TUGAS AKHIR – RE1599

SISTEM PENGEREMAN OTOMATIS MENGGUNAKAN KONTROL LOGIKA FUZZY

Sony Nikodemus Limbong
NRP 2212100185

Dosen Pembimbing
Dr. Muhammad Rivai, ST, MT.
Ir. Tasripan, MT.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2017

TUGAS AKHIR - TE141599

Sistem Pengereman Otomatis Menggunakan Kendali Logika Fuzzy

Sony Nikodemus Limbong
NRP 2212100185

Dosen Pembimbing
Dr. Muhammad Rivai, ST., MT.
Ir. Tasripan, MT.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2017

Halaman ini sengaja dikosongkan

Final Project - TE141599

Automatic Braking System Using Fuzzy Logic Control

Sony Nikodemus Limbong
NRP 2212100185

Supervisor
Dr. Muhammad Rivai, ST., MT.
Ir. Tasripan, MT.

ELECTRICAL ENGINEERING DEPARTMENT
FACULTY OF ELECTRICAL ENGINEERING
Institut Teknologi Sepuluh Nopember
Surabaya 2017

Halaman ini sengaja dikosongkan

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "Sistem Pengenreman Otomatis Menggunakan Kontrol Logika Fuzzy" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 14 Juli 2017

Sony Nikodemus Limbong
NRP. 2212 100 185

Halaman ini sengaja dikosongkan

SISTEM PENGEREMAN OTOMATIS MENGGUNAKAN KONTROL LOGIKA FUZZY

TUGAS AKHIR

Diajukan Guna Memenuhi Sabagian Persyaratan Untuk
Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Elektronika
Deartemen Teknik Elektro
Institut Teknologi Sepuluh Nopemeber

Menyetujui

Dosen Pembimbing I,

Dr. Muhammad Rivai, ST., MT.
Nip: 19690426199 031003

Dosen Pembimbing II,

Ir. Tasripan, MT.
Nip: 196204181990031004

SURABAYA
JULI 2017

Halaman ini sengaja dikosongkan

Sistem Pengereman Otomatis Menggunakan Kendali Logika Fuzzy

Nama : Sony Nikodemus Limbong
Pembimbing I : Dr. Muhammad Rivai, ST., MT.
Pembimbing II : Ir. Tasripan, MT.

ABSTRAK

Angka kecelakaan lalu lintas di kota besar yang ada di Indonesia saat ini termasuk tinggi dan setiap tahunnya meningkat secara signifikan. Rata-rata penyebab kecelakaan dikarenakan oleh faktor kelalaian dari pengguna kedaraan bermotor. Sistem pengereman otomatis merupakan salah satu cara yang dikembangkan oleh industri otomotif agar angka kecelakaan dapat diminimalisir. Pada penelitian ini telah dirancang dan dibuat sistem pengereman otomatis menggunakan kontrol logika fuzzy. Pada sistem ini digunakan motor DC yang sudah di lengkapi dengan rotary encoder yang berfungsi untuk membaca kecepatan dari prototype mobil ini. Pada alat ini juga dilengkapi sensor ultrasonik yang berfungsi untuk membaca jarak antara alat dengan benda yang ada di depan alat. Pada sistem ini digunakan mikrokontroller arduino yang berfungsi sebagai pusat sistem kontrol dan pembaca encoder. Pada penelitian ini dilakukan pengujian pengereman prototipe mobil dengan jarak pengereman yang bervariasi. Pada saat pwm 49 %, prototipe mobil dapat berhenti pada jarak sekitar 35,9 cm. Pada saat pwm 43%, prototipe mobil dapat berhenti pada saat jarak sekitar 45,5 cm. Sedangkan pada saat pwmnya 39, prototipe mobil dapat berhenti pada jarak sekitar 47 cm. Metode ini diharapkan dapat digunakan sistem pengereman pada mobil listrik sehingga mengurangi resiko kecelakaan.

Halaman ini sengaja dikosongkan

Automatic Braking System Using Fuzzy Logic Control

Nama : Sony Nikodemus Limbong
Pembimbing I : Dr. Muhammad Rivai, ST., MT.
Pembimbing II : Ir. Tasripan, MT.

ABSTRACT

The number of traffic accidents in major cities in Indonesia now is very high and increases significantly every years. The accidents are usually caused by human errors. Automatic braking system is a method developed by the automotive industries in order to minimize the number of accidents. In this research has been designed and made an automatic braking system using fuzzy logic control. In this system used DC motors equipped with a rotary encoder to measure the speed of the prototype of car. In this prototype also features ultrasonic sensor to measure the distance of the obstacle. In this system, the arduino microcontroller was used to serve as a main control system. In this study, the braking tests of the prototype were conducted with various distance. At the pwm of 49, 43, 39% the prototype can stop at a distance of about 35.9, 45.5, 47 cm, respectively. This method is expected to be used as a braking system on the electric cars so that can reduce the risk of accidents.

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas kasih dan rahmat-Nya penulis dapat menyelesaikan Tugas Akhir ini dengan judul :

Sistem Pengereman Otomatis Menggunakan Kendali Logika Fuzzy

Tugas Akhir ini merupakan persyaratan dalam menyelesaikan pendidikan program Strata-Satu di Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya.

Tugas Akhir ini dibuat berdasarkan teori-teori yang didapat selama mengikuti perkuliahan, berbagai literatur penunjang dan pengarahan dosen pembimbing dari awal hingga akhir pengerjaan Tugas Akhir ini.

Pada kesempatan ini, penulis ingin berterima kasih kepada pihak-pihak yang membantu pembuatan tugas akhir ini, khususnya kepada:

1. Dr. Muhammad Rivai, S.T., M.T. selaku dosen pembimbing pertama saya dan Ir. Tasripan, M.T. selaku dosen pembimbing kedua saya
2. Dr. Tri Arief Sardjono, ST., MT., Ir. Siti Halimah Baki, MT., Rachmad Setiawan, ST., MT. dan Dr. Astria Nur Irfansyah, ST., M.eng., selaku dosen penguji yang telah megoreksi Tugas Akhir ini.
3. Dr. Eng. Ardyono Priyadi, ST., M.Eng. selaku ketua Departemen Teknik Elektro ITS
4. Icshsan Pratam, S.T. yang telah banyak membantu saya dalam pengengerjaan Tugas Akhir ini
5. Bapak, Ibu, dan saudara-saudara saya yang telah membantu saya secara moril selama ini.
6. Teman-teman E-52 yang tidak bisa saya sebutkan satu persatu yang sudah membantu.

banyak hal yang dapat diperbaiki. Saran, kritik dan masukan dari semua pihak sangat membantu penulis untuk pengembangan lebih lanjut.

Halaman ini sengaja dikosongkan

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB I

PENDAHULUAN

1.1 Latar Belakang

Indonesia merupakan salah satu negara berkembang yang ada di dunia. Setiap tahun jumlah penduduk meningkat secara signifikan dan juga jumlah kendaraan bermotor yang ada juga meningkat setiap tahunnya.

Berdasarkan data badan pusat statistik pada tahun 2012 didapatkan bahwa pada tahun 2002 di Indonesia didapatkan 12.267 kasus kecelakaan lalu lintas. Sedangkan sepuluh tahun kemudian didapatkan kasus kecelakaan sebanya 117.949. Dari data di atas didapatkan bahwa terjadi peningkatan yang sangat besar antara tahun 2002-2012. Peningkatan yang didapatkan sebesar sepuluh kali lipat.

Berdasarkan data dari Korlantas Mabes Polri didapatkan bahwa persentase kecelakaan yang didasari oleh manusia adalah sebesar 65,67%. Oleh faktor prasarana jalan sebesar 12,80%, kelaikan jalan 10,74%, kelaiakan kendaraan sebesar 9,70%, dan oleh faktor dari alam sebesar 1,28%.

Dari data yang di atas dapat disimpulkan bahwa faktor yang mempengaruhi dari kecelakaan lalu lintas di Indonesia adalah faktor dari manusia itu sendiri.

Perancangan tugas akhir ini bertujuan agar dapat meminimalisir terjadinya kecelakaan karena faktor pengemudi tersebut. Dengan adanya Sistem Pengereman Otomatis diharapkan kecelakaan yang dikarenakan oleh manusia dapat berkurang. Dengan berkurangnya angka kecelakaan yang diakibatkan oleh manusia diharapkan angka kecelakaan lalu lintas di Indonesia dapat berkurang

Sistem ini menggunakan kontrol logika fuzzy dikarenakan pada logika fuzzy tidak diperlukan perhitungan matematikan yang rumit. Pada logika fuzzy juga kontrol dapat sesuai dengan keinginan pembuat sistem kontrol. Sehingga dapat sesuai dengan pengguna sistem kontrol tersebut

1.2 Perumusan Masalah

Permasalahan pada penelitian ini adalah sebagai berikut:

1. Bagaimana cara kerja sistem pengereman otomatis.
2. Jenis kontrol yang pada sistem pengereman.
3. Bagaimana cara sistem pengereman agar prototipe berhenti pada jarak aman.

1.3 Tujuan

Tujuan dari pembuatan tugas akhir ini adalah:

1. Menerapkan pengaturan *pulse width modulation* untuk sistem pengereman.
2. Penggunaan kontrol logika fuzzy dalam sistem pengereman.
3. Menggunakan sensor jarak sebagai input logika fuzzy.

1.4 Batasan Masalah

Batasan masalah pada tugas akhir ini adalah sebagai berikut:

1. Pada *range* kecepatan dan jarak yang tentu.
2. Menggunakan sensor ultrasonik.
3. Menggunakan kendaran *prototype* mobil.
4. Behenti pada jarak tang tertentu.
5. Sistem pengereman mulai berkerja pada jarak yang ditentukan.

1.5 Metodologi Penelitian

Metodologi yang digunakan pada penelitian tugas akhir ini adalah sebagai berikut:

1. Studi Literatur

Pada tahap studi literatur dilakukan pengumpulan dasar teori yang menunjang dalam penulisan tugas akhir. Dasar teori tersebut diambil dari artikel-artikel di internet, forum-forum diskusi dan buku literatur.

2. Perancangan Perangkat Keras

Pada tahap ini merancang perangkat keras yang akan digunakan. Kendali logika fuzzy akan di masukkan pada microkontroller.

Sehingga microkontroller ini dapat menggerakan driver motor yang akan menggerakan roda. Microkontroller ini juga dapat mengatur sistem pengereman dan menentukan kecepatan awalnya.

3. Perancangan Perangkat Lunak

Pada tahap ini merancang software yang akan mengatur kerja mikrokontroller dalam mengatur sistem pengereman kendaraan. Dalam tahap ini juga akan diatur kendali logika fuzzy yang akan digunakan dalam mikrokontroller.

4. Perancangan Sistem

Setelah melakukan riset dari referensi yang berkaitan dengan pengerjaan tugas akhir ini, langkah berikutnya adalah melaksanakan perancangan sistem yang akan digunakan dalam implementasi hardware. Pada tahap ini digabungkan antara perancangan hardware dan perancangan perangkat lunak.

5. Pengujian dan Perbaikan Sistem

Pada tahap ini, dilakukan pengujian terhadap robot dengan menjalankan keseluruhan fungsi. Analisis dilakukan untuk memperbaiki galat yang dijumpai.

6. Penulisan Laporan Tugas Akhir

Tahap penulisan laporan tugas akhir dilakukan beriringan pengerjaan.

1.6 Sistematika Penulisan

Dalam penulisan buku tugas akhir ini sesuai dengan format penulisan tugas akhir yang telah ditentukan oleh jurusan Teknik Elektro Institut Teknologi Sepuluh Nopember. Tugas Akhir ini terdiri dari lima bab yang akan diuraikan sebagai berikut

1. Bab 1 : Pendahuluan

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

2. Bab 2 : Dasar Teori

Bab ini menjelaskan tentang teori-teori pendukung yang digunakan. Teori-teori pendukung yang digunakan antara lain teori pengereman, kendali logika fuzzy, pengendali mikrokontroller dan arduino, sensor

ultrasonik, *rotary encoder*, *pulse width modulation*, motor dc, *liquid crystal*, dan jarak.

3. Bab 3 : Perancangan Sistem

Bab ini menjelasakan tentang perencanaan sistem baik perangkat keras (hardware) maupun perangkat lunak (software) untuk sistem pengukuran posisi target dan pengarahan senjata secara otomatis.

4. Bab 4 : Pengujian

Pada bab ini akan menjelaskan hasil uji coba sistem beserta analisanya.

5. Bab 5 : Penutup

Bagian ini merupakan bagian akhir yang berisikan kesimpulan yang diperoleh dari pembuatan Tugas Akhir ini, serta saran-saran untuk pengembangan lebih lanjut.

1.7 Relevansi

Dengan adanya tugas akhir ini diharapakan dapat memberikan mamfaat sebagai berikut :

1. Dapat membantu merancang sistem pengereman otomatis pada kendaraan bermotor
2. Dapat membantu penelitian selanjutnya tentang pengereman otomatis

BAB II

DASAR TEORI

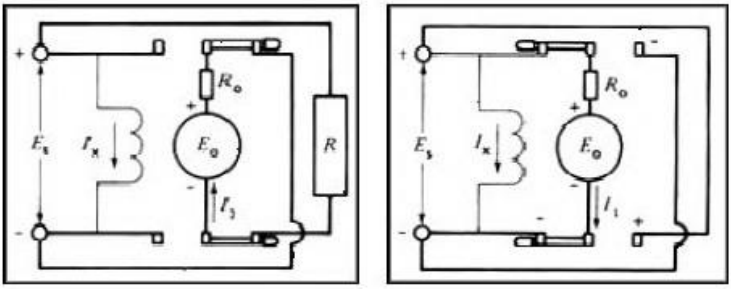
Pada bab ini akan dijelaskan macam-macam dasar teori yang digunakan dalam perancangan penelitian ini.

2.1 Teori pengereman

Sistem pengereman adalah sistem pada kendaraan yang berfungsi untuk memperlambat dan menghentikan kendaraan. Pada mobil listrik ada 2 tipe sistem pengereman yang biasa sering digunakan yaitu sistem pengereman dinamis dan sistem pengereman pluging. Pada sistem pengereman dinamis hal yang dilakukan adalah melepaskan jangkar yang berputar dari sumber tegangan dan memasangkan tahanan pada terminal jangkar [1]. Pada sistem pengereman *pluging* hal yang dilakukan adalah membalikan arus angker dengan membalik terminal sumber sehingga motor dapat berhenti lebih cepat seperti ditunjukan pada gambar 2.1.

Sistem pengereman dalam kendaraan dapat berfungsi dalam banyak hal antara lain :

1. Mengurangi kecepatan kendaraan.
2. Menghentikan kendaraan yang sedang berjalan.
3. Menjaga agar kendaraan tetap berhenti.

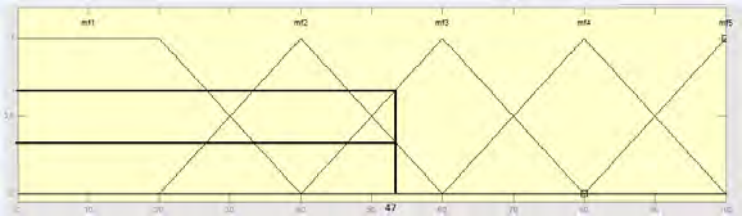


Gambar 2.1 Konfigurasi antara Armature Dan Sumber DC Es [1].

2.2 Kendali Logika Fuzzy

2.2.1 Fuzzifikasi

Fuzzifikasi yaitu suatu proses untuk mengubah suatu masukan dari bentuk tegas (crisp) menjadi fuzzy (variabel linguistik) yang biasanya disajikan dalam bentuk himpunan-himpunan fuzzy dengan suatu fungsi keanggotaannya masing-masing. Contoh dari proses Fuzzifikasi adalah seperti yang ditunjukkan di gambar 2.1. Sebuah sistem fuzzy untuk mengukur suhu mempunyai 5 buah membership function yang mempunyai label sangat dingin, dingin, hangat, panas, sangat panas. Kemudian input yang diperoleh dari crisp input adalah 47° maka pengambilan fuzzy input adalah seperti pada gambar 2.1 berikut.



Gambar 2.2 contoh fuzzifikasi

2.2.2 Metode Mamdani

Metode Mamdani sering juga dikenal dengan nama metode Max-Min. metode ini diperkenalkan oleh Ebrahim Mamdani pada tahun 1975. Untuk mendapatkan output diperlukan beberapa tahapan, antara lain:

1. Pembentukan himpunan fuzzy

   Pada Metode Mamdani, baik variabel input maupun variabel output dibagi menjadi satu atau lebih himpunan fuzzy.

2. Aplikasi fungsi implikasi

   Pada Metode Mamdani, fungsi implikasi yang digunakan adalah Min.

2.2.3 Komposisi aturan

Tidak seperti penalaran monoton, apabila sistem terdiri dari beberapa aturan, maka inferensi diperoleh dari kumpulan dan kolerasi

antar aturan. Ada 3 metode yang digunakan dalam melakukan inferensi sistem fuzzy, yaitu *max*, *additive* dan probabilistik *OR* (probor).

1. Metode *Max* (*Maximum*)

   Metode *Max* (*Maximum*) mengambil solusi himpunan fuzzy diperoleh dengan cara mengambil nilai maksimum aturan, kemudian menggunakannya untuk memodifikasi daerah fuzzy, dan mengaplikasikannya ke output dengan menggunakan operator *OR* (*union*). Jika semua proposisi telah dievaluasi, maka keluaran akan berisi suatu himpunan fuzzy yang merefleksikan kontribusi dari tiap-tiap proporsi. Secara umum dapat dituliskan seperti pada rumus 2.1:

   $$\mu_{sf}[x_i] \leftarrow \max(\mu_{sf}[x_i], \mu_{kf}[x_i] \quad \textbf{2.1}$$

   dengan:

   $\mu_{sf}[x_i]$ = nilai keanggotaan solusi fuzzy sampai aturan ke-i

   $\mu_{kf}[x_i]$ = nilai keanggotaan konsekuen fuzzy sampai aturan ke-i

2. Metode Additive (Sum)

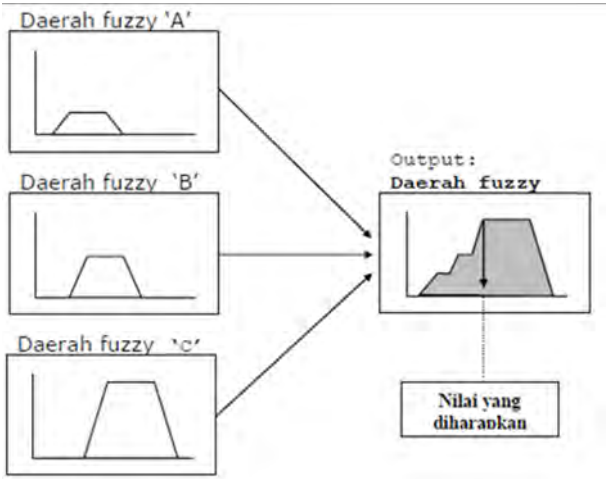
   Metode *Additive* (*Sum*) mengambil solusi himpunan fuzzy diperoleh dengan cara melakukan *bounded-sum* terhadap semua output daerah fuzzy.

3. Metode Probabilistik OR (probor)

   Metode Probabilitik *OR* (probor) mengambil solusi himpunan fuzzy diperoleh dengan cara melakukan *product* terhadap semua output daerah fuzzy.

2.2.4 Penegasan (defuzzyfikasi)

Input dari proses defuzzyfikasi adalah suatu himpunan fuzzy yang diperoleh dari komposisi aturan-aturan fuzzy, sedangkan output yang dihasilkan merupakan suatu bilangan pada domain himpunan fuzzy tersebut. Sehingga jika diberikan suatu himpunan fuzzy dalam *range* tertentu, maka keluarannya akan terlihat pada Gambar 2.3.



Gambar 2.3 defuzzifikasi atau penegasan

Ada beberapa metode defuzzifikasi pada komposisi aturan mamdani antara lain:

1. Metode Centroid (Composite Moment).

   Pada metode *centroid* solusi crispnya diperoleh dengan cara mengambil titik pusat daerah fuzzy. Secara umum dapat dilihat seperti pada rumus 2.2 :

$$z^* = \frac{\int_x^y z\mu(z)dz}{\int_x^y \mu(z)dz} \quad \text{..... (2.2a)}$$

$$z^* = \frac{\sum_{j=1}^{n} z_j\mu(z_j)}{\sum_{j=1}^{n} \mu(z_j)} \quad \text{..... (2.2b)}$$

2. Metode Bisektor

Pada metode bisektor solusi crispnya diperoleh dengan cara mengambil nilai pada domain yang memiliki nilai keanggotaan separo dari jumlah total nilai keanggotaan pada daerah fuzzy.

3. Metode *Mean of Maximum* (MOM)

Pada metode *mean of maximum* solusi crispnya diperoleh dengan cara mengambil nilai rata-rata domain yang memiliki nilai keanggotaan maksimum.

4. Metode *Largest of Maximum* (LOM)

Pada metode *largest of maximum* solusi crispnya diperoleh dengan cara mengambil nilai terbesar dari domain yang memiliki nilai keanggotaan maksimum.

5. Metode *Smallest of Maximum* (SOM).

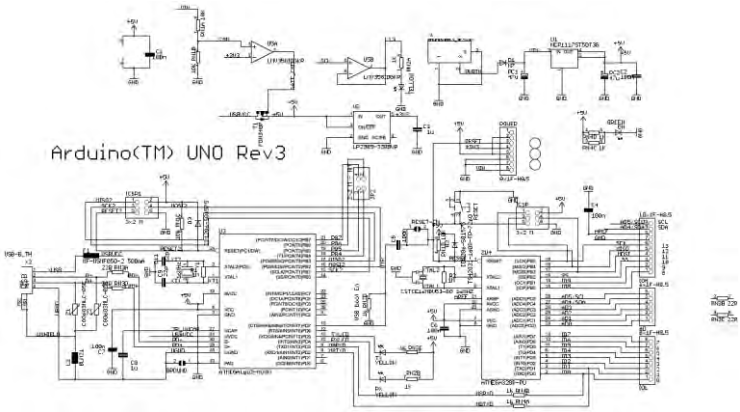
Pada metode *smallest of maximum* solusi crispnya diperoleh dengan cara mengambil nilai terkecil dari domain yang memiliki nilai keanggotaan maksimum.

2.3 Pengendali Mikrokontroler dan Arduino

Pengendali mikrokontroler adalah sebuah *chip* yang berfungsi sebagai pengontrol rangkaian elektronik dan umumnya dapat menyimpan program di dalamnya. Mikrokontroler adalah sebuah *chip* yang berfungsi sebagai pengontrol rangkaian elektronik dan umumnya dapat menyimpan program, dan umumnya terdiri dari CPU (*Central Processing Unit*), memori, I/O tertentu dan unit pendukung seperti *Analog-to-Digital Converter* (ADC) yang sudah terintegrasi di dalamnya. Kelebihan utama dari mikro kontroler ialah tersedianya RAM dan peralatan I/O pendukung sehingga ukuran *board* mikrokontroler menjadi sangat ringkas. Mikrokontroler adalah sebuah *chip* yang berfungsi sebagai pengontrol rangkaian elektronik dan umumnya dapat menyimpan program did MCS51 ialah mikrokomputer CMOS 8 bit dengan 4 KB *Flash* PEROM (*Programmable and Erasable Only Memory*) yang dapat dihapus dan ditulisi sebanyak 1000 kali. Mikrokontroler ini diproduksi dengan

menggunakan teknologi *high density non-volatile memory*. *Flash* PEROM *on-chip* tersebut memungkinkan memori program untuk diprogram ulang dalam sistem (*in-system programming*) atau dengan menggunakan *programmer non-volatile memory* konvensional. Kombinasi CPU 8 bit serba guna dan *Flash* PEROM, menjadikan mikrokontroler MCS51 menjadi *Micro Computer* handal yang fleksibel [2].

Pengendali mikro yang akan digunakan adalah Arduino. Arduino merupakan salah satu mikrokontroler yang berbasis *single-board* yang bersifat *open-source*. *Open-Source* berarti setiap orang dapat dengan bebas mengembangkan karena tidak terikat pada seseorang atau suatu perusahaan. *Processor* yang digunakan pada arduino adalah Atmel AVR. Arduino termasuk mikrokontroler yang mudah untuk digunakan karena memiliki kemiripan *syntax* dengan bahasa pemograman C. Selain itu pengemabangan dari arduino termasuk yang paling masif di Indonesia. Gambar 2.4 dibawah ini merupakan skematik dari arduino uno.

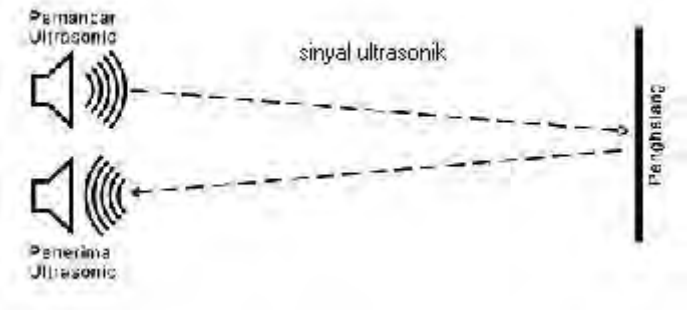


Gambar 2.4 skematik arduino

2.4 Sensor Ultrasonik

Sensor Ultrasonik adalah alat elektronika yang kemampuannya bisa mengubah dari energi listrik menjadi energi mekanik dalam bentuk gelombang suara ultrasonik. Sensor ini terdiri dari rangkaian pemancar Ultrasonik yang dinamakan *transmitter* dan penerima ultrasonik yang disebut *receiver*. Alat ini digunakan untuk mengukur gelombang ultrasonic. Gelombang ultrasonik adalah gelombang mekanik yang memiliki ciri-ciri longitudinal dan biasanya memiliki frekuensi di atas 20 Khz. Gelombang Utrasonik dapat merambat melalui zat padat, cair, maupun gas. Gelombang Ultrasonik adalah gelombang rambatan energi dan momentum mekanik sehingga merambat melalui ketiga element tersebut sebagai interaksi dengan molekul dan sifat inersia medium yang dilaluinya [4].

Ada beberapa penjelasan mengenai gelombang ultrasonik. Sifat dari gelombang ultrasonik yang melalui medium menyebabkan getaran partikel dengan medium amplitudo sama dengan arah rambat longitudinal sehingga menghasilkan partikel medium yang membentuk suatu rapatan atau biasa disebut *Strain* dan tegangan yang biasa disebut *Strees*. Proses lanjut yang menyebabkan terjadinya rapatan dan regangan di dalam medium disebabkan oleh getaran partikel secara periodik selama gelombang ultrasonik lainya. Gelombang ultrasonik merambat melalui udara dengan kecepatan 344 meter per detik, mengenai objek dan memantul kembali ke sensor. Gambar 2.5 dibawah merupakan cara kerja sensor ultrasonik

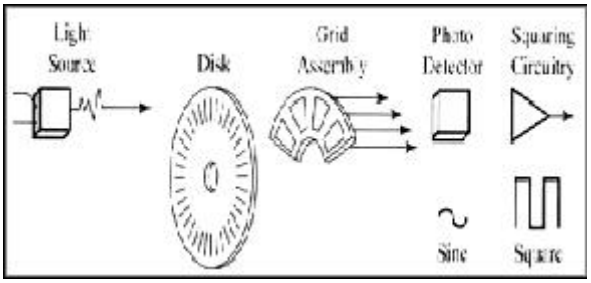


Gambar 2.5 Cara kerja sensor ultrasonik

Seperti yang telah umum diketahui, gelombang ultrasonik hanya bisa didengar oleh makhluk tertentu seperti kelelawar dan ikan paus. Kelelawar menggunakan gelombang ultrasonik untuk berburu di malam hari sementara paus menggunakanya untuk berenang di kedalaman laut yang gelap.
Perhitungan waktu yang diperlukan modul sensor Ping untuk menerima pantulan pada jarak tertentu mempunyai rumus S= (tIN x V) : 2. Rumus diatas mempunyai keterangan sebagai berikut. (S) adalah jarak antara sensor ultrasonik dengan objek yang terdeteksi. Kecepatan adalah cepat rambat gelombang ultrasonik di udara dengan kecepatan normal 344 meter per detik (tIN) adalah selisih waktu pemancaran dan penerimaan pantulan gelombang. Ada 3 prinsip kerja dari sensor ultrasonik yaitu, sinyal dipancarkan melalui pemancar gelombang ultrasonik. Sinyal yang dipancarkan akan merambat sebagai gelombang bunyi dengan kecepatan bunyi berkisar 344 m/s. Dan yang terakhir sinyal yang sudah diterima akan diproses untuk menghitung jaraknya.

2.5 *Rotary Encoder*

Rotary encoder adalah divais elektromekanik yang dapat memonitor gerakan dan posisi. *Rotary encoder* umumnya menggunakan sensor optik untuk menghasilkan serial pulsa yang dapat diartikan menjadi gerakan, posisi, dan arah. Sehingga posisi sudut suatu poros benda berputar dapat diolah menjadi informasi berupa kode digital oleh rotary encoder untuk diteruskan oleh rangkaian kendali. *Rotary encoder* umumnya digunakan pada pengendalian robot, motor drive.



Gambar 2.6 Blok penyusun rotary encoder

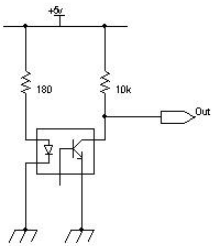
Rotary encoder tersusun dari suatu piringan tipis yang memiliki lubang-lubang pada bagian lingkaran piringan. LED ditempatkan pada salah satu sisi piringan sehingga cahaya akan menuju ke piringan. Di sisi yang lain suatu *photo-transistor* diletakkan sehingga *photo-transistor* ini dapat mendeteksi cahaya dari LED yang berseberangan. Piringan tipis tadi dikopel dengan poros motor, atau divais berputar lainnya yang ingin diketahui posisinya, sehingga ketika motor berputar piringan juga akan ikut berputar. Apabila posisi piringan mengakibatkan cahaya dari LED dapat mencapai *photo-transistor* melalui lubang-lubang yang ada, maka *photo-transistor* akan mengalami saturasi dan akan menghasilkan suatu pulsa gelombang persegi. Gambar 2.6 diatas menunjukkan bagan skematik sederhana dari *rotary encoder*. Semakin banyak deretan pulsa yang dihasilkan pada satu putaran menentukan akurasi rotary encoder tersebut, akibatnya semakin banyak jumlah lubang yang dapat dibuat pada piringan menentukan akurasi *rotary encoder* tersebut [5].

Rangkaian penghasil pulsa pada Gambar 2.7 dibawah yang digunakan umumnya memiliki output yang berubah dari +5V menjadi 0.5V ketika cahaya diblok oleh piringan dan ketika diteruskan ke *photo-transistor*. Karena divais ini umumnya bekerja dekat dengan motor DC maka banyak noise yang timbul sehingga biasanya output akan dimasukkan ke *low-pass* filter dahulu. Apabila *low-pass* filter digunakan, frekuensi *cut-off* yang dipakai umumnya ditentukan oleh jumlah slot yang ada pada piringan dan seberapa cepat piringan tersebut berputar, dinyatakan dengan rumus 2.3 :

$$f_c = \frac{s_w n}{60} \quad \text{..... 2.3}$$

Dimana f_c adalah frekuensi *cut-off* filter, s_w adalah kecepatan piringan dan n adalah jumlah slot pada piringan.

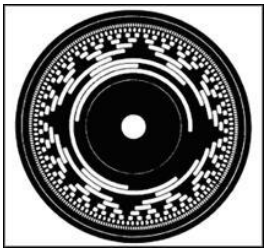
Terdapat dua jenis rotary encoder yang digunakan, *Absolute Rotary Encoder* dan *Incremental Rotary Encoder*. Masing-masing *rotary encoder* ini akan dipaparkan pada bagian berikutnya.



Gambar 2.7 Rangkaian tipikal penghasil pulsa pada rotary encoder

2.5.1 *ABSOLUTE ROTARY ENCODER*

Absolute encoder menggunakan piringan dan sinyal optik yang diatur sedemikian sehingga dapat menghasilkan kode digital untuk menyatakan sejumlah posisi tertentu dari poros yang dihubungkan padanya. Piringan yang digunakan untuk absolut enkoder tersusun dari segmen-segmen cincin konsentris yang dimulai dari bagian tengah piringan ke arah tepi luar piringan yang jumlah segmennya selalu dua kali jumlah segmen cincin sebelumnya. Cincin pertama di bagian paling dalam memiliki satu segmen transparan dan satu segmen gelap, cincin kedua memiliki dua segmen transparan dan dua segmen gelap, dan seterusnya hingga cincin terluar. Sebagai contoh apabila absolut enkoder memiliki 16 cincin konsentris maka cincin terluarnya akan memiliki 32767 segmen. Gambar 2.8 menunjukkan pola cincin pada piringan absolut encoder yang memiliki 16 cincin.

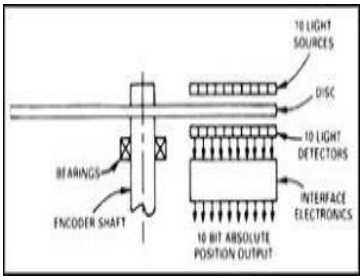


Gambar 2.8 pola 16 cincin konsentris pada absolut enkoder

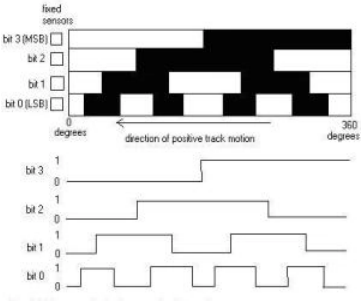
Karena setiap cincin pada piringan absolut enkoder memiliki jumlah segmen kelipatan dua dari cincin sebelumnya, maka susunan

ini akan membentuk suatu sistem biner. Untuk menghasilkan sistem biner pada susunan cincin maka diperlukan pasangan LED dan *photo-transistor* sebanyak jumlah cincin yang ada pada absolut enkoder tersebut.

Sistem biner yang untuk menginterpretasi posisi yang diberikan oleh absolut enkoder dapat menggunakan kode *gray* atau kode biner biasa, tergantung dari pola cincin yang digunakan. Untuk lebih jelas, dilihat contoh *absolute encoder* yang hanya tersusun dari 4 buah cincin untuk membentuk kode 4 bit. Apabila enkoder ini dihubungkan pada poros, maka *photo-transistor* akan mengeluarkan sinyal persegi sesuai dengan susunan cincin yang digunakan. Gambar 2.10 dan 2.11 menunjukkan contoh perbedaan diagram keluaran untuk *absolute encoder* tipe *gray code* dan tipe *binary code*.

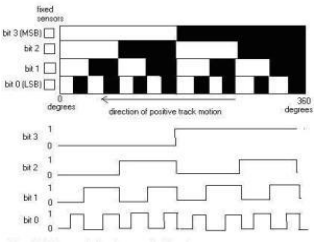


Gambar 2.9 Contoh piringan dengan 10 cincin dan 10 LED – photo-transistor untuk membentuk sistem biner 10 bit



Gambar 2.10 Contoh diagram keluaran absolut encoder 4-bit tipe gray code

Dengan absolute encoder 4-bit ini maka akan didapatkan 16 informasi posisi yang berbeda yang masing-masing dinyatakan dengan kode biner atau kode *gray* tertentu. Tabel 2.1 menyatakan posisi dan output biner yang bersesuaian untuk absolut encoder 4-bit. Dengan membaca output biner yang dihasilkan maka posisi dari poros yang diukur dapat diketahui untuk diteruskan ke rangkaian pengendali. Semakin banyak bit yang dipakai maka posisi yang dapat diperoleh akan semakin banyak.



Gambar 2.11 Contoh diagram keluaran absolut encoder 4-bit tipe binary code

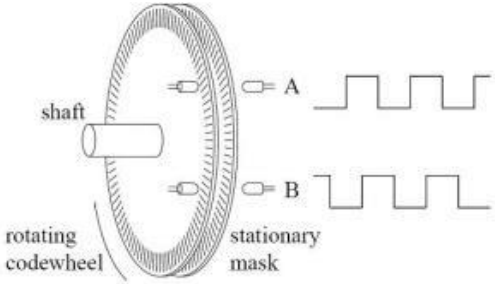
Tabel 2.1 posisi dan output biner yang bersesuaian untuk absolut encoder 4-bit

desimal	rentang putar	kode biner	kode gray
0	0-22,5	0000	0000
1	22,5-45	0001	0001
2	45-67,5	0010	0011
3	67,5-90	0011	0010
4	90-112,5	0100	0110
5	112,5-135	0101	0111
6	135-157,5	0110	0101
7	157,5-180	0111	0100
8	180-202,5	1000	1100
9	202,5-225	1001	1101
10	225-247,5	1010	1111
11	247,5-270	1011	1110
12	270-292,5	1100	1010
13	293,5-315	1101	1011
14	315-37,5	1110	1001
15	337,5-360	1111	1000

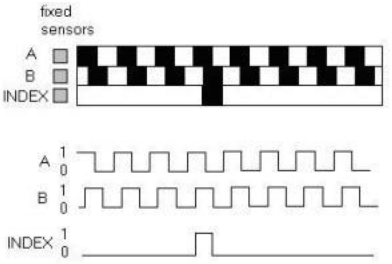
2.5.1 *INCREMENTAL ROTARY ENCODER*

Incremental encoder terdiri dari dua track atau single track dan dua sensor yang disebut channel A dan B yang ditunjukkan pada Gambar 2.12. Ketika poros berputar, deretan pulsa akan muncul di masing-masing *channel* pada frekuensi yang proporsional dengan kecepatan putar sedangkan hubungan fasa antara channel A dan B menghasilkan arah putaran. Dengan menghitung jumlah pulsa yang terjadi terhadap resolusi piringan maka putaran dapat diukur. Untuk mengetahui arah putaran, dengan mengetahui *channel* mana yang *leading* terhadap channel satunya dapat ditentukan arah putaran yang terjadi karena kedua channel tersebut akan selalu berbeda fasa seperempat putaran (*quadrature signal*). Seringkali terdapat output channel ketiga, disebut *INDEX*, yang menghasilkan satu pulsa per putaran berguna untuk menghitung jumlah putaran yang terjadi.

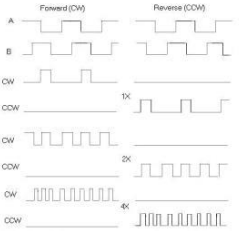
Contoh pola diagram keluaran dari suatu *incremental encoder* ditunjukkan pada Gambar 2.13. Resolusi keluaran dari sinyal *quadrature* A dan B dapat dibuat beberapa macam, yaitu 1X, 2X dan 4X. Resolusi 1X hanya memberikan pulsa tunggal untuk setiap siklus salah satu sinya A atau B, sedangkan resolusi 4X memberikan pulsa setiap transisi pada kedua sinyal A dan B menjadi empat kali resolusi 1X. Arah putaran dapat ditentukan melalui level salah satu sinyal selama transisi terhadap sinyal yang kedua. Pada contoh resolusi 1X, A = arah bawah dengan B = 1 menunjukkan arah putaran searah jarum jam, sebaliknya B = arah bawah dengan A = 1 menunjukkan arah berlawanan jarum jam.



Gambar 2.12 susunan piringan untuk incremental enkoder



Gambar 2.13 Contoh pola keluaran incremental encoder



Gambar 2.14 output dan arah putaran pada resolusi yang berbeda-beda

Pada *incremental encoder*, beberapa cara dapat digunakan untuk menentukan kecepatan yang diamati dari sinyal pulsa yang dihasilkan.

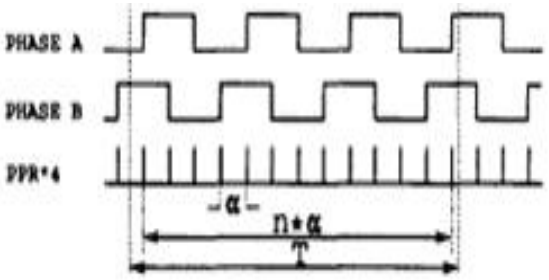
Diantaranya adalah menggunakan *frequencymeter* dan periodimeter. Cara yang sederhana untuk menentukan kecepatan dapat dengan *frequencymeter*, yakni menghitung jumlah pulsa dari enkoder, n, pada selang waktu yang tetap, T, yang merupakan periode loop kecepatan seperti pada Gambar 2.14 di atas. Apabila α adalah sudut antara pulsa encoder, maka sudut putaran pada suatu periode adalah seperti pada rumus 2.4 :

$$\alpha_f = n\alpha \quad \text{.....} \quad \textbf{2.4}$$

Sehingga kecepatan putar akan didapatkan adalah seperti pada rumus 2.5 :

$$\omega_T = \frac{\alpha_f}{T} \quad \text{.....} \quad \textbf{2.5}$$

Kelemahan yang muncul pada cara ini adalah pada setiap periode sudut α_f yang didapat merupakan kelipatan integer dari α. Ini akan dapat menghasilkan *quantification error* pada kecepatan yang ingin diukur.



Gambar 2.15 Sinyal keluaran encoder untuk pengukuran kecepatan dengan frequencymeter

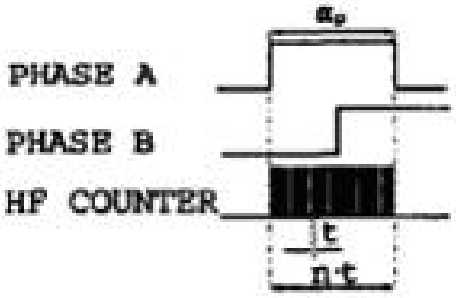
Cara yang lain adalah dengan menggunakan periodimeter. Dengan cara ini akan diukur kecepatan tidak lagi dengan menghitung jumlah pulsa enkoder tetapi dengan menghitung clock frekuensi tinggi (*HF Clock*) untuk sebuah pulsa dari encoder yaitu mengukur periode pulsa dari enkoder seperti pada Gambar 2.15. Apabila α_p adalah sudut dari pulsa enkoder, t adalah periode dari HF clock, dan n adalah jumlah pulsa HF yang terhitung pada counter. Maka waktu untuk sebuah pulsa enkoder, T_p, adalah seperti pada rumus 2.6:

$$Tp = nt \quad \text{..... 2.6}$$

Sehingga kecepatan yang akan diukur dapat diperoleh adalah seperti pada rumus 2.7 :

$$\omega_T = \frac{\alpha_p}{T_p} \quad \text{..... 2.7}$$

Seperti halnya pada frequencymeter, disini juga muncul *quantification error* karena waktu T_p akan selalu merupakan perkalian integer dengan t.



Gambar 2.16 Pengukuran kecepatan dengan menggunakan Periodimeter

2.6 *Pulse Width Modulation*

2.6.1 Pengertian PWM

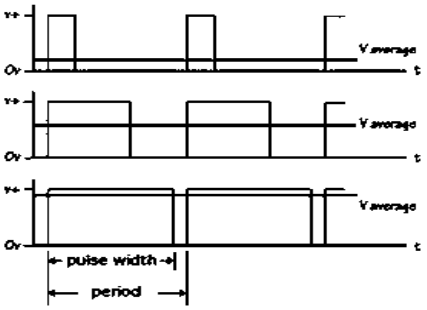
Pulse Width Modulation (PWM) secara umum adalah sebuah cara memanipulasi lebar sinyal yang dinyatakan dengan pulsa dalam satu periode, untuk mendapatkan tegangan rata-rata yang berbeda. Bebarapa contoh aplikasi PWM adalah pemodulasian data untuk telekomunikasi, pengontrolan daya atau tegangan yang masuk ke beban, regulator tegangan, *audio effect* dan penguatan, serta aplikasi-aplikasi lainnya [6].

Aplikasi PWM berbasis mikrokontroller biasanya berupa pengendalian kecepatan motor DC, pengendalian motor servo, dan pengaturan nyala terang LED. Oleh karena itu diperlukan pemahaman terhadap konsep PWM itu sendiri.

2.6.2 Konsep Dasar PWM

Sinyal PWM pada umumnya memiliki amplitude dan frekuensi dasar yang tetap, namun memiliki lebar pulsa yang bervariasi. Lebar pulsa PWM berbanding lurus dengan amplitude sinyal asli yang belum termodulasi. Artinya, sinyal PWM memiliki frekuensi gelombang yang tetap namun *duty cycle* bervariasi antara 0% hingga 100%.

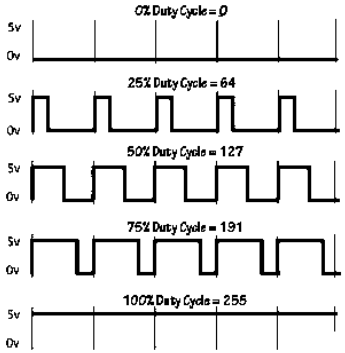
Dari persamaan diatas, diketahui bahwa perubahan duty cycle akan merubah tegangan output atau tegangan rata-rata seperti gambar 2.17 dibawah ini.



Gambar 2.17 pengaruh perubahan *duty cycle* terhadap tagangan

PWM merupakan salah satu teknik untuk mendapatkan sinyal analog dari sebuah piranti digital. Sebenarnya sinyal PWM dapat dibangkitkan dengan banyak cara, secara analog menggunakan IC op-amp atau secara digital.

Secara analog setiap perubahan PWM-nya sangat halus, sedangkan secara digital setiap perubahan PWM dipengaruhi oleh resolusi PWM itu sendiri. Resolusi adalah jumlah variasi perubahan nilai dalam PWM tersebut. Misalkan suatu PWM memiliki resolusi 8 bit, berarti PWM ini memiliki variasi perubahan nilai sebanyak 256 variasi mulai dari 0 – 225 perubahan nilai yang mewakili duty cycle 0% – 100% dari keluaran PWM tersebut sperti pada gambar 2.18



Gambar 2.18 perubahan *duty cycle*

2.7 Motor DC

2.7.1 Pengertian

Motor DC adalah motor listrik yang memerlukan suplai tegangan arus searah pada kumparan medan untuk diubah menjadi energi gerak mekanik seperti pada gambar 2.19 . Kumparan medan pada motor dc disebut stator (bagian yang tidak berputar) dan kumparan jangkar disebut rotor (bagian yang berputar). Motor arus searah, sebagaimana namanya, menggunakan arus langsung yang tidak langsung/*direct-unidirectional*. Motor DC memiliki 3 bagian atau komponen utama untuk dapat berputar sebagai berikut [7].

1. Motor DC sederhana memiliki dua kutub medan: kutub utara dan kutub selatan. Garis magnetik energi membesar melintasi ruang terbuka di antara kutub-kutub dari utara ke selatan. Untuk motor yang lebih besar atau lebih komplek terdapat satu atau lebih elektromagnet.

2. Dinamo yang berbentuk silinder, dihubungkan ke as penggerak untuk menggerakan beban. Untuk kasus motor DC yang kecil, dinamo berputar dalam medan magnet yang dibentuk oleh kutub-kutub, sampai kutub utara dan selatan magnet berganti lokasi.

3. Komponen ini terutama ditemukan dalam motor DC. Kegunaannya adalah untuk transmisi arus antara dinamo dan sumber daya.

Gambar 2.19 motor dc

Keuntungan utama motor DC adalah sebagai pengendali kecepatan, yang tidak mempengaruhi kualitas pasokan daya. Motor ini dapat dikendalikan dengan mengatur:

1. Tegangan dinamo – meningkatkan tegangan dinamo akan meningkatkan kecepatan

2. Arus medan – menurunkan arus medan akan meningkatkan kecepatan.

Hubungan antara kecepatan, flux medan dan tegangan dinamo ditunjukkan dalam persamaan berikut:

$$E = K\Phi N \quad \text{.....} \quad \textbf{2.8}$$

$$T = K\Phi I_a \quad \text{.....} \quad \textbf{2.9}$$

2.7.1 Jenis-Jenis Motor DC

1. Motor DC sumber daya terpisah/ Separately Excited
   Jika arus medan dipasok dari sumber terpisah maka disebut motor DC sumber daya terpisah/*separately excited*.

2. Motor DC Tipe *Shunt*
   Pada motor shunt, gulungan medan (*medan shunt*) disambungkan secara paralel dengan gulungan dinamo (A). Oleh karena itu total arus dalam jalur merupakan penjumlahan arus medan dan arus dinamo.Karakter kecepatan motor DC tipe *shunt* adalah :

   a. Kecepatan pada prakteknya konstan tidak tergantung pada beban (hingga torsi tertentu setelah kecepatannya berkurang) dan oleh karena itu cocok untuk penggunaan komersial dengan beban awal yang rendah, seperti peralatan mesin.

   b. Kecepatan dapat dikendalikan dengan cara memasang tahanan dalam susunan seri dengan dinamo (kecepatan berkurang) atau dengan memasang tahanan pada arus medan (kecepatan bertambah).

3. Motor DC Tipe Seri
   Dalam motor seri, gulungan medan (*medan shunt*) dihubungkan secara seri dengan gulungan dinamo (A). Oleh karena itu, arus medan sama dengan arus dinamo.
   Karakter kecepatan dari motor DC tipe seri adalah :
   1. Kecepatan dibatasi pada 5000 RPM
   2. Harus dihindarkan menjalankan motor seri tanpa ada beban sebab motor akan mempercepat tanpa terkendali.

4. Motor DC Tipe Kompon/Gabungan
   Motor Kompon DC merupakan gabungan motor seri dan shunt. Pada motor kompon, gulungan medan (*medan shunt*) dihubungkan secara paralel dan seri dengan gulungan dinamo (A). Sehingga, motor kompon memiliki torque penyalaan awal yang bagus dan kecepatan yang stabil.
   Karakter dari motor DC tipe kompon/gabungan ini adalah, makin tinggi persentase penggabungan (yakni persentase gulungan medan yang dihubungkan secara seri), makin tinggi pula torque penyalaan awal yang dapat ditangani oleh motor ini.

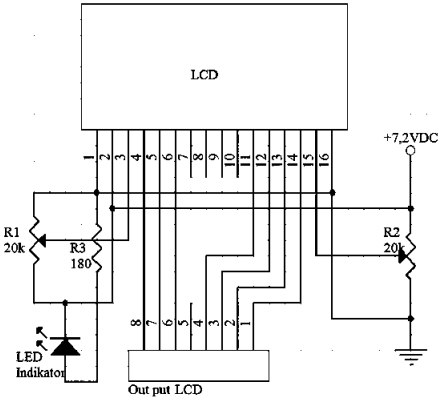
2.8 Liquid Crystal Display

Display elektronik adalah salah satu komponen elektronika yang berfungsi sebagai tampilan suatu data, baik karakter, huruf ataupun grafik. LCD (*Liquid Cristal Display*) adalah salah satu jenis display elektronik yang dibuat dengan teknologi *CMOS logic* yang bekerja dengan tidak menghasilkan cahaya tetapi memantulkan cahaya yang ada di sekelilingnya terhadap front-lit atau mentransmisikan cahaya dari back-lit. LCD (*Liquid Cristal Display*) berfungsi sebagai penampil data baik dalam bentuk karakter, huruf, angka ataupun grafik [8].

2.8.1 Material LCD (Liquid Cristal Display)

LCD adalah lapisan dari campuran organik antara lapisan kaca bening dengan elektroda transparan indium oksida dalam bentuk tampilan seven-segment dan lapisan elektroda pada kaca belakang seperti pada gambar 2.20 . Ketika elektroda diaktifkan dengan medan listrik (tegangan), molekul organik yang panjang dan silindris menyesuaikan diri dengan elektroda dari segmen. Lapisan *sandwich* memiliki *polarizer* cahaya vertikal depan dan *polarizer* cahaya

horisontal belakang yang diikuti dengan lapisan reflektor. Cahaya yang dipantulkan tidak dapat melewati molekul-molekul yang telah menyesuaikan diri dan segmen yang diaktifkan terlihat menjadi gelap dan membentuk karakter data yang ingin ditampilkan.



Gambar 2.20 Skematik lcd 16x2

2.8.2 Pengendali / Kontroler LCD (Liquid Cristal Display)

Dalam modul LCD (*Liquid Cristal Display*) terdapat microcontroller yang berfungsi sebagai pengendali tampilan karakter LCD (*Liquid Cristal Display*). Microntroller pada suatu LCD (*Liquid Cristal Display*) dilengkapi dengan memori dan register. Memori yang digunakan microcontroler internal LCD adalah :

1. DDRAM (*Display Data Random Access Memory*) merupakan memori tempat karakter yang akan ditampilkan berada.

2. CGRAM (*Character Generator Random Access Memory)* merupakan memori untuk menggambarkan pola sebuah karakter dimana bentuk dari karakter dapat diubah-ubah sesuai dengan keinginan.

3. CGROM (*Character Generator Read Only Memory*) merupakan memori untuk menggambarkan pola sebuah karakter dimana pola tersebut merupakan karakter dasar yang sudah ditentukan secara permanen oleh pabrikan pembuat LCD (*Liquid Cristal Display*) tersebut sehingga pengguna tinggal mangambilnya sesuai alamat

memorinya dan tidak dapat merubah karakter dasar yang ada dalam CGROM.

Register control yang terdapat dalam suatu LCD diantaranya adalah :

1. Register perintah yaitu register yang berisi perintah-perintah dari mikrokontroler ke panel LCD (*Liquid Cristal Display*) pada saat proses penulisan data atau tempat status dari panel LCD (*Liquid Cristal Display*) dapat dibaca pada saat pembacaan data.

2. Register data yaitu register untuk menuliskan atau membaca data dari atau ke DDRAM. Penulisan data pada register akan menempatkan data tersebut ke DDRAM sesuai dengan alamat yang telah diatur sebelumnya.

Kegunaan pin-pin yang ada dalam suatu LCD (*Liquid Cristal Display*) diantaranya adalah :

1. Pin data adalah jalur untuk memberikan data karakter yang ingin ditampilkan menggunakan LCD (Liquid Cristal Display) dapat dihubungkan dengan bus data dari rangkaian lain seperti mikrokontroler dengan lebar data 8 bit.

2. Pin RS (Register Select) berfungsi sebagai indikator atau yang menentukan jenis data yang masuk, apakah data atau perintah. Logika *low* menunjukan yang masuk adalah perintah, sedangkan logika *high* menunjukan data.

3. Pin R/W (*Read Write*) berfungsi sebagai instruksi pada modul jika *low* tulis data, sedangkan *high* baca data.

4. Pin E (*Enable*) digunakan untuk memegang data baik masuk atau keluar.

5. Pin VLCD berfungsi mengatur kecerahan tampilan (kontras) dimana pin ini dihubungkan dengan trimpot 5 KΩ, jika tidak digunakan dihubungkan ke ground, sedangkan tegangan catu daya ke LCD sebesar 5 Volt.

2.8 Jarak

Jarak adalah angka yang menunjukkan seberapa jauh suatu benda berubah posisi melalui suatu lintasan tertentu. Dalam fisika atau dalam pengertian sehari-hari, jarak dapat berupa estimasi jarak fisik dari dua buah posisi berdasarkan kriteria tertentu (misalnya jarak tempuh antara Jakarta-Bandung). Dalam bidang matematika, jarak haruslah memenuhi kriteria tertentu [9].

Berbeda dengan koordinat posisi, jarak tidak mungkin bernilai negatif. Jarak merupakan besaran skalar, sedangkan perpindahan merupakan besaran vektor.

Jarak yang ditempuh oleh kendaraan (biasanya ditunjukkan dalam odometer), orang, atau objek, haruslah dibedakan dengan jarak antara titik satu dengan lainnya.

Dalam ilmu fisika, jarak adalah panjang lintasan yang ditempuh oleh suatu objek yang bergerak, mulai dari posisi awal dan selesai pada posisi akhir. Konsep ini seringkali dipetukarkan dengan konsep perpindahan. Jarak dapat dituliskan pada rumus 2.8.

$$s = \int_{\overrightarrow{r2}}^{\overrightarrow{r2}} \left|\vec{v}(t)\left[\frac{dt}{dr}\right]\right| . d\vec{r} \quad \text{.....} \mathbf{2.10}$$

yang dapat dibaca sebagai panjang lintasan yang menghubungkan titik dan menggunakan kecepatan.

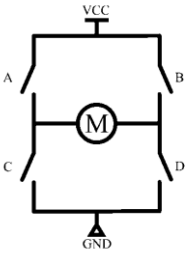
2.9 Driver Motor

2.9.1 Pengertian

Driver motor merupakan alat yang digunakan untuk menggerakan motor dc, mengatur kecepatan motor dc dan mengatur arah putar motor DC. Driver motor dapat dikontrol menggunakan mikrokontroller. Agar dapat mangatur arah putar motor DC driver motor harus mempunyai rangkaian *H-Bridge* di dalamnya.

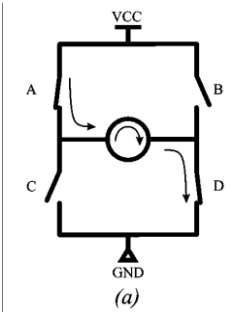
2.9.1 Rangkaian H-Bridge

H-Bridge merupakan suatu rangkian elektronika yang terdiri dari 4 saklar yang membentuk huruf H yang ditengah-tengahnya diberikan sebuah motor DC. Motor DC ini akan diatur arah putarnya oleh rangkaian *H-Bridge* ini.

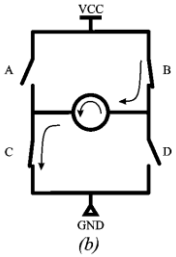


Gambar 2.21 Rangkaian *H-Bridge*

Cara kerja dari rangkaian *H-Bridge* untuk pengendalian motor ada 2 yaitu pada saat saklar A dan D aktif dan pada saat saklar B dan C aktif. Pada saat saklar A dan D aktif, arus akan mengalir dari Vcc menuju A menuju motor menuju D dan menuju *ground* sehingga motor akan berputar maju seperti pada gambar 3.6. Pada saat saklar B dan C yang aktif maka arus akan mengalir dari Vcc menuju B menuju motor menuju D dan menuju *ground* sehingga motor berputar mudur seperti pada gambar 3.7.



Gambar 2.22 motor maju



Gambar 2.23 motor mundur

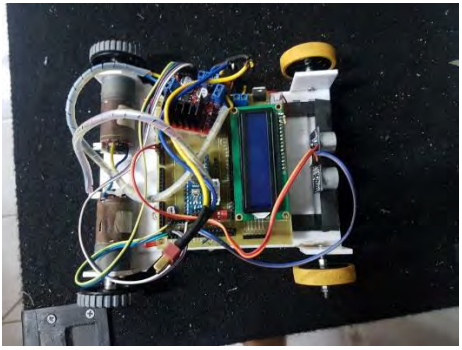
BAB III
PERANCANGAN SYSTEM

3.1 Perancangan perangkat keras

Pada bagian ini akan dibahas tentang bagian-bagian perangkat keras yang digunakan dan kegunaannya

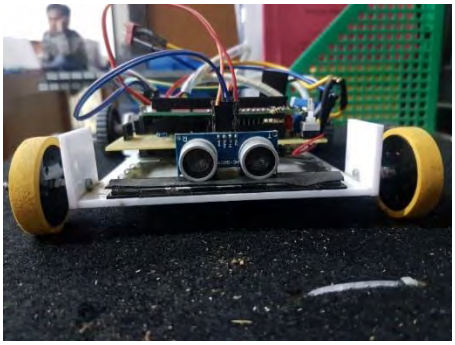
3.1.1 Gambaran perangkat keras

Pada *prototipe* mobil yang sudah dibuat ini digunakan motor DC 25GA370. Motor DC ini merupakan motor DC 12V 1000RPM dengan encoder. Encoder pada motor DC ini digunakan untuk membaca kecepatan. Motor DC ini di letakkan pada bagian belakang prototype mobil. Baterai yang digunakan untuk men*supply* prototype mobil ini adalah baterai *lithium polymer* (LiPo) 1100 mAh 11.1 V 3 cell diletakkan di atas mobil.



Gambar 3.1 tampak atas

Motor DC ini menggunakan driver motor L298N. Kelebihan driver motor L298N ini adalah presisi dalam mengontrol dan gampang untuk dikontrol. Driver motor ini juga dapat digunanakan untuk mengontrol 2 buah motor DC. Driver ini di letakkan di atas prototype motor di dekat Arduino mega. Terdapat juga 1 buat sensor ultrasonik HC-SR04 yang diletakan di bagian depan prototipe mobil. Sensor ini digunakan untuk membaca jarak.



Gambar 3.2 tampak depan

Pada prototipe ini mempunyai 2 buah mikrokontroller yaitu arduino nano dan arduino mega. Arduino nano di letakkan di atas arduino mega.

Gambar 3.3 tampak samping

3.1.2 Driver Motor L298N

Driver motor L298N merupakan driver motor berbasis pada *H-Bridge.* Sehingga pada motor ini dapat memutar maju dan memutar

mundur motor DC. Dalam IC driver motor ini terdapat 2 buah rangkaian *H-Bridge* sehingga driver motor ini dapat digunakan untuk 2 buah motor DC. Driver ini mampu menahan arus sebesar 2A. Pada driver ini output ke motor diberikan dioda agar dapat menahan arus balik yang datang dari motor DC. Input driver motor berasal dari mikrokontroler utama, untuk MOT 1A dan MOT 1B untuk menggerakan motor 1, ENABLE 1 untuk mengatur kecepatan motor 1 menggunakan PWM, selanjutnya untuk MOT 2A dan MOT 2B untuk menggerakan motor 2, ENABLE 2 untuk mengatur kecepatan motor 2 menggunakan PWM.

Gambar 3.4 Driver motor L298N

3.1.3 Arduino mega

Pada sistem pengereman otomatis menggunakan kontrol logika fuzzy ini, arduino berfungsi sebagai kontroller utama dalam sistem ini. Arduino mega bertugas membaca data jarak, menggerakan motor dan melakukan kontrol fuzzy terhadap sistem

Gambar 3.5 Arduino mega [10].

3.1.4 Arduino nano

Arduino nano pada sistem pengereman otomatis menggunkan kontrol logika fuzzy ini berfungsi untuk membaca nilai dari encoder. Nilai enkoder yang didapatkan dari arduino nano ini selanjutnya akan dikirimkan dengan pengiriman *serial* ke arduino mega.

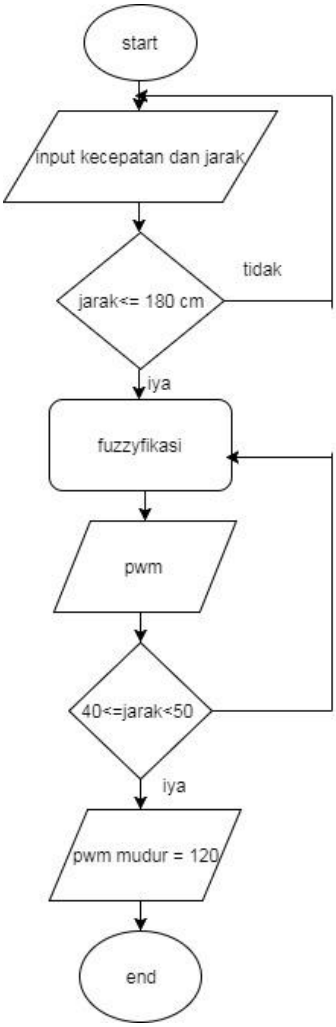
Gambar 3.9 arduino nano [11].

3.2 Perancangan Perangkat Lunak

Pada bagian ini akan dibahas tentang penyusunan perangkat lunak meliputi perangkat lunak yang ada pada mikrokontroller.

3.2.1 Alur Kerja Sistem

Pada sistem ini kecepatan diukur menggunakan enkoder dan jarak halangan di depannya diukur menggunakan sensor ultrasonik. Pembacaan jarak dan sensor ini dimasukkan kedalam fuzzy sebagai input. Output dari fuzzy pada sistem berupa nilai pwm. Fuzzy tidak akan berjalan sebelum jarak 180 cm dan akan berhenti antara jarak 50 sampai 30 cm dari depan halangan. Dibawah ini merupakan Gambar 3.6 yang merupakan gambar dari alur kerja sistem(*flowchart).*



Gambar 3.6 flowchart sistem

3.2.2 Pembacaan Sensor Ultrasonik

Sensor ultrasonik memiliki satu buah pin *trigger*, satu buah pin *echo,* satu buah pin Vcc dan satu buah pin *ground*. Pin *trigger*

berfungsi sebagai pemancar dan pin *echo* berfungsi sebagai penerima. Ketika menghidupkan pin *trigger* selama 10uS maka akan terbangkit 8 step sinyal dengan frekuensi 40kHz. Sinyal ini mempunyai kecepatan 340 m/s. Sinyal ini bila bersentuhan dengan sebuah benda akan memantul. Sinyal pantulan ini yang akan diterima oleh pin *echo.* Dengan adanya selesih waktu pantulan dari sinyal maka dapat mengetahui jarak prototipe mobil dengan benda. Rumus yang digunakan adalah sebagai berikut :

$$s = \frac{340 * t}{2} \quad \text{.....3.1}$$

Pada gambar 3.7 merupakan gambar sensor ultrasonik

Gambar 3.7 sensor ultrasonik

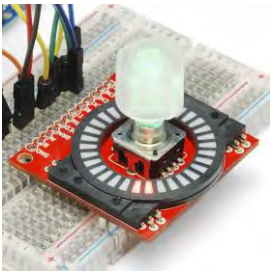
3.2.3 Pembacaan Encoder

Rotary encoder disini berfungsi untuk mengetahui kecepatan dari motor DC. Rotary encoder yang digunakan merupakan incerement encoder. Pada satu putaran terdapat 50 *increment*. Perhitungan kecepatan dapat dilakukan dengan rumus. f adalah jumlah pulsa dalam 1 detik, n adalah jumlah pulsa dan m adalah jumlah lubang dalam setiap detik

$$v = f * 2\pi r \quad \text{..... 3.2}$$

$$f = n * m \quad \text{..... 3.3}$$

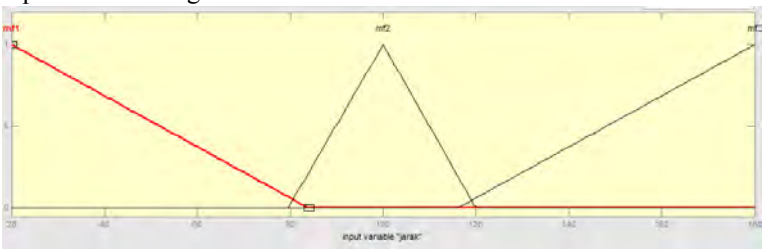
Pada gambar 3.12 merupakan gambar rotary encoder



Gambar 3.8 rotary encoder

3.2.4 Perancangan Kontrol Logika Fuzzy

Pada perancangan kontrol logika fuzzy ini menggunakan metode mamdani. Kelebihan metode Mamdani dibandingkan metode sistem penalaran fuzzy yang lain, diantaranya adalah karena bersifat intuitif, mencakup bidang yang luas, dan sesuai dengan proses input informasi manusia. Sistem penalaran fuzzy metode Mamdani dikenal juga dengan nama metode *Max-Min*. Alasan kenapa sistem penalaran Mamdani lebih menyerupai pola pikir manusia karena fungsi implikasi antara *antecedent* dengan *consequent* sama-sama dalam himpunan fuzzy. Pada fuzzy ini mempunyai dua input dan satu output. Input pertama dari fuzzy ini adalah nilai dari sensor jarak yang benilai dari 0 – 200. Input kedua adalah nilai dari encoder yang berupa kecepatan yang bernilai dari 0 – 255. *Membership function* dari kedua input adalah sebagai berikut :

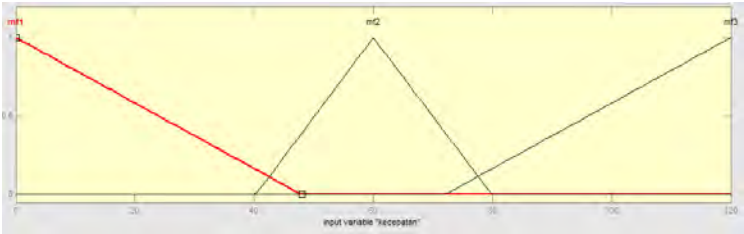


Gambar 3.9 membership jarak

Kemudian rule yang digunakan dalam kontrol fuzzy ini adalah sebagai berikut :

Tabel 3.10 tabel rule

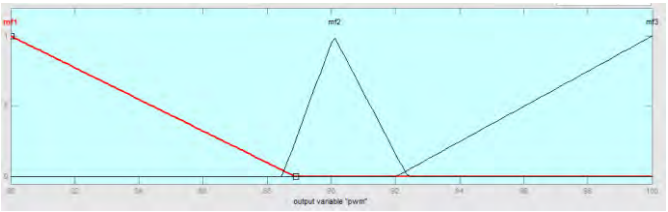
jarak \kecepatan	lambat	sedang	jauh
dekat	1	1	2
sedang	1	2	2
jauh	1	2	3



Gambar 3.11 membership kecepatan

Untuk proses defuzzzyfikai digunakan metode centroid dengan rumus :

$$FD = \frac{\sum \mu D}{\sum \mu} = \frac{\mu_u D_u + \mu_{lh} D_{LH} + \cdots}{\mu_u + \mu_{LH} + \cdots} \quad \text{.....3.4}$$



Gambar 3.12 membership output

BAB IV

PENGUKURAN DAN ANALISIS

Pada bab ini akan dibahas mengenai pengukuran yang sudah dilakukan, pengujian alat yang sudah diuji dan analisis dari pengukuran dan pengujian.

4.1 Pengujian Encoder

Pada bagian akan diketahui pengujian encoder yang telah di lakukan. Diameter roda yang digunakan adalah 4,6 cm sehingga jari-jari rodanya adalah 2,3 cm. Sehingga di dapatkan keliling dari rodanya adalah sekitar 14,4 cm. Encoder yang mempunyai 50 increment persatu putaran. Dengan menggunakan rumus dalam jarak 40 cm didapatkan nilai dari encodernya adalah 138,5041. Sedangkan dengan mengukur dengan menggunakan enkodernya didapatkan hasil pengukuran rata-ratanya adalah 137,5.

Tabel 4.1 pengujian encoder

diameter = 4,6 cm										
jari - jari = 2,3 cm										
increment per 1 putaran = 50										
keliling lingkaran = 14,4 cm										
perhitungan dengan rumus untuk pengukuran dengan jarak 40 cm					pengkuran langsung dengan jarak 40 cm				error	error(%)
138,5041					138				0,00364	0,36396
138,5041					137				0,01086	1,085961
138,5041					136				0,01808	1,807961
138,5041					134				0,03252	3,251961
138,5041					137				0,01086	1,085961
138,5041					140				0,0108	1,08004
138,5041					140				0,0108	1,08004
138,5041					138				0,00364	0,36396
138,5041					138				0,00364	0,36396
138,5041					137				0,01086	1,085961
138,5041					137,5				0,00725	0,72496

Dari pengukuran yang telah dilakukan dapat lihat bahwa ada terjadi sedikit perbedaan antara menggunkan rumus dengan mengukur

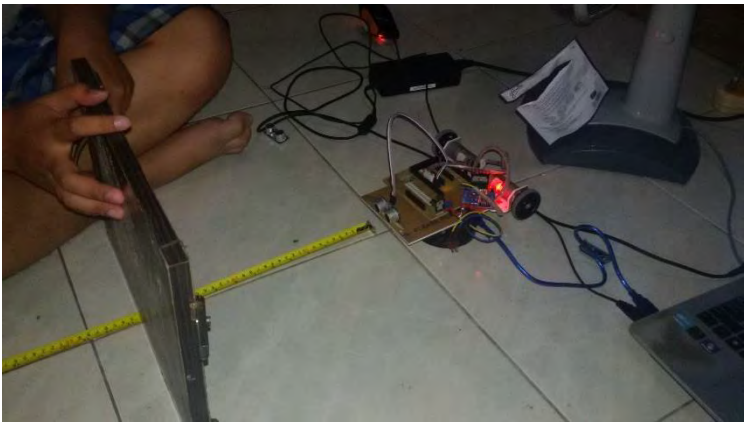
secara langsung. Error yang terjadi berasal dari 2 hal yaitu kesalahan dari enkoder itu sendiri dan kesalahan dari pengukur. Kesalahan dari pengukur terjadi karena pada saat pengukuran, jarak yang digunakan tidak lurus.

4.2 Pengujian Pengukuran Jarak

Pengukuran jarak ini menggunakan sensor ultrasonik HC-SR04 karakteristik dari sensor ini mempunyai resolusi 1cm, range baca mulai dari 2 cm sampai antara 400 – 500 cm, tegangan kerjanya 5V dan frekuensinya kerjanya 40kHz.

Tabel 4.2 pengujian sensor jarak

Jarak Asli(cm)	Jarak dari Sensor(cm)	error(%)
10	9	10,0
20	19	5,0
30	27	10,0
40	38	5,0
50	49	2,0
60	56	6,7
70	66	5,7
80	76	5,0
90	86	4,4
100	96	4,0
110	105	4,5
120	113	5,8
130	124	4,6
140	132	5,7
150	145	3,3
160	153	4,4
170	163	4,1
180	172	4,4
190	182	4,2
200	190	5,0
210	205	2,4
220	211,5	3,9
230	219,5	4,6
240	230	4,2
250	219	12,4
260	237	8,8
270	244,5	9,4
280	244	12,9
290	238	17,9
300	235	21,7



Gambar 4.1 pengujian sensor ultrasonik

Dari data di atas dapat dilihat bahwa error yang dihasilkan dari pembacaan sensor ultrasonik lumayan besar. Pada sensor ultrasonik di atas dapat dilihat *range* maksimum pembacaan dengan error yang lumayan kecil adalah pada jarak 200 cm. Sedangkan pada *datasheet*nya range maksimumnya adalah 400-500 cm. Sehingga dapat disimpulkan jarak maksimum efektif yang dapat dibaca oleh sensor ultrasonik HC-SR04 adalah 200 cm.

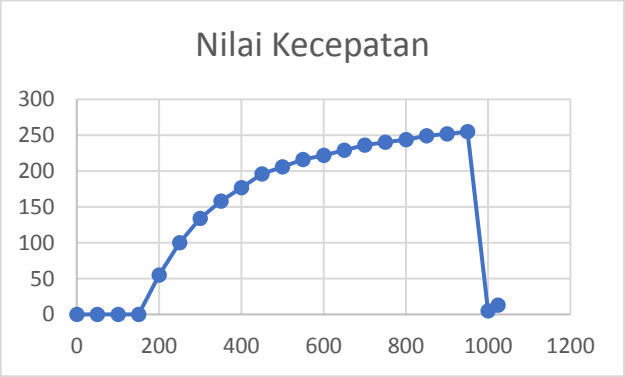
4.3 Pembacaan kecepatan motor DC

Motor DC ini merupakan motor DC 25GA370. Motor DC ini dilengkapi dengan encoder sebagai pembaca kecepatannya. Perhitungan kecepatan menggunakan rumus

$v = nilai\ encoder\ \times (40 \div 138)$ ..... (**4.1**)

Tabel 4.3 pengujian kecepatan motor

Nilai PWM	Nilai Kecepatan
1024	13
1000	5
950	255
900	252
850	249
800	244
750	240
700	236
650	229
600	222
550	216
500	206
450	196
400	177
350	158
300	134
250	100
200	55
150	0
100	0
50	0
0	0



Gambar 4.2 Grafik kecepatan motor

Dari data di atas dapat disimpulkan bahwa kenaikan kecepatannya tidak linier. Pada data terakhir terjadi error karena data yang dikirim dari arduino nano ke arduino mega telah melebihi 255 sehingga kembali ke angka 0 dan terjadi pengulangan.

4.4 Pengujian Alat

Pada bagian ini akan ditampilkan dan dijelaskan tentang pengujian yang telah dilakukan.

Pada pengujian kali ini ada 3 kecepatan berdasarkan pwmnya yaitu kecepatan cepat pwm yang digunakan 125, kecepatan cepat pwm yang digunakan 110, dan kecepatan lambat pwm yang digunakan 100.

Pengujian pada saaat PWM awal 125 seperti pada tabel 4.4 ada 4 jarak yang digunakan yaitu pada 200, 180, 110, dan 80. Pada jarak 200 fuzzybelum berkerja. Fuzzy mulai berkerja pada saat jarak 180. Pada saat jarak 200 rata-rata berhentinya adalah 33,8. Pada saat 180 rata-rata berhentinya ada 42. Pada saat 110 rata-rata berhentinya adalah 34. Pada saat 80 rata-rata berhentinya adalah 33,8. Rata-rata keseluruhannya berhenti adalah 35,9.

Tabel 4.4 pengujian pada saat pwm 125

Kecepatan pada saat pwm 49%						rata-rata
jarak mulai jalan		berhenti pada				
200	31	30	35	40	33	33,8
180	40	41	42	44	43	42
110	32	33	31	40	34	34
80	31	30	35	40	33	33,8

Pengujian pada saaat PWM awal 110 seperti pada tabel 4.5 ada 4 jarak yang digunakan yaitu pada 200, 180, 110, dan 80. Pada jarak 200 fuzzybelum berkerja. Fuzzy mulai berkerja pada saat jarak 180. Pada saat jarak 200 rata-rata berhentinya adalah 50. Pada saat 180 rata-rata berhentinya ada 41,8. Pada saat 110 rata-rata berhentinya adalah 43. Pada saat 80 rata-rata berhentinya adalah 46,6. Rata-rata keseluruhannya berhenti adalah 45,5.

Tabel 4.5 pengujian pada saat pwm 110

Kecepatan pada saat pwm 43%						rata-rata
jarak mulai jalan		berhenti pada				
200	51	50	50	52	51	50,8
180	40	41	42	42	44	41,8
110	40	45	50	36	44	43
60	46	47	46	47	47	46,6

Pengujian pada saaat PWM awal 105 seperti pada tabel 4.4 ada 4 jarak yang digunakan yaitu pada 200, 180, 110, dan 80. Pada jarak 200 fuzzybelum berkerja. Fuzzy mulai berkerja pada saat jarak 180. Pada saat jarak 200 rata-rata berhentinya adalah 51. Pada saat 180 rata-rata berhentinya ada 50,6. Pada saat 110 rata-rata berhentinya adalah 43,4. Pada saat 80 rata-rata berhentinya adalah 43. Rata-rata keseluruhannya berhenti adalah 47.

Tabel 4.6 pengujian pada saat pwm 105

Kecepatan pada saat pwm 39%						rata-rata
jarak mulai jalan		berhenti pada				
200	52	51	52	50	50	51
180	45	50	55	52	51	50,6
110	42	42	40	50	43	43,4
80	45	43	42	42	43	43

BAB V

PENUTUP

5.1 Kesimpulan

Pada penelitian ini telah dibuat sistem pengereman otomatis menggunakan kendali logika fuzzy. Pada sistem ini digunakan 2 motor DC yang dilengkapi dengan *rotary encoder*. Pembacaan nilai encoder yang ada pada motor DC 25GA370 cukup akurat dan presisi. Error yang dihasilkan adalah sekitar 1,11 % masih dalam batasan yang wajar. Sehingga tidak terlalu berpengaruh dalam kerja alat. Kalibrasi nilai kecepatan dengan encoder cukup baik. Pembacaan nilai dari sensor ultrasonik hanya akurat pada jarak antara 10-200 cm. Nilai error dari pembacaan antara 10-200 cm masih dibawah 5%. Pada saat pwm bernilai 49, 43, 39 % prototipe akan berhenti pada saat jarak rata-rata 35.9, 45.5, 47 cm secara berturut-turut.

5.2 Saran

Ada beberapa saran yang dapat ditambahkan ke dalam sistem ini agar berkerja lebih baik.

1. Sensor jarak yang digunakan dapat diganti dengan sensor yang lebih akurat dan presisi.
2. Sistem pengeremannya dapat ditambahkan pengereman mekanik untuk pengeremannya mendadaknya.
3. Jumlah *rules inputs* dan *rule decison* agar sistem dapat berjalan lebih mulus.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] Agus Purnama. Metode Pengereman Pada Motor Listrik http://elektronika-dasar.web.id/metode-pengereman-pada-motor-listrik/, Diakses oada tanggal 15 juli 2017

[2] Naziq, Ahmad. Pengertian Mikrokontroller – Informasi Anyar. https://sites.google.com/site/informasiterbarusekali/pengertian-mikrokontroller, Diakses pada tanggal 9 September 2016

[3] <URL : https://id.wikipedia.org/wiki/Arduino> 9 Septetmber 2016

[4] Hari Santoso. Sensor ultrasonik http://www.elangsakti.com /2015/05/sensor-ultrasonik.html, Diakses pada 1 Juni 2017

[5] Arwindra Rizqiawan. Sekilas Rotary encoder. https://konversi.wordpress.com/2009/06/12/sekilas-rotary-encoder/. Diakses pada tanggal 7 Mei 2017

[6] Andri Marzui. *Pulse Width Modulation*(PWM). http://andri_mz.staff.ipb.ac.id/pulse-width-modulation-pwm/. Diakses pada tanggal 1 Juni 2017

[7] Agus Purnama. Teori Motor DC Dan Jenis-Jenis Motor DC http://elektronika-dasar.web.id/teori-motor-dc-dan-jenis-jenis-motor-dc/. Diakses pada tanggal 1 Juni 2017.

[8] Agus Purnama. LCD (Liquid Cristal Display). http://elektronika-dasar.web.id/lcd-liquid- cristal-display/. Diakses pada tanggal 10 Juni 2017

[9] <URL : https://id.wikipedia.org/wiki/Jarak> 5 Juni 2017

[10] Arduino. https://www.arduino.cc/en/uploads/Main/Arduino Mega.jpg. Diakses Pada tanggal 18 Juli 2017

[11] Arduino. https://store.arduino.cc/usa/arduino-nano. Diakses pada tanggal 18 Juli 2017

Halaman ini sengaja dikosongkan

LAMPIRAN

POTONGAN PROGRAM ARDUINO MEGA

```
//*************************************************
**********
// Matlab .fis to arduino C converter v2.0.0.29032014
// - Karthik Nadig, USA
// Please report bugs to: karthiknadig@gmail.com
//*************************************************
**********

#include "fis_header.h"
#include <LiquidCrystal.h>
LiquidCrystal lcd(12,11,6,5,4,3);

// Number of inputs to the fuzzy inference system
const int fis_gcI = 2;
// Number of outputs to the fuzzy inference system
const int fis_gcO = 1;
// Number of rules to the fuzzy inference system
const int fis_gcR = 9;


// Declare variable
long duration, kecepatan, kecepatan1;
int distance, pwmOut, nilai, pwm;
byte motorPin[5] = {2,46,48,50,52}; // Enable, maju mundur
byte echoPin = A1, trigPin = A0;        // echo,trig

FIS_TYPE g_fisInput[fis_gcI];
FIS_TYPE g_fisOutput[fis_gcO];

// Setup routine runs once when you press reset:
void setup()
{
   nilai= analogRead(A10);
   pwmOut= nilai /4 ;
   delay(2000);
```

```
    // initialize the Analog pins for input.
    // Pin mode for Output: Motor
    for (int i=7; i<=13; i++){
      pinMode(i,OUTPUT);
    }
    lcd.begin(16,2);
    pinMode(trigPin,OUTPUT);
    pinMode(echoPin, INPUT);

    // initialize the Analog pins for output.
    // Pin mode for Output: output1
    pinMode( 2, OUTPUT);

    // Start Serial
    Serial.begin(115200, SERIAL_8N2);
    Serial1.begin(115200,SERIAL_8N2);

    // Initializing Vcc/Ground pin
    pinMode(motorPin[1],OUTPUT);
    pinMode(motorPin[2],OUTPUT);
    pinMode(motorPin[3],OUTPUT);
    pinMode(motorPin[4],OUTPUT);

    //digitalWrite(motorPin[1],LOW);
    //digitalWrite(motorPin[2],HIGH);
    //digitalWrite(motorPin[3],LOW);
    //digitalWrite(motorPin[4],HIGH);

}

// Loop routine runs over and over again forever:
void loop()
{

    int waktu = millis();

    // Read kecepatan

    // Read jarak
```

```
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in
microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance= duration*0.034/2;
  distance = constrain(distance,0,200);
  // Read Input: kecepatan
  g_fisInput[0] = kecepatan1;
  // Read Input: jarak
  g_fisInput[1] = distance;

  g_fisOutput[0] = 0;

  fis_evaluate();

  // Set output vlaue: output1
if (distance <= 50 && distance > 40)
{
pwmOut = 0;
mundur(pwmOut);
}
else if (distance <= 40 && distance >= 20)
{
pwmOut = 70;
mundur(pwmOut);
}
else if (distance < 20)
{
pwmOut = 0;
maju(pwmOut);
}
```

```
else if (distance <= 180 )
{
pwmOut = g_fisOutput[0];
maju(pwmOut);
}
else
{
maju(pwmOut);
}

//if (distance > 30 && pwmOut <= 65)
//{
  //pwm = 65;
  //maju(pwm);
//}

//if (distance = 50)
//{
//pwmOut = 80  ;
//mundur(pwmOut);
//}

    //pwmOut = (100 - (g_fisOutput[0]/1))*pwmOut/100;
    //analogWrite(motorPin[0] , pwmOut);

    // Print !!
    lcd.clear();
    lcd.setCursor(0,0); lcd.print("v: ");
    lcd.setCursor(2,0); lcd.print(kecepatan1);
    lcd.setCursor(5,0); lcd.print("cm/s");
    lcd.setCursor(0,1); lcd.print("s: ");
    lcd.setCursor(2,1); lcd.print(distance);
    lcd.setCursor(5,1); lcd.print("cm");
    lcd.setCursor(9,0); lcd.print("b: ");
    lcd.setCursor(12,0); lcd.print(pwm);
    lcd.setCursor(15,0); lcd.print("");
    lcd.setCursor(9,1); lcd.print("PWM: ");
    lcd.setCursor(13,1); lcd.print(pwmOut);
```

```
    Serial.print(kecepatan1);Serial.print(' ');
    Serial.print(distance); Serial.print(' ');
    Serial.print(g_fisOutput[0]); Serial.print("% ");
    Serial.print(pwmOut); Serial.print(' ');
    Serial.println(waktu/1000);
    //delay(100);

    if (Serial1.available() > 0){
      kecepatan = Serial1.read();
      Serial.println(kecepatan);
    }
    kecepatan1 = kecepatan;
}

void maju(int x)
{
  analogWrite(motorPin[0],x);
  digitalWrite(motorPin[1], HIGH);
  digitalWrite(motorPin[2], LOW);
  digitalWrite(motorPin[3], HIGH);
  digitalWrite(motorPin[4], LOW);
}
void mundur(int x)
{
  analogWrite(motorPin[0],x);
  digitalWrite(motorPin[1], LOW);
  digitalWrite(motorPin[2], HIGH);
  digitalWrite(motorPin[3], LOW);
  digitalWrite(motorPin[4], HIGH);
}


//*************************************************
**********
// Support functions for Fuzzy Inference System
```

```
//*****************************************************
*****************
// Triangular Member Function
FIS_TYPE fis_trimf(FIS_TYPE x, FIS_TYPE* p)
{
   FIS_TYPE a = p[0], b = p[1], c = p[2];
   FIS_TYPE t1 = (x - a) / (b - a);
   FIS_TYPE t2 = (c - x) / (c - b);
   if ((a == b) && (b == c)) return (FIS_TYPE) (x == a);
   if (a == b) return (FIS_TYPE) (t2*(b <= x)*(x <= c));
   if (b == c) return (FIS_TYPE) (t1*(a <= x)*(x <= b));
   t1 = min(t1, t2);
   return (FIS_TYPE) max(t1, 0);
}

FIS_TYPE fis_min(FIS_TYPE a, FIS_TYPE b)
{
   return min(a, b);
}

FIS_TYPE fis_max(FIS_TYPE a, FIS_TYPE b)
{
   return max(a, b);
}

FIS_TYPE    fis_array_operation(FIS_TYPE    *array,    int    size,
_FIS_ARR_OP pfnOp)
{
   int i;
   FIS_TYPE ret = 0;

   if (size == 0) return ret;
   if (size == 1) return array[0];

   ret = array[0];
   for (i = 1; i < size; i++)
   {
      ret = (*pfnOp)(ret, array[i]);
   }
```

```
   return ret;
}


//*****************************************************
*****************
// Data for Fuzzy Inference System
//*****************************************************
*****************
// Pointers to the implementations of member functions
_FIS_MF fis_gMF[] =
{
   fis_trimf
};

// Count of member function for each Input
int fis_gIMFCount[] = { 3, 3 };

// Count of member function for each Output
int fis_gOMFCount[] = { 3 };

// Coefficients for the Input Member Functions
FIS_TYPE fis_gMFI0Coeff1[] = { -44, 20, 84 };
FIS_TYPE fis_gMFI0Coeff2[] = { 36, 100, 164 };
FIS_TYPE fis_gMFI0Coeff3[] = { 116, 180, 244 };
FIS_TYPE*      fis_gMFI0Coeff[]      =      {      fis_gMFI0Coeff1,
fis_gMFI0Coeff2, fis_gMFI0Coeff3 };
FIS_TYPE fis_gMFI1Coeff1[] = { -48, 0, 48 };
FIS_TYPE fis_gMFI1Coeff2[] = { 40.2383531960997, 60, 79.9 };
FIS_TYPE fis_gMFI1Coeff3[] = { 72, 120, 168 };
FIS_TYPE*      fis_gMFI1Coeff[]      =      {      fis_gMFI1Coeff1,
fis_gMFI1Coeff2, fis_gMFI1Coeff3 };
FIS_TYPE** fis_gMFICoeff[] = { fis_gMFI0Coeff, fis_gMFI1Coeff
};

// Coefficients for the Input Member Functions
FIS_TYPE fis_gMFO0Coeff1[] = { 53.6, 64, 75.5633802816901 };
FIS_TYPE fis_gMFO0Coeff2[] = { 75, 77.1, 80.1 };
```

```
FIS_TYPE fis_gMFO0Coeff3[] = { 79.6, 90, 100.4 };
FIS_TYPE* fis_gMFO0Coeff[] = { fis_gMFO0Coeff1, fis_gMFO0Coeff2, fis_gMFO0Coeff3 };
FIS_TYPE** fis_gMFOCoeff[] = { fis_gMFO0Coeff };

// Input membership function set
int fis_gMFI0[] = { 0, 0, 0 };
int fis_gMFI1[] = { 0, 0, 0 };
int* fis_gMFI[] = { fis_gMFI0, fis_gMFI1};

// Output membership function set
int fis_gMFO0[] = { 0, 0, 0 };
int* fis_gMFO[] = { fis_gMFO0};

// Rule Weights
FIS_TYPE fis_gRWeight[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1 };

// Rule Type
int fis_gRType[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1 };

// Rule Inputs
int fis_gRI0[] = { 1, 1 };
int fis_gRI1[] = { 2, 1 };
int fis_gRI2[] = { 3, 1 };
int fis_gRI3[] = { 1, 2 };
int fis_gRI4[] = { 2, 2 };
int fis_gRI5[] = { 3, 2 };
int fis_gRI6[] = { 1, 3 };
int fis_gRI7[] = { 2, 3 };
int fis_gRI8[] = { 3, 3 };
int* fis_gRI[] = { fis_gRI0, fis_gRI1, fis_gRI2, fis_gRI3, fis_gRI4, fis_gRI5, fis_gRI6, fis_gRI7, fis_gRI8 };

// Rule Outputs
int fis_gRO0[] = { 1 };
int fis_gRO1[] = { 1 };
int fis_gRO2[] = { 1 };
int fis_gRO3[] = { 1 };
int fis_gRO4[] = { 2 };
```

```
int fis_gRO5[] = { 2 };
int fis_gRO6[] = { 2 };
int fis_gRO7[] = { 2 };
int fis_gRO8[] = { 3 };
int* fis_gRO[] = { fis_gRO0, fis_gRO1, fis_gRO2, fis_gRO3,
fis_gRO4, fis_gRO5, fis_gRO6, fis_gRO7, fis_gRO8 };

// Input range Min
FIS_TYPE fis_gIMin[] = { 20, 0 };

// Input range Max
FIS_TYPE fis_gIMax[] = { 180, 120 };

// Output range Min
FIS_TYPE fis_gOMin[] = { 64 };

// Output range Max
FIS_TYPE fis_gOMax[] = { 90 };

//*********************************************************
*****************
// Data dependent support functions for Fuzzy Inference System
//*********************************************************
*****************
FIS_TYPE fis_MF_out(FIS_TYPE** fuzzyRuleSet, FIS_TYPE x, int
o)
{
   FIS_TYPE mfOut;
   int r;

   for (r = 0; r < fis_gcR; ++r)
   {
      int index = fis_gRO[r][o];
      if (index > 0)
      {
         index = index - 1;
         mfOut          =          (fis_gMF[fis_gMFO[o][index]])(x,
fis_gMFOCoeff[o][index]);
      }
```

```
      else if (index < 0)
      {
         index = -index - 1;
         mfOut = 1 - (fis_gMF[fis_gMFO[o][index]])(x,
fis_gMFOCoeff[o][index]);
      }
      else
      {
         mfOut = 0;
      }

      fuzzyRuleSet[0][r] = fis_min(mfOut, fuzzyRuleSet[1][r]);
   }
   return fis_array_operation(fuzzyRuleSet[0], fis_gcR, fis_max);
}

FIS_TYPE fis_defuzz_centroid(FIS_TYPE** fuzzyRuleSet, int o)
{
   FIS_TYPE step = (fis_gOMax[o] - fis_gOMin[o]) /
(FIS_RESOLUSION - 1);
   FIS_TYPE area = 0;
   FIS_TYPE momentum = 0;
   FIS_TYPE dist, slice;
   int i;

   // calculate the area under the curve formed by the MF outputs
   for (i = 0; i < FIS_RESOLUSION; ++i){
      dist = fis_gOMin[o] + (step * i);
      slice = step * fis_MF_out(fuzzyRuleSet, dist, o);
      area += slice;
      momentum += slice*dist;
   }

   return ((area == 0) ? ((fis_gOMax[o] + fis_gOMin[o]) / 2) :
(momentum / area));
}

//**********************************************************
*****************
```

```
// Fuzzy Inference System
//*******************************************************
*****************
void fis_evaluate()
{
   FIS_TYPE fuzzyInput0[] = { 0, 0, 0 };
   FIS_TYPE fuzzyInput1[] = { 0, 0, 0 };
   FIS_TYPE* fuzzyInput[fis_gcI] = { fuzzyInput0, fuzzyInput1, };
   FIS_TYPE fuzzyOutput0[] = { 0, 0, 0 };
   FIS_TYPE* fuzzyOutput[fis_gcO] = { fuzzyOutput0, };
   FIS_TYPE fuzzyRules[fis_gcR] = { 0 };
   FIS_TYPE fuzzyFires[fis_gcR] = { 0 };
   FIS_TYPE* fuzzyRuleSet[] = { fuzzyRules, fuzzyFires };
   FIS_TYPE sW = 0;

   // Transforming input to fuzzy Input
   int i, j, r, o;
   for (i = 0; i < fis_gcI; ++i)
   {
      for (j = 0; j < fis_gIMFCount[i]; ++j)
      {
         fuzzyInput[i][j] =
            (fis_gMF[fis_gMFI[i][j]])(g_fisInput[i],
fis_gMFICoeff[i][j]);
      }
   }

   int index = 0;
   for (r = 0; r < fis_gcR; ++r)
   {
      if (fis_gRType[r] == 1)
      {
         fuzzyFires[r] = FIS_MAX;
         for (i = 0; i < fis_gcI; ++i)
         {
            index = fis_gRI[r][i];
            if (index > 0)
               fuzzyFires[r]          =          fis_min(fuzzyFires[r],
fuzzyInput[i][index - 1]);
```

```
            else if (index < 0)
                fuzzyFires[r] = fis_min(fuzzyFires[r], 1 - fuzzyInput[i][-
index - 1]);
            else
                fuzzyFires[r] = fis_min(fuzzyFires[r], 1);
        }
    }
    else
    {
        fuzzyFires[r] = FIS_MIN;
        for (i = 0; i < fis_gcI; ++i)
        {
            index = fis_gRI[r][i];
            if (index > 0)
                fuzzyFires[r]          =          fis_max(fuzzyFires[r],
fuzzyInput[i][index - 1]);
            else if (index < 0)
                fuzzyFires[r] = fis_max(fuzzyFires[r], 1 - fuzzyInput[i][-
index - 1]);
            else
                fuzzyFires[r] = fis_max(fuzzyFires[r], 0);
        }
    }

    fuzzyFires[r] = fis_gRWeight[r] * fuzzyFires[r];
    sW += fuzzyFires[r];
}

if (sW == 0)
{
    for (o = 0; o < fis_gcO; ++o)
    {
        g_fisOutput[o] = ((fis_gOMax[o] + fis_gOMin[o]) / 2);
    }
}
else
{
    for (o = 0; o < fis_gcO; ++o)
    {
```

```
            g_fisOutput[o] = fis_defuzz_centroid(fuzzyRuleSet, o);
        }
    }
}
```

LAMPIRAN PROGRAM ARDUINO NANO

```
int val;
int encoder0PinA = 3;
int encoder0PinB = 2;
int encoder0Pos = 0;
int encoder0PinALast = LOW;
int n = LOW,wkt,kec,encoder0PosLast;

void setup() {
  pinMode (encoder0PinA,INPUT);
  pinMode (encoder0PinB,INPUT);
  Serial.begin (9600);
}

void loop() {
  wkt = millis();
  n = digitalRead(encoder0PinA);
  if ((encoder0PinALast == LOW) && (n == HIGH)) {
    if (digitalRead(encoder0PinB) == LOW) {
      encoder0Pos++;
    }
    else {
      encoder0Pos--;
    }
  }
  if (wkt%100 == 0){
    kec = encoder0Pos-encoder0PosLast;
    Serial.write(kec*10);
    delay(1);
    encoder0PosLast = encoder0Pos;
  }
  encoder0PinALast = n;}
```

Halaman ini sengaja dikosongkan

BIOGRAFI PENULIS

Sony Nikodemus Limbong dilahirkan di Palangka Raya, 27 Maret 1995. Penulis memulai jenjang pendidikan di SDK St Yohanes Don Bosco Palangka Raya, melanjutkan ke SMPK St Paulus Palangka Raya. Kemudian melanjutkan ke SMAK Kolese Santu Yusup Malang. Penulis melanjutkan studi di Institut Teknologi Sepuluh Nopember jurusan Teknik Elektro dengan konsentrasi di bidang studi Elektronika. Alamat email yang penulis gunakan untuk keperluan sehari-hari adalah limbong103@gmail.com