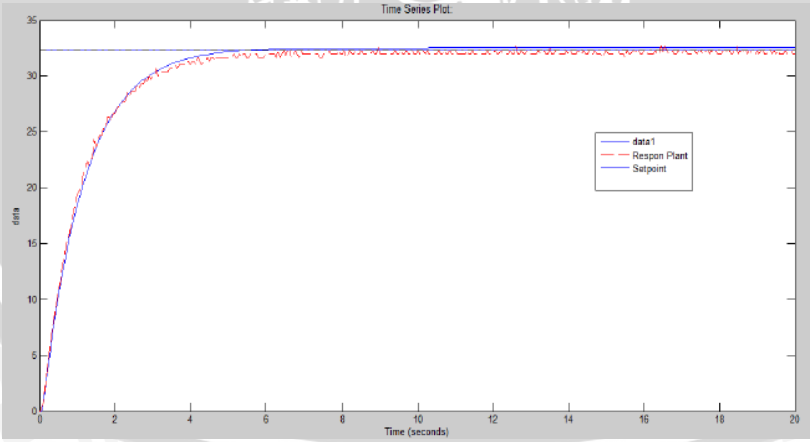
$$\frac{Y(s)}{U(s)} = \frac{70.35}{s^2+17.85s+71.61} \qquad (3.4)$$

7. Dengan menganalisis fungsi alih didapat nilai frekuensi alamiah 5,23 dan koefisien redaman 2,95 yang berarti sistem teredam. Dengan memberikan masukan *unit step* pada program MATLAB R2014a didapatkan respon pada Gambar 3.18, di dapat nilai *settling time* 0.692 second, *rise time* 0.395 second, *error steady state* 0%.



Gambar 3.18 Plotstep Respon Fungsi Alih Hasil PRBS

8. Setelah didapat fungsi alih, diperlukan validasi. Dalam melakukan validasi fungsi alih motor dilakukan dengan cara membandingkan respon fungsi alih dan respon kecepatan motor DC yang dapat dari pembacaan *rotary encoder* dengan memberikan masukan pulsa *unit step*. Berikut perbandingan kedua respon yang didapat dengan menggunakan MATLAB R2014a (Gambar 3.19)



Gambar 3.19 Validasi Respon Fungsi Alih hasil PRBS dengan *Plant*

3.10 Desain Kontroler Proporsional Integral (PI)

Dalam kawasan waktu kontroler PI dapat dinotasikan dengan persamaan

$$c(t) = Kp\left(e(t) + \frac{1}{T_i}\int_0^t e(t)\right)dt \qquad (3.5)$$

Dimana c(t) adalah keluaran kontroler, Kp adalah *gain* proporsional, Ti adalah waktu konstan Integal atau *reset time* dan e(t) adalah *error* yang terjadi. Dari persamaan di atas dapat diubah menjadi kawasan frekuensi dengan Transformasi Laplace sehingga menjadi persamaan berikut

$$C(s) = Kp\left(1 + \frac{1}{T_i\, s}\right)E(s) \qquad (3.6)$$

Persamaan di atas belum bisa dimasukkan kedalam mikrokontroler maka dari itu persamaan kontunyu diatas harus diubah kedalam bentuk diskrit melalui Transformasi Z. Dalam Transformasi Z dibutuhkan waktu *sampling* (Ts). Digunakan metode *Billinear Transform* sehingga nilai notasi s pada Laplace setara dengan

$$s = \frac{2}{Ts}\left(\frac{1-z^{-1}}{1+ z^{+1}}\right) \qquad (3.7)$$

Maka persamaan (3.2) disubstitusikan ke dalam persamaan (3.3) menjadi

$$C(z) = \left[Kp + \frac{KpTs\,(1+ z^{-1})}{2T_{i(1- z^{-1})}}\right]E(z) (3.8)$$

Berikutnya adalah memodifikasi persamaan agar dapat disederhanakan. Kedua ruas dikalikan dengan $1 - z^{-1}$

$$C(z)(1 - z^{-1}) = Kp\,E(z)(1 - z^{-1}) + \frac{Kp\,Ts}{2T_i}\,E(z)(1 + z^{-1}) (3.9)$$

Disusun kembali persamaan diatas menjadi keluaran kontroler

$$C(z) = C(z)(z^{-1}) + Kp\big(E(z) - E(z)(z^{-1})\big) + \frac{Kp\,Ts}{2T_i}(E(z) + E(z)(z^{-1})) (3.10)$$

Dari persamaan (3.6) diubah ke dalam persamaan beda sehingga didapatkan persamaan dibawah ini

$$C(k) = C(k-1) + Kp\,\big(E(k) - E(k-1)\big) + \frac{Kp\,Ts}{2T_i}(E(k) + E(k-1)) (3.11)$$

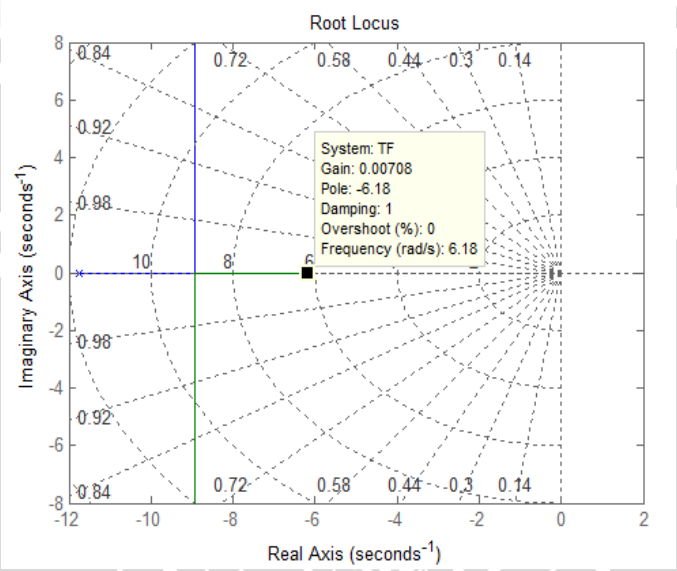
Dimana k-1 adalah kondisi sebelumnya. Persamaan diatas kemudian dimasukkan kedalam program pada mikrokontroler.

3.11 Penentuan Parameter Kontroler PI

Untuk mmemenuhi tujuan performansi *loop* yang diinginkan, maka perlu ditambahkan kontroler pada sistem tersebut. Kontroler yang dipilih ialah kontroler PI.

Setelah didapatkan fungsi alih sisitem yaitu F(s) = $\frac{70.35}{s^2+17.85s+71.61}$. Selanjutnya adalah menetukan parameter kontroler PI dengan menggunakan metode *root locus*. Hal pertama yang dilakukan yaitu menentukan letak simpul *loop* tertutup.

Berdasarkan fungsi alih *loop* tertutup dapat diketahui sistem berorde dua. Nilai parameter kontroler PI ditentukan oleh pemilihan *pole* pada diagram *root locus.* Pada penelitian ini digunakan s_1 = -6.18. Penentuan letak *pole* diagram *root locus* terlihat dalam Gambar 3.20.



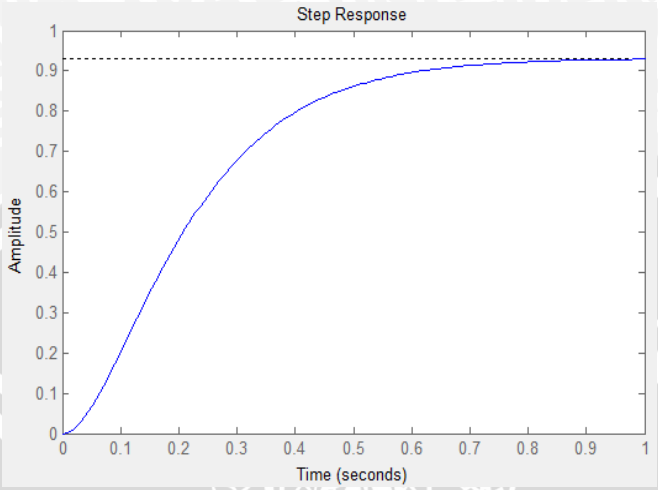
Gambar 3.20 Letak *Pole* pada Diagram *Root Locus*

Setelah ditemukan letak *pole* yang diinginkan kemudian dengan mensubstitusikan nilai s_1 dan nilai fungsi alih sistem serta memvariasikan nilai Ki akan didapatkan parameter PI pada Tabel 3.4. Pencarian parameter Kp dan Ki dengan menggunakan MATLAB R2014a.

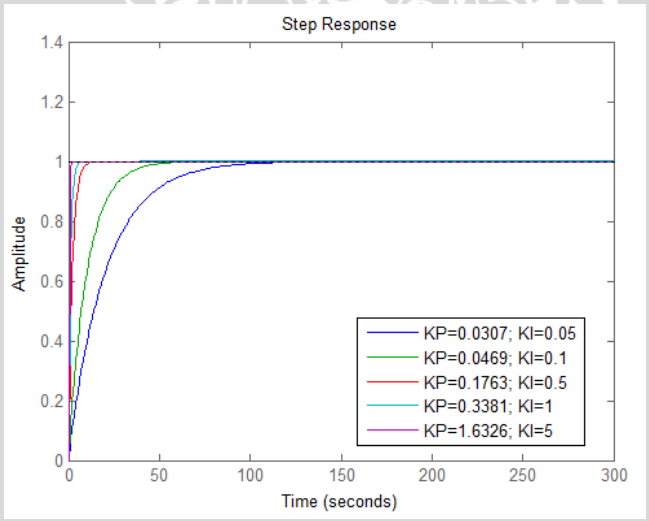
Tabel 3.4 Parameter PI dengan s_1 = -6.18

No.	Kp	Ki	*Pole* 1	*Pole* 2
1	0.0307	0.05	-11.7965	-6.18
2	0.0469	0.1	-11.8407	-6.18
3	0.1763	0.5	-12.2074	-6.18
4	0.3381	1	-12.6981	-6.18
5	1.6326	5	-17.7573	-6.18

Setelah didapatkan nilai Kp, dan Ki hasil perhitungan kemudian dilakukan pengujian terhadap sistem dengan parameter yang sesuai dengan sistem. Berikut grafik respon sistem tanpa kontroler PI yang ditunjukkan pada Gambar 3.21 dan respon sistem dengan kontroler PI yang ditunjukkan pada Gambar 3.22.



Gambar 3.21 Respon sistem tanpa kontroler PI

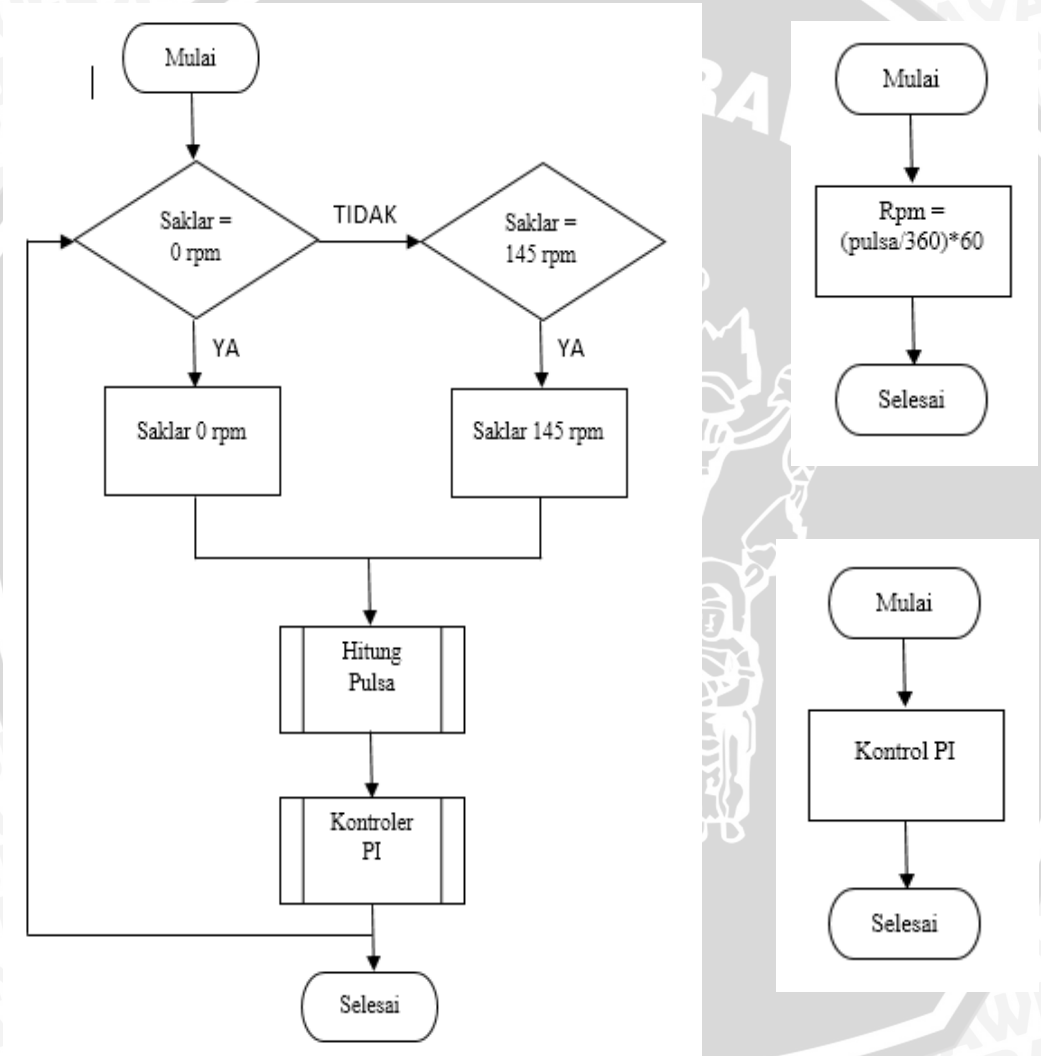


Gambar 3.22 Respon sistem dengan kontroler PI

Dari Gambar 3.21 dapat diketahui bahwa respon sistem tanpa menggunakan kontroler PI lebih lama menuju *steady state* dan melebihi *setpoint* yang diinginkan. Dengan digunakannya kontroler PI hasil *tuning* didapatkan respon yang lebih cepat dari pada respon tanpa menggunakan kontroler PI, serta dapat mencapai *setpoint* yang diinginkan seperti tertera pada Gambar 3.22. Dari lima jenis jenis parameter PI yang didapat dipilih nilai PI yang memiliki respon terbaik yaitu Kp = 1.6326 dan Ki = 5.

3.12 Diagram Alir

Diagram alir program (Gambar 3.23) merupakan gambaran alur proses program yang dilakukankan oleh kontroler pada saat implementasi, kontroler yang digunakan adalah mikrokontroler, pada diagram alir penelitian ini akan dibagi, pertama diagram alir program utama (program sistem secara keseluruhan), kedua diagram alir rutin eksternal interrupt (program yang akan bekerja karena picuan dari luar, dalam program ini yang memicu adalah sensor *rotary encoder*), dan yang ketiga adalah diagram alir kontroler PI. Berikut diagram alir program:



Gambar 3.23 Diagram Alir Sistem Keseluruhan

BAB IV

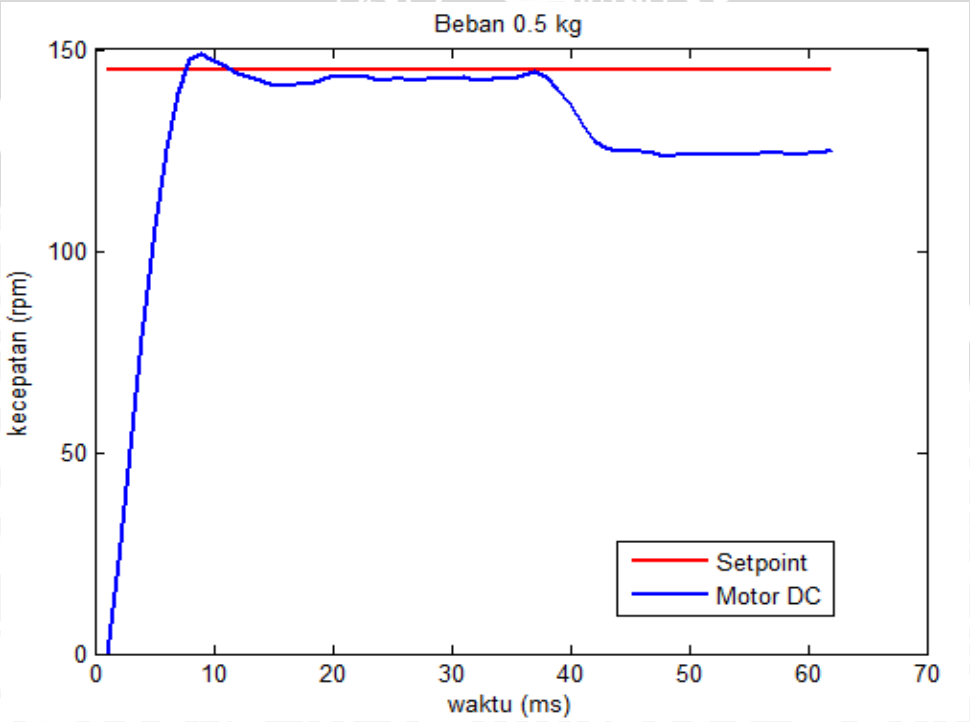
HASIL DAN PEMBAHASAN

Pada bab ini membahas hasil dari pengujian sistem secara keseluruhan baik dengan kontroler maupun tanpa diberi kontroler. Analisis ini dilakukan bertujuan untuk mengetahui apakah sistem yang dirancang pada pembahasan sebelumnya mampu bekerja sesuai perancangan atau tidak.

4.1 Pengujian Sistem Tanpa Kontroler PI

Pengujian ini dilakukan untuk mengetahui seberapa besar gangguan yang terjadi bila konveyor barang diberi beban berupa timbel dengan berat 0.5 kg, 1 kg, dan 1.5 kg. Dengan diberi *setpoint* kecepatan sebesar 145 rpm, bagaimanakah respon dari sistem tersebut tanpa menggunakan kontroler PI. Berikut hasil dari pengujian sistem secara keseluruhan tanpa diberi kontroler yang ditunjukkan pada Gambar 4.1, Gambar 4.2, dan Gambar 4.3.

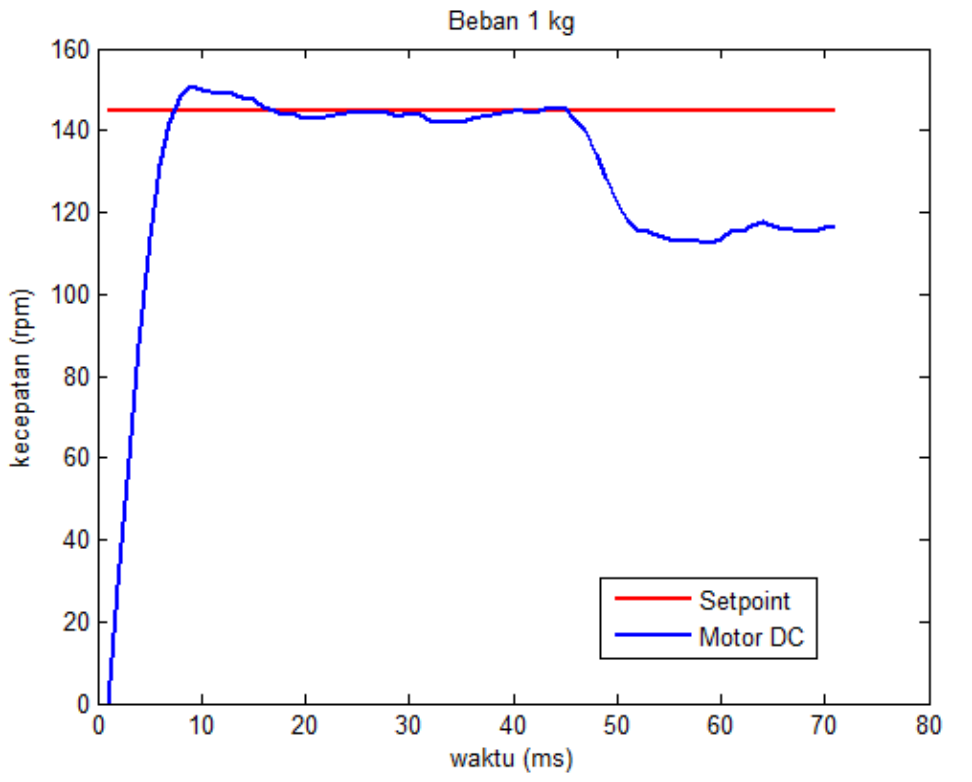
a) Pengujian Sistem Tanpa Kontroler PI dengan Beban 0.5 kg



Gambar 4.1 Grafik respon sistem dengan gangguan 0.5 kg

Dari grafik respon sistem yang ditunjukkan dalam Gambar 4.1 dapat disimpulkan bahwa pada saat sebelum diberi gangguan, sistem terdapat osilasi dan kurang dari *setpoint* yang diinginkan, t_s (waktu kerja) sebesar 12.4 ms, dan terdapat *overshoot* sebesar 4.8%. Setelah diberi gangguan berupa timbel dengan berat 0.5 kg, kecepatan laju konveyor melambat dengan kecepatan putar motor DC menjadi 123 rpm.

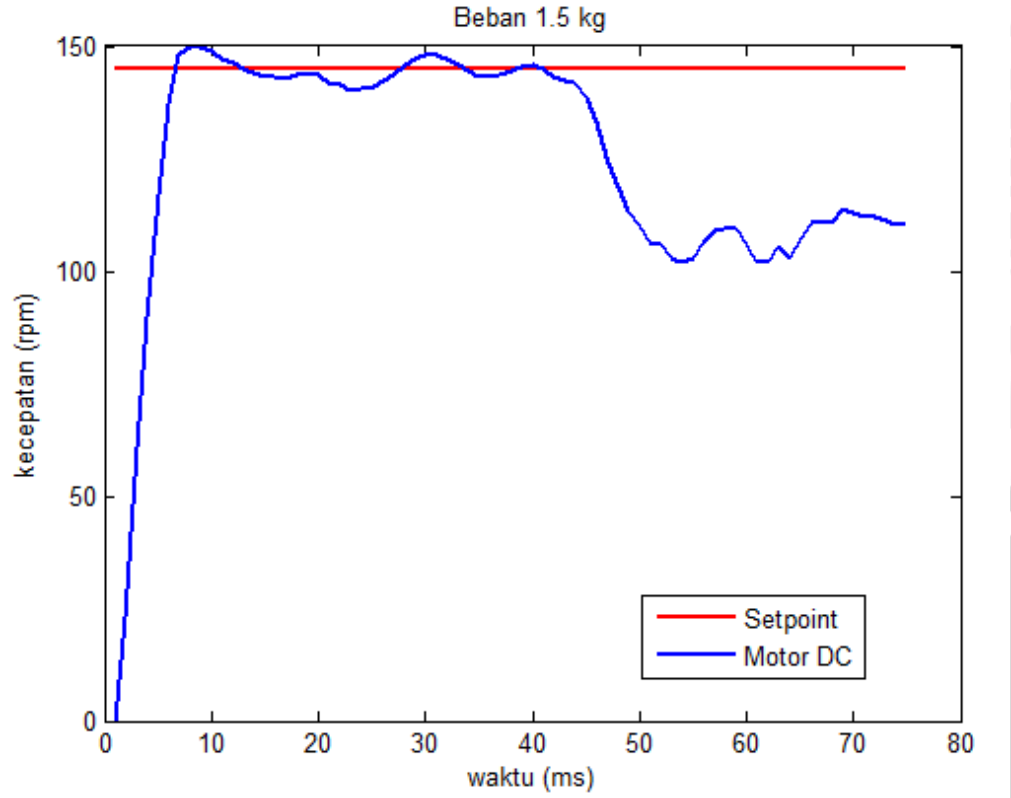
b) Pengujian Sistem Tanpa Kontroler PI dengan Beban 1 kg



Gambar 4.2 Grafik respon sistem dengan gangguan 1 kg

Dari grafik respon sistem yang ditunjukkan dalam Gambar 4.2 dapat disimpulkan bahwa pada saat sebelum diberi gangguan, sistem terdapat osilasi, t_s (waktu kerja) sebesar 16.8 ms, dan terdapat *overshoot* sebesar 5.5%. Setelah diberi gangguan berupa timbel dengan berat 1 kg, kecepatan laju konveyor melambat dengan kecepatan putar motor DC menjadi 110 rpm.

c) Pengujian Sistem Tanpa Kontroler PI dengan Beban 1.5 kg



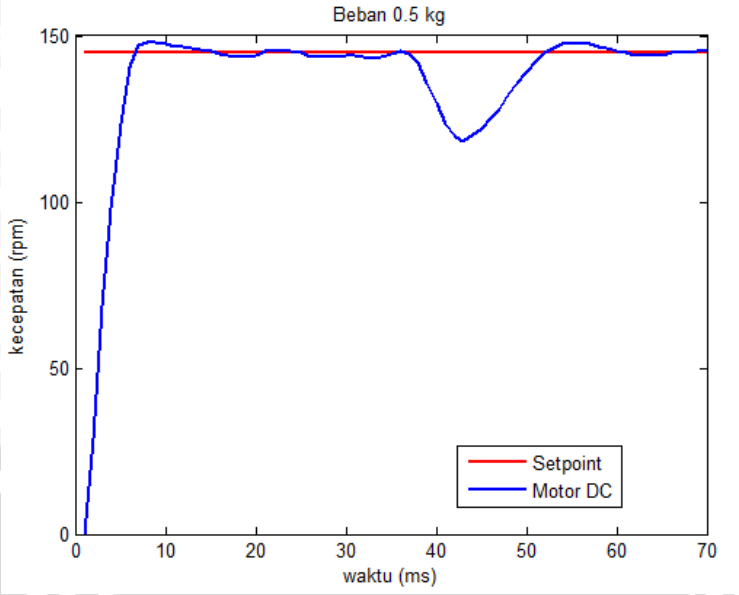
Gambar 4.3 Grafik respon sistem dengan gangguan 1.5 kg

Dari grafik respon sistem yang ditunjukkan dalam Gambar 4.3 dapat disimpulkan bahwa pada saat sebelum diberi gangguan, sistem terdapat osilasi, t_s (waktu kerja) sebesar 21.7 ms, dan terdapat *overshoot* sebesar 6.2%. Setelah diberi gangguan berupa timbel dengan berat 1.5 kg, kecepatan laju konveyor melambat dengan kecepatan putar motor DC menjadi 95 rpm.

4.2 Pengujian Sistem Dengan Kontroler PI

Pengujian ini dilakukan untuk mengetahui apakah respon dapat kembali dalam keadaan *steady state* sesuai *setpoint* ketika diberi gangguan. Pada pengujian ini *setpoint* yang diberikan sebesar 145 rpm dan gangguan yang diberikan berupa timbel dengan berat sebesar 0.5 kg, 1 kg, dan 1.5 kg pada saat respon sudah mencapai keadaan *steady state*. Sehingga diketahui apakah nilai parameter PI yang sudah didapat sesuai dengan *setpoint* yang diinginkan. Hasil pengujian sistem dengan kontroler PI ditunjukkan pada Gambar 4.4, Gambar 4.5, dan Gambar 4.6.

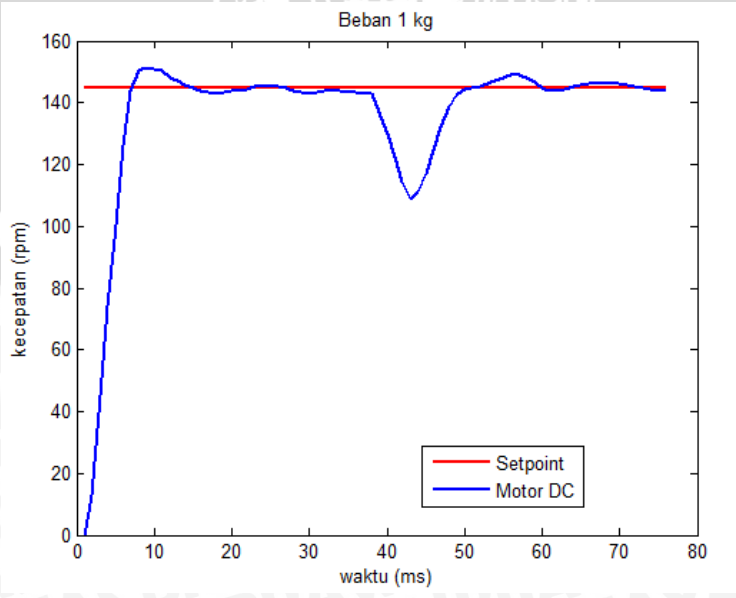
a) Pengujian Sistem Dengan Kontroler PI dengan Beban 0.5 kg



Gambar 4.4 Grafik respon sistem dengan kontroler PI dan gangguan 0.5 kg

Dari grafik respon sistem yang ditunjukkan dalam Gambar 4.4 dapat disimpulkan bahwa pada saat sebelum diberi gangguan, sistem terdapat osilasi namun masih dalam toleransi 2-5 %, t_s (waktu kerja) sebesar 17 ms, dan terdapat *overshoot* sebesar 3.4%. Setelah diberi gangguan, kecepatan putar motor DC berkurang hingga mencapai 115 rpm, kemudian kembali naik menjadi *setpoint* yang ditentukan sebelumnya dengan *recovery time* sebesar 8 ms. Sehingga parameter PI yang diimplementasikan pada alat dengan *setpoint* 145 rpm dan gangguan sebesar 0.5 kg memiliki respon yang cukup baik.

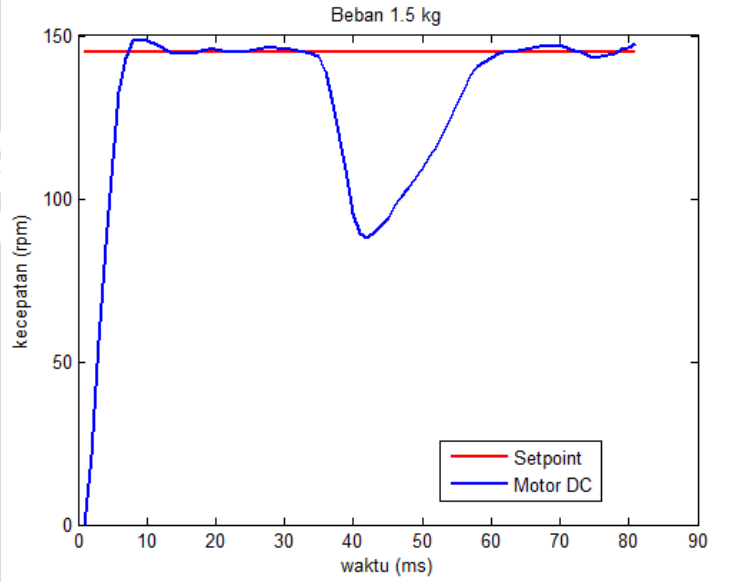
b) Pengujian Sistem Dengan Kontroler PI dengan Beban 1 kg



Gambar 4.5 Grafik respon sistem dengan kontroler PI dan gangguan 1 kg

Dari grafik respon sistem yang ditunjukkan dalam Gambar 4.5 dapat disimpulkan bahwa pada saat sebelum diberi gangguan, sistem terdapat osilasi namun masih dalam toleransi 2-5 %, t_s (waktu kerja) sebesar 18 ms, dan terdapat *overshoot* sebesar 6.9%. Setelah diberi gangguan, kecepatan putar motor DC berkurang hingga mencapai 105 rpm, kemudian kembali naik menjadi *setpoint* yang ditentukan sebelumnya dengan *recovery time* sebesar 10 ms. Sehingga parameter PI yang diimplementasikan pada alat dengan *setpoint* 145 rpm dan gangguan sebesar 1 kg memiliki respon yang cukup baik.

c) Pengujian Sistem Dengan Kontroler PI dengan Beban 1.5 kg



Gambar 4.6 Grafik respon sistem dengan kontroler PI dan gangguan 1.5 kg

Dari grafik respon sistem yang ditunjukkan dalam Gambar 4.6 dapat disimpulkan bahwa pada saat sebelum diberi gangguan, sistem terdapat osilasi namun masih dalam toleransi 2-5 %, t_s (waktu kerja) sebesar 15 ms, dan terdapat *overshoot* sebesar 3.5%. Setelah diberi gangguan, kecepatan putar motor DC berkurang hingga mencapai 86 rpm, kemudian kembali naik menjadi *setpoint* yang ditentukan sebelumnya dengan *recovery time* sebesar 19 ms. Sehingga parameter PI yang diimplementasikan pada alat dengan *setpoint* 145 rpm dan gangguan sebesar 1.5 kg memiliki respon yang cukup baik

Berdasarkan grafik respon sistem yang ditunjukkan dalam Gambar 4.4, Gambar 4.5, dan Gambar 4.6 dapat disimpulkan bahwa setelah diberi gangguan, sistem dapat kembali pada keadaan *steady steate* sesuai dengan *setpoint*. Sehingga dapat diketahui bahwa kontroler bekerja dengan baik dan sistem yang sangat responsif.

Halaman ini sengaja dikosongkan

BAB V
PENUTUP

5.1 Kesimpulan

1. Berdasarkan data respon sistem yang diperoleh dari pengujian dengan menggunakan sinyal *Pseudo-Random Binary Sequence* (PRBS) didapat nilai fungsi alih F(s) = $\frac{70.35}{s^2+17.85s+71.61}$ dengan nilai *best fit* sebesar 85.84%. Berdasarkan respon sistem yang diperoleh dari pengujian dengan menggunakan metode *root locus* didapat nilai parameter kontroler PI dengan penguatan sebesar K_p = 1.6325 dan K_i = 5.
2. Hasil pengujian sistem dengan menggunakan kontroler PI menunjukkan bahwa, respon motor DC dengan nilai 145 rpm dengan gangguan 0.5 kg, 1 kg, dan 1.5 kg mengalami perlambatan, kemudian sistem dapat kembali pada keaadaan *steady state* sesuai *setpoint* dengan *recovery time* yang sangat cepat masing-masing adalah 8 ms, 10 ms dan 19 ms. Walapun sistem terdapat osilasi namun masih dalam toleransi 2-5%. Sehingga dapat diketahui bahwa kontroler bekerja dengan baik dan sistem bersifat responsif.

5.2 Saran

Saran yang dapat diberikan untuk penelitian selanjutnya adalah dengan mengimplementasikan motor DC sebagai aktuator lainnya seperi lift barang, pompa dan sebagainya. Serta mengunakan sensor kecepatan yang lebih teliti, sehingga data yang didapatkan mendekati dengan nilai yang sebenarnya

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

Yana, Ade A. 2014. *Metode Root Locus Untuk Mencari Parameter PID Dalam Pengendalian Posisi Stamping Rod Berbasis Pneumatic Menggunakan Ardino Uno*. Malang: Skripsi Teknik Elektro Universitas Brawijaya Malang.

Nugraha, Rifqi P. 2016. *Pengontrolan Tegangan Generator Set DC (Type 734 11) LEYBOLD Dengan Struktur State Feedback Control*. Malang: Skripsi Teknik elektro Universitas Brawijaya Malang.

Prabarianto, Bayu. 2016. *Metode Root Locus Untuk Mencari Parameter PID pada Pengendalian Kecepatan Motor DC D-6759 dengan Menggunakan Arduino Mega 2560*. Malang: Skripsi Teknik Elektro Universitas Brawijaya Malang.

Morris, S Alan. 2011. *Measurement and Instrumentation Principles,* Oxford: Butterworth-Heinemann.

Ogata, Katsuhiko. 1997. *Teknik Kontrol Automatik (Sistem Pengaturan)*. Jakarta: Erlangga.

Ogata, Katsuhiko. 1996. *Teknik Kontrol Automatik.Edisi kedua*.Diterjemahkan oleh: Leksono, Edi. Jakarta: Erlangga.

Phillips, Charles L., H. Troy Nagle. 1995. *Digital Control System Analysis and Design.*Engelwood Cliffs, NJ: Prentice-Hall, Inc.

Future Electronics.*HIGH CURRENT 30 A DC MOTOR DRIVER*.

William, S.Levine.1996.*The Control Handbook.* United State of America:CRC Press

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

Sar, S.K. dan lillie D. 2014. *MRAC Based PI Controller for Speed Control of DC Motor Using Lab View*. WSEAS Transactions on Systems and Control.Wain, Y. Suban. https://asro.wordpress.com/2009/01/16/diskritisasi/(diakses pada 23 Maret 2016).

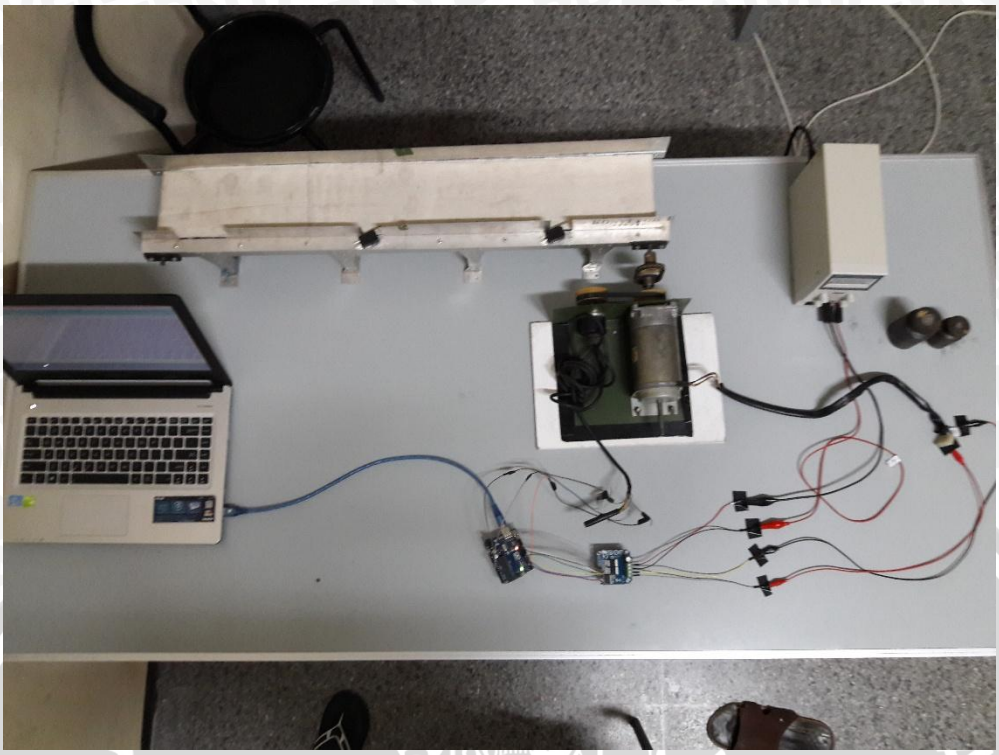
Gopal, Dr.M. 1998. *Digital Control Engineering*. New Delhi: New Age International Pulisher, Ltd.

Bolton, W. 2004. *Sistem Instrumentasi Dan Sistem Kontrol*. Jakarta : Penerbit Erlangga.

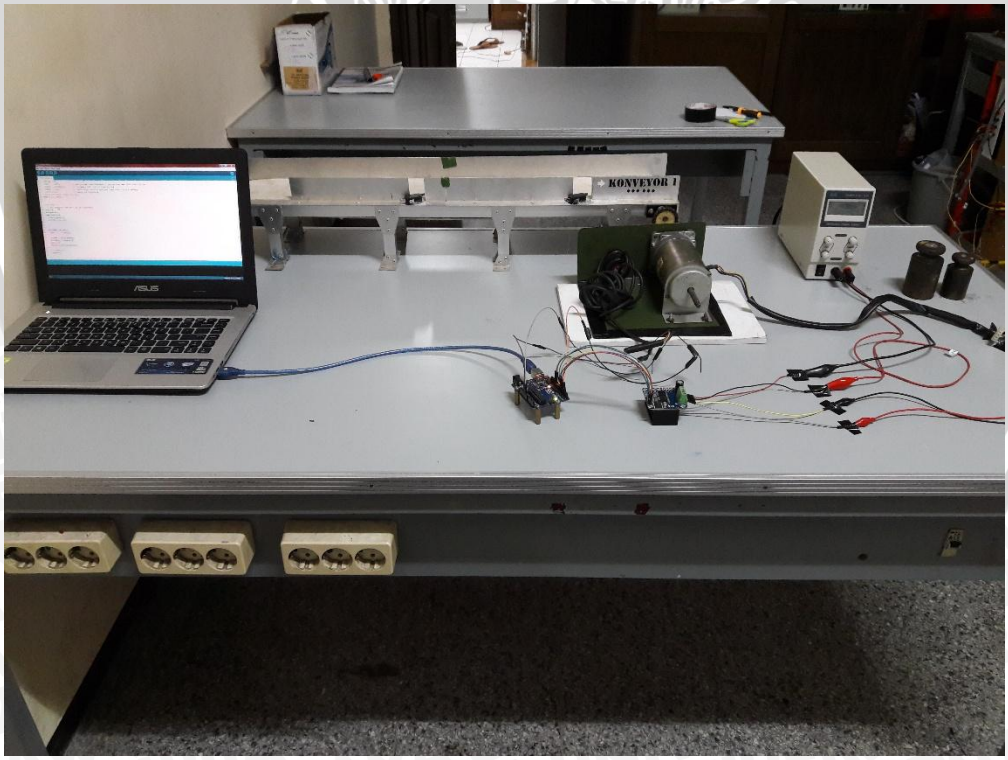
Halaman ini sengaja dikosongkan

LAMPIRAN

Foto Alat



Alat tampak atas



Alat tampak samping

Motor DC yang telah dikopel dengan *rotary encoder*

Konveyor yang telah dikopel dengan motor DC

Gangguan yang d

LAMPIRAN

Listing Program

MATLAB Program

```
%=======================================================
% PROGRAM MENCARI Parameter Kontroler PI 20 April 2016
%=======================================================

%nilai pole yang ditentukan dari Gambar root locus

s1=-6.18
KI=[0.05 0.1 0.5 1 5]

plant_num=[0 0 70.35];
plant_den=[1 17.85 71.61];

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

for k =1:5

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

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

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

    r = roots(Tden)
    step (Tnum,Tden,t)
    %step (plant_num,plant_den,t)
    hold on
end;
hold off
figure, rlocus(Tnum,Tden)
```

Arduino Program

```
////////////////////////////////////////////////
/// PRBS by Faisol, modified by Ray
////////////////////////////////////////////////

//deklarasi variable//////////////////////
volatile int pulse      = 0;     //untuk mendeteksi encoder
float total_pulse       = 0;     //untuk menyimpan jumlah pulsa
float total_encoder     = 360;   //jumlah kotak hitam encoder
float putaran           = 0 ;    //banyak putaran
float w                 = 0;     // Kecepatan
int pwm                 = 0;     //pwm

//////////////////////////////////////////
//Deklarasi variable utk PRBS//////////////////
int ledState          = LOW;
unsigned char i     = 0;
int cnt               = 0;
int j                 = 0;
int k                 = 0;
int temp              = 0;
int berhenti          = 0 ;
unsigned char data[8]={1,1,1,1,1,1,1,1}, prbs[8];
int pwm_prbs=0;
float vin = 0;

////////////////////////////////////////////
void setup() {
 // Persiapan Timer1
 noInterrupts();
 TCCR1A  = 0;
 TCCR1B  = 0;
 TCNT1     = 64753; //49911;//45536;     //(max preload timer-konstanta)// preload timer
     max 65536-16MHz/256/2Hz
 TCCR1B  = 0b00000101;       //  prescaler 000 - 101 // 8,64,256,1024
 TIMSK1  = 0x05;          // enable timer overflow interrupt (pake cvavr aja biar gampang)
 interrupts();             // enable all interrupts

 // Persiapan interrupt eksternal
 attachInterrupt(0, count, CHANGE);      // 0 = pin2 ; 1 = pin3

 //pin driver
 pinMode(12, OUTPUT); //MIN1 Vcc, driver pin7
 pinMode(13, OUTPUT); //MIn2 gnd, driver pin8
 pinMode(5, OUTPUT);   //MIN1 Vcc motor1 pwm, driver pin5
 pinMode(6, OUTPUT);   //MIn2 gnd motor1 pwm, driver pin6
 pinMode(7, OUTPUT);   //MEN high, driver pin7
 pinMode(8, OUTPUT);   //MSLP high, drver pin8
```

```
 digitalWrite(12, HIGH);
 digitalWrite(13, LOW);
 digitalWrite(6, LOW);
 digitalWrite(7, HIGH);
 digitalWrite(8, HIGH);
 Serial.begin(9600);
}

// Jika terjadi interrupt eksternal akan memanggil fungsi:
void count() {
 pulse++;
 total_pulse++;
}

//Interrupt Timer1
ISR(TIMER1_OVF_vect) {
 TCNT1      = 64753; //49911;//45536;          // preload timer (preload timer-konstanta)
 putaran    = total_pulse / total_encoder;
 w          = putaran * 20 ;
 total_pulse = 0;
 cnt++;
 cetak();
}

void prbs_f(){ //membuat white noise sebagai input
      temp=data[1]^data[2]^data[3]^data[7]; //output bit 2,3,4,8 di xor
   for (j=7;j>=1;j--) //geser bit
      {
      prbs[j]=data[j-1];
      }
      prbs[0]=temp; //isi register data dengan prbs selanjutnya
   for (k=0;k<=7;k++){
      data[k]=prbs[k];}
}

void cetak (){
 if (cnt<2001 && berhenti==0){
        prbs_f();
        Serial.print(pwm_prbs);
        Serial.print("\t \t");
       // Serial.print(vin);
       // Serial.print("\t");
        Serial.print(w);
        Serial.print("\t");
    if (data[0]==1){
          pwm_prbs=”pwm”; //prbs high
          Serial.println("vin \t ");}
    else{
          pwm_prbs=”pwm”; //prbs low
          Serial.println("vin \t ");}}
```

```
  else if(cnt==2001 && berhenti == 0){
     pwm_prbs=0;
    Serial.println("PRBS SELESAI");
    cnt = 257;
    berhenti = 1;}
}

void loop() {
  analogWrite(5,pwm_prbs);
  //vin = 24 *(pwm_prbs/255);
}
```

```
/*********************************
--------Uji Sistem Keseluruhan--------
BISMILLAHIROHMANIRROHIM

Baca Putaran      ==> PIN 2
PWM               ==> PIN 5
LED               ==> PIN 13
setpoint 0 rpm    ==> PIN 9
setpoint 145 rpm  ==> PIN 10
.**********************/
#define led 13
#define pwm 5

int set_point;        // Set Point
float y;              // Nilai kecepatan Motor DC
float error;          // set_point - y
float last1_error;    // error sebelumnya
float pulsa=0;        // pulsa rotary encoder
float Ts;             // Time Sampling Model Referensi
float pwmMotor;       // PWM Motor
float Prop;           // Sinyal kontrol Kontroler P
float Intg;           // Sinyal kontrol Kontroler I
float last1_Intg;     // Sinyal kontrol Kontroler I sebelumnya
double MV;            // Sinyal Kontrol Kontroler PI
float Kp;             // Nilai konstanta Kp
float Ki;             // Nilai konstanta Ki
int as,ap;

void setup()
{
 pinMode(7, OUTPUT);
 pinMode(8, OUTPUT);
 pinMode(pwm, OUTPUT);
 pinMode(led, OUTPUT);
 pinMode(9, INPUT_PULLUP);
 pinMode(10, INPUT_PULLUP);

 digitalWrite(7, HIGH);
 digitalWrite(8, HIGH);

 //set_point=145;
 Ts=0.01;
 Kp=1.6236;   // parameter yang diubah
 Ki=5;        // parameter yang diubah

 last1_error=0; last1_Intg=0;

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

```
  TCNT1   = 0;
  OCR1A   = 3124;           // compare match register 16HMHz/256/50Hz/50ms
  TCCR1B  |=(1 << WGM12);    // CTC mode
  TCCR1B  |=(1 << CS12);     // 256 prescaller
  TIMSK1  |=(1 << OCIE1A);   // enable timer compare interrupt

  TCCR2A = 0b00000010;
  TCCR2B = 0b00000111;
  TIMSK2 |= (1 <<OCIE2A);    // enable timer compare interrupt
  OCR2A  = 156;             // compare match register 16MHz/64/25KHz/10ms

  attachInterrupt(0, hitung_pulsa, FALLING);
  interrupts();             // enable all interrupt
  Serial.begin(9600);
}

//Timer Rotary
ISR(TIMER2_COMPA_vect)        // timer compare interrupt service routine
{
  Rot_switch();
  if(as==5)
  {
    as=0;
    y = (pulsa*1200)/360;
    pulsa = 0;
  }
  as++;
}

//Timer Controller and Serial Monitor
ISR(TIMER1_COMPA_vect)        // timer compare interrupt service routine
{
  error = set_point - y;
  Kontroler();

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

void Kontroler()
{
  Prop = Kp * error;
  Intg = last1_Intg + (Ki * Ts * error);
  MV = Prop + Intg;

 last1_Intg = Intg;
 last1_error = error;
```

```
 if(MV<0)MV = 0;
 pwmMotor = MV*0.046;
}

void Rot_switch()
{
 if(digitalRead(9)==LOW)
 {
  set_point=0; pwmMotor=0; MV=0;
 }
 else if(digitalRead(10)==LOW)
 {
  set_point=145;
 }
 else;
}

void loop()
{
 analogWrite(pwm, 153);
}

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