PERANCANGAN DAN IMPLEMENTASI KONTROLER PID BERDASAR MODEL UNTUK KENDALI POSISI SUDUT MENGGUNAKAN SCILAB ARDUINO

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Teknik Mesin

Disusun Oleh :

Nama : Muhammad Devan Fadhila
No. Mahasiswa : 16525058
NIRM 2016060867

JURUSAN TEKNIK MESIN
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2021

Pernyataan Keaslian

“Dengan ini saya menyatakan bahwa karya tulis ilmiah yang saya buat merupakan karya sendiri dan bukan hasil plagiasi dari karya tulis orang lain. Semua referensi dan kutipan yang terdapat pada karya tulis ini telah saya cantumkan sitasi dan sumber pustakanya mengikuti tata cara pengutipan karya ilmiah yang benar. Apabila dikemudian hari saya dianggap melakukan pelanggaran hak kekayaan intelektual dan terbukti melanggar hak tersebut, maka saya bersedia menerima sanksi sesuai dengan peraturan yang berlaku.”

Yogyakarta, 15 Januari 2022

Muhammad Devan Fadhila

LEMBAR PENGESAHAN DOSEN PEMBIMBING

PERANCANGAN DAN IMPLEMENTASI KONTROLER PID BERDASAR MODEL UNTUK KENDALI POSISI SUDUT MENGGUNAKAN SCILAB ARDUINO

TUGAS AKHIR

Disusun Oleh :

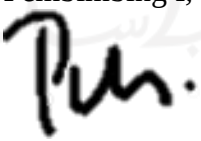
Nama : Muhammad Devan Fadhila

No. Mahasiswa : 16525058

NIRM 2016060867

Yogyakarta, 12 Desember 2021

Pembimbing I,



Purtojo, S.T., M.Sc.

LEMBAR PENGESAHAN DOSEN PENGUJI

PERANCANGAN DAN IMPLEMENTASI KONTROLER PID BERDASAR MODEL UNTUK KENDALI POSISI SUDUT MENGGUNAKAN SCILAB ARDUINO

TUGAS AKHIR

Disusun Oleh :

Nama : Muhammad Devan Fadhila
No. Mahasiswa : 16525058
NIRM 2016060867

Tim Penguji

Purtojo, ST., M.Sc.	Tanggal : 13 / 01 / 2022
Agung Nugroho Adi, ST., MT.	Tanggal : 6/1/2022
Donny Suryawan, ST., M.Eng	Tanggal : 07 / 01 / 2022

Mengetahui

Ketua Jurusan Teknik Mesin

Dr. Eng. Risdiyono, ST., M.Eng.

HALAMAN PERSEMBAHAN

"Sesungguhnya bersama kesulitan ada kemudahan. Maka apabila engkau telah selesai (dari sesuatu urusan urusan), tetaplah bekerja keras (untuk urusan yang lain). Dan hanya kepada Tuhanmulah engkau berharap."

(QS. Al-Insyirah: 6-8)

Saya ucapkan terimakasih kepada:

Bapak dan Ibu tercinta

Ir. H. Septin Elka dan Dra. Hj. Yulminarti, M.Si

Sebagai tanda bukti atas amanah yang diberikan kepada saya untuk melanjutkan pendidikan ke jenjang S1. Terimakasih telah berusaha mendidik dan mengantarkan saya mencapai gelar sarjana. Semoga gelar ini dapat menjadi jembatan saya untuk membanggakan kalian, dan segala proses perjuangan kalian dapat menjadi pahala yang bermanfaat di akhirat kelak.

Aamiin ya rabbal alamin

Kakak, **Fadhila Devani, S.Psi**

Terimakasih atas segala petuah dan semangat yang telah diberikan. Doakan aku bisa menjadi apa yang engkau harapkan kedepanya. Jangan lelah untuk terus memberikan nasehat kepadaku.

HALAMAN MOTTO

Rasulullah SAW bersabda, barang siapa menempuh jalan untuk mendapatkan ilmu, Allah SWT akan memudahkan baginya jalan menuju surga.

(Hadist Riwayat Muslim)

Work hard in silence, let success be your noise.

(Frank Ocean)

A smooth sea never made a skilled sailor

(Franklin D Roosevelt)

Silence in your procces, Show them your success

(Anonymous)

KATA PENGANTAR ATAU UCAPAN TERIMA KASIH

Assalamu'alaikum warrahmatullahi wabarakatuh,

Alhamdulillahirabbil'alamin, puji syukur dipanjatkan atas kehadirat Allah SWT yang telah memberikan rahmat, hidayah, dan karunia-Nya sehingga tugas akhir yang berjudul "**Perancangan dan Implementasi Kontroler PID Berbasis Model Untuk Kendali Posisi Sudut Menggunakan Scilab Arduino**" dapat diselesaikan dengan sebaik-baiknya. Tidak lupa shalawat serta salam dipanjatkan kepada junjungan kita Nabi Muhammad SAW.

Tidak lupa pula ucapan terimakasih kepada pihak-pihak yang selalu mendukung dan memberi bantuan baik secara moril ataupun materil sehingga tugas akhir ini dapat diselesaikan dengan sebaik-baiknya. Terkhusus kepada:

1. Septin Elka dan Yulminarti, selaku kedua orang tua penulis yang dengan usaha serta doa-nya lah penulis dapat sampai dititik ini. Terimakasih banyak atas kasih saying dan nasihat yang diberikan kepada penulis sampai akhirnya penulis dapat menyelesaikan tugas akhir ini.
2. Bapak Purtojo, S.T.,M.Sc., sebagai dosen pembimbing yang telah berjasa meluangkan waktunya untuk membimbing, mengarahkan, memberikan ilmu serta mendukung penulis selama pembuatan dan penyusunan tugas akhir ini. Semoga ilmu yang diberikan dapat menjadi berkah yang berguna dikemudian hari. *Aamiin ya rabbal alamin.*
3. Fadhila Devani, selaku saudara kandung penulis. Terimakasih atas dukungan dan doa kakak selama ini.
4. Mutiara Nurul Salsabila, selaku *support system* hidup penulis. Sudah mau menampung keluh kesah penulis, sudah menjadi orang yang berpengaruh dalam hidup penulis. Terimakasih sudah menamani

penulis sampai saat ini. Terimakasih juga atas segala dukungan moral dan doa-mu selama ini.

5. Terimakasih untuk seluruh keluarga besar penulis baik dari keluarga ayah ataupun keluarga ibu, yang selalu siap sedia membantu dan memberi dukungan baik dari segi rohani ataupun jasmani.
6. Terimakasih untuk sahabat dan teman-teman penulis yang selalu memberikan candaan dan solusi atas masalah-masalah yang dihadapi.
7. Terimakasih untuk semua orang yang memiliki jasa atas kehidupan penulis. Tentunya penulis tidak akan bisa sampai dititik ini tanpa jasa-jasa dari kalian.
8. *Last but not least, I wanna thank me, I wanna thank me for believing in me, I wanna thank me for doing all this hardwork, I wanna thank me for having no days off, I wanna thank me for never quitting, I wanna thank me for always being a giver and tryna give more than I receive, I wanna thank me for tryna do more right than wrong, I wanna thank me for just being me at all times.*

Semoga amal dan perbuatan kalian dapat dibalas dengan kebaikan oleh Allah SWT. Penulis berharap pembaca laporan ini dapat memberikan kritik dan saran yang membangun untuk perkembangan tugas akhir ini. Semoga tugas akhir dapat memberikan manfaat bagi masyarakat umum.

***Wassalamualaikum warrahmatullahi wabarakatuh*.**

Yogyakarta, 08 November 2021

Penulis,

MUHAMMAD DEVAN FADHILA

PERANCANGAN DAN IMPLEMENTASI KONTROLER PID BERDASAR MODEL UNTUK KENDALI POSISI SUDUT MENGGUNAKAN SCILAB ARDUINO

Muhammad Devan Fadhila

ABSTRAK

Secara umum, sistem kendali adalah suatu usaha atau proses untuk mengolah masukan agar mendapatkan nilai pengeluaran. Penggunaan sistem kendali sudah banyak digunakan di segala bidang, baik bidang industri manufaktur maupun bidang lainya. Pengembangan sistem kendali juga sejalan dengan perkembangan teknologi, dimana saat ini teknologi sudah menggunakan otomatisasi dan kendali kontrol untuk mempermudah manusia dalam penggunaanya. Sistem kendali juga membutuhkan kendali kontrol atau kontroler agar nilai masukan yang ditentukan sesuai dengan nilai keluaran yang diinginkan. Pada tugas akhir kali ini, penulis melakukan penelitian terkait kendali kontrol untuk kendali posisi sudut, menggunakan prototipe rotary servo based unit. Komponen utama pada alat rotary servo based unit adalah motor DC dan sensor enkoder. Motor DC sebagai penggerak untuk menunjukan posisi sudut, sensor enkoder digunakan untuk pembacaan pergerakan posisi dari motor DC. Perangkat lunak yang digunakan dalam penelitian ini adalah Scilab Xcos untuk merancang model sistem dan mesimulasi sistem kendali. Metode sistem kendali yang digunakan merupakan metode kendali PID (Propotional, Integral, Derivative). Penggunaan kontroler PID bertujuan agar nilai masukan posisi sudut sesuai dengan nilai keluaran yang diinginkan.

***Kata kunci** : rotary servo based unit, kecepatan sudut, PID*

DESIGN AND IMPLEMENTATION PID CONTROLLER MODEL BASED FOR ANGULAR POSITION USING SCILAB ARDUINO

Muhammad Devan Fadhila

ABSTRACT

In general, control system is an attempt or process the specified input in order to get the value of output. The application of control system has been widely applied in all fields, such as in the manufacturing industry or in any other fields. In development of control systems is also in line with technological developments, which is currently using automation and controls to make it easier for humans to use it. The control system is also requires controller so that specified input value is in accordance with the desire output value. In this final project, the writer conducts research related to control for angular position, using a rotary servo based unit prototype. The main components of the rotary servo based unit are the DC motor and encoder sensor. DC motor as a driver to indicate the angular position, the encoder sensor used to read the position of the motor DC. The software used is Scilab Xcos that used for design the model system and simulate the control system. The PID (Propotional, Integral, Derivative) control is used as a control system method. The Purpose of using the PID controller is to make the input values for angular position match the desire output values.

***Keyword** : rotary servo based unit, angular position, PID*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR NOTASI

PID	: *Proportional*, *integral*, dan *derivative*	
PD	: *Proportional*, dan *derivative*	
K_P	: Konstanta proporsional pada motor DC	
K_I	: Konstanta integral pada motor DC	
K_D	: Konstanta derivatif pada motor DC	
K_{Ps}	: Konstanta proporsional pada kendali posisi	
K_{Ds}	: Konstanta derivatif pada kendali posisi	
P(s)	: Fungsi transfer motor DC	
Jm	: Momen inersia beban	(□□. □²)
B	: Koefisien gesek	(□□/[□□□/□])
□□□	: Konstanta torsi motor	(Nm/A)
□□	: Arus motor	(A)
ω	: Kecepatan motor	(rad/s)
T_r	: *Rise Time*	
T_s	: *Settling Time*	
SSE	: *Steady State Error*	

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi sangat berpengaruh pada revolusi industri, dimana revolusi industri pada abad ke-20 ini sudah mencapai tahap ke-4 yaitu industri 4.0. Pada revolusi industri 4.0 ini, peran manusia sangat sedikit dalam mengendalikan dan mengoperasikan mesin, dimana rata-rata semuanya sudah dikendalikan dan dioperasikan secara otomatis oleh mesin itu sendiri. Terutama pada bidang industri manufaktur. Karena banyaknya kebutuhan dalam mengendalikan objek kendali otomatis, hal ini juga sejalan dengan banyaknya penelitian mengenai perancangan sistem kendali.

Salah satu penggunaan sistem kendali dapat diterapkan pada pengendalian arah gerak sayap pesawat dan kapal. Dimana kedua hal tersebut memiliki kerja sistem kendali yang sama. Apabila *input* nilai sudut berbeda dengan respon pergerakanya, maka arah gerak pesawat maupun kapal tersebut berbeda arah tidak sesuai perintah. Untuk menghindari hal tersebut maka dibutuhkanya kontroler untuk sistem kendali.

Sistem pengendali sudut yang terdapat pada sayap pesawat dan kapal cukup kompleks dan rumit. Pada penelitian kali ini akan dibuat kontroler sistem kendali yang lebih sederhana.

Industri 4.0 sangat bergantung pada komputasi sistem/objek kendali. Mesin/robot mengendalikan hampir semua proses kerja agar waktu produksi jauh lebih efesien dan mengurangi terjadinya kesalahan *human error*. Suatu mesin dikendalikan oleh sistem/objek kendali yang dirancang untuk melakukan suatu perintah(*input*) dan perintah pengeluaranya(*output*) kerja mesin dapat berupa posisi sudut dan kecepatan rotasi motor.

Dalam prakteknya, sistem kendali masih dapat terjadi kesalahan, salah satu kesalahan utamanya yaitu perintah masukanya(*input*) tidak sesuai dengan pengeluaranya(*output*). Oleh karena itu, dibutuhkanya kontroler pada sistem

kendali agar presisi dimana perintah *input*-nya sama dengan *output*-nya dan tidak terjadi kesalahan pada kerja mesin.

Banyak penelitian yang sudah dilakukan terkait kontroler pada sistem kendali seperti "Analisis Penalaan Kontrol PID pada Simulasi Kendali Kecepatan Putaran Motor DC berbeban menggunakan Metode Heuristik" (Waluyo, Fitriansyah, & Syahrizal, 2013) dimana penulis menggunakan Metode Heuristik untuk analisis penalaan kontroler-nya. Dan "*DC Motor Angular Position Control using PID Controller with Friction Compesation*" (Maung, Latt, & Myat, 2018) dimana penulis menggunakan kompensasi gesekan pada alat untuk kontroler-nya. Pada penelitian "Implementasi Pengendali Posisi Motor DC Berbasis PID dengan Interface Mikrokontroler dan Matlab pada Laboratorium Sistem Kendali Digital" (Prasetyo & Tarmukan, 2017) dijelaskan pengendali posisi sudut menggunakan kontroler PID menggunakan teknik automasi pada aplikasi Matlab. Pada penelitian kali ini menggunakan aplikasi Sci-Lab dan metode hitungan manual untuk pencarian nilai PID.

Namun bagaimana kontroler pada sistem kendali posisi sudut yang sederhana? Nah, pada penelitian kali ini penulis akan merancang kontroler untuk alat prototipe sederhana pada penelitian yang sudah dilakukan sebelumnya yaitu "Pemodelan dan Pembuatan Prototipe Sistem Kendali Posisi dan Kecepatan Sudut" (Pratama, 2020) Karena pada penelitian tersebut, kinerja masih belum selesai pada alat prototipe dimana nilai *input* pada kendali posisi masih belum sesuai dengan nilai pengeluaranya(*output*). Komponen utama pada prototipe tersebut ialah motor DC dan enkoder yang digunakan untuk membaca pergerakan motor. Pembacaan untuk pergerakan sudut pada motor DC nanti menggunakan penunjuk(*pointer*) yang terhubung pada motor DC.

Perangkat lunak yang digunakan pada penelitian saya kali ini yaitu Scilab Xcos yang berfungsi untuk melakukan simulasi kerja sistem kendali dan merancang model sistematis dari sistem kendali maupun kontroler-nya.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah disampaikan, maka didapatkan rumusan masalah sebagai berikut :

1. Bagaimana merancang model kendali posisi sudut menggunakan kontroler/sistem kendali PID?
2. Bagaimana mengimplementasikan kontroler/sistem kendali PID pada kendali posisi sudut?
3. Bagaimana mengevaluasi kinerja kontroler/sistem kendali PID pada kendali posisi sudut?

1.3 Batasan Masalah

Penelitian ini dilakukan

1. Alat uji yang digunakan adalah alat prototipe yang telah dirancang pada penelitiannya tentang "Pemodelan dan Pembuatan Prototipe Sistem Kendali Posisi dan Kecepatan Sudut" (Pratama, 2020).
2. Kontroler PID digunakan sebagai metode kendali untuk me-presisikan hasil agar nilai outputnya sesuai dengan nilai inputnya.
3. Penelitian yang dilakukan sampai dengan implementasi kontroler PID pada sistem kendali posisi sudut pada prototipe.
4. Perangkat lunak yang digunakan adalah Sci-lab versi 5.5.2 dan Arduino IDE.
5. Pemrograman Arduino menggunakan arduino *toolbox v3*.
6. Arduino mega 2560 digunakan sebagai mikrokontroller.
7. Jenis enkoder yang digunakan adalah *rotary encoder* 400 pulsa per revolusi.
8. Motor DC yang digunakan memiliki kecepatan 255 rpm.

1.4 Tujuan Penelitian atau Perancangan

Tujuan yang ingin dicapai dari penelitian ini adalah:

1. Merancang sistem kendali posisi menggunakan kontroler/sistem kendali PID.
2. Mengimplementasikan sistem kendali menggunakan kontroler PID pada alat prototipe.
3. Mengevaluasi kinerja kontroler/sistem kendali PID apakah nilai input sudah sama dengan outputnya.

1.5 Manfaat Penelitian atau Perancangan

Manfaat dari penelitian yang dilakukan penulis adalah dapat membandingkan kinerja objek kendali yang menggunakan kontroler dengan kinerja objek kendali yang tidak menggunakan kontroler. Peneliti juga mampu menerapkan pembelajaran yang telah dipelajari pada masa kuliah. Penelitian ini yang nantinya diharapkan dapat dilakukan penelitian lebih lanjut dan pembaruan perangkat lunak dan keras.

1.6 Sistematika Penulisan

Sistematika penulisan pada tugas akhir ini dibagi menjadi lima bab, diantaranya:

BAB I PENDAHULUAN

Bab ini terdiri dari latar belakang penelitian, rumusan masalah yang dihadapi, batasan masalah, tujuan serta manfaat dari pelaksanaan penelitian, dan sistematika penulisan laporan.

BAB II TINJAUAN PUSTAKA

Bab ini terdiri dari landasan teori untuk melakukan penelitian terkait seperti rumus-rumus yang akan digunakan dan juga referensi-referensi dari penelitian sebelumnya.

BAB III METODE PENELITIAN

Bab ini membahas tentang alur penelitian yang akan dikerjakan, perangkat keras yang digunakan, serta pengaturan perangkat lunak seperti pengaturan pada aplikasi arduino dan scilab x-cos.

BAB IV HASIL DAN PEMBAHASAN

Bab ini menunjukkan dan membahas hasil dari penelitian yang telah dilakukan.

BAB V PENUTUP

Bab ini berisi tentang kesimpulan dari penelitian serta kritik dan saran untuk penelitian selanjutnya.

BAB 2
TINJAUAN PUSTAKA

2.1 Kajian Pustaka

Sudah banyak penelitian tentang penerapan sistem kendali posisi dan kecepatan sudut dan sudah banyak pula yang sudah dikembangkan..Namun, belum banyak penelitian yang membahas tentang bagaimana cara mengatur sistem kendali tersebut agar nilai outputnya sesuai dengan nilai inputnya. Salah satu cara mengatur sistem kendali tersebut agar sesuai ialah dengan menggunakan kontroler PID. Pada penelitan ini bertujuan untuk menambahkan kontroler PID pada sistem kendali posisi dan kecepatan sudut yang sudah dilakukan oleh peneliti sebelumnya.

Oleh karena itu, diperlukan kajian pustaka yang dapat dijadikan referensi untuk penelitian ini. Didapatkan beberapa referensi yang membahas tentang kontroler PID pada sistem kendali. Seperti pada jurnal yang berjudul "Implementasi Pengendali Posisi Sudut Motor DC Berbasis PID dengan Interface Mikrokontroler dan Matlab pada Laboratorium Sistem Kendali Digital" (Prasetyo & Tarmukan, 2017) yang membahas tentang kontrol PID pada sistem kendali posisi motor DC.

Pada penelitian kali ini menggunakan metode sistem kendali PID(PD) untuk mengendalikan motor agar pergerakan untuk membentuk sudut yang sesuai dengan masukanya. Skema sistem kendali PID dari jurnal tersebut menyerupai skema objek kendali pada penelitian kali ini. Sehingga skema sistem kendali pada jurnal tersebut dapat digunakan sebagai acuan pada penelitian kali ini. Perbedaan dan kelebihan penelitian ini dibandingkan dengan jurnal diatas adalah skema penggunaan PID jauh lebih sederhana dan pencarian nilai konstanta PID nya cukup mudah dengan ketentuan yang sudah ada.

Penelitian sebelumnya membahas perancangan objek kendali *motor servo base unit* dan pengimplementasianya secara langsung (Pratama, 2020). Penelitian ini membahas bagaimana merancang objek kendali *motor servo base unit* dengan komponen yang mudah didapatkan dan dengan karakteristik yang sudah

ditentukan. Pembahasan pada penelitian ini hingga objek kendali *motor servo base unit* selesai dibuat dan dapat beroperasi dengan baik.

Hasil dari kedua referensi diatas ialah kontroler PID dapat mengatur agar *output* dari sistem kendali dapat sesuai dengan nilai input-nya. Tetapi kedua jurnal tersebut terdapat perbedaan pada metode yang digunkan untuk men-*tuning* nilai PID nya.

2.2 Dasar Teori

2.2.1 Motor DC

Motor DC merupakan suatu alat atau perangkat yang mengubah energi listrik menjadi energi kinetik atau gerak. DC Motor memiliki dua terminal dan memerlukan tegangan arus searah atau DC (*Direct Current*) untuk dapat menggerakanya. Motor DC ini menghasilkan sejumlah putaran per menit (*RPM*) dan dapat dibuat berputar searah jarum jam maupun sebaliknya, tergantung pada polaritas listrik yang diberikan pada motor DC tersebut. Besar atau kecilnya tegangan yang diberikan kepada kedua terminal menentukan kecepatan motor.

Gambar 2-1 Motor DC

2.2.2 Rotary Encoder

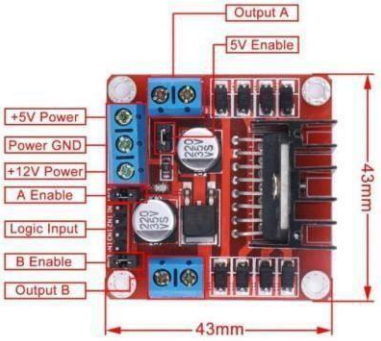
Rotary Encoder adalah perangkat elektro mekanik yang mengubah posisi sudut atau gerakan poros menjadi sinyal keluaran analog atau digital. Jenis *rotary encoder* yang digunakan adalah optikal enkoder. Secara umum, enkoder ini menggunakan sensor optik untuk menghasilkan serial pulsa yang dapat diartikan gerakan dan posisi. Sehingga sinyal yang masuk dapat diolah menjadi informasi berupa kode digital.

Gambar 2-2 *Rotary Encoder*

2.2.3 Driver Motor

Pada penelitian kali ini driver motor yang digunakan yaitu seri L298N sebagai pengatur kerja motor DC. Driver L298N ini mampu mengontrol dan menggerakan 2 motor DC sekaligus. *Integated Circuit* (IC) L298N memiliki kemampuan untuk menggerakan motor DC sampai arus 2A dan tegangan maksimal 40V.

Pada driver motor ini, tersedia pin-pin dengan kegunaanya masing-masing. Pin *output* A dan B untuk mengendalikan kecepatan motor, pin 5V dan 12V merupakan pin untuk *input* voltase, pin *output* IC 13 dihubungkan ke motor DC yang sebelumnya melalui diode yang disusun secara H-Bridge, dan pin GND untuk koneksi ke *ground*. Pengaturan kecepatan motor digunakan teknik PWM (*Pulse Width Modulation*) yang dimasukan dari mikrokontroler.



Gambar 2-3 Driver Motor L298N.

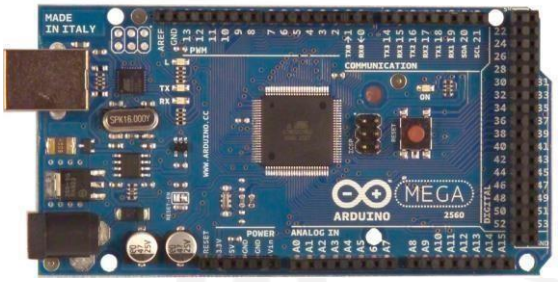
2.2.4 Arduino

Arduino adalah suatu papan mikrokontroler *open source hardware* yang dapat digunakan untuk membuat proyek berbasis pemrograman. Arduino banyak

jenisnya, pada penilitian ini menggunakan jenis arduino MEGA. *Hardware* arduino memiliki prosesor tersendiri yaitu ATMega 2560, dan *software* bernama arduino IDE sebagai aplikasi pemrogramanya. Arduino menggunakan bahasa C++ sebagai bahasa pemrogramanya. Arduino banyak penggunaanya, dengan cara kerja menerima *input* dari berbagai sensor atau tombol (sensor cahaya, suhu, inframerah, ultrasonik, jarak, tekanan, dan kelembapan) dan dapat mengontrol perangkat lainya seperti mengontrol kecepatan motor, arah putar motor, menyalakan LED, mengatur waktu permesinan, dan lain sebagainya. Berikut spesifikasi lengkap dari arduino MEGA 2560 dan gambarnya.

Tabel 2-1 Spesifikasi Arduino MEGA 2560

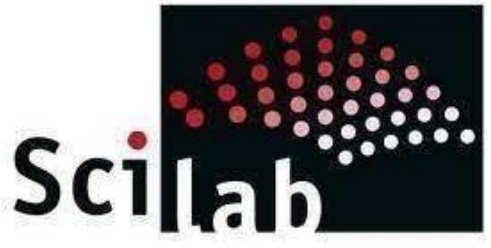
Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz



Gambar 2-4 Arduino MEGA 2560

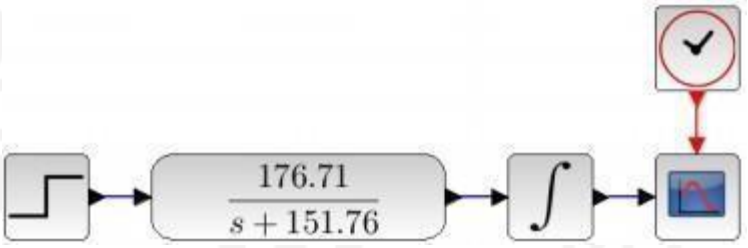
2.2.5 Scilab Xcos

Scilab adalah perangkat lunak (*software*) yang *open source* yang digunakan untuk komputasi numerik atau permodelan matematis seperti kalkulasi integral, diferensial, aljabar, statistik dan lain sebagainya. Karena Scilab merupakan aplikasi *open source* oleh karena itu aplikasi ini dapat di *download* secara gratis dan digunakan oleh semua kalangan.



Gambar 2-5 *Software* Scilab

Pada penelitian kali ini, fitur yang digunakan ialah Xcos untuk melakukan simulasi model matematis. Xcos merupakan bagian dari scilab GUI (*Graphic User Interface*). Xcos memiliki permodelan blok matematis yang memiliki fungsinya masing-masing yang dihubungkan antara satu dengan yang lainya untuk membuat diagram sistem kendali.



Gambar 2-6 Contoh permodelan pada Xcos Scilab

2.2.6 Power Supply (Catu Daya)

Power supply atau catu daya merupakan sebuah alat yang menjadi sumber utama listrik energi untuk satu atau lebih beban listrik. Prinsip kerja catu daya lebih kurang sama seperti baterai atau accu sebagai sumber utama energi. Prinsip rangkaian catu daya terdiri atas komponen utama yaitu: transformator, dioda, dan kondensator.

Gambar 2-7 Catu Daya

2.2.7 Model Matematika pada Motor DC

Model matematika pada motor DC berupa fungsi transfer. Fungsi transfer suatu sistem adalah model matematika karena merupakan metode operasional untuk mengekspresikan persamaan diferensial maupun integral yang menghubungkan variabel *input* ke variabel *output* (Ogata, 2010). Terdapat 2 fungsi transfer yang bisa dicari yaitu dengan nilai dari *datasheet* motor DC itu sendiri dan dari fenomena eksperimen motor DC. Jika fungsi transfer suatu sistem sudah diketahui, maka *output* atau respon dapat dipelajari untuk berbagai bentuk *input* dengan pandangan untuk memahami sifat sistem. Motor DC digunakan sebagai alat penggerak pada alat *motor servo base unit* dimana motor DC terhubung dengan lengan penunjuk yang dikoneksikan dengan *pulley* dan *belt* untuk menggerakan ke sudut tertentu yang ditentukan.

2.2.7.1 Fungsi Transfer Eksperimen

Motor sangat diperlukan dalam banyak sistem kendali. Motor digunakan untuk memutar antena, teleskop, kemudi kapal, dan sayap pesawat; untuk membuka dan menutup katup; dan rol di pabrik baja. Terdapat beberapa tipe motor: DC, AC stepper, dan hidrolik. Medan magnet motor DC dapat tereksitasi oleh rangkaian yang terhubung secara seri dengan yang tidak(magnet permanen). Motor DC digunakan juga untuk mengendalikan sistem yang disebut dengan motor servo yang memiliki ciri-ciri rasio torsi inersia rotor besar, ukuran kecil, dan karakteristik linier yang baik.

Field-Controlled DC Motor

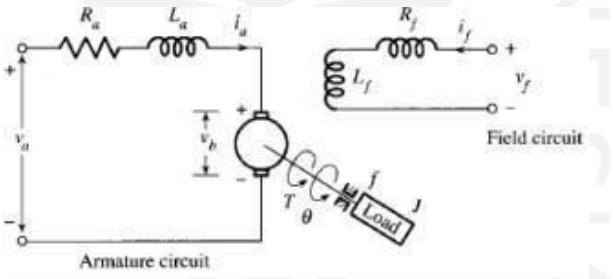
Sebagian besar motor DC digunakan pada sistem kendali yang dimodelkan dalam persamaan (2.1). terdapat dua rangkaian yang disebut dengan rangkaian medan dan rangkaian *armature*. Arus medan □□, □□ arus *armature*, dan torsi T yang dihasilkan oleh motor dituliskan dalam persamaan berikut

$$□(□) = □□□(□)□□(□) \tag{2.1}$$

dimana *k* adalah konstanta. Torsi yang dihasilkan digunakan untuk menggerakan beban melalui poros yang diasumsikan kaku. Analisis disederhanakan dengan hanya mempertimbangkan gesekan antara poros dan *bearing*. Total momen inersia beban, poros, dan rotor motor (*J*); perpindahan sudut beban θ; dan koefisien gesekan *bearing* (*f*) sehingga dapat ditulis pesamaanya sebagai berikut.

$$□(□) = □\frac{□^2□(□)}{□□^2} + □\frac{□□(□)}{□□} \tag{2.2}$$

Persamaan tersebut merupakan hubungan antara torsi motor dan perpindahan beban angular.



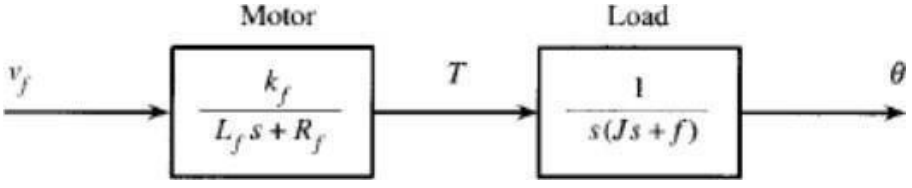
Gambar 2-8 Sirkuit Motor DC

Jika arus *armature* tetap konstan dan tegangan *input u(t)* diaplikasikan pada rangkaian medan, motor disebut *field-controlled dc motor*. Pada motor tersebut, arus $i_\alpha(t)$ adalah konstan. Oleh karena itu (2.1) dapat direduksi menjadi

$$□(□) = □□□(□)□□ = □□(□)□□(□) \tag{2.3}$$

Dimana $k_f = i_\alpha$.

Motor dan beban dapat dimodelkan seperti gambar (2-9).



Gambar 2-9 Blok diagram *field-controlled* DC motor

Terdapat dua blok pada gambar: blok sebelah kiri mempresentasikan motor dan blok sebelah kanan beban. Penambahan atau pengurangan beban tidak mempengaruhi fungsi transfer.

Fungsi transfer *field controlled dc motor* dengan beban dituliskan dalam persamaan berikut:

$$\square(\square) = \frac{\square_{\square}}{(\square_{\square}\square+1)\square(\square_{\square}\square+1)} \tag{2.4}$$

dimana $\square$ $\frac{\square_{\square}}{\square_{\square}\square_{\square}}$, $\square$ $\frac{\square_{\square}}{\square_{\square}}$, dan $\square$ $\frac{\square_{\square}}{\square_{\square}\square_{\square}}$. Konstanta $\square$ hanya bergantung pada rangkaian elektrik yang dapat disebut sebagai *electrical time constant* dan konstanta $\square_{\square}$ hanya tergantung pada beban yang dapat disebut dengan *mechanical time constant*. Jika *electrical time constant* lebih kecil dari *mechanical time constant*, seperti yang sering terjadi pada praktiknya, fungsi transfer dapat dirumuskan sebagai berikut:

$$\square(\square) = \frac{\square_{\square}}{\square(\square_{\square}\square+1)} \tag{2.5}$$

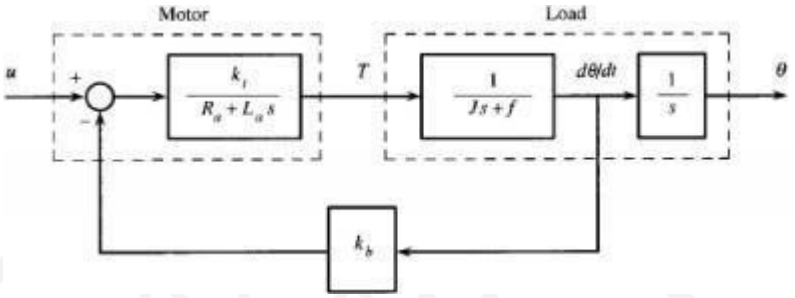
Field controlled dc motor digunakan untuk menggerakan beban. Perbedaan beban akan menyebabkan perbedaan $\square_{\square}$; bahkan untuk beban yang sama, hal tersebut tidak akan menjadi konstanta. Oleh karena itu, permasalahan pembebanan dieleminasi dengan menjaga konstanta $\square_{\square}$. Dalam penerapanya, menjaga konstanta tersebut sangatlah sulit, jadi *field controlled dc motor* jarang digunakan.

Armature-Controlled DC Motor

Berdasarkan sirkuit motor DC pada gambar (2-8), jika arus medan $\square_{\square}(\square)$ tetap konstan atau rangkaian medan diubah menjadi medan magnet permanen dan jika tegangan *input* diterapkan ke rangkaian *armature motor* disebut *armature controlled motor dc*. Fungsi transfer dikembangkan dan jika $\square_{\square}(\square)$ adalah konstanta, maka menjadi

$$\square_{\square}(\square) = \square \frac{\square\square(\square)}{\square\square} \tag{2.6}$$

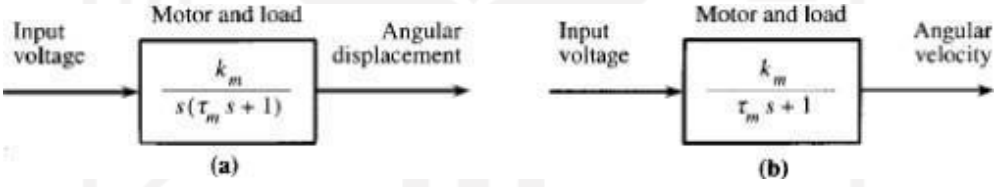
Untuk *armature-controlled dc motor* dapat digambarkan ke dalam blok diagram seperti berikut:



Gambar 2-10 Blok diagram *armature controlled DC motor*

Meskipun dapat digambarkan blok untuk motor dan blok untuk beban, dua blok tidak terpisah seperti pada *field-controlled dc motor*. Sinyal dari dalam beban adalah umpan balik ke *input* motor. Pada dasarnya hal ini dapat mengatasi masalah pembebanan, lebih mudah untuk menggabungkan motor dan beban ke dalam satu blok dengan fungsi transfer.

Time constant tergantung pada beban serta *armature circuit*. Fungsi transfer *armature-controlled dc motor* dapat digambarkan sebagai berikut:



Gambar 2-11 Fungsi transfer *armature-controlled dc motor*

Dari persamaan fungsi transfer motor diatas, tegangan yang diterapkan untuk kendali posisi adalah:

$$\frac{\square(\square)}{\square(\square)} = \frac{\square\square}{\square\square\square+1} \qquad (2.7)$$

Pada fungsi transfer motor (2.7), penting untuk menentukan *input* dan *output*. Perbedaan spesifikasi menyebabkan fungsi transfer yang berbeda untuk sistem yang sama.

Pengukuran Fungsi Transfer Motor

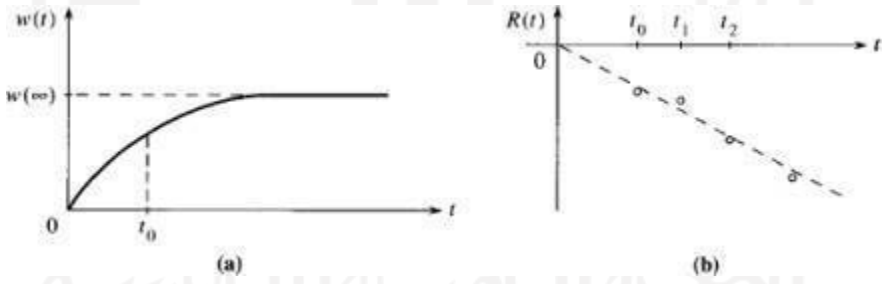
Fungsi transfer motor yang menggunakan perhitungan □□ dan □□ dimana untuk menghitung keduanya, dibutuhkan □□, □□, □□, □□, *J*, dan *f*. Meskipun empat konstanta pertama mungkin disediakan oleh manufaktur motor, konstanta lain yang dibutuhkan adalah *J* dan *f*. Momen inersia *J* beban tidak mudah dihitung jika mempunyai bentuk regular dan *f* tidak dapat diperoleh secara analistis. Oleh karena itu, untuk memperoleh *J* dan *f* dilakukan pengukuran. Pengukuran juga dapat digunakan untuk mengetahui fungsi transfer secara langsung. Berikut ini cara menghitung fungsi transfer motor.

Tegangan listrik dan kecepatan angular digunakan untuk pengukuran, kemudian fungsi transfer sudah disederhanakan menjadi (2.7). jika tegangan listrik α kecepatan dapat dirumuskan

$$□(□) = □_□□ - □_□□□^{-□/□□} \quad (2.8)$$

Persamaan diatas sudah bentuk invers transformasi Laplace. Untuk $t \geq 0$, karena □□ adalah positif, hubungan □^−□/□□ mendekati nol seperti *t* mendekati tak terbatas dan respon *w(t)* adalah seperti pada gambar (2-10(a)). Jadi, dapat dituliskan

$$□(\infty) = □_□□ \quad (2.9)$$



Gambar 2-12 (a) Respon motor; (b) Estimasi □□

Hal ini disebutkan kecepatan akhir atau *steady-state speed*. Jika tegangan listrik diketahui besarnya α dan ukuran kecepatan akhir *w(∞),* kemudian *motor gain constant* dapat ditulis

$$-\frac{□}{□□} = \ln\left(1 - \frac{□(□)}{□(\infty)}\right) = □(□) \quad (2.10)$$

Jika kecepatan *t* berapapun, asumsikan *t* = □0 kemudian dari (2.10) diperoleh

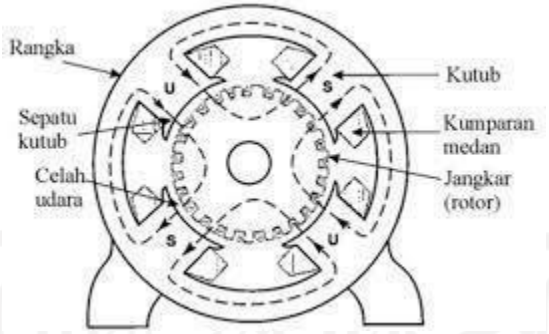
$$\square\square = \frac{-\square 0(\,)}{\ln(1-\frac{\square\square}{\square(\infty)})} \qquad (2.11)$$

Jadi, *time motor constant* dapat diperoleh dari kecepatan akhir dan satu tambahan pengukuran pada *t* berapapun.

Pada (2.11) selain kecepatan akhir hanya digunakan satu datum untuk mengukur □□, untuk □□ yang lebih akurat dapat diperoleh dari data yang lebih detail. Kemiringan garis lurus sama dengan 1/□□. Hal ini lebih akurat untuk memperoleh *motor time constant*. Metode ini juga dapat untuk memeriksa fungsi transfer (2.7) apakah bisa digunakan atau tidak.

Masalah dalam menentukan □□ dan □□ pada (2.7) dari pengukuran diasumsikan diketahui parameternya, tetapi kenyataan parameter tersebut tidak diketahui. Parameter yang tidak diketahui sebelumnya tersebut dapat diketahui dengan fenomena yang terjadi pada pergerakan motor DC. Hal ini adalah kasus khusus dari masalah identifikasi dimana bentuk maupun parameternya diasumsikan belum diketahui sebelumnya.

2.2.7.2 Fungsi Transfer *Datasheet*



Gambar 2-13 Rangka dalam motor DC

Pada sistem kontrol, motor DC memiliki nilai fungsi transfer sendiri untuk memformulakan karakteristiknya. Data yang dibutuhkan dalam perumusan nilai fungsi transfer didapatkan dari *datasheet* motor DC.

$$\square(\square) = \frac{1}{\square(\square\square\square+\square)} \qquad (2.12)$$

Dari perumusan diatas, diketahui *P(s)* adalah fungsi transfer yang akan digunakan; □□ adalah momen inersia beban (□□. □²); dan *B* adalah koefisien gesek

(□□/[□□□/□]). Nilai momen inersia beban tergantung pada beban yang diberkan kepada motor DC. Sedangkan untuk nilai koefisien gesek dapat dicari menggunakan perumusan sebagai berikut:

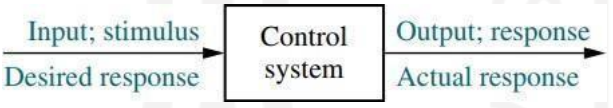
$$\square = \frac{\square_{\square\square} \cdot \square_{\square}}{\square} \quad (2.13)$$

Nilai $\square_{\square\square}$ adalah konstanta torsi motor (*Nm/A*); $\square_{\square}$ adalah arus motor (*A*); dan ω adalah kecepatan motor (*rad/s*). Konstanta atau nilai yang dibutuhkan dalam perumusan (2.12) dan (2.13) didapatkan dari *datasheet* motor DC (Toochinda, 2016).

Data yang digunakan dalam penelitian kali ini semua berada pada kolom 24V pada *datasheet*, karena tegangan yang digunakan oleh alat *motor servo base unit* ini ialah 24V. Karakteristik *datasheet* dari setiap motor DC juga berbeda-beda, hal ini juga dipengaruhi oleh ukuran dimensi motor DC dan perusahaan yang memproduksinya.

2.2.8 Sistem Kontrol

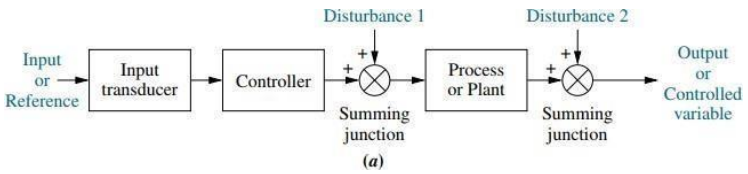
Sistem kontrol memiliki sebuah sistem atau proses yang bertujuan untuk mendapatkan *output* yang diinginkan dengan nilai *input* yang ditentukan(Nise, 2015). Berikut Gambar 2-14 menunjukan penjelasan sistem kontrol sederhana.



Gambar 2-14 Sistem kontrol sederhana

Dari gambar diatas, nilai input yang dimasukan lalu diproses oleh sistem kontrol dan menghasilkan *output*. Terdapat 2 jenis sistem kontrol yaitu sistem kontrol *open loop* dan sistem kontrol *closed loop*.

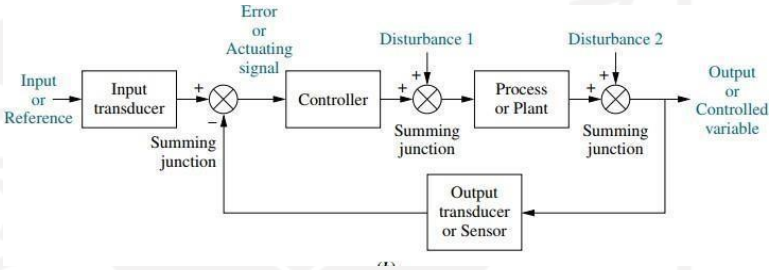
2.2.8.1 Sistem Kontrol *Open Loop*



Gambar 2-15 Sistem kontrol *open loop*

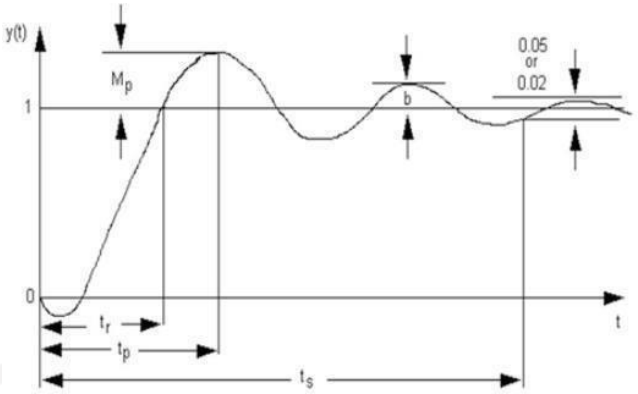
Dari Gambar 2-15 diatas adalah alur kerja sistem kontrol *open loop*. Nilai *input* akan diterima oleh *input transducer*, lalu akan diteruskan ke kontroler. Dalam kontroler , data yang di *input* kan tadi akan di proses sehingga menghasilkan nilai *output.* Dalam proses yang terjadi, terdapat beberapa gangguan yang dapat mempengaruhi nilai *output* nya. Hal tersebutlah yang menjadi kekurangan sistem *open loop* ini, tidak mampu mendeteksi gangguan yang terjadi selama pemrosesan. Tetapi kelebihanya ialah sistem ini lebih *simple* dibandingkan sistem *closed loop*.

2.2.8.2 Sistem Kontrol *Closed Loop*



Gambar 2-16 sistem kontrol *closed loop*

Pada sistem *closed loop*, sistem mampu mendeteksi gangguan yang terjadi selama proses pemrosesan. Hal ini dapat terjadi karena pada sistem *closed loop* ini memiliki sensor yang mengembalikan data hasil ketika tidak sesuai dengan nilai *output* yang diinginkan. Sensor ini mengembalikan data ke bagian *swimming junction* , ketika hasil sudah sesuai baru dapat diteruskan ke proses berikutnya atau nilai akhir *output* nya. Pengembalian data kembali dari *swimming junction* pada sistem kontrol *closed loop* ini dikenal dengan istilah *feedback. Feedback* ini yang berfungsi untuk melacak apakah performa sistem sudah sesuai dan juga dapat mendeteksi *error* sistemnya (Toochinda, 2016).



Gambar 2-17 Grafik sistem kontrol *closed loop*

Berdasarkan Gambar 2-17 adalah karakteristik dari sistem *closed loop*. *Rise time* (□□) adalah waktu yang dibutuhkan untuk terjadinya respon pada sistem, *time peak* (□□) adalah waktu ketika sistem berapa pada puncak, *settling time* (□□) waktu ketika sistem sudah berada pada kondisi *steady state*, dan *overshoot* sistem (□□). Dari beberapa karakteristik tersebut, dapat ditentukan respon sistem yang diinginkan dengan menggunakan kontroler.

2.2.9 PID

PID merupakan singakatan dari *Propotional, Integral, Derivative*. PID merupakan salah satu kontroler yang paling sering digunakan dalam sistem untuk mengatur atau mengontrol objek kendali agar respon sistem sesuai dengan yang diinginkan. PID mempunyai persamaan umumnya sebagai berikut

$$□(□) = □ \; □(□) + □ \int_{□}^{□} □(□)□□ + □ \frac{□□(□)}{□□} \qquad (2.14)$$

PID mempunyai 3 variabel nilai utama dalam perhitunganya untuk mengatur objek kendali yaitu konstanta *error proposional* (□□); konstanta *error integral* (□□); dan konstanta *error derivative* (□□). Ketiga variabel nilai tersebut dapat didefinisikan dengan perbandingan sebagai berikut

$$□□ = □ \; ; □ = \frac{□}{□□} \; ; □ = \frac{}{□□□} \qquad (2.15)$$

Masing-masing variabel nilai / konstanta PID tersebut memiliki pengaruh untuk mengendalikan respon sistem. Kontroler PID dapat mempengaruhi respon sistem, berikut tabel pengaruh dari setiap konstanta PID

Tabel 2-2 Karakteristik konstanta PID

Respon closed loop	Rise time	Overshoot	Settling time	SS error
Kp	Berkurang	Bertambah	Berubah sedikit	Berkurang
Ki	Berkurang	Bertambah	Bertambah	Lenyap
Kd	Berubah sedikit	Berkurang	Berkurang	Berubah sedikit

Tabel 2-2 diatas adalah pengaruh konstanta PID terhadap *sistem closed loop*. Nilai dari setiap konstanta PID dapat dicari menggunakan beberapa metode. Metode yang digunakan dalam penelitian kali ini ialah metode ZNFD dan Cohen-Coon. Terdapat keunikan kedua metode *tuning* PID tersebut, ialah metode ZNFD tipe 1 karakteristik pencarian yang dibutuhkan sama seperti metode Cohen-Coon Tipe 2, hanya berbeda pada rumus pencarianya saja. Begitu pula sebaliknya, metode ZNFD tipe 2 sama dengan metode Cohen-Coon Tipe 1, hanya berbeda pada rumus pencarianya saja.

2.2.9.1 Metode ZNFD

Metode ZNFD atau *Ziegler-Nichols Frequency Domain* adalah metode yang paling umum dan yang paling sering digunakan sebagai kontroler untuk beberapa objek kendali. Terdapat 2 tipe metode ZNFD yaitu tipe 1 dan tipe 2. Perbedaan dari keduanya adalah bergantung pada respon objek kendalinya, ZNFD tipe 1 digunakan ketika respon sistem berhenti ketika sudah mencapai nilai *steady state*-nya. Sedangkan untuk ZNFD tipe 2 digunakan ketika respon sistem bergerak konstan ketika sudah mencapai nilai *steady state*-nya atau hasil grafiknya berosilasi.

ZNFD Tipe 1

Tuning PID menggunakan metode ZNFD tipe 1 memiliki karakteristik respon sistem seperti berikut

Gambar 2-18 Kurva karakteristik ZNFD tipe 1

Berdasarkan gambar grafik diatas, diketahui L adalah nilai *rise time* yang bersinggungan dengan garis; T adalah nilai saat memasuki kondisi *steady state*; dan K adalah nilai *steady state* aktual. Selanjutnya nilai yang sudah diketahui tersebut, dimasukan kedalam perhitungan konstanta nilai PID menggunakan rumus sebagai berikut:

Tabel 2-3 Perhitungan nilai PID metode ZNFD tipe 1

Bentuk kontroller	□□	□□	□□
P	□/□	-	0
PI	0,9 □/□	□/0,3	0
PID	1,2 □/□	2□	0,5□

2.2.9.2 Metode Cohen-Coon

Metode Cohen-Coon ini adalah metode *tuning* PID alternatif ketika menggunakan metode ZNFD nilai PID yang diinginkan tidak ditemukan. Sama seperti metode ZNFD, metode Cohen Coon juga memliki 2 tipe yaitu tipe 1 dan tipe 2. Perbedaan dari kedua tipe tersebut juga sama dengan metode ZNFD yaitu pada respon objek kendalinya.

Cohen-Coon Tipe 2

Tuning PID menggunakan metode Cohen-Coon tipe 2 memiliki karakteristik yang sama dengan metode ZNFD tipe 1 seperti pada Gambar 2-18. Setelah nilai dari karakteristik sudah diketahui, masukkan kedalam rumus untuk mencari nilai kontanta PID nya menggunakan rumus berikut:

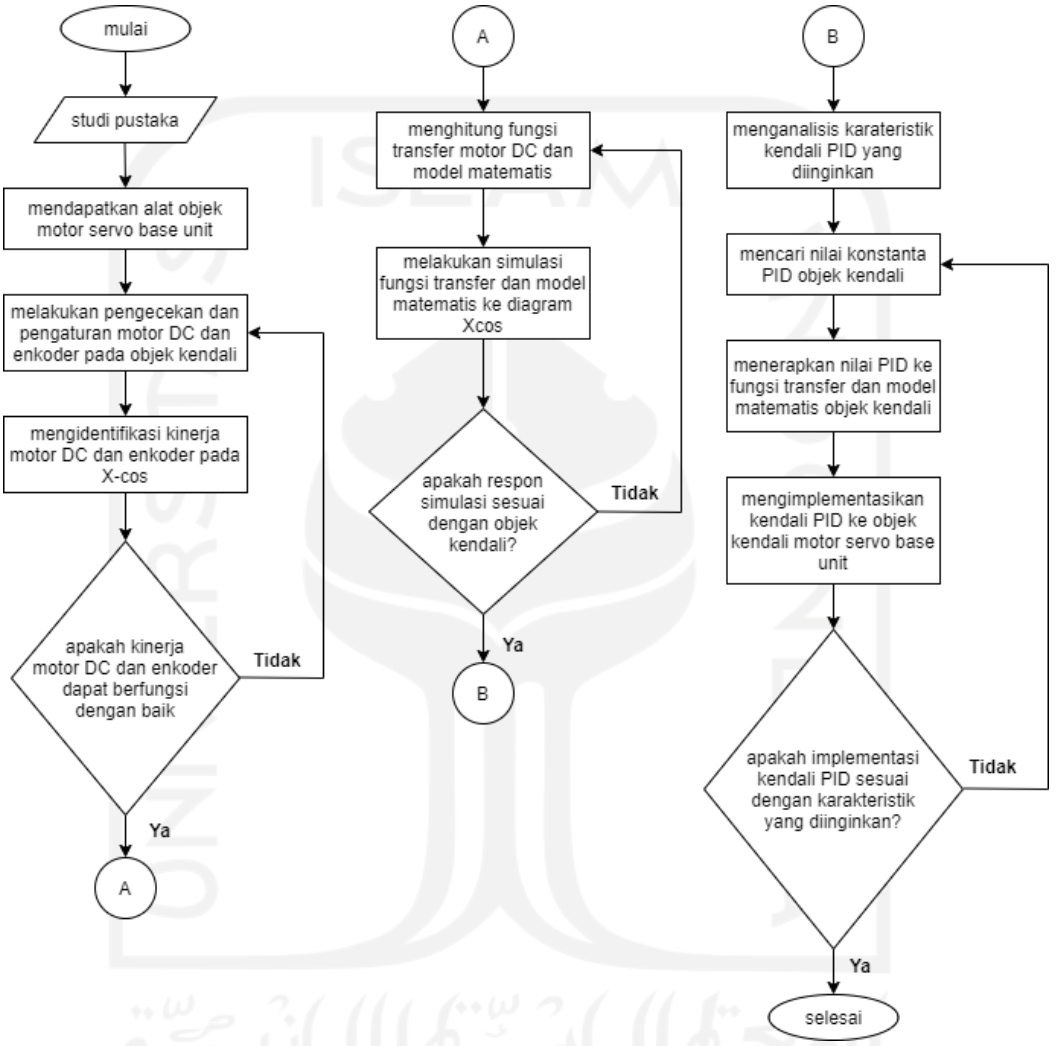
Tabel 2-4 Perhitungan nilai PID metoode Cohen-Coon tipe 2

Bentuk kontroller	K_c	τ_I	τ_D
P	$\frac{1}{K}\left(\frac{\tau}{\theta}\right)\left[1+\frac{\theta}{3\tau}\right]$	-	-
PI	$\frac{1}{K}\left(\frac{\tau}{\theta}\right)\left[0{,}9+\frac{\theta}{12\tau}\right]$	$\theta\left[\frac{30+3\left(\frac{\theta}{\tau}\right)}{9+2\left(\frac{\theta}{\tau}\right)}\right]$	
PD	$\frac{1}{K}\left(\frac{\tau}{\theta}\right)\left[\frac{5}{4}+\frac{1}{6}\left(\frac{\theta}{\tau}\right)\right]$	-	$\theta\left[\frac{6-2\left(\frac{\theta}{\tau}\right)}{22+3\left(\frac{\theta}{\tau}\right)}\right]$
PID	$\frac{1}{K}\left(\frac{\tau}{\theta}\right)\left[\frac{4}{3}+\frac{1}{4}\left(\frac{\theta}{\tau}\right)\right]$	$\theta\left[\frac{32+6\left(\frac{\theta}{\tau}\right)}{13+8\left(\frac{\theta}{\tau}\right)}\right]$	$\theta\left[\frac{4}{11+2\left(\frac{\theta}{\tau}\right)}\right]$

BAB 3

METODE PENELITIAN

3.1 Alur Penelitian



Gambar 3-1 Alur penelitian

3.2 Perancangan

Perancangan pengendali posisi sudut pada penelitian kali ini terbagi menjadi dua yaitu perancangan perangkat keras dan perancangan perangkat lunak. Objek kendali *motor servo base unit* yang digunakan pada penelitian kali ini sudah terlebih dahulu dirancang pada penelitian sebelumnya.

3.2.1 Perancangan Perangkat Keras

Perancangan perangkat keras yang *motor servo base unit* menggunakan referensi dari Quanser sebagai acuan bentuk objek kendali. Kerangka susunan perangkat keras seperti pada gambar dibawah ini.

Gambar 3-2 Model rancangan perangkat keras *motor servo base unit*.

Berdasarkan gambar 3-2 rancangan perangkat keras diatas, susunan alat dibuat 2 alas bertingkat. Motor DC dan enkoder terletak pada alas pertama(atas) dimana kedua komponen utama tersebut dihubungkan dengan 2 buah *pulley* yang berdiameter sama dan *belt*. Lalu *pulley* tersebut dipasangkan lengan penunjuk untuk menunjukan hasil sudut. Alas kedua(bawah) terdiri dari *power supply*, mikrokontroler Arduino Mega 2560, *motor driver* L298. Komponen pada perangkat keras *motor servo base unit* ini saling terhubung menggunakan kabel

jumper dengan mekanisme atau perancangan elektriknya seperti gambar dibawah ini.

Gambar 3-3 Skema rangkaian elektrik *motor servo base unit*.

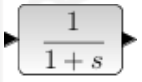
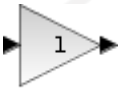
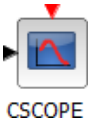
Berdasarkan gambar 3-3, motor DC dan *rotary encoder* beroperasi pada tegangan 24 V yang didapatkan dari catu daya. Arduino mega 2560 mengirimkan perintah kendali posisi ke *driver motor* L298N melalui pin digital 8 dan 9, lalu dari *driver motor* tadi diteruskan ke Motor DC. Arduino tidak langsung terhubung ke motor DC, karena harus menggunakan *driver motor* L298N agar kecepatan dan arahnya dapat diatur. Perputaran *rotary encoder* dengan motor DC yang terhubung dengan *pulley* dan *belt* akan dibaca kembali oleh Scilab melalui pin *digital* 20 dan 21 pada arduino MEGA. Gambar 3-4 menunjukkan hasil rangkaian perangkat keras *motor servo base unit*.

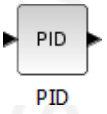
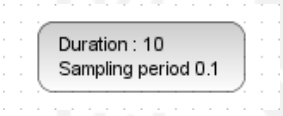
Gambar 3-4 Perangkat keras *motor servo base unit*.

3.2.2 Perancangan Perangkat Lunak

Perancangan perangkat lunak dilakukan untuk mengetahui respon dari sistem yang dibuat oleh skema blok di Xcos Scilab, koneksi kan objek kendali *motor servo base unit* ke perangkat lunak Scilab. Langkah pertama yang dilakukan ialah mengunggah program *loader toolbox* arduino V3 ke dalam perangkat lunak Scilab versi 5.5.2. Setelah program *loader* tersebut berhasil di unggah, selanjutnya dapat membuka Xcos untuk merancang blok diagram sistem kendali. Blok Blok yang digunakan pada penelitian kali ini terdapat pada Tabel 3-1 dibawah ini.

Tabel 3-1 Penjelasan blok Xcos Scilab yang digunakan

Blok diagram	Nama	Fungsi
	STEP_FUNCTION	Blok ini berfungsi untuk memberikan input berupa step. Bisa dalam nilai voltase, derajat dan juga radian.
	CLR	Berfungsi untuk menyatakan nilai fungsi transfer pada Xcos.
	GAINBLK	Sebagai blok pengkali nilai pada Xcos.
	SUMMATION	Berfungsi untuk melakukan perbandingan antara nilai input dan output pada sistem kendali.
	MUX	Berfungsi untuk menyatukan beberapa input menjadi satu output.
	CSCOPE	Berfungsi untuk menampilkan respon sistem simulasi.

	SATURATION	Blok ini berfungsi untuk memberi batasan sinyal input yang masuk ke sistem.
	DERIV	Blok ini berfungsi untuk melakukan proses turunan pada input.
	INTEGRAL_m	Blok ini berfungsi untuk melakukan proses integral pada input.
	CLOCK_c	Berfungsi untuk mengatur periode waktu sistem kendali.
	PID	Berguna untuk memasukkan nilai kendali PID pada sistem.
	TIME_SAMPLE	Berfungsi untuk mengatur waktu dan periode pengambilan sampel pada arduino.
	ARDUINO_SETUP	Melakukan pengaturan pembacaan port arduino yang akan dibaca oleh sistem Xcos.
	ARDUINO_SCOPE	Menunjukkan grafik keluaran dari sistem yang menggunakan arduino
	DCMOTOR_SB	Berfungsi untuk mengatur dan menjalankan motor DC yang terhubung dengan arduino.
	ENCODER_SB	Berfungsi mengatur dan membaca data dari sensor encoder yang terhubung dengan arduino.

Dari blok-blok diagram diatas, objek sistem kendali *motor servo base unit* dapat dilakukan uji coba respon untuk mengetahui performa dari motor DC dan pembacaan sensor enkoder.

BAB 4
HASIL DAN PEMBAHASAN

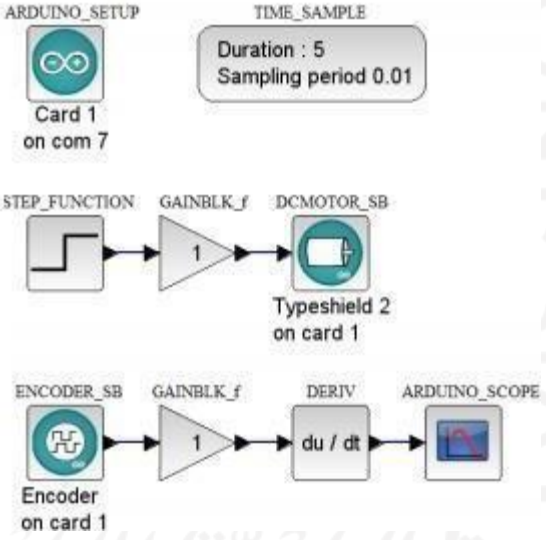
4.1 Hasil Perancangan

Hasil perancangan terdiri dari hasil perancangan perangkat lunak, perhitungan fungsi transfer (eksperimen dan *datasheet*), dan hasil perhitungan nilai PID.

4.1.1 Perancangan Perangkat Lunak

Terdapat 3 skema blok yang digunakan pada perancangan perangkat lunak seperti berikut:

a. Skema blok pengukuran fungsi transfer eksperimen

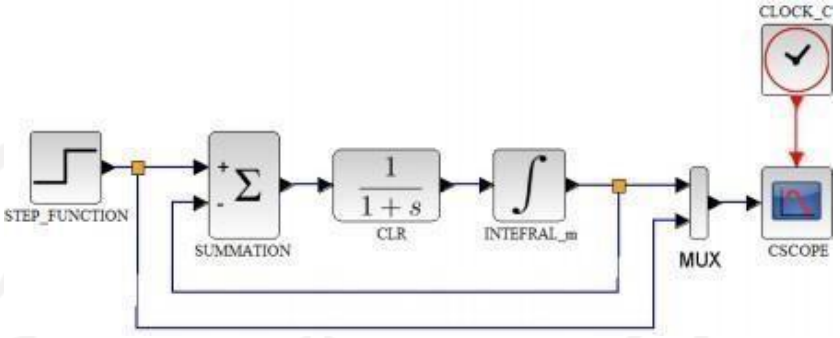


Gambar 4-1 Skema blok pengukuran fungsi transfer eksperimen

Pada gambar 4-1, blok *STEP_FUNCTION* merupakan blok untuk memasukkan nilai berupa tegangan (*volt*) yang selanjutnya akan diproses menuju blok *GAINBLK_f* yang berisi nilai maksimal pwm untuk memutar motor DC pada blok *DCMOTOR_SB*. Blok *ENCODER_SB* membaca putaran poros motor DC melalui *rotary encoder* yang kemudian menuju blok *GAINBLK_f*, blok tersebut berisi perbandingan antara sudut satu putaran dengan jumlah pulsa perangkat keras enkoder(konversi pulsa ke radian). Blok *DERIV* akan menurunkan dari

posisi(radian) menjadi kecepatan sudut (*rad/s*) yang responya akan ditampilkan pada blok *ARDUINO_SCOPE*.

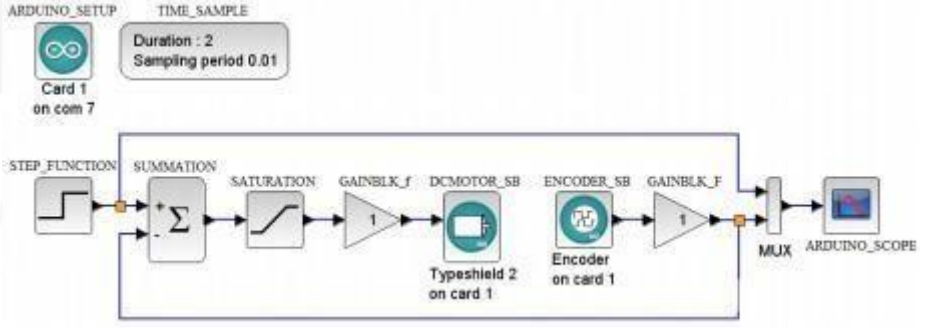
b. Skema blok pengendali posisi sudut



Gambar 4-2 Skema blok pengendali posisi

Pada gambar 4-2, blok *STEP_FUNCTION* merupakan blok untuk memasukkan nilai berupa posisi (*rad*) yang selanjutnya akan diproses menuju blok *CLR*. Hasil proses fungsi transfer tadi akan di integralkan untuk fungsi transfer orde 1, apabila hasil fungsi transfer sudah orde 2 tidak perlu lagi di integralkan karena pengendali posisi itu adalah hasil respon dari orde 2. Proses tersebut mengalami pengulangan (*looping*) karena sistem *closed loop*. Blok *MUX* digunakan untuk membandingkan nilai masukan posisi (*rad*) dengan proses.

c. Skema blok pengendali posisi pada perangkat keras



Gambar 4-3 Skema blok pengendali posisi pada perangkat keras

Pada gambar 4-3, blok *STEP_FUNCTION* merupakan blok untuk memasukan nilai berupa posisi (radian) yang selanjutnya akan diproses menuju blok *SATURATION* untuk membatasi nilai maksimal dan minimal *input*, *GAINBLK_f* yang berisi nilai pwm, dan *DCMOTOR_SB* yang menggerakkan

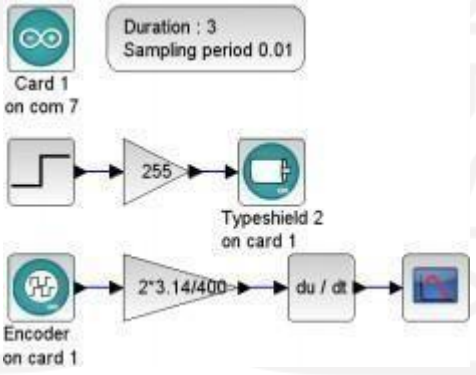
motor DC sesuai nilai posisi (radian) yang dimasukkan. Blok *ENCODER_SB* membaca putaran poros motor DC yang kemudian menuju blok *GAINBLK_f*, blok tersebut berisi perbandingan antara jumlah sudut satu putaran dengan jumlah pulsa enkoder (konversi pulsa ke radian). Proses ini mengalami perulangan (*looping*) karena *closed loop*. Blok *MUX* digunakan untuk membandingkan nilai masukan posisi (radian) dengan respon perangkat keras.

4.1.2 Perhitungan Fungsi Transfer

Fungsi transfer yang digunakan pada penelitian ini ada 2 yaitu fungsi transfer eksperimen dan *datasheet*.

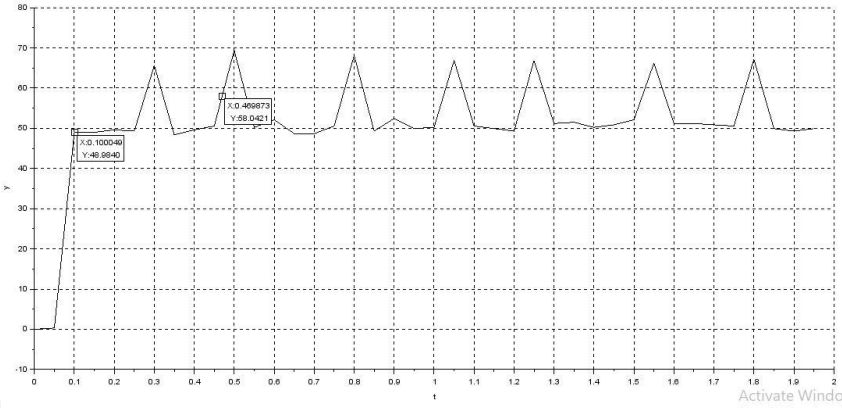
4.1.2.1 Fungsi Transfer Eksperimen

Perhitungan fungsi transfer eksperimen menggunakan persamaan (2.7). Data yang digunakan pada perhitungan fungsi transfer ini didapatkan dari hasil skema blok pengukuran fungsi transfer eksperimen pada gambar 4-1.



Gambar 4-4 Hasil perancangan skema blok pengukuran fungsi transfer eksperimen

Pada gambar 4-6, blok *STEP_FUNCTION* memiliki nilai 24. *Type of shield* pada blok *DCMOTOR_SB* adalah L298 *motor based* dengan pengaturan *direction pin* 8 dan *enable (speed)* pin 9. Pada blok *ENCODER_SB counting mode* dengan pengaturan *pin channel 20* dan *pin direction* 21.



Gambar 4-5 Respon skema blok pengukuran fungsi transfer eksperimen

Berdasarkan gambar 4-5 diperoleh $w(\infty)$ = 58,04 rad/s, $w(t)$ = 49 rad/s, dan t_0 = 0,1 detik.

Nilai k_m diperoleh dari perhitungan persamaan (2.9)

$$\square_{\square} = \frac{\square(\infty)}{\square}$$

$$\square_{\square} = \frac{58.04}{24}$$

$$k_m = 2.42$$

dan nilai $\square_{\square}$ diperoleh dari perhitungan persamaan (2.11)

$$\square_{\square} = \frac{-\square_0}{\ln\left(1 - \frac{\square(\square)}{\square(\infty)}\right)}$$

$$\square_{\square} = \frac{-0.1}{\ln\left(1 - \frac{49}{58.04}\right)}$$

$$\square_{\square} = 0.06 \text{ s}$$

Sehingga fungsi transfernya adalah

$$\frac{\square(\square)}{\square(\square)} = \frac{\square_{\square}}{\square_{\square}\square + 1}$$

$$\frac{\square(\square)}{\square(\square)} = \frac{2.42}{0.06\square + 1}$$

$$\frac{\square(\square)}{\square(\square)} = \frac{40.3}{\square + 16.7}$$

4.1.2.2 Fungsi Transfer *Datasheet*

Perhitungan fungsi transfer *datasheet* menggunakan persamaan (2.12) dengan asumsi bahwa beban yang bertumpu pada motor DC diabaikan, sehingga nilai momen inersia menggunakan nilai dari *datasheet* motor (terlampir). Nilai koefisien gesek (*B*) terlebih dahulu dicari menggunakan persamaan (2.13). Nilai yang digunakan dari datasheet menggunakan nilai dengan voltase 24V.

J_m = Momen Inersia beban = 5,68 gcm^2 = 5,68 × 10^{-7} kgm^2

K_{tm} = Konstanta torsi motor = 30,7 mNm/A = 0,03 *Nm/A*

I_m = Arus motor = 0,42 *A*

ω = Kecepatan motor $= \frac{\text{putaran motor per hour}}{\text{jumlah detik per hour}}$

$= \frac{7190 \text{ rpm}}{504,84}$

= 14,24 *rpm* = 1,49 *rad/s*

Setelah diketahui unsur apa saja yang dibutuhkan dalam *datasheet*, maka nilai koefisien gesek (*B*) dapat dihitung

$B = \frac{K_t \cdot I_m}{\omega}$ $= \frac{0,03 \frac{Nm}{A} \cdot 0,42 \text{ A}}{1,49 \frac{rad}{s}}$ = 0,0084 *Nm/(rad/s)*

Ketika nilai koefisien gesek sudah didapatkan, maka fungsi transfernya adalah

$G(s) = \frac{1}{s(J_m s + B)}$ $= \frac{1}{s(5,68 \times 10^{-7} s + 0,0084)}$

$= \frac{1}{(0,00000057 s^2 + 0,0084 s)}$

$= \frac{1754386}{s^2 + 14737 s}$

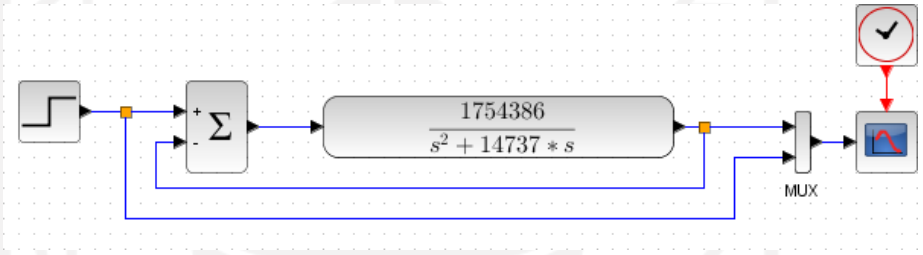
Fungsi transfer eksperimen sudah didapatkan, tetapi fungsi transfer tersebut hanya bisa digunakan untuk kendali posisi sudut karena masih orde 2. Untuk kendali kecepatan sudut maka fungsi transfer orde 2 tersebut diturunkan menjadi orde 1

$$G(s) = \frac{1754386}{s + 14737}$$

4.2 Hasil Pengujian

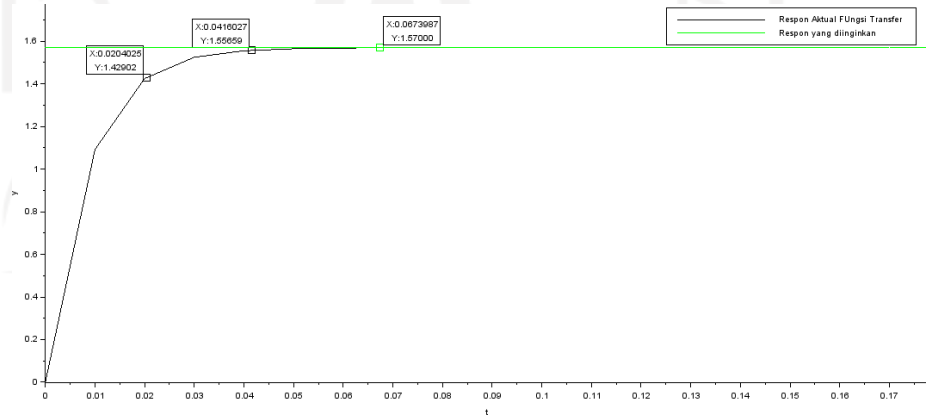
Setelah sebelumnya sudah melakukan perhitungan untuk mencari fungsi transfer eksperimen dan fungsi transfer *datasheet*, selanjutnya fungsi transfer tersebut kita masukan kedalam program Xcos untuk dilihat kinerja nya pada perangkat lunak. Terdapat 2 hasil perancangan perangkat lunak dan respon skema blok serta 3 hasil perancangan dan pengujian skema blok pengendali posisi sudut pada perangkat keras seperti berikut :

4.2.1 Perancangan perangkat lunak dan respon kendali posisi *datasheet*



Gambar 4-6 Hasil perancangan skema blok kendali posisi *datasheet*

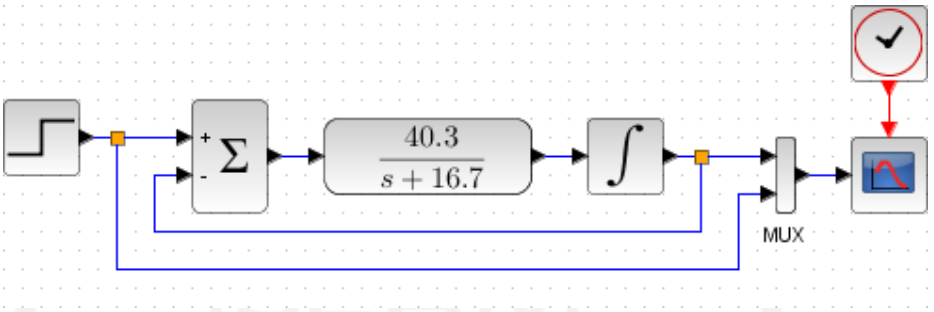
Pada gambar 4-6, blok *STEP_FUNCTION* memiliki nilai 1,57, blok *CLR* berisi fungsi transfer eksperimen. Blok *CSCOPE* diatur dengan *Y max* 2 dan *Y min* 0 dengan *refresh period* 1. Blok *CLOCK_c* diatur dengan *period* 0.01, simulasi dilakukan selama 0,2 detik.



Gambar 4-7 Respon Skema blok kendali posisi *datasheet*

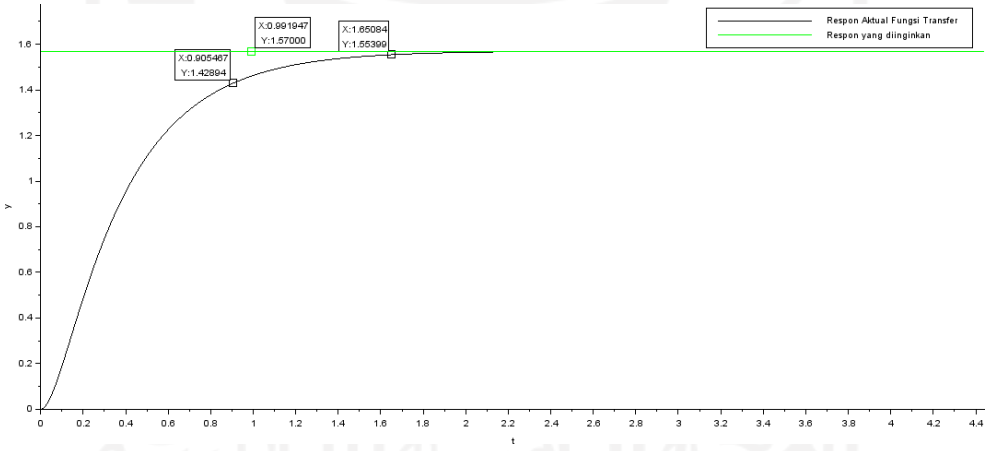
Berdasarkan 4-7, diperoleh data T_r = 0.02 detik; T_s = 0.04 detik; dan SSE = 0%

4.2.2 Perancangan perangkat lunak dan respon kendali posisi eksperimen



Gambar 4-8 hasil perancangan skema blok kendali posisi eksperimen

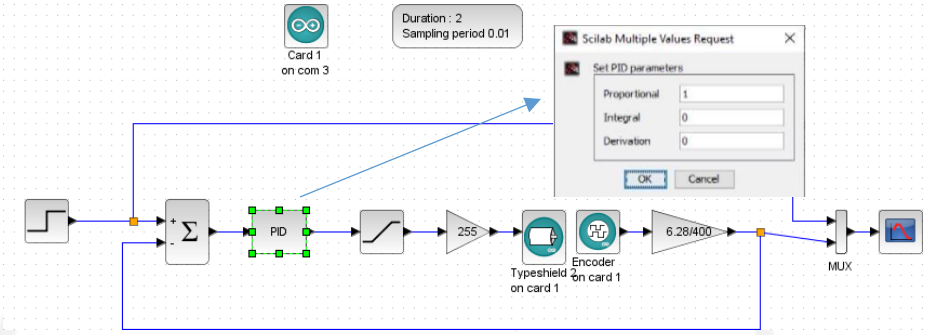
Pada gambar 4-8, blok *STEP_FUNCTION* memiliki nilai 1.57, blok *CLR* berisi fungsi transfer eksperimen, blok *INTEGRAL* berfungsi untuk menaikkan fungsi transfer dari orde 1 menjadi orde 2, karena untuk kendali posisi sudut membutuhkan fungsi transfer orde 2. Blok *CSCOPE* diatur dengan *Y max* 2 dan *Y min* 0 dengan *refresh period* 1. Blok *CLOCK_c* diatur dengan *period* 0.1, simulasi dilakukan selama 5 detik.



Gambar 4-9 Respon skema blok kendali posisi eksperimen

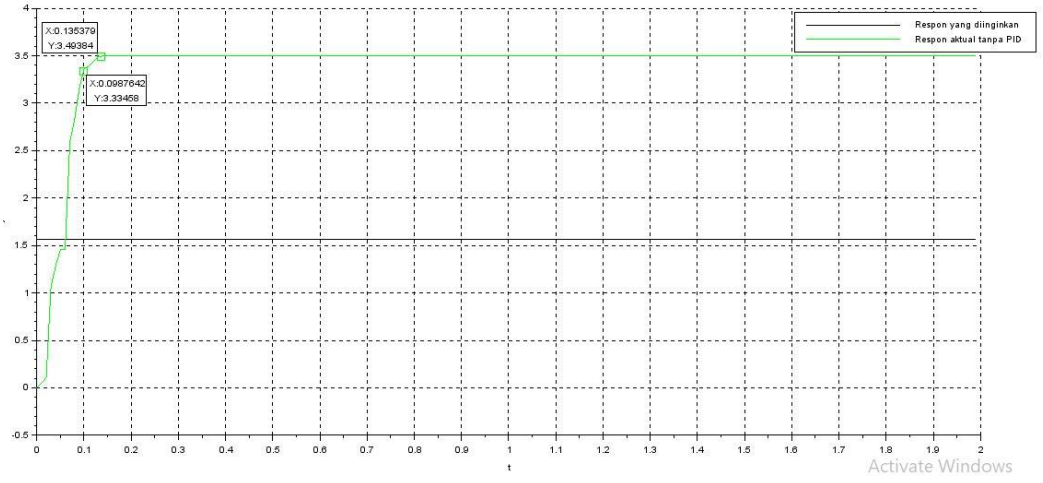
Berdasarkan 4-9, diperoleh data T_r = 0.9 detik; T_s = 1.65 detik; dan SSE = 0%

4.2.3 Implementasi kendali posisi pada perangkat keras tanpa PID



Gambar 4-10 Hasil skema blok pengendali posisi pada perangkat keras tanpa PID

Pada gambar 4-10, blok *STEP_FUNCTION* memiliki nilai 1.57, blok *SATURATION* memiliki *upper limit* 24. Blok *PID* memiliki nilai konstanta integral dan derivatif adalah 0 dan nilai konstanta *proposional* adalah 1, cara ini digunakan agar nilai PID tidak terbaca oleh sistem. *Type of shield* pada blok *DCMOTOR_SB* adalah PMODHB5 or L298 *based* dengan pengaturan *direction pin* 8 dan *enable (speed) pin* 9. Pada blok *ENCODER_SB counting mode* adalah *up chantA* (1) dengan pengaturan *pin channel* A 20 dan *pin direction* 21. Blok *GAINBLK_f* berisi 6.28/400. Simulasi dilakukan selama 2 detik.



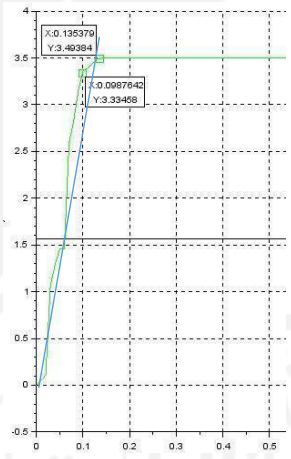
Gambar 4-11 Respon skema blok pengendali posisi tanpa PID

Respon diatas menunjukan pergerakan motor DC tanpa kendali PID. *Input* yang diberikan adalah 1,57 *rad*. Garis hitam pada gambar menunjukan nilai *input*, dan garis hijau menunjukan respon aktual motor DC.

Dari gambar 4-11 diperoleh data T_r = 0,08 detik, T_s = 0,135 detik, dan nilai *steady state* = 3.49 rad, dimana respon tersebut menunjukan nilai *Steady State Error* (SSE) = 110-115%. Berdasarkan data tersebut diketahui bahwa nilai posisi *steady state* aktual motor DC sangat jauh berbeda dari respon yang diinginkan. Oleh karena itu kita membutuhkan kontrol PID agar respon aktual motor DC sesuai dengan yang diinginkan.

4.2.4 Implementasi kendali posisi pada perangkat keras dengan PID metode ZNFD tipe 1

Pengujian selanjutnya adalah mencari nilai konstanta PID menggunakan metode ZNFD tipe 1. Karena respon motor DC menunjukan fenomena kurva S, sehingga memenuhi syarat untuk menggunakan metode ini untuk mencari nilai konstanta PID.



Gambar 4-12 Mencari nilai konstanta PID

Berdasarkan gambar 4-12 adalah hasil respon skema blok pengedali posisi tanpa kendali PID yang sudah diberi garis untuk membantu penentuan nilai konstanta PID nya. Dari gambar tersebut, dapat menentukan nilai L = 0,02, T = 0,08, dan K = 3,14. Untuk nilai PID dapat dihitung sesuai dengan tabel 2-3.

□ □ = 1,2 □ / □

$$□□ = 1{,}2\frac{0{,}02}{0{,}08}$$

$$□□ = 1{,}2(0{,}25)$$

$$□□ = 0{,}3$$

$$□□ = 2□$$

$$□□ = 2(0{,}02)$$

$$□□ = 0{,}04$$

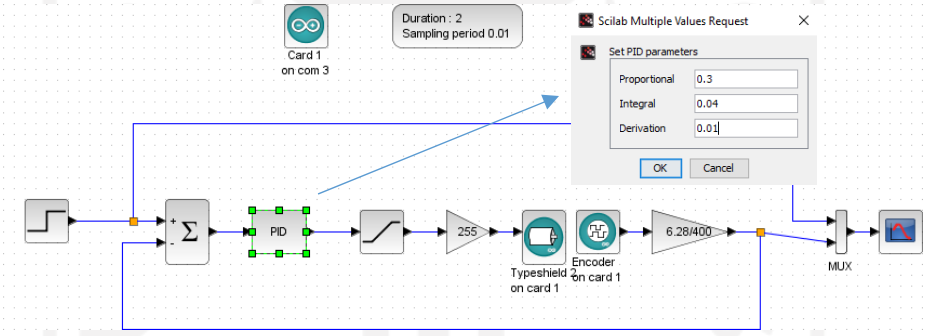
$$□□ = 0{,}5□$$

$$□□ = 0{,}5(0{,}02)$$

$$□□ = 0{,}01$$

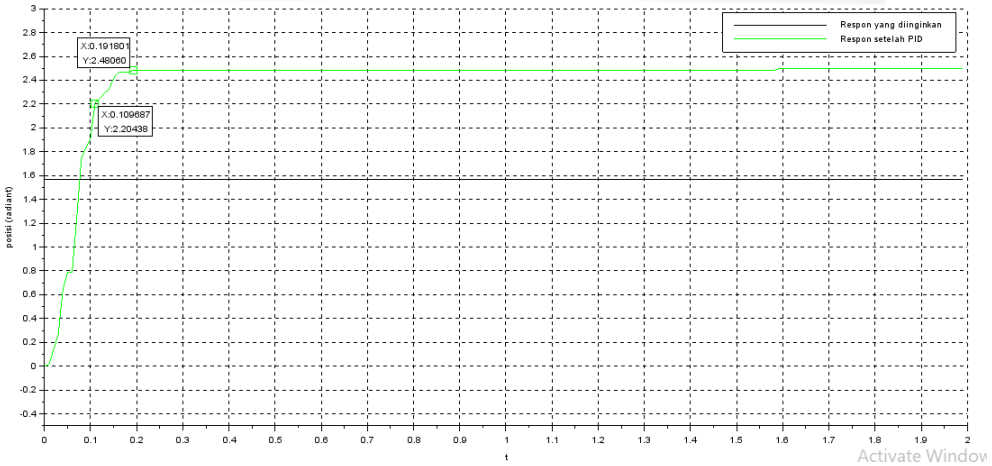
Hasil perhitungan nilai PID menggunakan metode ZNFD tipe 1 menunjukan bahwa nilai KP = 0.3; KI = 0.04 dan KD = 0.01.

Masukan nilai tersebut kedalam blok Xcos untuk implementasi nilai PID.



Gambar 4-13 Hasil skema blok pada perangkat keras menggunakan PID

Pada blok PID, masukan nilai *Propotional* = 0.3; nilai *Integral* = 0.04 dan nilai *Derivation* = 0.1. Dan berikut respon hasil grafik dari blok Xcos diatas.



Gambar 4-14 Respon skema blok pengendali posisi setelah PID

Berdasarkan grafik respon diatas, diketahui bahwa nilai *steady state* berkurang menjadi 2.48 rad menggunakan PID metode ZNFD tipe 1. Nilai *steady state error* (SSE) = 40% , sedangkan nilai T_r = 0,1 detik dan T_s = 0,2 detik.

4.2.5 Implementasi kendali posisi pada perangkat keras dengan PID(PD) metode Cohen-Coon tipe 2

Pengujian ketiga adalah mecoba mencari nilai konstanta PID menggunakan metode Cohen-Coon Tipe 2. Karena respon motor DC menunjukan fenomena kurva S, sehingga memenuhi syarat untuk penggunaan metode Cohen-Coon tipe 2 untuk mencari nilai konstanta PID, sama seperti metode ZNFD tipe 1.

Dari gambar 4-12, dapat menentukan nilai L = 0,02, T = 0,08, dan K = 3,14. Sebelumnya, untuk metode Cohen-Coon tipe 2 ini sudah dilakukan pencarian nilai PID nya itu sendiri dan ternyata hasil respon implementasinya sama saja dengan tanpa kendali PID atau respon masih sangat jauh hasilnya dari respon yang diinginkan, oleh karena itu untuk metode Cohen-Coon tipe 2 yang digunakan adalah nilai PD nya karena nilai PD yang hasil responnya sesuai dengan respon yang diinginkan. Untuk nilai PD dapat dihitung sesuai dengan tabel 2-4.

$$K_p = \frac{1}{K}\left(\frac{T}{L}\right)\left[\frac{5}{4} + \frac{1}{6}\left(\frac{T}{L}\right)\right]$$

$$K_p = \frac{1}{3,14}\left(\frac{0,08}{0,02}\right)\left[\frac{5}{4} + \frac{1}{6}\left(\frac{0,08}{0,02}\right)\right]$$

$$K_p = 0,3(0,25)[1,25 + 0,16(0,25)]$$

$$K_p = 0,3(0,25)[1,25 + 0,16(0,25)]$$

$$K_p = 0,075[1,25 + 0,04]$$

$$K_p = 0,075[1,29]$$

$$K_p = 0,1$$

$$K_d = K\left[\frac{6 - 2\left(\frac{T}{L}\right)}{22 + 3\left(\frac{T}{L}\right)}\right]$$

$$\square\square = 0{,}02\left[\frac{6 - 2\left(\frac{0{,}02}{0{,}08}\right)}{22 + 3\left(\frac{0{,}02}{0{,}08}\right)}\right]$$

$$\square\square = 0{,}02\left[\frac{6 - 2(0{,}25)}{22 + 3(0{,}25)}\right]$$

$$\square\square = 0{,}02\left[\frac{6 - 0{,}5}{22 + 0{,}75}\right]$$

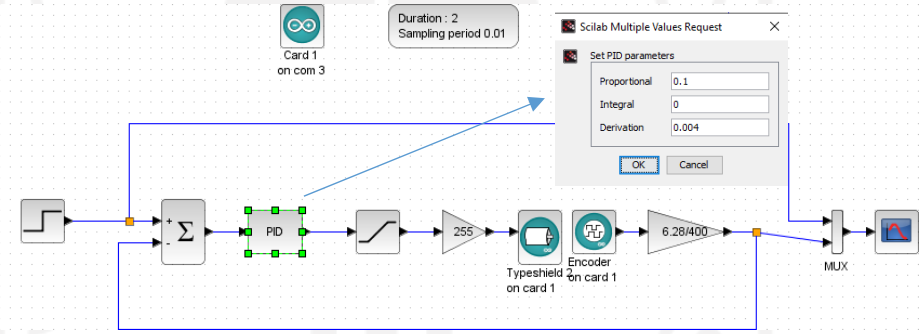
$$\square\square = 0{,}02\left[\frac{5{,}5}{22{,}75}\right]$$

$$\square\square = 0{,}02[0{,}2]$$

$$\square\square = 0{,}004$$

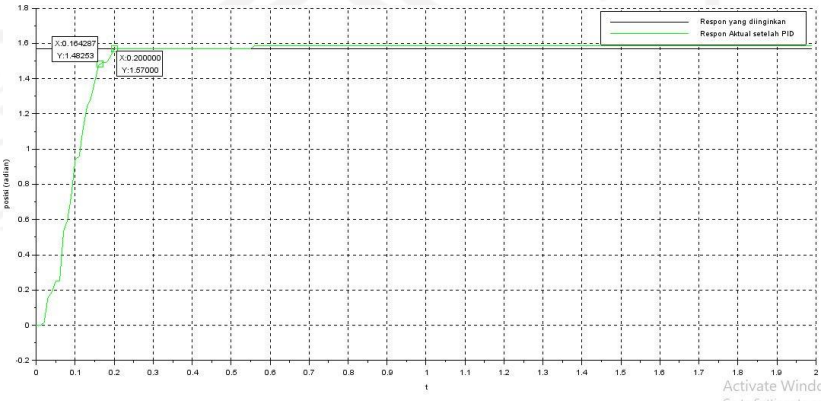
Hasil perhitungan nilai PD menggunakan metode Cohen-Coon tipe 2 menunjukan bahwa nilai KP = 0.1 dan nilai KD = 0.004.

Masukan nilai tersebut kedalam blok Xcos untuk implementasi nilai PID(PD)



Gambar 4-15 hasil skema blok pada perangkat keras menggunakan PD

Pada blok PID, masukan nilai *Propotional* = 0.1; nilai *Integral* = 0 dan nilai *Derivation* = 0.004. Dan berikut respon hasil grafik dari blok Xcos diatas.



Gambar 4-16 Respon skema blok pengendali posisi setelah PD

Berdasarkan grafik respon diatas, diketahui bahwa nilai *steady state* sangat jauh berkurang menjadi sama dengan respon yang diinginkan yaitu 1.57 rad

menggunakan PD metode Cohen-Coon tipe 2. Nilai *steady state error* (SSE) = 0-5% saja, sedangkan nilai T_r = 0,16 detik dan T_s = 0,2 detik.

4.3 Analisis dan Pembahasan

Bedasarkan pengujian, dilakukan tiga uji coba untuk mengendalikan alat agar posisi sudut sesuai dengan yang diinginkan. Ketiga uji coba tersebut adalah, pengujian tanpa kendali PID, pengujian kendali PID dengan metode ZNFD tipe 1 dan pengujian kendali PID(PD) dengan metode Cohen-Coon tipe 2. Selanjutnya, ketiga hasil tersebut juga dibandingkan dengan respon berdasarkan *datasheet* dan respon berdasarkan eksperimen.

Berikut ini adalah tabel data respon skema blok hasil perancangan perangkat lunak pengendali posisi sudut.

Tabel 4-1 Perbandingan respon pengendali posisi sudut

Parameter	Respon fungsi transfer berdasarkan *datasheet*	Respon fungsi transfer berdasarkan eksperimen
□□ (detik)	0.02s	0.9s
T_s (detik)	0.04s	1.65s
SSE (persen)	1.57(0%)	1.57(0%)
Parameter	Respon berdasarkan implementasi tanpa PID	Respon berdasarkan implementasi dengan PID metode ZNFD tipe 1
□r (detik)	0.08s	0.1s
T_s (detik)	0.135s	0.2s
SSE (persen)	3.5(115%)	2.48(40%)
Parameter	Respon berdasarkan Implementasi PD metode Cohen-Coon tipe 2	-
□r (detik)	0.16s	-
T_s (detik)	0.2s	-
SSE (persen)	1.57(0-5%)	-

Berdasarkan Tabel 4-1 nilai T_r (*rise time*) terkecil adalah respon fungsi transfer berdasarkan *datasheet*, nilai T_s (*settling time*) terkecil adalah respon fungsi transfer berdasarkan *datasheet* juga, dan SSE (*steady state error*) terkecil adalah respon fungsi transfer berdasarkan *datasheet* dan eksperimen. Respon sistem yang

baik adalah tidak memiliki selisih atau nilai selisihnya hanya kecil dari respon aktual dengan respon yang diinginkan. Dari tabel 4-1 dapat disimpulkan bahwa respon sistemnya sudah baik karena memiliki nilai selisih yang kecil.

Pada pengujian fungsi transfer berdasarkah *datasheet* dan eksperimen menunjukan hasil atau respon yang berbeda. Perbedaan respon skema blok fungsi transfer berdasarkan *datasheet* dan eksperimen disebabkan karena beberapa kemungkinan seperti pada perhitungan keduanya ada parameter yang belum teridentifikasi. Alasan lainya adalah spesifikasi motor DC yang ada pada *datasheet* yang diberikan oleh perusahaan tidak sesuai dengan kinerja motor DC. Kinerja motor DC juga berbeda setiap merknya.

Respon pada implementasi ketiga percobaan yaitu tanpa PID, PID metode ZNFD tipe 1 dan dengan metode PID(PD) Cohen-Coon tipe 2 tidak memiliki selisih respon nilai *rise time* dan *settling time* yang besar, tetapi selsisih nilai *steady state error*nya cukup besar. Hal tersebut berarti bahwa skema blok pengendali posisi pada perangkat keras memang harus menggunakan kontroler. Penggunaan kontroler PID(PD) metode Cohen-Coon tipe 2 adalah kontroler yang paling tepat karena nilai respon aktual motor DC sesuai dengan nilai yang diinginkan.

Hasil respon aktual motor DC menggunakan metode PD Cohen-Coon tipe 2 lebih baik dibandingkan menggunakan metode ZNFD tipe 1 dikarenakan metode ZNFD cakupanya masih luas, bisa juga untuk sistem kendali *open loop*. Sedangkan metode Cohen-Coon hanya dikhususkan untuk sistem kendali *closed loop* saja, dimana pada penelitian ini sistem kendali yang digunakan adalah *closed loop*. Selain itu, perumusan pada metode Cohen-Coon jauh lebih kompleks dibandingkan metode ZNFD. Hal ini yang membuat pencarian nilai PID menggunakan metode Cohen-Coon lebih valid.

BAB 5
PENUTUP

5.1 Kesimpulan

Kesimpulan dari penilitian ini adalah:

1. Sistem kendali pada objek kendali *motor servo base* unit merupakan sistem kendali *closed loop* dan kendali PID berhasil dirancang sesuai karakteristik tersebut. Pada awalnya sistem kendali masih tidak sesuai dengan apa yang diinginkan, karena sistem kendali *closed loop* dan dibutuhkanya kontroler untuk mengatur respon yang sesuai.
2. Simulasi objek kendali dan kontroler PID telah dilakukan dengan merancang blok-blok fungsi pada Xcos. Implementasi dilakukan dengan cara melakukan simulasi sehingga diketahui respon dari objek kendali *motor servo base unit*.
3. Berdasarkan hasil implementasi sistem kendali pada alat, disimpulkan bahwa alat memang membutuhkan pengendalian untuk mencapai respon yang diinginkan. Metode kendali ZNFD tipe 1 masih kurang tepat untuk digunakan dalam sistem kendali posisi sudut, meskipun nilai *steady state* sudah lebih baik dan nilai SSE sudah berkurang tetapi respon masih belum sesuai dengan apa yang diinginkan. Penggunaan metode Cohen-Coon tipe 2 adalah metode kontroler PID yang paling tepat untuk sistem kendali posisi sudut karena respon sudah sesuai dengan yang diinginkan. Hal ini dikarenakan setiap model kendali memiliki karakteristik, spesifikasi dan perumusan yang berbeda. Karakteristik dan perumusan pencarian nilai konstanta PID pada metode Cohen-Coon tipe 2 adalah yang paling sesuai untuk sistem kendali posisi sudut objek kendali *motor servo base unit*.

5.2 Saran atau Penelitian Selanjutnya

Saran untuk penilitian selanjutnya ialah:

1. Penelitian selanjutnya dapat menggunakan dapat menggunakan perangkat lunak Matlab jika memungkinkan, hal ini dikarenakan perangkat lunak Matlab lebih sering digunakan dibidang industri secara umum. Pencarian konstanta PID menggunakan Matlab juga jauh lebih mudah karena pada perangkat lunak tersebut terdapat fitur *auto tuning* untuk kendali PID.
2. Penelitian selanjutnya dapat melakukan perbaikan terkait komponen pada alat prototipe yang berantakan sehingga dapat menyebabkan *part* atau perangkat terkadang tidak berfungsi atau tidak terbaca oleh sistem
3. Penelitian selanjutnya dapat dilakukan pengembangan terkait kendali PID untuk kecepatan sudut.
4. Penelitian selanjutnya juga dapat dikembangkan lagi terkait metode kendalinya, dapat berupa Fuzzy, Direct Synthesis atau metode yang lain.

DAFTAR PUSTAKA

Ahmed, & Alnaib, I. (2019). DC Motors.

Handayani, D. N., Pramudya, Y., Suparwoto, & Muchlas. (2018). The Application of Scilab Software in Frequency Mode Simulation on the Circular Membrane. *Journal of Physics: Theories and Applications*, 83-94.

Kurniawan, E., Simbolon, R. S., & MN, N. (2020). Analysis and Simulation of PI and PID Control System Using Xcos Scilab. *Journal of Technomaterials*, 108-116.

Maung, M., Latt, M., & Myat, C. (2018). DC Motor Angular Position Control using PID Controller with Friction Compesation.

Nise, N. S. (2015). *Control System Engineering.* John Willey & Sons, Inc.

Ogata, K. (2010). *Modern Control Engineering Fifth Edition.*

Prasetyo, H., & Tarmukan. (2017). Implementasi Pengendali Posisi Motor DC berbasis PID dengan Interface Mikrokontroler dan Matlab pada Laboratorium Sistem Kendali Digital.

Pratama, V. I. (2020). *Permodelan dan Pembuatan Prototipe Sistem Kendali Posisi dan Kecepatan Sudut.* Yogyakarta: Universitas Islam Indonesia.

Rokhmah, N. N. (2018). Kendali Kecepatan Motor DC dengan Metode PID berbasis Arduino Uno.

Technology, H. (t.thn.). *L298N Dual H-Bridge Motor Driver*.

Toochinda, V. (2016). *Feedback Control with Scilab and Arduino.* Bangkok: Varodom Toochinda, Ph,D.

Waluyo, F., & Aditya, S. (2013). Analisis Penalaan Kontrol PID pada Simulasi Kendali Kecepatan Putaran Motor DC berbeban menggunakan Metode Heuristik.

Waluyo, Fitriansyah, A., & Syahrizal. (2013). Analisis Penalaan Kontrol PID pada Simulasi Kendali Kecepatan Putaran Motor DC Berbeban menggunakan Metode Heuristik.

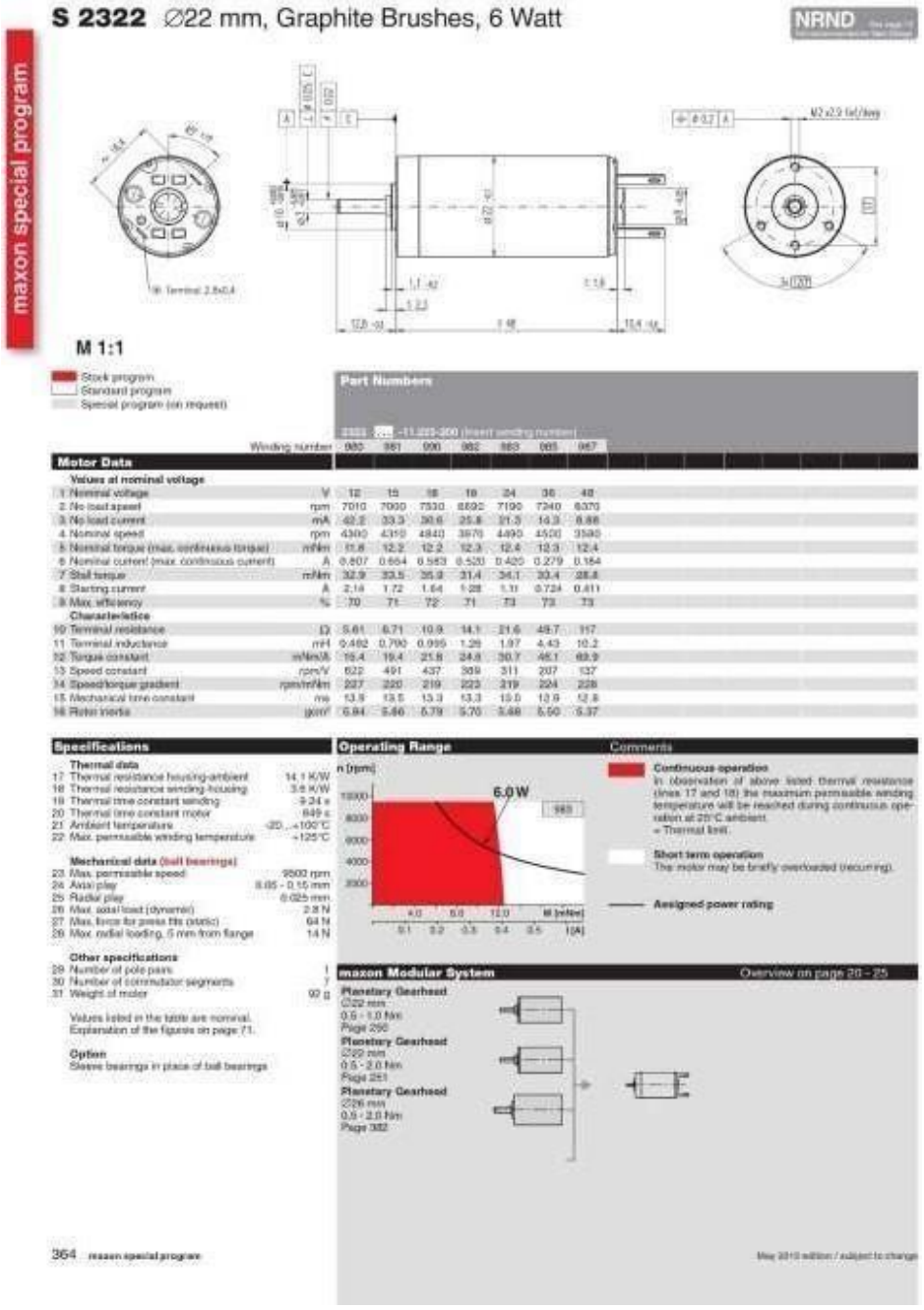
Wicaksono, H. (2013). Analisa Perfomansi dan Robutness beberapa Metode Tuning Kontroler PID pada Motor DC.

Zheng, D., Zhang, S., Wang, S., Hu, C., & Zhao, X. (2015). *A Capacitive Rotary Encoder Based on Quadrature Modulation and Demodulation*.

LAMPIRAN

Lampiran 1

Datasheet Motor DC Maxon S2322



S 2322 ∅22 mm, Graphite Brushes, 6 Watt

NRND

maxon special program

M 1:1

Stock program
Standard program
Special program (on request)

Part Numbers

Winding number: 983 | 981 | 990 | 982 | 983 | 985 | 987

Motor Data

	Values at nominal voltage							
1	Nominal voltage	V	12	15	18	19	24	36	48
2	No load speed	rpm	7010	7000	7530	6892	7190	7240	6370
3	No load current	mA	42.2	33.3	30.6	25.8	21.5	14.3	8.88
4	Nominal speed	rpm	4300	4310	4840	3970	4490	4520	3580
5	Nominal torque (max. continuous torque)	mNm	11.8	12.2	12.2	12.3	12.4	12.3	12.4
6	Nominal current (max. continuous current)	A	0.807	0.664	0.583	0.520	0.420	0.279	0.184
7	Stall torque	mNm	32.9	33.5	35.9	31.4	34.1	30.4	28.8
8	Starting current	A	2.14	1.72	1.64	1.28	1.11	0.724	0.411
9	Max. efficiency	%	70	71	72	71	73	73	73
	Characteristics								
10	Terminal resistance	Ω	5.61	6.71	10.9	14.1	21.6	49.7	117
11	Terminal inductance	mH	0.482	0.700	0.995	1.26	1.87	4.43	10.2
12	Torque constant	mNm/A	15.4	19.4	21.8	24.6	30.7	42.1	69.9
13	Speed constant	rpm/V	622	491	437	389	311	207	137
14	Speed/torque gradient	rpm/mNm	227	220	219	223	219	224	228
15	Mechanical time constant	ms	13.8	13.5	13.3	13.3	13.0	12.9	12.8
16	Rotor inertia	gcm²	5.84	5.86	5.79	5.70	5.68	5.50	5.37

Specifications

	Thermal data	
17	Thermal resistance housing-ambient	14.1 K/W
18	Thermal resistance winding-housing	3.6 K/W
19	Thermal time constant winding	9.24 s
20	Thermal time constant motor	649 s
21	Ambient temperature	-20...+100°C
22	Max. permissible winding temperature	+125°C
	Mechanical data (ball bearings)	
23	Max. permissible speed	9500 rpm
24	Axial play	0.05 - 0.15 mm
25	Radial play	0.025 mm
26	Max. axial load (dynamic)	2.8 N
27	Max. force for press fits (static)	64 N
28	Max. radial loading, 5 mm from flange	14 N
	Other specifications	
29	Number of pole pairs	1
30	Number of commutator segments	7
31	Weight of motor	92 g

Values listed in the table are nominal.
Explanation of the figures on page 71.

Option
Sleeve bearings in place of ball bearings

Operating Range

Comments

Continuous operation
In observation of above listed thermal resistance (lines 17 and 18) the maximum permissible winding temperature will be reached during continuous operation at 25°C ambient.
= Thermal limit.

Short term operation
The motor may be briefly overloaded (recurring).

Assigned power rating

maxon Modular System Overview on page 20 – 25

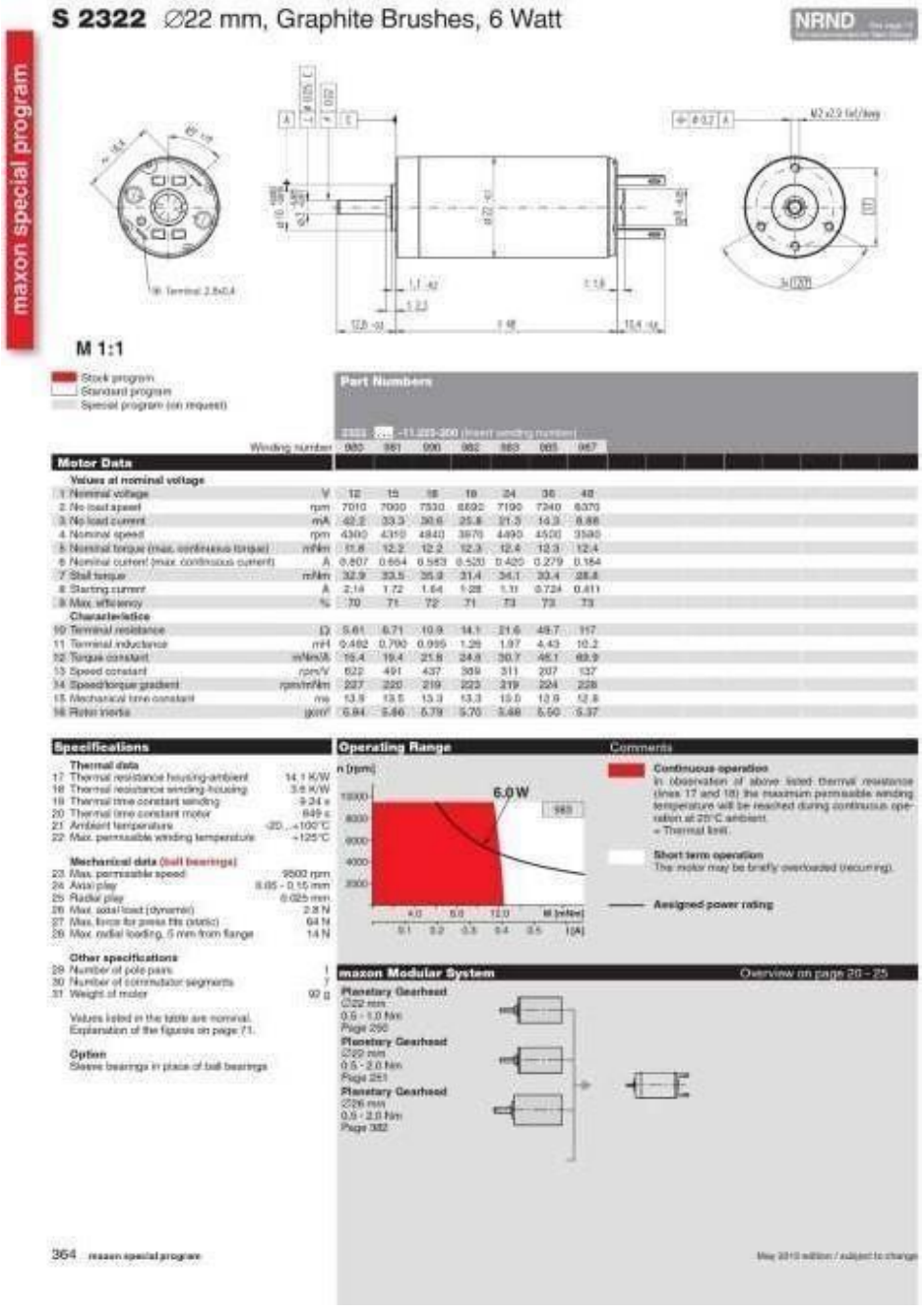
Planetary Gearhead
Ø22 mm
0.5 - 1.0 Nm
Page 250

Planetary Gearhead
Ø22 mm
0.5 - 2.0 Nm
Page 251

Planetary Gearhead
Ø26 mm
0.5 - 2.0 Nm
Page 382

Lampiran 2

Rotary Encoder E40-S



S 2322 ∅22 mm, Graphite Brushes, 6 Watt

NRND

maxon special program

M 1:1

Stock program
Standard program
Special program (on request)

Part Numbers

Winding number: 980 981 990 982 983 985 987

Motor Data

Values at nominal voltage								
1 Nominal voltage	V	12	15	18	19	24	36	48
2 No load speed	rpm	7010	7000	7530	6892	7190	7240	6370
3 No load current	mA	42.2	33.3	30.6	25.8	21.5	14.3	8.88
4 Nominal speed	rpm	4300	4310	4840	3970	4490	4530	3580
5 Nominal torque (max. continuous torque)	mNm	11.8	12.2	12.2	12.3	12.4	12.3	12.4
6 Nominal current (max. continuous current)	A	0.807	0.664	0.583	0.520	0.420	0.279	0.184
7 Stall torque	mNm	32.9	33.5	35.2	31.4	34.1	33.4	28.8
8 Starting current	A	2.14	1.72	1.64	1.28	1.11	0.724	0.411
9 Max. efficiency	%	70	71	72	71	73	73	73
Characteristics								
10 Terminal resistance	Ω	5.61	8.71	10.9	14.1	21.6	49.7	117
11 Terminal inductance	mH	0.492	0.700	0.995	1.26	1.97	4.43	10.2
12 Torque constant	mNm/A	15.4	19.4	21.8	24.6	30.7	46.1	69.9
13 Speed constant	rpm/V	622	491	437	389	311	207	137
14 Speed/torque gradient	rpm/mNm	227	220	219	223	219	224	228
15 Mechanical time constant	ms	13.9	13.5	13.3	13.3	13.0	13.9	12.8
16 Rotor inertia	gcm²	6.84	5.86	5.79	5.70	5.48	5.50	5.37

Specifications

Thermal data		
17 Thermal resistance housing-ambient	14.1 K/W	
18 Thermal resistance winding-housing	3.6 K/W	
19 Thermal time constant winding	9.24 s	
20 Thermal time constant motor	849 s	
21 Ambient temperature	-20...+100°C	
22 Max. permissible winding temperature	+125°C	
Mechanical data (ball bearings)		
23 Max. permissible speed	9500 rpm	
24 Axial play	0.05 - 0.15 mm	
25 Radial play	0.025 mm	
26 Max. axial load (dynamic)	2.8 N	
27 Max. force for press fits (static)	64 N	
28 Max. radial loading, 5 mm from flange	14 N	
Other specifications		
29 Number of pole pairs	1	
30 Number of commutator segments	7	
31 Weight of motor	92 g	

Values listed in the table are nominal.
Explanation of the figures on page 71.

Option
Sleeve bearings in place of ball bearings

Operating Range

Comments

Continuous operation
In observation of above listed thermal resistance (lines 17 and 18) the maximum permissible winding temperature will be reached during continuous operation at 25°C ambient.
= Thermal limit.

Short term operation
The motor may be briefly overloaded (recurring).

Assigned power rating

maxon Modular System Overview on page 20 - 25

Planetary Gearhead
Ø22 mm
0.5 - 1.0 Nm
Page 250

Planetary Gearhead
Ø22 mm
0.5 - 2.0 Nm
Page 251

Planetary Gearhead
Ø26 mm
0.5 - 2.0 Nm
Page 382

May 2010 edition / subject to change

Lampiran 3

Loader Scilab