

**SISTEM PEMOSISIAN SINAR UVC PADA ROBOT AUMR
MENGUNAKAN SENSOR MAGNET**

**POSITIONING SYSTEM UVC RAY ON THE AUMR ROBOT USING
MAGNETIC SENSOR**

TUGAS AKHIR

Disusun untuk memenuhi salah satu syarat untuk menyelesaikan
Program Studi S-1 Teknik Elektro

Disusun oleh :

Achmad Riyadi

1102174274



**FAKULTAS TEKNIK ELEKTRO
UNIVERSITAS TELKOM
BANDUNG
2021**

LEMBAR PENGESAHAN

TUGAS AKHIR

**SISTEM PEMOSISIAN SINAR UVC PADA ROBOT AUMR
MENGUNAKAN SENSOR MAGNET**

**POSITIONING SYSTEM UVC RAY ON THE AUMR ROBOT USING
MAGNETIC SENSOR**

Telah disetujui dan disahkan sebagai Tugas Akhir

Program Studi S-1 Teknik Elektro

Fakultas Teknik Elektro

Universitas Telkom

Oleh


Achmad Riyadi

1102174274

Bandung, 6 Agustus 2021

Pembimbing I

Pembimbing II



Angga Rusdinar, S.T., M.T., Ph. D.

NIP. 07740023

Yusuf Nur Wijayanto S.T, M, Eng, Ph.D

NIP. 19800002

ABSTRAK

Sinar UV (Ultraviolet) saat ini sangat dibutuhkan sebagai sterilisasi ruangan dimasa pandemi Covid-19 yang melanda seluruh dunia pada tahun 2020. Sinar UV yang digunakan untuk memberikan kerusakan kepada virus dan bakteri merupakan sinar UVC (Ultraviolet tipe C) yang dimana memiliki panjang gelombang 190-280 nm. UVC juga berbahaya bagi manusia maka penggunaan sinar UVC tersebut juga harus efektif dan tepat mengenai lingkungan sekitarnya.

Untuk dapat mengatasi masalah efektifitas dan ketepatan penyinaran sinar UVC, akan dirancang sebuah sistem yang berupa robot AUMR (*Autonomous UVC Mobile Robot*) yang dimana berfungsi untuk membawa lampu UVC tersebut untuk mensterilisasi ruangan. AUMR tersebut dapat berjalan otomatis mengikuti garis magnet berupa *magnetic tape* dengan dibaca oleh *magnetic sensor* serta menggunakan sensor *rotary encoder* untuk membaca posisi robot. Untuk pergerakan dan pemosisian robot digunakannya metode *fuzzy logic* untuk menyesuaikan posisi robot dengan *magnetic tape* dan jarak kapan lampu UVC harus dihidupkan.

Kata Kunci – UVC, *Autonomous UVC Mobile Robot*, *Fuzzy Logic*.

ABSTRACT

UV (Ultraviolet) rays are currently needed for room sterilization during the Covid-19 pandemic priode that hit the whole world in 2020. The UV rays used for give the damage to viruses and bacteria is UVC (Ultraviolet type C) rays which have a wavelength of 190-280 nm. UVC also dangerous for humans, so the application from UVC rays must be effective and precise on the surrounding environment

To be able to solve the problem of effectiveness and accuracy of UVC light irradiation, a system called AUMR (Autonomous UVC Mobile Robot) will be design which the functions to carry the UVC lamp to sterilize the room. The AUMR possible run automatically following a magnetic line in the form of a magnetic tape by being read by a magnetic sensor and also using a rotary encoder sensor to read the position of the robot. For the movement and positioning of the robot, the fuzzy logic method is used to adjust the movement and positioning the robot with magnetic tape and the distance when the UVC lamp must be turn on.

Keywords – UVC, Autonomous UVC Mobile Robot, Fuzzy Logic

KATA PENGANTAR

Assalamu'alaikum Wr.Wb.

Puji syukur kehadiran Allah SWT karena dengan rahmatnya saya sebagai penulis dapat menyelesaikan Tugas Akhir dengan judul Sistem Pemosisian Sinar UVC Pada Robot AUMR Menggunakan Sensor Magnet. Buku ini disusun sebagai salah satu syarat dalam menyelesaikan pendidikan S1 Teknik Elektro pada Fakultas Teknik Elektro Universitas Telkom.

Pada proses dan hasil dari penulisan buku ini disadari bahwa terdapat banyak kesalahan dan kekurangan didalamnya. Maka oleh karena itu sangat diharapkan kritik dan saran yang dapat membangun penulis sangatlah penting.

Diharapkan dengan hadirnya buku ini dapat bermanfaat untuk pembaca dan dapat memberi ilmu serta dikembangkan menjadi lebih baik.

Wassalamu'alaikum Wr.Wb

Bandung, 6 Agustus 2021

Achmad Riyadi

UCAPAN TERIMA KASIH

Segala puji bagi Allah SWT, karena berkat ridho-Nya saya sebagai penulis dapat menyelesaikan Tugas Akhir ini. Dengan adanya kelebihan dan kekurangan dalam Tugas Akhir ini penulis tidak lupa untuk mengucapkan terima kasih telah menerima dan membantu dengan baik kepada berbagai pihak dan mohon maaf jika tidak tersebut dalam ucapan ini. Adapun pihak-pihak yang diucapkan terima kasih oleh penulis adalah sebagai berikut:

1. Allah SWT, karena ridho dan rahmat-Nya saya sebagai penulis Insha Allah dapat menyelesaikannya dengan baik.
2. Kepada kedua orang tua, Bapak Syafrudin H.A dan Mamah Almh.Nunghayati serta Kaka, Sri Hermawati, Puput Safitri, dan Ari Apriyandi yang selalu memberikan dukungan baik materi dan non-materi dengan memberikan motivasi dan doa dalam setiap kegiatan, sehingga penulis dapat sampai pada jenjang ini.
3. Bapak Muhammad Zakiyullah Romdlony, S.T., M.T., Ph.D selaku dosen wali yang dimana sangat membantu dalam melaksanakan kegiatan perkuliahan dengan memberikan saran dan bantuan yang sangat berarti dan tidak tergantikan.
4. Bapak Angga Rusdinar, S.T., M.T., Ph.D selaku dosen pembimbing I yang telah memberikan kesempatan, ilmu, saran dan dukungan sehingga penulis dapat menyelesaikan tugas akhir ini.
5. Bapak Yusuf Nur Wijayanto, S.T., M,Eng, Ph.D selaku dosen pembimbing II yang telah memberikan arahan, kesempatan, ilmu dan bimbingan dalam menyelesaikan tugas akhir ini.
6. Seluruh Dosen yang telah memberikan pembelajaran dan ilmu kepada penulis baik akademik maupun non-akademik.
7. Kepada sdr. Risnanda Satriatama, S.T. selaku senior yang telah membantu dalam memberikan arahan dan saran dalam menyelesaikan tugas akhir ini.
8. Seluruh asisten Laboratorium EIRRG yang dimana sebagai tempat penulis mendapatkan implementasi ilmu, tempat belajar, bercerita, berbagi, dan

istirahat. Segala kegiatan yang dilakukan bersama rekan-rekan sangat berharga.

9. Seluruh keluarga ROOBICS is Fum, terlebih pada tim ROOBICS 2019 dan 2020, bang risnanda, bang rully, bang azka, tama, adnan, felix, dea, wisnu, faisal, nurul. Terima kasih telah berjuang untuk bersama dan selalu *break the limit* untuk mengharumkan Telkom University dalam Kontes Robot Indonesia.
10. Rekan-rekan ESC yang telah bersama dari 2018 dari peserta menjadi panitia sungguh pengalaman dan cerita yang tidak bisa dilupakan dengan segala kejadian suka dan duka.
11. Kepada Kelas EL-41-04 yang dimana merupakan kelas yang sangat menakjubkan yang selalu membantu dan memberikan dukungan kepada penulis dimasa perkuliahan.
12. Kepada Putri Ananda Nadilla telah hadir menjadi tempat cerita dan menjadi seseorang yang sangat bisa diandalkan.
13. Kosan Burung, Bayu, Akmal, Athar, Alfi, Risqi, dan Teguh sebagai tempat cerita dan bercanda dikala tidak ada perkuliahan.
14. Rekan-rekan Narutindo, bang Firman, bang Risannda, bang Arif, adnan, tama, widya yang telah membantu dan menjadi tempat untuk mengembangkan diri serta membangun mimpi bersama dengan memabangun dan mengembangkan Start-Up Narutindo.
15. Telkom University karena telah memfasilitasikan sarana dan prasarana kegiatan akademik maupun non-akademik kepada penulis yang mencukupi.

Semoga Allah SWT membalas semua kebaikan yang telah diberikan secara langsung maupun tidak langsung kepada semua pihak yang telah membantu dalam menyelesaikan perkuliahan dan Tugas Akhir ini. Aamiin.

DAFTAR ISI

LEMBAR PENGESAHAN.....	ii
ABSTRAK	iii
ABSTRACT	iv
KATA PENGANTAR	v
UCAPAN TERIMA KASIH	vi
DAFTAR ISI	viii
DAFTAR GAMBAR	xi
DAFTAR TABEL	xii
DAFTAR SINGKATAN	xiii
BAB I PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Rumusan Masalah.....	2
1.3. Tujuan	2
1.4. Manfaat	2
1.5. Batasan Masalah	2
1.6 Metode Penelitian	3
1.7 Sistematika Penulisan	4
BAB II TINJAUAN PUSTAKA	5
2.1. Prinsip Kerja Konsep	5
2.2. <i>Autonomous UVC Mobile Robot</i>	5
2.3. Fuzzy Logic Controller (FLC)	6
2.4. <i>Differential Drive and Forward Kinematics</i>	9
2.4.1 <i>Forward Kinematics for Differential Drive</i>	10
2.5. Sensor Magnet	11
2.6. Rotary Encoder	12
2.7. Mikrokontroler	13
2.8. Motor DC	14

BAB III	PERANCANGAN SISTEM.....	15
3.1	Desain Sistem	15
3.1.1	Diagram Blok	15
3.1.2	Diagram Alir	16
3.2	Desain Perangkat Keras	18
3.2.1	Desain Mekanik Sistem	19
3.2.2	Spesifikasi Komponen	19
3.2.2.1	Arduino Mega	19
3.2.2.2	Magnetic Sensor MGS1600GY.....	20
3.2.2.3	<i>Incremental Rotary Encoder</i> E4056-600-3T-24	21
3.2.2.4	Driver Motor BTS7960.....	22
3.2.2.5	Brushed Motor K90 60W.....	22
3.2.2.6	Baterai Aki 12V 18Ah	23
3.2.2.7	Baterai Aki 12V 70Ah.....	24
3.2.2.8	Inverter 900VA 12V	25
3.2.2.9	UBEC 5V 3A	25
3.2.2.1	Relay 2 Channel	26
3.2.2.2	Flysky FS-I6S.....	27
3.3	Desain Sistem Perangkat Lunak.....	27
3.3.1	Perancangan FLC (Fuzzy Logic Controller).....	27
3.3.2	Perancangan <i>Forward Kinematics for Differential Drive</i>	30
BAB IV	HASIL DAN ANALISA	31
4.1	Pengambilan Data.....	31
4.2	Pengujian <i>Incremental Rotary Encoder</i>	31
4.3	Pengujian Sensor <i>Magnetic Sensor</i> MGS1600GY	33
4.4	Pengujian <i>Fuzzy Logic Control</i> pada Matlab dan Robot	34

4.5	Pengujian <i>Forward Kinematic</i> terhadap lintasan	36
4.6	Pengujian Sinar UVC Menggunakan UV Meter	37
BAB V KESIMPULAN DAN SARAN.....		39
5.1	Kesimpulan.....	39
5.2	Saran	40
DAFTAR PUSTAKA		41
LAMPIRAN		43
❖	Lampiran 1 (Source Code).....	43
❖	Lampiran 2 Penurunan Rumus FLC.....	81
❖	Lampiran 3 Penurunan persamaan <i>Forward Kinematics of Differential Drive</i>	89
❖	Lampiran 4 Gambar alat	90

DAFTAR GAMBAR

Gambar II-1: Diagram Fungsi Sistem	5
Gambar II-2: Robot AUMR	6
Gambar II-3: Proses Fuzzy Logic Controller	7
Gambar II-4: Diagram fungsi keanggotaan segitiga fuzzy	8
Gambar II-5: Diagram fungsi keanggotaan trapesium fuzzy	8
Gambar II-6: <i>Differential Drive Kinematics</i>	9
Gambar II-7: <i>Forward Kinematics for Differential Drive</i>	11
Gambar II-8: Sensor Magnet MGS1600GY	11
Gambar II-9: Sistem kerja rotary encoder.....	13
Gambar II-10: Mikrokontroler ATmega 2566	14
Gambar III-1: Diagram Blok Sistem	15
Gambar III-2: Diagram Blok Kontrol Posisi Robot	16
Gambar III-3: Diagram Alir Sistem Utama	17
Gambar III-4: Diagram Alir Proses	18
Gambar III-5: Desain Mekanisme Robot AUMR	19
Gambar III-6: Mikrokontroler Arduino Mega	19
Gambar III-7: Magnetic Sensor MGS1600GY	20
Gambar III-8: Incremental Rotary Encoder	21
Gambar III-9: Driver Motor BTS7960	22
Gambar III-10: Motor DC K9D60N2	22
Gambar III-11: Aki 12V 18Ah	23
Gambar III-12: Aki 12V 70Ah	24
Gambar III-13: Inverter 900VA Input 12V	25
Gambar III-14: DC to DC UBEC 5V 3A	25
Gambar III-15: Relay 5V 2 Channel	26
Gambar III-16: FlySky FS-I6S	27
Gambar III-17: Output Nilai Sensor	28
Gambar III-18: Fungsi Keanggotaan error (e)	28
Gambar III-19: Fungsi Keanggotaan delta error (Δe)	29
Gambar III-20: Fungsi Keanggotaan keluaran PWM_	29

DAFTAR TABEL

Tabel III-1: Mekanik Sistem	19
Tabel III-2: Spesifikasi Arduino Mega	19
Tabel III-3: Spesifikasi Magnetic Sensor MGS1600GY	20
Tabel III-4: Spesifikasi Incremental Rotary Encoder	21
Tabel III-5: Spesifikasi Driver Motor BTS7960	22
Tabel III-6: Spesifikasi Motor DC K9D60N2	23
Tabel III-7: Spesifikasi Aki 12V 18Ah	24
Tabel III-8: Spesifikasi Aki 12V 17Ah	24
Tabel III-9: Spesifikasi Inverter 900VA Input 12V	25
Tabel III-10: Spesifikasi UBEC 5V 3A	25
Tabel III-11: Spesifikasi Relay 5V 2 Channel	26
Tabel III-12: Spesifikasi FlySky FS-I6S	27
Tabel III-13: Rules FLC	28
Tabel IV-1: Pengujian <i>Incremental Rotary Encoder</i>	31
Tabel IV-2: Pengujian Keluaran FLC Pada Matlab dan MotorDC	34
Tabel IV-3: Pengujian UVC Terhadap Waktu	37

DAFTAR SINGKATAN

UVC	: Ultraviolet <i>type C</i>
AUMR	: <i>Autonomous UVC Mobile Robot</i>
FLC	: <i>Fuzzy Logic Controller</i>
Kg	: Kilogram
DNA	: Deoxyribonucleic acid
LED	: Light Emitting Dioda
DC	: Direct Current
PWM	: Pulse Width Modulation
Ah	: Ampere Hour
W	: Watt
V	: Voltage
KHz	: Kilo Hertz
MHz	: Mega Hertz
Rpm	: Revolution per Minute
PCB	: Printed Circuit Board
KB	: Kilo Byte
Nm	: Newton Meter
VAC	: Voltage Alternating Current
nm	: nano meter
cm	: Centi Meter
V _r	: Velcoity Right
V _l	: Velocity Left
ICC	: <i>Instantaneous Center of Curvature</i>
$\mu\text{W}/\text{cm}^2$: mikroWatt per-Centi meter kuadrat
BPFK	: Balai Pengaman Fasilitas Kesehatan

BAB I

PENDAHULUAN

1.1. Latar Belakang

Pada tanggal 2 Maret 2020 merupakan kasus COVID-19 pertama di Indonesia yang dimana menjadikan kecemasan publik, kemudian peningkatan setiap harinya semakin tinggi terhitung dalam waktu satu bulan yaitu pada tanggal 31 Maret 2020 sudah terdapat 1528 kasus positif yang terkonfirmasi[3].

UVC atau disebut Ultraviolet tipe C, merupakan sebuah lampu yang memiliki panjang gelombang 190-280 nm yang dimana dapat memiliki dampak kerusakan DNA pada virus dan bakteri[1]. Sehingga UVC merupakan salah satu alternatif untuk melakukan sterilisasi ruangan dengan tidak meninggalkan bekas secara langsung terhadap lingkungan sekitar. Tetapi dibalik manfaat yang dimiliki oleh UVC tersebut terdapat dampak buruk bagi manusia jika terpapar langsung dengan radiasi sinar UVC tersebut dapat memicu kanker kulit pada manusia [1][2]. Maka dari pada itu penggunaan sinar UVC perlu dijalankan dari jarak jauh salah satunya dengan robot.

Penggunaan robot ataupun alat untuk menggantikan fungsi manusia yang dimana jika UVC dibawa oleh manusia dapat mengakibatkan kanker kulit[1]. Maka robot pembawa lampu UVC ini menjadi riset untuk membantu penanganan COVID-19, maka di buatnya AUMR (*Autonomous UVC Mobile Robot*) yang dimana berfungsi untuk membawa 6 buah lampu UVC 36 watt dan 1 buah UVC 10 watt yang dapat dikontrol secara otomatis. Dalam proses sterilisasi ruangan, robot AUMR memerlukan jarak yang baik serta waktu yang tepat agar efektif yang dimana membutuhkan waktu 10-15 menit. Komponen dan yang digunakan untuk mengoptimalkan posisi robot adalah dengan menggunakan sensor Magnet MGS1600 dan dengan metode kontrol *Fuzzy Logic*[3].

Pengerjaan Tugas Akhir ini, merupakan rancangan bagaimana memposisikan robot AUMR yang membawa lampu UVC sebagai pemancar dari sinar UVC dengan menggunakan sensor magnet dengan metode kontrol *fuzzy logic*. Hasil dari sistem ini dapat digunakan sebagai inovasi untuk efektifitas robot AUMR.

1.2. Rumusan Masalah

Rumusan masalah yang dibahas dalam Tugas Akhir adalah sebagai berikut:

1. Bagaimana mengolah data sensor magnet untuk navigasi AUMR
2. Bagaimana mengimplementasikan metode Fuzzy Logic dengan sensor magnet pada robot AUMR
3. Bagaimana perancangan pengiriman data dari robot ke *microkontroller*.
4. Bagaimana menghitung jarak efektifitas sinar UVC dengan waktu yang dibutuhkan untuk memberikan kerusakan DNA pada virus.

1.3. Tujuan

1. Melakukan pengambilan nilai dari *Magnetic Tape* dengan menggunakan sensor magnet.
2. Mengolah nilai dari sensor magnet yang didapat dengan menggunakan metode kontrol *Fuzzy Logic*.
3. Data akan diterima oleh *microcontroller* yang sudah terintegrasi dengan sensor magnet.
4. Melakukan pengambilan data jarak menggunakan sensor *rotary encoder* pada motor DC.

1.4. Manfaat

Manfaat dari tugas akhir ini adalah semoga dapat memberikan manfaat bagi penanganan COVID-19 maupun universitas sebagai berikut:

1. Dapat digunakan sebagai sumber informasi dan menjadi referensi yang nantinya ada untuk pengembangan ilmu pengetahuan
2. Dapat menjadi sebuah tolak ukur pada bidang-bidang yang sesuai dengan program studi yang telah dipelajari
3. Dapat sebagai inovasi untuk percepatan penanganan COVID-19 yang akan membantu tenaga kesehatan.

1.5. Batasan Masalah

Membatasi cakupan pembahasan masalah pada Tugas Akhir ini untuk memperjelas pembahasan. Maka diberikan batasan-batasan sebagai berikut:

1. Robot yang digunakan merupakan robot AUMR yang memiliki bentuk dan ukuran sebagai mana mestinya dengan menggunakan sensor Magnet MGS1600.
2. Robot diuji pada ruangan *indoor* ber-lokasi di gedung D, BTP.
3. Robot menggunakan *Mikrocontroller Arduino Mega* sebagai pemrosesannya.
4. Metode atau Algoritma menggunakan *Fuzzy Logic*.
5. Dalam perhitungan jarak robot menggunakan sensor *Rotary Incremental Encoder*

1.6 Metode Penelitian

Metode penelitian pada proses penyelesaian Tugas Akhir ini adalah sebagai berikut:

1. Identifikasi Masalah
Mencari dan melakukan identifikasi masalah yang ada berdasarkan hasil pengamatan secara langsung berdasarkan topik permasalahan.
2. Analisa Sistem dan Komponen
Mempelajari dan menganalisa sistem serta komponen dengan mengaplikasikan berdasarkan masalah yang akan dibahas.
3. Studi Literatur
Pencarian sumber-sumber teori dari riset atau inovasi sebelumnya yang berkaitan tentang masalah yang dibahas serta mempelajari studi literatur tentang komponen yang dibutuhkan seperti:
 - Mempelajari Sensor Magnet MGS1600
 - Mempelajari Fuzzy Logic
 - Mempelajari Mikrokontroler ATmega2566
 - Mempelajari UVC
 - Mempelajari parameter Rotary Incremental Encoder
4. Pembuatan Alat
Pembuatan alat yang dimana menggunakan robot AUMR yang sebelumnya sudah jadi kemudian ditambahkan sensor magnet MGS1600 dan untuk kontrol navigasinya menggunakan metode Fuzzy Logic.
5. Pengujian Sistem

Pada tahap ini pengujian sistem berguna untuk mengetahui target serta hasil dan kesimpulan dari penggunaan sensor magnet MGS1600 dan metode kontrol Fuzzy Logic dalam memposisikan sinar UVC pada robot AUMR.

1.7 Sistematika Penulisan

Sistematika penulisan untuk Tugas Akhir ini dibagi menjadi beberapa bagian yang dimana adalah sebagai berikut:

1. **BAB I**

Pada bab ini berisikan penjelasan bagaimana gambaran isi dari Tugas Akhir ini yang dimana terdiri dari latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah, metode penelitian, dan sistematika penulisan pada Tugas Akhir ini,

2. **BAB II**

Pada bab ini membahas tentang teori-teori dasar yang mendukung yang dimana sebagai penunjang Tugas Akhir ini.

3. **BAB III**

Pada bab ini membahas desain sistem yang dimana di dalamnya menjelaskan tentang gambaran cara kerja sistem, spesifikasi komponen yang digunakan, dan perancangan perangkat lunak.

4. **BAB IV**

Pada bab ini berisikan hasil dari analisa dan evaluasi dari perancangan dari sistem yang dibuat yaitu Sistem Pemosisian Sinar UVC Pada Robot AUMR Menggunakan Sensor Magnet.

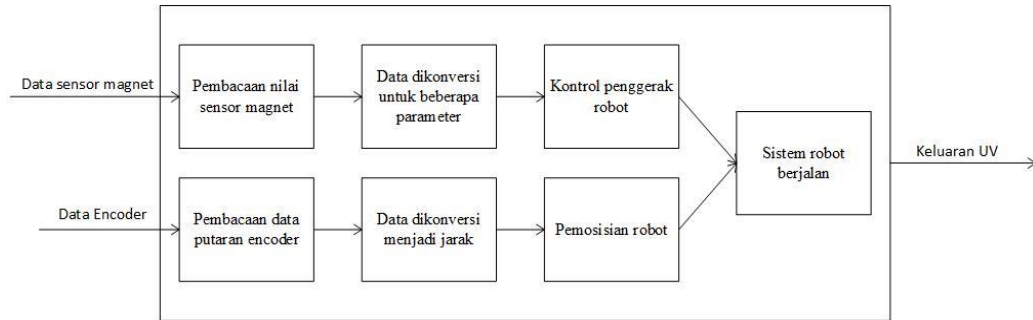
5. **BAB V**

Pada bab ini berisikan kesimpulan dan saran, kesimpulan berisikan hasil dari kerja yang dilakukan dan saran untuk memberikan perbaikan ataupun pengembangan selanjutnya.

BAB II

TINJAUAN PUSTAKA

2.1. Prinsip Kerja Konsep



Gambar II-1: Diagram Fungsi Sistem

Prinsip kerja dalam sistem pemosisian sinar UVC pada robot AUMR menggunakan sensor magnet. Berikut penjelasannya.

1. Robot diaktifkan (secara manual), kemudian secara otomatis sensor magnet akan aktif dan menerima *raw* data dari *magnetic tape*.
2. Setelah data terbaca dikirimkan ke mikrokontroller bagian sensor yang dimana berfungsi untuk mengolah data dari setiap sensor yang digunakan.
3. Sambil robot berjalan ada juga sensor *rotary encoder* yang membaca putaran roda untuk menghitung jarak tempuh robot.
4. Setelah semua data sensor diolah dan dikirim dari mikrokontroller sensor ke bagian mikrokontroller master dibagian ini diberikan perintah pergerakan robot dan kontrol robot.
5. Setiap 6 meter robot akan berhenti untuk menyalahkan lampu UVC yang dimana jarak tersebut dibaca dari sensor *rotary encoder*

2.2. *Autonomous UVC Mobile Robot*

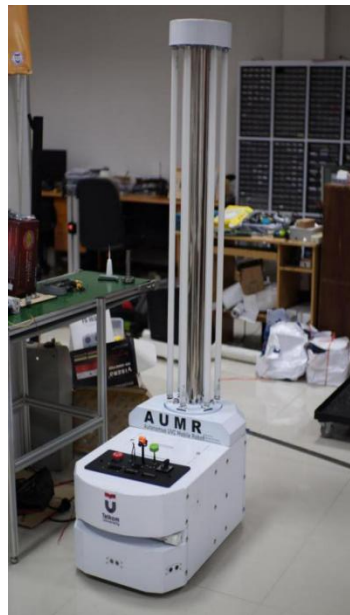
Autonomous UVC Mobile Robot (AUMR) merupakan sebuah *mobile robot* yang dimana membawa 7 lampu ultraviolet tipe C dengan panjang gelombang setiap lampu UVC 190-280 nm yang dimana dapat menonaktifkan DNA dari bakteri dan virus yang dimana dilakukan ujicoba dengan virus[1]. AUMR umumnya digunakan sebagai robot disinfektan ruangan menggunakan lampu ultraviolet tipe C dan tidak memerlukan operator untuk mengemudikannya karena

sinar ultraviolet tersebut sangat bahaya jika terpapar langsung dengan manusia yang dimana dapat menyebabkan kanker kulit. AUMR juga umumnya digunakan pada rumah sakit rujukan COVID-19, perkantoran serta beberapa institusi yang membutuhkan sterilisasi ruangan setelah digunakan.

AUMR telah dilakukan tes alat kesehatan di BPFK (Balai Pengaman Fasilitas Kesehatan) dengan hasil pada jarak 2 meter didapatkan radiasi dari UVC tersebut bernilai $154 \mu\text{W}/\text{cm}^2$ dengan minimal radiasi $40 \mu\text{W}/\text{cm}^2$.

AUMR mempunyai beberapa tipe navigasi yang dimana diantaranya :

1. Tipe *magnetic tape*, pada tipe ini digunakannya *magnetic tape* atau sebuah garis yang berbahan magnet dilantai sebagai referensi nilai dari sensor magnet yang dipasang dirobot dengan memiliki 2 kutub selatan dan utara dari sudut ini bisa diatur untuk *marker* dan *line tracking*.
2. Tipe Laser, pada tipe navigasi ini menggunakan *Laser Range Finder* yang dimana terdapat sensor berupa LiDAR yang akan mengeluarkan laser yang nantinya akan dipancarkan ke titik-titik tertentu kemudian pantulan dari titik-titik tersebut akan memberikan gambaran bagi AUMR untuk memetakan lingkungan yang nantinya dapat digambarkan berupa map.



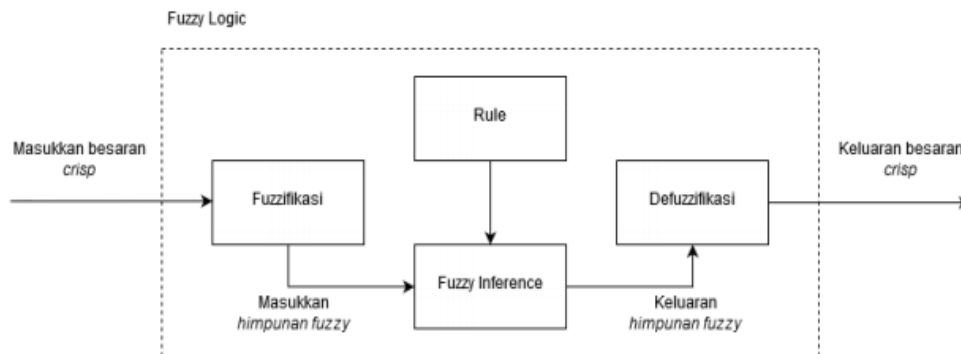
Gambar II-2: Robot AUMR

2.3. Fuzzy Logic Controller (FLC)

Fuzzy Logic Controller (FLC) adalah logika yang dimana digunakan untuk

menjelaskan hal yang samar dalam pengambilan keputusan, yang dimana himpunan *fuzzy* diperkenalkan oleh Lotfi A. Zadeh pada tahun 1965. Pada teori himpunan *fuzzy* terdapat derajat keanggotaan 0 hingga 1 yang dimana pada daerah 0 hingga 1 merupakan daerah yang disebut samar. Masukan pada *fuzzy logic* berupa *error* (e) posisi dan *delta error* (Δe) [6]. (Pada himpunan *fuzzy* setiap anggota himpunan memiliki derajat keanggotaan, yang dimana merupakan nilai yang menunjukkan seberapa besar tingkat keanggotaan pada suatu elemen (x) dalam suatu himpunan A [4].

Fuzzy logic controller dibagi menjadi empat elemen, *fuzzyfication*, *rule*, *fuzzy inference*, *defuzzification* yang dapat dilihat pada Gambar II-3.



Gambar II-3: Proses *Fuzzy Logic Controller*

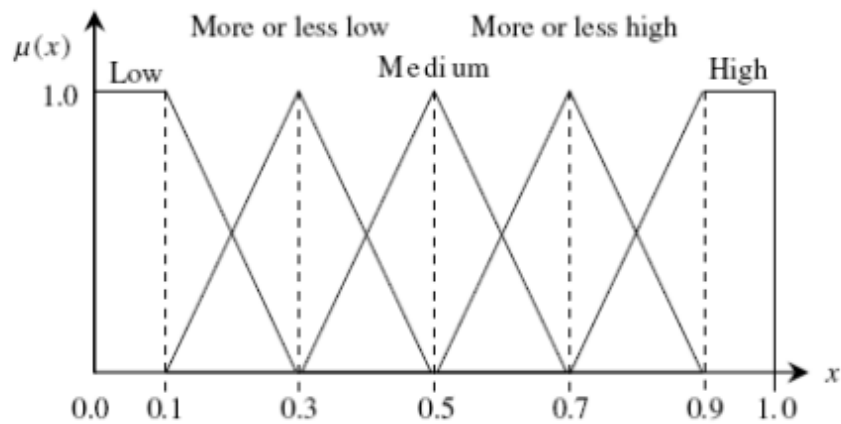
Gambar ... merupakan blok diagram dari proses *fuzzy logic controller*, yang dimana hanya ada 3 proses utama yaitu *fuzzyfication*, *fuzzy inference*, dan *defuzzification*[4].

2.3.1. *Fuzzyfication*

Pada proses *fuzzyfication* nilai yang masuk berupa nilai tegas (*crisp*) kemudian dikonversikan menjadi bentuk variabel linguistik berdasarkan fungsi keanggotaannya. Terdapat beberapa fungsi keanggotaan himpunan *fuzzy* dan yang paling sering digunakan adalah segitiga, dan trapesium. Berikut contoh fungsi keanggota himpunan *fuzzy*.

a. Fungsi segitiga

Pada himpunan *fuzzy* fungsi segitiga mengilustrasikan bangun datar segitiga yang dimana fungsi segitiga dapat dilihat pada Gambar II-4.

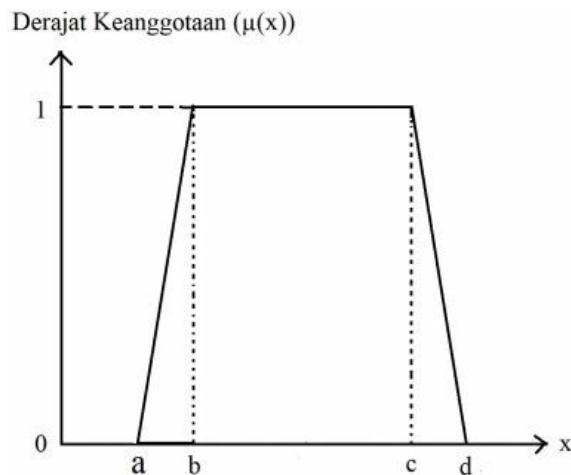


Gambar II-4: Diagram fungsi keanggotaan segitiga *fuzzy* [8]

Gambar II-4 fungsi segitiga dimana $\mu(x)$ menunjukkan derajat keanggotaan, dan nilai 0.0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0 merupakan nilai dari domain x .

b. Fungsi Trapesium

Pada himpunan *fuzzy* fungsi trapesium mengilustrasikan bangun datar segitiga yang dimana fungsi segitiga dapat dilihat pada Gambar II-5.



Gambar II-5: Diagram fungsi keanggotaan trapesium *fuzzy* [4]

Gambar II-5 fungsi trapesium dimana $\mu(x)$ menunjukkan derajat keanggotaan, dan a, b, c, d , merupakan nilai dari domain x .

2.3.2. Fuzzy Inference

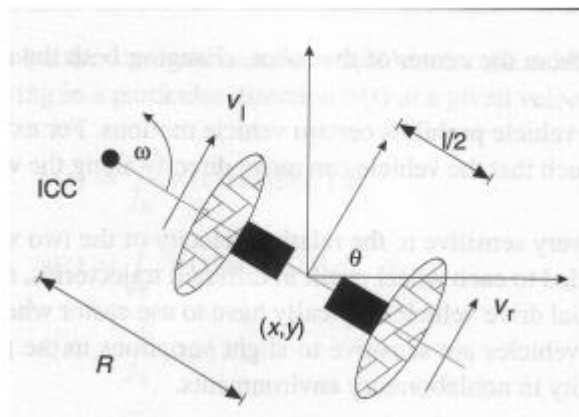
Fuzzy inference system adalah kerangka komputasi yang dimana didasari pada rule himpunan *fuzzy* dan aturan yang berupa IF-THEN. Masukan pada *fuzzy inference* berupa nilai tegas (*crisp*).

2.3.3. Defuzzification

Defuzzification adalah tahap terakhir pada FLC mengubah keluaran nilai *fuzzy* menjadi nilai tegas (*crisp*)[9]. Output yang akan digunakan pada *defuzzification* pada sistem ini berupa nilai PWM untuk mengendalikan kecepatan motor DC

2.4. Differential Drive and Forward Kinematics

Differential drive adalah sebuah mekanisme dari *mobile robot* yang dimana terdapat 2 roda pada sumbu yang sama, dimana setiap roda dapat digerakan secara independen baik ke depan maupun ke belakang[11]. Pada *differential drive* terdapat parameter seperti ICC (*Instantaneous Center of Curvature*) yang dimana merupakan titik dimana robot melakukan rotasi. Contoh *differential drive kinematics* dapat dilihat pada Gambar II-6.



Gambar II-6: *Differential Drive Kinematics*[11]

Terdapat juga parameter *Velocity* yang dimana merupakan variasi untuk kecepatan dua roda tersebut, dari memvariasikan kecepatan pada roda tersebut dapat juga untuk memvariasikan lintasan yang telah dilalui oleh robot. Karena laju putaran pada roda sama dengan ICC untuk kedua roda maka dapat Persamaan II-1 berikut:

$$\begin{aligned}\omega (R + l/2) &= V_r \\ \omega (R + l/2) &= V_l\end{aligned}\tag{II-1}$$

Pada persamaan diatas 1 adalah jarak antara pusat kedua roda dan V_r , V_l merupakan kecepatan kedua roda kanan dan kiri, dan R adalah jarak antara titik ICC ke titik tengah antara roda[11]. Didapatkan untuk Persamaan II-2 R dan ω sebagai berikut.

$$\omega = \frac{V_r + V_l}{W} \quad (II-2)$$

$$R = \frac{W}{2} \cdot \frac{V_l + V_r}{V_r - V_l}$$

Terdapat tiga kasus yang digunakan untuk menentukan pergerakan robot sebagai berikut:

1. Jika $V_l = V_r$, maka didapatkan gerakan maju dalam garis lurus. R menjadi tak terbatas[11].
2. Jika $V_l = -V_r$, maka $R = 0$ dan didapatkan rotasi[11].
3. Jika $V_l = 0$, maka didapatkan rotasi dari roda kiri[11].

2.4.1 Forward Kinematics for Differential Drive

Pada Gambar II-, asusmsikan robot pada posisi (x,y) yang sama. Menuju ke arah yang membuat sudut θ dengan sumbu x . Dengan memanipulasi *velocity* dari V_r dan V_l , didapatkan robot bergerak ke posisi dan orientasi yang berbeda[11].

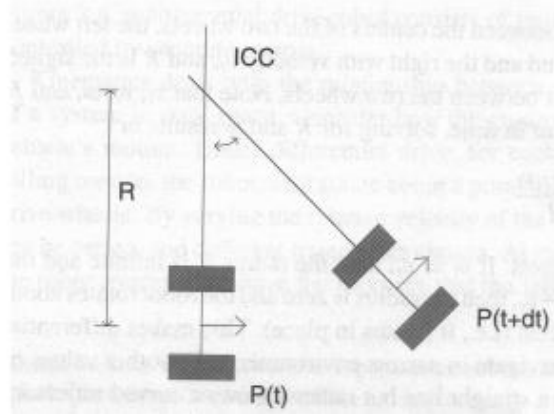
Setelah didapatkannya R , dan robot dalam posisi (x,y,θ) kita bisa menggunakan persamaan 3 kasus pada *differential drive* dan didapatkan ICC dengan Persamaan II-3[12]:

$$ICC = [x - R \sin(\theta), y + R \cos(\theta)] \quad (II-3)$$

Dapat diketahui berapa lama (δt) robot dapat bergerak dengan kecepatan yang sama antara dua roda, kemudian dilakukan menghitung posisi robot yang baru (X', Y', θ') yang dimana adalah sebuah rotasi dengan ICC yang telah didapatkan sebelumnya[12]. Perubahan posisi (X', Y', θ') didefinisikan pada Persamaan II-4 sebagai berikut:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos(\omega\delta t) & -\sin(\omega\delta t) & 0 \\ \sin(\omega\delta t) & \cos(\omega\delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - ICC_x \\ y - ICC_y \\ \theta \end{bmatrix} + \begin{bmatrix} ICC_x \\ ICC_y \\ \omega\delta t \end{bmatrix} \quad (II-4)$$

Pada Persamaan II-3 menjelaskan tentang pergerakan robot yang berputar sejauh jarak R pada ICC dengan kecepatan sudut dari ω [11]. Dapat dilihat pada Gambar II-7:



Gambar II-7: *Forward Kinematics for Differential Drive*[11]

2.5. Sensor Magnet

Sensor Magnet adalah sebuah pendeteksi medan magnet yang berasal dari sebuah medan magnet yang dimana disini digunakannya *magnetic tape* yang dapat menghasilkan medan magnet yang akan dibaca oleh sensor magnet. Disini menggunakan sensor magnet MGS1600GY.



Gambar II-8: Sensor Magnet MGS1600GY [5]

Pada sensor magnet MGS1600GY jarak maksimal antara sensor dan *magnetic tape* adalah 25mm dan kutub antara sensor dan manetic tape harus berbeda antara utara dan selatan ataupun selatan dan utara. konektor yang digunakan adalah DB-15 yang dimana terdapat I/O yang berupa parameter-parameter yang akan digunakan untuk pendeteksian serta pembacaan nilai *magnetic tape* [5], berikut parameter-parameter yang digunakan:

- PWM Out

PWM Out harus selalu aktif digunakan untuk membawa semua informasi sensor, termasuk deteksi lebar pita untuk pengontrol kecepatan motor[5].

- Analog Out

Analog Out harus selalu aktif berfungsi untuk memberikan informasi posisi tape ke sensor dengan cara memvariasikan tegangan 1.5V saat tape berada di tengah sensor, 0 dan 3V saat tape berada di salah satu ujung sensor[5].

- Track Present

Track Present berfungsi untuk mendeteksi *magnetic tape* yang dimana akan mengeluarkan informasi berupa keluaran 5V jika sensor membaca *magnetic tape* dan memberikan keluaran 0V jika keluar dari jangkauan *magnetic tape* [5].

- Left and Right Markers Output

Markers berfungsi untuk memberikan nilai berbeda dari *magnetic tape* yang digunakan sebagai tanda dan dipasang sama dengan kutub sensor

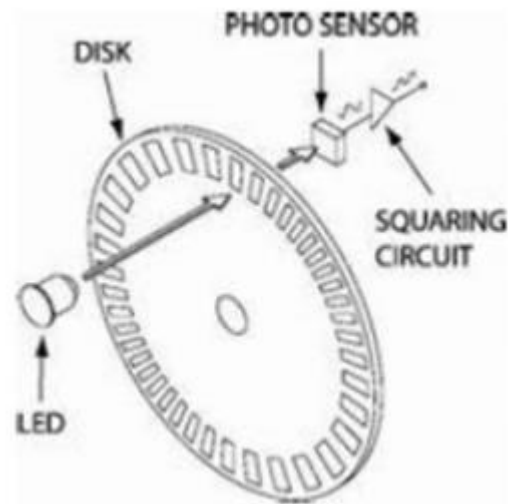
- Fork Left and Fork Right Inputs[5].

Fork digunakan untuk memilih *magnetic tape* bagian kiri atau kanan yang harus menjadi keluaran pada PWM dan analog, digunakan untuk percabangan[5].

2.6. Rotary Encoder

Rotary Encoder merupakan sebuah perangkat elektromekanik yang dapat memonitor gerakan posisi[7]. Umumnya para *rotary encoder* terdapat optik atau hall sensor untuk menghasilkan nilai pulsa yang dimana dikonversi menjadi posisi, gerakan, dan arah. Pada *rotary encoder* terdapat sebuah disk dan detektor yang dimana disk merupakan sebuah piringan berlubang dan detektor berfungsi untuk mendeteksi dengan cara memanfaatkan dari disk yang berputar tersebut[4].

Terdapat dua jenis *rotary encoder* yaitu *absolute rotary encoder* dan *incremental rotary encoder*. *Incremental rotary encoder* adalah jenis *rotary encoder* yang digunakan yang dimana memiliki kelebihan lebih sedikit detektor sehingga penggunaan pin pada mikrokontroler akan semakin sedikit juga. Pada *incremental rotary encoder* memiliki dua kanal detektor, kanal A dan B. Diantaranya merupakan detektor yang berupa LED dan berupa phototransistor untuk menerima cahaya LED tersebut.



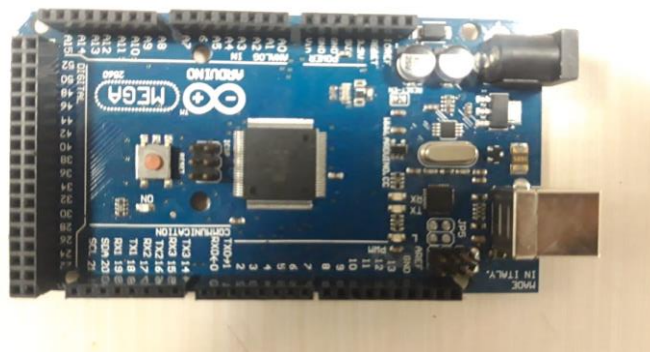
Gambar II-9: Sistem kerja *rotary encoder* [4]

Pada Gambar II-7 terdapat LED dan phototransistor yang saling berhadapan digunakan untuk, saat disk berputar dan LED memancarkan cahaya disaat itu phototransistor menerima cahaya dari LED melalui lubang dari *disk* dan menghasilkan phototransistor bernilai 1. kemudian saat cahaya terhalangi oleh putaran selanjutnya maka phototransistor bernilai 0, sehingga menghasilkan gelombang persegi. dari nilai tersebut menghasilkan pulsa informasi yang dapat dikonversi dan diolah menjadi arah, jarak, dan posisi[4].

2.7. Mikrokontroler

Mikrokontroler adalah sebuah perangkat berupa chip yang berfungsi untuk melakukan pengontrolan sistem, yang dimana terdapat komponen pendukung untuk menjalankan chip tersebut. Mikrokontroler biasa digunakan sebagai prototype sistem dan jarang sekali ditemukan pada sebuah industri. Mikrokontroler terdiri dari Memori, I/O, CPU (*Central Processing Unit*), dan ADC (*Analog-to-Digital Converter*)[10].

Penggunaan mikrokontroler disini disesuaikan dengan kebutuhan alat yang dimana dibutuhkannya 54 digital pin dan 4 serial pin untuk komunikasi antar dua mikrokontroler yang diintegrasikan dengan sensor magnet MGS1600GY, relay, sensor *rotary encoder*, Motor DC. maka dari pada itu dengan kebutuhan dan fitur tersebut mikrokontroler yang cocok adalah mikrokontroler ATmega2566 yang dimana memiliki 54 digital pin dan 4 fitur komunikasi serial. Dikarenakan ATmega2566 tidak bisa berdiri sendiri maka dari pada itu dibutuhkannya komponen pendukung atau disebut sistem minimum yang dirangkai dengan kebutuhan dan menjadi sebuah *board* mikrokontroller.



Gambar II-10: Mikrokontroler ATmega 2566

Mikrokontroller ini berfungsi untuk kontrol utama yang dimana menerima masukan data sensor magnet, mengendalikan kecepatan motor DC, *rotary encoder*, dan kontrol kecepatan motor DC menggunakan *Fuzzy Logic Controller* (FLC)

2.8. Motor DC

Motor DC adalah sebuah penggerak ataupun aktuator yang digunakan sebagai penggerak utama robot AUMR. Motor DC bekerja ketika dialirkan arus sehingga kumparan didalamnya akan menghasilkan medan magnet. Medan magnet yang dihasilkan tersebut akan menghasilkan gaya putar yang berlawanan dengan kutub medan magnet dan akan terjadi secara terus menerus. untuk mengatur arah putaran motor DC tergantung pada arah aliran arus yang akan mengubah arah medan magnetnya. kecepatan motor DC bergantung dengan tegangan yang diberikan, tegangan yang digunakan pada motor DC pada AUMR adalah 24V dikarenakan berat AUMR yang mencapai 100kg diperlukannya spesifikasi motor DC yang mempunyai kebutuhan tersebut.

BAB III

PERANCANGAN SISTEM

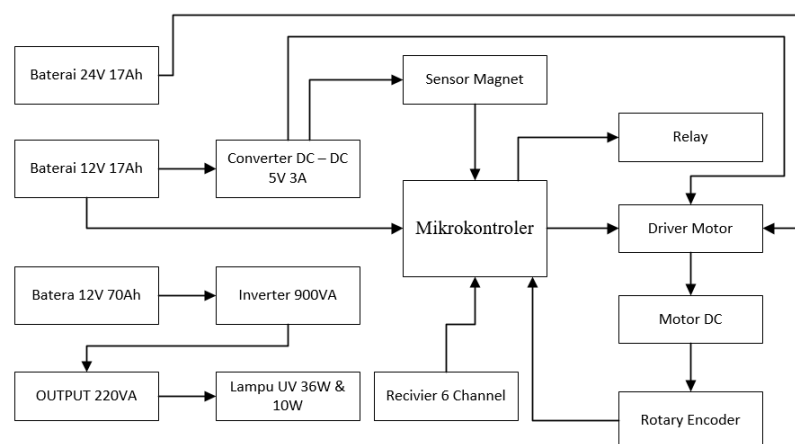
3.1 Desain Sistem

Pada bab ini membahas bagaimana perancangan desain sistem pemosisian sinar UVC pada robot AUMR berbasis sensor magnet. Pembahasan disini mengenai desain sistem, desain perangkat keras, desain perangkat lunak.

Pertama dari sistem ini yaitu pembacaan *magnetic tape* oleh sensor magnet MGS1600GY yang dimana berfungsi sebagai navigasi. Pada navigasi tersebut nilai dari *magnetic tape* dibandingkan dengan nilai *set point* untuk mendapatkan nilai *error* berupa posisi, nilai *error* tersebut akan diolah menggunakan FLC. Keluaran yang dihasilkn untuk kontrol kecepatan pada motor DC. Kemudian menentukan posisi robot dengan sensor *rotary encoder* yang dimana pada setiap 6 meter robot akan berhenti dan menyalahkan lampu UVC selama 15 menit dengan efektifitas sinar UVC adalah 3 meter.

3.1.1 Diagram Blok

Secara keseluruhan sistem dapat dilihat dengan diagram blok seperti yang ditunjukkan digambar berikut:



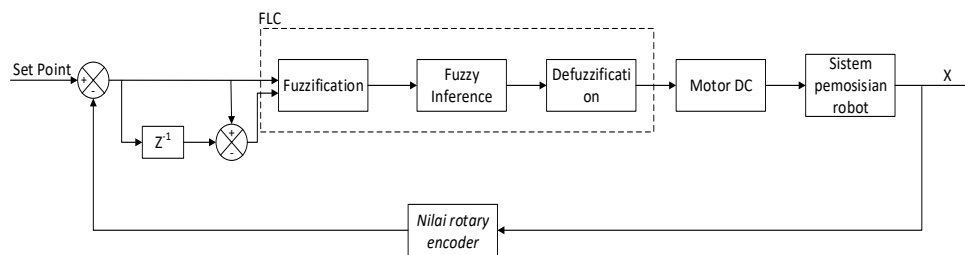
Gambar III-1: Diagram Blok Sistem

Pada sistem ini menggunakan tiga buah catu daya dari baterai aki yang dimana 12V 17Ah untuk catu daya sistem kemudian dibagi untuk diturunkan

menjadi 5V 3A untuk catu daya sensor dan driver motor, 12V 70Ah untuk catu daya inverter dan lampu UVC, dan catu daya 24V 17Ah untuk catu daya Motor DC.

Mikrokontroler disini di fungsikan sebagai kontrol utama yang dimana menerima masukan dari sensor magnet, sensor *rotary encoder*, dan masukan dari *recivier 6 channel* untuk remot kontrol robot. Mikrokontroler tidak hanya menerima tapi juga mengelola keluaran yaitu relay, PWM dan direksi untuk putaran motor DC yang dikontrol melalui driver motor.

Masukan yang diterima dengan metode FLC adalah *error* (e) yang dimana merupakan seberapa jauh posisi simpangan yang melenceng dari *set point* yang sudah ditetapkan, dan *delta error* (Δe) posisi yang dimana merupakan selisih antara *error* terhadap *error* sebelumnya. Dimana pada FLC disini keluaran yang akan dikontrol adalah nilai PWM. Berikut diagram blok kontrol posisi robot dengan metode FLC pada Gambar...

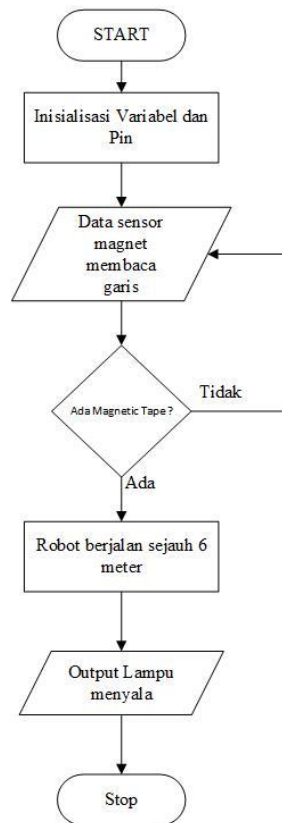


Gambar III-2: Diagram Blok Kontrol Posisi Robot

Nilai awal merupakan sebuah *set point* yang telah ditentukan. Nilai dari *rotary encoder* pada motor DC pada diagram blok merupakan sebuah *feedback*, dan juga terdapat blok FLC yang didalamnya terdapat *fuzzyfication*, *fuzzy inference*, dan *defuzzyfication*.

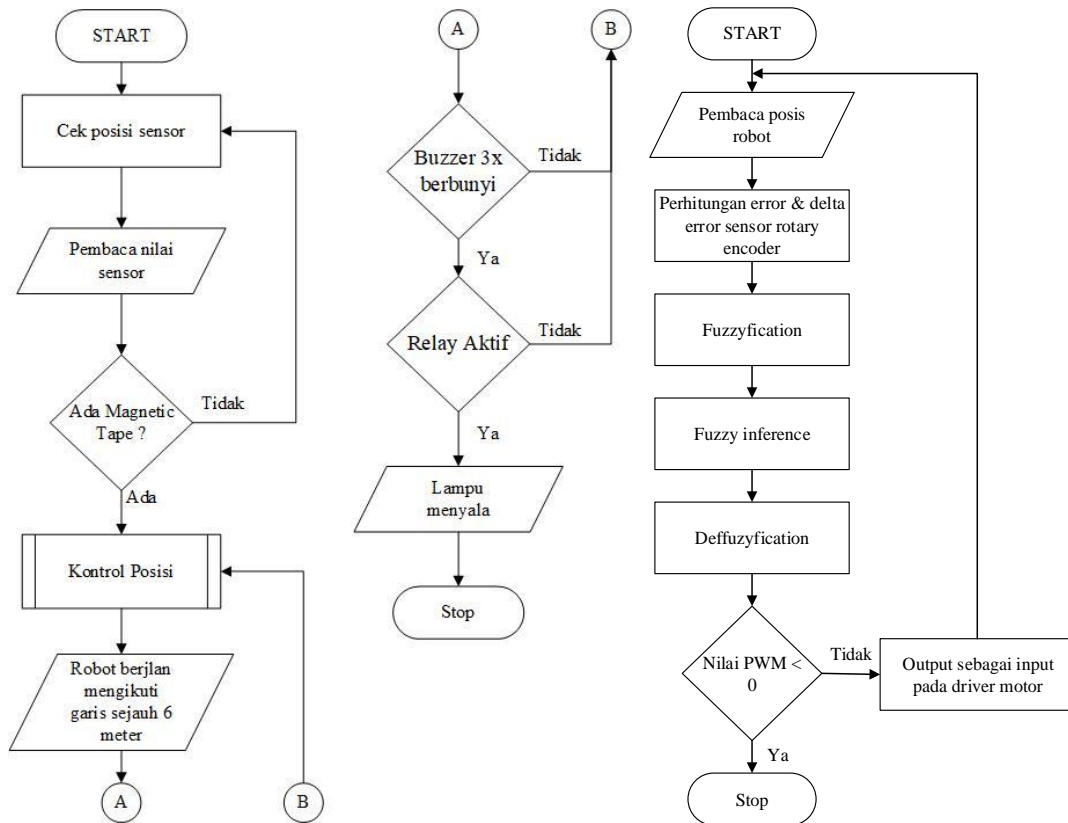
3.1.2 Diagram Alir

Diagram alir sistem yang telah dirancang dilihat pada Gambar III-3 sebagai berikut.



Gambar III-3: Diagram Alir Sistem Utama

Diagram alir sistem utama ini proses pertama dilakukan insialisai variabel dan pin dari parameter-parameter yang digunakan dari sensor. Kemudian sensor magnet membaca nilai dari *magnetic tape* yang dikirimkan ke mikrokontroler. Jika terdapat *magnetic tape* maka robot akan berjalan sejauh 6 meter yang dimana diketahui dengan sensor *rotary encoder* dari putran motor DC maka robot akan berhenti dan lampu akan menyala, kemudian jika tidak terbaca *magnetic tape* robot akan kembali ke step pembacaan *magnetic tape* kembali.



Proses Aktivasi Robot

Proses Kontrol

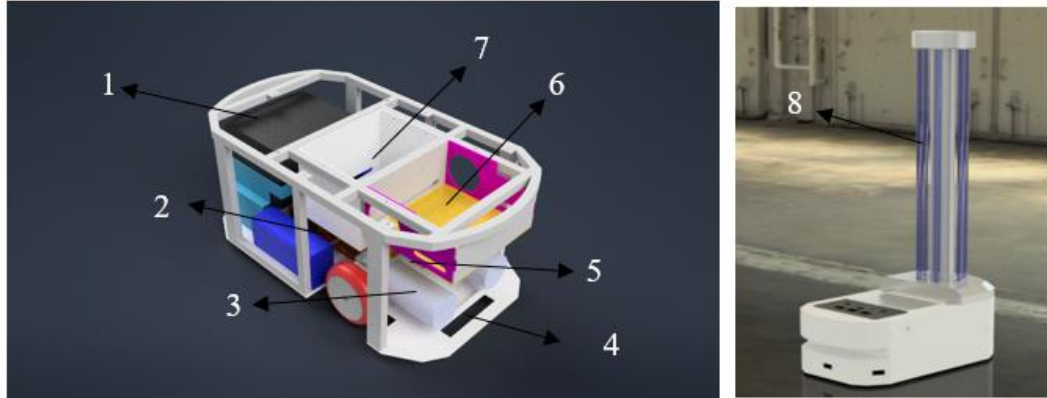
Gambar III-4: Diagram Alir Proses

3.2 Desain Perangkat Keras

Desain perangkat keras disini meliputi desain mekanik sistem robot, spesifikasi komponen pada sisistem, desain dan spesifikasi elektronika sistem. Penjelasan sebagai berikut:

3.2.1 Desain Mekanik Sistem

Berikut Gambar III-5 merupakan gambar dari desain 3D serta pemosisian komponen pada robot AUMR serta penjelasan di Tabel III-1.



Gambar III-5: Desain Mekanisme Robot AUMR

Tabel III-1: Mekanik Sistem

No.	Keterangan
1	Penempatan <i>Accu</i> 12V 70Ah
2	Inverter 900VA 12V
3	Motor DC <i>Brushed</i> 60W 3000Rpm
4	Penempatan sensor magnet MGS1600GY
5	<i>Rotary Encoder</i>
6	Kotak elektronika, berisi PCB
7	Kotak <i>Accu</i> 12V 18Ah
8	Lampu UVC 36W

Dilihat dari Gambar III-5, terdapat 2 buah motor DC untuk penggerak yang dimana didalamnya sudah terpasang sensor *rotary encoder* yang dimana ketika motor bergerak maka encoder juga akan bergerak.

3.2.2 Spesifikasi Komponen

Pada spesifikasi komponen berisi komponen-komponen yang digunakan berdasarkan kebutuhan yang telah disebutkan diawal.

3.2.2.1 Arduino Mega

Arduino Mega merupakan sebuah *board* mikrokontroler dengan *chip* ATmega2566. Berikut gambar dari Arduino Mega dan spesifikasi pada Tabel...



Gambar III-6: Mikrokontroller Arduino Mega

Tabel III-2: Spesifikasi Arduino Mega

Tegangan Input	7 – 12V (<i>recommended</i>)
Tegangan Operasi	5V
Digital I/O	54 (15 PWM)
Pin Input Analog	16
Flash Memory	256 KB, dan 8 KB digunakan untuk bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

3.2.2.2 Magnetic Sensor MGS1600GY

Magnetic Sensor MGS1600GY merupakan sebuah sensor yang membaca medan magnet yang dipancarkan oleh *magnetic tape*. Bentuk ilustrasi sensor magnet dengan *magnetic tape* -nya ditunjukkan oleh Gambar III-7 :



Gambar III-7: *Magnetic Sensor MGS1600GY*

Pada magnetic sensor jenis MGS1600GY terdapat beberapa parameter yang digunakan. Dijelaskan dalam Tabel III-3.

Tabel III-3: Spesifikasi *Magnetic Sensor* MGS1600GY

Tegangan masukan	4.5V ke 30V DC
Fork Right	Memilih track kanan
Fork Left	Memilih track kiri
Analog Out	0-3V (1.5V posisi tengah) nilai analog sesuai posisi <i>track</i> ke sensor
PWM Out	Keluaran PWM berdasarkan posisi <i>track</i>
Left Marker	Pendeteksian Left Marker
Right Marker	Pendeteksian Right Marker
Track Present	Pendeteksian Track

Dari Tabel III-3 tersebut Tegangan masukan yang digunakan adalah 5V karena ketika digunakan 12V dan ada penurunan kapasitas baterai nilai pembacaan sensor akan berubah.

3.2.2.3 *Incremental Rotary Encoder* E4056-600-3T-24

Incremental Rotary Encoder dimana digunakan sebagai pembacaan posisi robot ketika bergerak jenis *Incremental Rotary Encoder* yang digunakan adalah E4056-600-3T-24. *Incremental Rotary Encoder* E4056-600-3T-24 dapat dilihat pada Gambar III-8.



Gambar III-8: *Incremental Rotary Encoder*

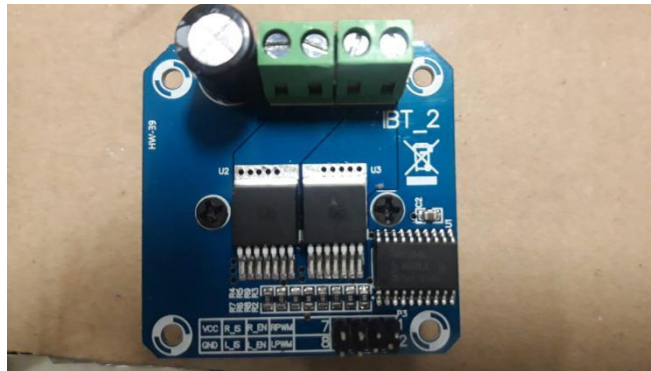
Untuk Spesifikasi *Incremental Rotary Encoder* E4056-600-3T-24 dapat dilihat pada Tabel III-4.

Tabel III-4: Spesifikasi *Incremental Rotary Encoder*

Tegangan Input	12V – 24V
Resolusi	600 <i>counter/Rotation</i>
Kanal	Kanal A dan B

3.2.2.4 Driver Motor BTS7960

Driver motor BTS7960 adalah sebuah pengatur masukan tegangan untuk sebuah Motor DC. Driver motor BTS7960 dapat dilihat pada Gambar III-9 dan spesifikasinya pada Tabel III-5.



Gambar III-9: Driver Motor BTS7960

Tabel III-5: Spesifikasi Driver Motor BTS7960

Tegangan Masukan	6V – 27V DC
Maksimum Arus	43 Ampere
Over-Voltage	Lock Out
Under-Voltage	Shut Down
Control Mode	PWM or level

3.2.2.5 Brushed Motor K90 60W

Motor DC yang digunakan pada robot AUMR adalah dua buah Brush DC produksi GGM dengan seri K9D60N2. Berikut Tabel III-6 spesifikasi dan Gambar III-10 dari Motor DC K9D60N2.



Gambar III-10: Motor DC K9D60N2

Tabel III-6: Spesifikasi Motor DC K9D60N2

Keluaran	60W
Tegangan masukan	24V
Kecepatan	3000 rpm
Torsi	0.19 Nm
Torsi Awal	2.73 Nm
Arus	4.6A
Arus Awal	60A

Spesifikasi tersebut cocok dengan beban dari robot itu sendiri yang dimana masa dari robot tersebut lebih dari 100kg

3.2.2.6 Baterai Aki 12V 18Ah

Baterai Aki 12V dengan kapasitas 18Ah digunakan sebagai catu daya untuk menghidupkan sistem yang dimana masukan sistem membutuhkan 12V yang akan ke mikrokontroler dan sensor, kemudian untuk mendapatkan sumber tegangan 24V dua buah baterai di seri untuk keperluan Motor DC. Seri baterai yang digunakan adalah VRLA SMT1218 yang dimana dapat dilihat pada Gambar III-11 dan spesifikasinya pada Tabel III-7 sebagai berikut:



Gambar III-11: Aki 12V 18Ah

Tabel III-7: Spesifikasi Aki 12V 18Ah

Keluaran Tegangan	12V
Kapasitas	18Ah
Resistansi	$\leq 16m\Omega$

3.2.2.7 Baterai Aki 12V 70Ah

Baterai Aki 12V 70Ah diperuntuhkan untuk catu daya inverter yang dimana bebannya adalah 6 buah lampu dengan daya $36W = 216W$ serta 1 buah lampu dengan daya 10W. Seri aki yang digunakan adalah VRLA AGM Kijo 70Ah-12V UPS yang dimana dapat dilihat dari Gambar III-12 dan spesifikasi pada Tabel III-8 sebagai berikut:



Gambar III-12: Aki 12V 70Ah

Tabel III-8: Spesifikasi Aki 12V 17Ah

Keluaran Tegangan	12V
Kapasitas	70Ah
Resistansi	$\leq 7.0m\Omega$

3.2.2.8 Inverter 900VA 12V

Inverter 900VA 12V digunakan untuk mengubah DC menjadi AC yang dimana diperlukan untuk menghidupkan lampu UVC dengan beban 226W. Seri inverter yang digunakan adalah Inverter UPS Zelio 900VA 12V yang dimana terdapat pada Gambar III-13 dan spesifikasi pada Tabel III-9 sebagai berikut:



Gambar III-13: Inverter 900VA *Input* 12V

Tabel III-9: Spesifikasi Inverter 900VA *Input* 12V

Masukan Tegangan	12V DC
Daya	900VA
Keluaran Tegangan	220V AC
Pengaman	Alarm jika, <i>Overload</i> , Baterai Habis, salah pengkabelan, <i>Short Circuit</i> , <i>Over temperature</i>

3.2.2.9 UBEC 5V 3A

UBEC 5V merupakan komponen untuk menurunkan catuan daya dari baterai 12V menjadi 5V dan menghasilkan arus 3A untuk kebutuhan sensor dan driver motor. Seri yang digunakan adalah 3 Amp Switch-Mode UBEC yang dapat dilihat pada Gamba III-14 dan spesifikasinya pada Tabel III-10.



Gambar III-14: DC to DC UBEC 5V 3A

Tabel III-10: Spesifikasi UBEC 5V 3A

Keluaran Tegangan dan Arus	5V/3A
Masukan Tegangan	5.5V-25V

3.2.2.1 Relay 2 Channel

Relay 2 Channel digunakan sebagai saklar otomatis untuk menghidupkan lampu UVC dari jarak jauh disini digunakan sebanyak digunakannya 2 channel dikarenakan kebutuhan 2 jenis lampu UVC bagian depan dan belakang. Seri relay yang digunakan berupa Relay Modul 2 Channel SPDT 5pin Outocoupler yang dimana dapat dilihat pada Gambar III-15 dan spesifikasi pada Tabel III-11.



Gambar III-15: Relay 5V 2 Channel

Tabel III-11: Spesifikasi Relay 5V 2 Channel

Masukan Tegangan kontrol Relay	5V – 7.5V
Maksimal masukan Tegangan dan Arus pada Relay	250VAC 10A/ 30V DC 10A

Pengaman	Optocoupler
----------	-------------

3.2.2.2 Flysky FS-I6S

Flysky FS-I6S merupakan remote kontrol yang digunakan untuk mengontrol robot dari jarak jauh. komunikasi yang digunakan berupa radio dan pada remote ini digunakan juga untuk mengganti mode otomatis dan manual. Flysky FS-I6S Dapat dilihat pada Gambar III-16 dan spesifikasinya pada Tabel III-12 berikut.



Gambar III-16: FlySky FS-I6S

Tabel III-12: Spesifikasi FlySky FS-I6S

Cahnnel	6
Radio Frekuensi	2.408 – 2.475GHz
Radio Frekuensi Power	Kurang dari 20 dBm
Bandwidth	500KHz
Power Input	4.3V – 6.0V

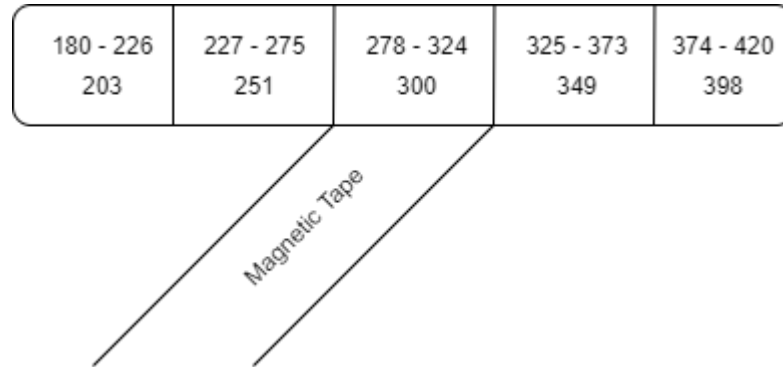
3.3 Desain Sistem Perangkat Lunak

Desain system perangkat lunak ini membahas bagaimana metode FLC digunakan dalam sistem pemosisian sinar UVC pada robot AUMR menggunakan sensor magnet.

3.3.1 Perancangan FLC (Fuzzy Logic Controller)

Perancangan FLC yang dimana meliputi *fuzzyfication*, *fuzzy inference*, dan *defuzzyfication*. Masukan pada sistem yang digunakan adalah *error* (e) posisi dan

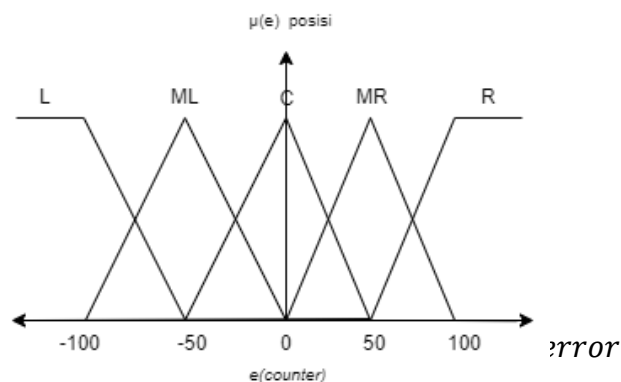
Δe posisi. Perancangan *fuzzyfication* berguna untuk memetakan nilai *crisp* dan *error* (e) posisi dari Δe posisi ke himpunan *fuzzy*. Berikut *variable linguistic* yang dirancang pada sistem.



Gambar III-17: Output Nilai Sensor

Berdasarkan dari Gambar III-17 merupakan output nilai dari magnetic sensor terhadap *magnetic tape* yang dimana direpresentasikan sebagai *error* (e) posisi. Pada saat nilai 180 – 420 merupakan nilai dari posisi *magnetic tape* terhadap posisi *sensor* yang dimana pada 203 merupakan titik tengah posisi sensor ketika *magnetic tape* berada pada posisi paling kiri, 251 merupakan titik tengah dari posisi sensor ketika *magnetic tape* berada pada posisi kiri, 300 merupakan titik tengah dari sensor ketika *magnetic tape* berada pada posisi tengah, 349 merupakan titik tengah dari posisi sensor ketika *magnetic tape* berada pada posisi kanan, dan 398 merupakan titik tengah dari posisi sensor ketika *magnetic tape* berada pada paling kanan.

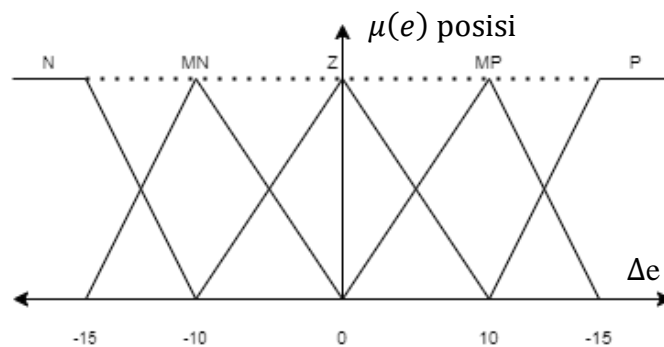
Dari persamaan pengambilan data tersebut didapatkan 5 variabel linguistic pada *input error* (e) posisi, yaitu *left* (L), *most left* (ML), *center* (C), *most right* (MR), dan *right* (R). Dengan bentuk fungsi keanggotaan berupa segitiga dan fungsi keanggotaan *error* (e) posisi dapat dilihat pada Gambar III-18



Gambar III-18: Fungsi Keanggotaan *error* (e)

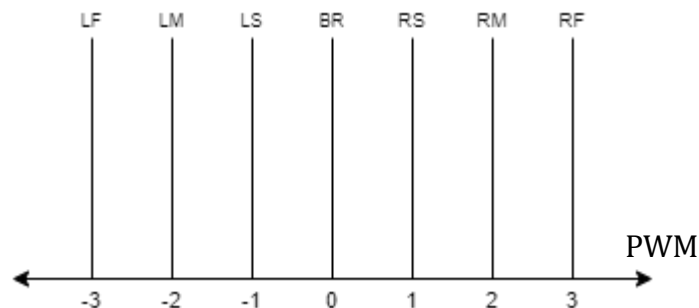
Dari Gambar III-18 terdapat 5 fungsi keanggotaan pada himpunan *fuzzy* yaitu *left* (L), *most left* (ML), *center* (C), *most right* (MR), dan *right* (R). kemudian nilai tersebut didapatkan berdasarkan hasil keluaran nilai sensor magnet terhadap *magnetic line*.

Himpunan fungsi keanggotaan *delta error* (Δe) didapatkan dari percobaan pada sistem yang diperoleh dari kecepatan perpindahan posisi sensor terhadap *magnetic tape*. Akhirnya terdapat 5 fungsi keanggotaan pada himpunan *fuzzy delta error* (Δe) posisi *Negative* (N), *Most Negative* (ME), *Zero* (Z), *Most Positive* (MP), *Positive* (P) dengan bentuk fungsi keanggotaan segitiga dapat dilihat pada Gambar III-19



Gambar III-19: Fungsi Keanggotaan *delta error* (Δe)

Keluaran dari FLC adalah PWM yang dimana dapat dibuat himpunan fungsi keanggotaan dari PWM yang terdiri dari 7 fungsi keanggotaannya yaitu *Left Fast* (LF), *Left Medium* (LM), *Left Slow* (LS), *Forward* (FR), *Right Slow* (RS), *Right Medium* (RM), dan *Right Fast* (RF) yang dapat ditunjukkan pada Gambar III-20 berikut.



Gambar III-20: Fungsi Keanggotaan keluaran PWM

Nilai PWM tersebut diperoleh dari pengujian sistem yang dimana akan dibahas pada BAB IV.

Kemudian menentukan perancangan tahap *fuzzy inference* yang dimana nilai dari *fuzzyfication* menjadi sebuah aturan-aturan (*rules*). *Rules* tersebut yang dimana akan menjadi keluaran dan respon dari sistem tersebut. Berikut Table III-13 *rules* yang didapatkan.

Tabel III-13: Rules FLC

e	L	ML	C	MR	R
NE	RM	RS	LS	LS	LF
MNE	LM	LS	LS	RS	RM
Z	LF	LM	BR	RM	RF
MPE	LM	LS	RS	RS	RM
PE	RM	RS	RS	LS	NM

Kemudian tahap terakhir yaitu *Defuzzyfication* yang dimana pada tahap ini nilai *fuzzy* diolah menjadi tegas (*crisp*). Keluaran dari tahap *defuzzyfication* berupa nilai PWM untuk control kecepatan motor DC.

3.3.2 Perancangan *Forward Kinematics for Differential Drive*

Perancaan *forward kinematics* dimana metode untuk menentukan arah dan kecepatan robot dengan mengubah input posisi berupa koordinat x,y dan derajat θ menjadi kecepatan pada setiap roda. Dengan sensor *incremental rotary encoder* sebagai referensi untuk membaca jarak pada robot.

Berdasarkan pada Persamaan II-2 dan Persamaan II-3 dapat diturunkan kedalam persamaan untuk posisi x,y dan derajat θ dan didapatkan sebagai berikut pada Persamaan III-1:

$$\begin{aligned}
 X &= R [\sin(\theta) + \sin(\omega\delta t)] + ICCx \\
 Y &= R [\cos(\theta) - \cos(\omega\delta t)] + ICCy \\
 \theta &= \omega\delta t
 \end{aligned}
 \tag{III-1}$$

Dari Persamaan III-1 diadapatkan turunan persamaan posisi *forward kinematics* untuk x,y dan θ dari ICC yang telah diketahuni pada Persamaan II-3.

BAB IV

HASIL DAN ANALISA

4.1 Pengambilan Data

Pengambilan data pada robot menggunakan komunikasi *serial* antara mikrokontroller dan laptop dengan menggunakan *software* PLX-DAQ. Data yang diambil merupakan *DeltaError*, *Error*, *xPositionInCM*, *yPositionInCM*, *Right PWM*, *Left Pwm*. Kemudian untuk mengukur dan mendapatkan data intensitas UVC menggunakan UVC *Light Meter Lutron* UVC254SD. Proses pengambilan data dibagi menjadi dua yaitu pada robot dan pada UVC, pada proses pengambilan data di robot dilakukan dengan cara sambungkan laptop dengan mikrokontroller untuk membaca data serial dari parameter *Fuzzy Logic* dan *Kinematics* yang dimana data serial tersebut akan dimasukkan kedalam format *.xlsx* menggunakan *software* PLX-DAQ, pada proses pengambilan data pada intensitas UVC menggunakan alat UVC *Light Meter* yang dimana data akan langsung keluar pada LCD yang terdapat pada alat tersebut.

4.2 Pengujian *Incremental Rotary Encoder*

Pengujian dilakukan untuk mengetahui nilai dan akurasi dari sensor yang digunakan yaitu *incremental rotary encoder*, pada pengujian ini diuji dengan cara menjalankan robot pada *magnetic tape* kemudian nilai sensor ditampilkan menggunakan serial monitor yang dimana nilai pada sensor tersebut sudah dikonversi menjadi satuan cm. Hasil dari nilai encoder yang sudah dikonversi akan dibandingkan dengan panjang lintasan dari *magnetic tape* sejauh 300cm yang diukur menggunakan meteran berjalan. Berikut hasil pengujian dari sensor *incremental encoder* dapat dilihat pada Tabel IV-1 berikut.

Tabel IV-1 Pengujian *Incremental Rotary Encoder*.

No	Posisi Encoder Kiri (Counter)	Posisi Encoder Kanan (Counter)	Nilai Pada Meteran (cm)	Nilai Rata-rata Dari Sensor (cm)	Error (%)	Error (%)
1	127.40	127.40	10	10.088	0.088	0.088

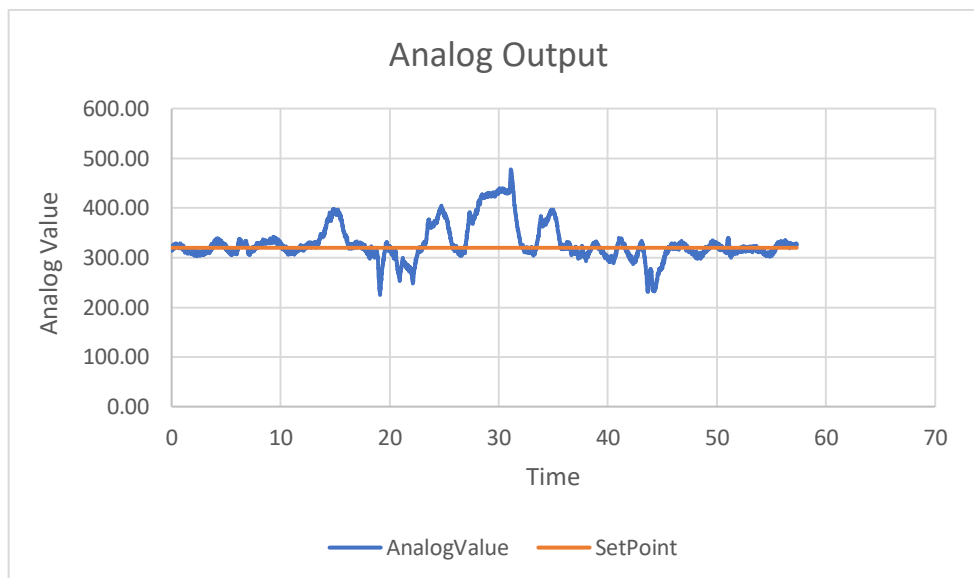
2	249.60	249.60	20	19.826	0.174	0.174
3	374.20	374.20	30	29.72	0.28	0.28
4	498.40	502.80	40	39.578	0.422	0.422
5	626.00	632.80	50	49.76	0.24	0.24
6	750.00	761.00	60	59.732	0.268	0.268
7	876.60	885.20	70	69.648	0.352	0.352
8	1003.40	1017.00	80	79.87	0.13	0.13
9	1128.60	1147.60	90	89.98	0.02	0.02
10	1254.20	1274.00	100	99.944	0.056	0.056
11	1378.40	1401.60	110	109.896	0.104	0.104
12	1505.60	1526.40	120	119.858	0.142	0.142
13	1639.40	1653.40	130	130.17	-0.17	0.17
14	1766.20	1773.20	140	139.916	0.084	0.084
15	1895.40	1903.00	150	150.156	- 0.156	0.156
16	2019.20	2027.20	160	159.96	0.04	0.04
17	2146.40	2158.00	170	170.162	- 0.162	0.162
18	2274.20	2283.80	180	180.184	- 0.184	0.184
19	2404.60	2413.00	190	190.448	- 0.448	0.448
20	2517.80	2534.60	200	199.732	0.268	0.268
21	2654.60	2660.80	210	210.128	- 0.128	0.128
22	2786.80	2789.60	220	220.444	- 0.444	0.444
23	2907.00	2915.60	230	230.178	- 0.178	0.178
24	3033.60	3043.60	240	240.24	-0.24	0.24

25	3160.60	3174.00	250	250.418	- 0.418	0.418
26	3284.80	3297.20	260	260.198	- 0.198	0.198
27	3412.40	3423.80	270	270.246	- 0.246	0.246
28	3539.40	3548.40	280	280.192	- 0.192	0.192
29	3669.80	3672.40	290	290.252	- 0.252	0.252
30	3802.60	3799.60	300	300.528	- 0.528	0.528
Nilai Rata-rata <i>Error</i> (%)						0.220%
Akurasi (%)						99.75%

Pada Tabel IV-1 pembacaan nilai pada encoder dan nilai *error* kemudian dapat dihitung nilai rata-rata *error* didapatkan 0.220% dan akurasi 99.75% yang dimana dengan menggunakan *incremental rotary encoder* E4056-600-3T-24 mendapatkan *error* yang kecil dan akurasi yang baik.

4.3 Pengujian Sensor *Magnetic Sensor* MGS1600GY

Pada pengujian *Magnetic Sensor* hal yang dibandingkan adalah nilai keluaran analog dengan *setpoint* yang ditetapkan pada bagian tengah sensor terhadap *magnetic tape* sebagai jalurnya, dan didapatkan berapa persen rata-rata *error*-nya. Pengambilan data menggunakan PLX-DAQ dengan membaca data *serial* dari mikorkontroller ke laptop dan divisualisasikan dengan grafik. Berikut Gambar IV-1 menunjukkan pembacaan nilai sensor dengan *setpoint* 320 pada keluaran analog sensor.



Gambar IV-1: Analog Value

Bedasarkan dari data grafik pada Gambar IV-1 hasil pengujian menggunakan *magnetic* sensor didapatkan saat grafik naik dan turun merupakan dalam keadaan berbelok karena dilakukan pada lintasan berbelok dan memutar. Kemudian pengujian juga dilakukan saat dihidupkannya lampu UV dan hasilnya tidak berpengaruh pada nilainya kemudian untuk kontrol posisi menggunakan FLC.

4.4 Pengujian *Fuzzy Logic Control* pada Matlab dan Robot

Pengujian dilakukan dengan membandingkan nilai *output* pada *software* matlab dengan kondisi sebenarnya pada robot yang menggunakan Arduino. Proses pengujian dilakukan dengan cara membaca nilai sensor magnet yang dimana menjadi referensi untuk FLC tersebut yang dimana keluarannya akan berupa PWM pada motor DC kiri dan kanan. kemudian melakukan pengujian menggunakan matlab untuk FLC untuk membandingkan simulasi dengan kenyataan untuk mengetahui berapa persen *error* dan akurasi sistem. Berikut Tabel IV-2 Pengujian Keluaran FLC Pada Matlab dan MotorDC.

Tabel IV-2: Pengujian Keluaran FLC Pada Matlab dan MotorDC

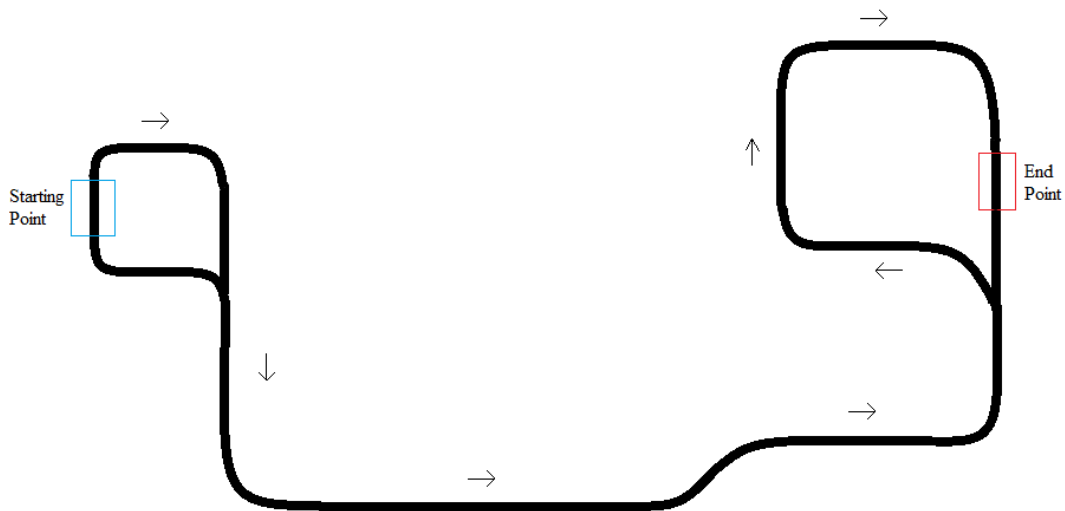
No.	Error (Counter)	Delta Error (Counter)	PWM Motor DC Kanan	PWM Motor DC Kiri	PWM Matlab Kanan	PWM Matlab Kiri	AvgError (%)
1	34,00	0	13	-14	12.3	-12.3	1.20
2	33,00	-1	11	-11	10.3	-10.3	0.70

3	33.00	0	13	-14	12	-12	1.50
4	34.00	1	13	-13	12.3	-12.3	0.70
5	34.00	0	13	-14	12.3	-12.3	1.20
6	34.00	0	13	-14	12.3	-12.3	1.20
7	34.00	0	13	-14	12.3	-12.3	1.20
8	34.00	0	13	-14	12.3	-12.3	1.20
9	34.00	0	13	-14	12.3	-12.3	1.20
10	34.00	0	13	-14	12.3	-12.3	1.20
11	34.00	0	13	-14	12.3	-12.3	1.20
12	30.00	-4	6	-7	6.36	-6.36	0.14
13	30.00	0	12	-12	11.2	-11.2	0.80
14	35.00	5	11	-12	10.5	-10.5	1.00
15	35.00	0	14	-14	12.6	-12.6	1.40
16	25.00	-10	0	0	-0.13	0.13	0.13
17	25.00	0	10	-10	9.86	-9.86	0.32
18	28.00	3	10	-11	10.6	-10.6	-0.10
19	28.00	0	11	-12	10.7	-10.7	0.80
20	20.00	-8	1	-2	1.53	-1.53	-0.03
21	20.00	0	8	-8	8.53	-8.53	-0.53
22	26.00	6	10	-10	9.99	-9.99	0.01
23	26.00	0	10	-11	10.1	-10.1	0.40
24	13.00	-13	-7	-6	-3.07	-3.07	-6.50
25	13.00	0	5	-6	6.44	6.44	-0.94
26	6.00	-7	-3	2	-2.41	-2.41	-0.09
27	6.00	0	2	-3	3.66	3.66	-1.16
28	2.00	-4	-3	2	-2.78	-2.78	0.28
29	2.00	0	0	-1	1.42	1.42	-0.92
30	1.00	-1	-1	0	-0.601	-0.601	-0.10
Error Rata-rata (%)							0.25%
Akurasi (%)							93.88%

Dari Tabel IV-2 didapatkan *error* rata-rata sebesar 0.25% dan akurasi sebesar 93.88% yang dimana membuktikan algoritma FLC yang dibuat mendekati benar dengan membandingkan hasil dari simulasi pada Matlab.

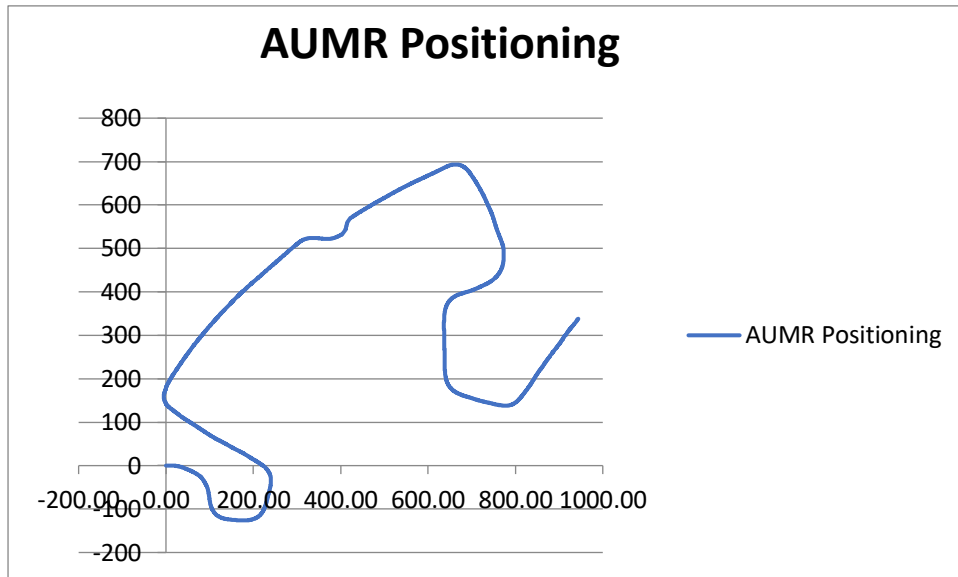
4.5 Pengujian *Forward Kinematic* terhadap lintasan

Pengujian dilakukan untuk mengetahui posisi robot berdasarkan lintasannya dan nilai pada parameter x, y serta sudutnya menggunakan sensor *encoder*. Pada pengujian ini dilakukan dengan cara menjalankan robot pada *magnetic tape* kemudian dengan menggunakan algoritma *Forward Kinematic* didapatkannya nilai x, y dan θ sudutnya, yang dimana secara teori ketika robot berjalan lurus maka nilai x akan bertambah tetapis nilai y dan θ akan bernilai 0, kemudian ketika berbelok θ sudut akan berubah dan nilai y akan berubah sesuai arah belok, jika kekiri maka nilai akan menjadi ke arah negatif dan sebaliknya. Berikut gambar IV-1 merupakan lintasan *magnetic tape*.



Gambar IV-2: Lintasan *magnetic tape*

Dari gambar IV-1 yang dimana merupakan lintasan yang seharusnya didapatkan pada nilai posisi x,y dari algoritma *forward kinematic* berikut merupakan gambar grafik terbaik dari 10 kali percobaan menggunakan *forward kinematic*.



Gambar IV-3: Output grafik *forward kinematic*

Dari gambar IV-2 didapatkan grafik tidak sesuai dengan kondisi asli pada lintasan uji coba.

4.6 Pengujian Sinar UVC Menggunakan UV Meter

Pengujian sinar UVC yang teradapat pada robot yang dimana akan berhenti ketika perpindahan posisi (*resultant*) mecapai 300cm yang dimana nilai x,y sudah dikonversi menjadi cm pada proses *forward kinematic*. Proses pengujian dilakukan dengan meletakan alat UV Meter sejauh 300cm dari jarak lampu UV pada robot tanpa adanya penghalang. Pengujian mengambil data berapa lama waktu yang dibutuhkan untuk radiasi UV mencapai nilai $40 \mu\text{W}/\text{cm}^2$ akan ditunjukkan pada Tabel IV-3 Pengujian UVC Terhadap Waktu.

Tabel IV-3: Pengujian UVC Terhadap Waktu

Percobaan	Radias UV ($\mu\text{W}/\text{cm}^2$)	Waktu Mencapaian $40 \mu\text{W}/\text{cm}^2$ (Detik)
1	45	1.01
2	45	0.25
3	51	0.724
4	45	0.750
5	56	0.75

6	58	0.75
7	56	0.799
8	44	1
9	59	1.249
10	45	1.249
Rata-rata Waktu (detik)		0.8531

Dari Tabel IV-3 didapatkan nilai rata-rata waktu untuk mencapai nilai radiasi lebih besar sama dengan $40 \mu\text{W}/\text{cm}^2$ adalah 0.8531 detik, yang dimana berarti pada jarak 300cm dengan waktu 0.8531 sudah efektif untuk menonaktifkan virus dan bakteri.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan dari pada hasil pengujian dan analisis, didapatkan kesimpulan pada Tugas Akhir adalah sebagai berikut.

1. AUMR terbukti efektif untuk menonaktifkan virus dan bakteri dalam jarak 300cm dengan waktu rata-rata 0.8531 detik dalam 10 kali percobaan untuk mencapai radias $40 \mu\text{W}/\text{cm}^2$.
2. Pengujian sensor *incremental rotary encoder* dapat digunakan untuk mengetahui posisi yang nilainya dapat diubah dalam satuan cm pada robot dengan memiliki nilai *error* 0.220% dan akurasi sebesar 99.75% setelah di rata-ratakan dari 5 kali percobaan pada jarak 300cm
3. Sensor *magnetic* MGS1600GY dapat digunakan baik pada lingkungan *indor*. Keluaran nilai analog dari *magnetic* sensor tersebut tidak terpengaruh terhadap lingkungannya seperti intensitas cahaya ketika dihidupkannya lampu UVC jadi berjalan dengan baik untuk sistem.
4. *Fuzzy Logic Control* yang dirancang terdapat 2 masukan yaitu *error* (e) posisi dan *delta error* (Δe), dengan masukan tersebut udah cukup untuk melakukan kontrol kecepatan antara dua roda pada robot menggunakan sensor *magnetic* sebagai referensi.
5. Pengujian *Fuzzy Logic Control* pada robot dan simulasi dapat digunakan dengan diperolehnya *error* pada keluaran kecepatan antara dua roda tersebut sejumlah 0.25% dan akurasi yang cukup akurat dengan nilai sebesar 93.88%
6. *Forward Kinematic* yang dirancang dengan masukan berupa nilai dari encoder dengan beberapa parameter-parameter mekanik robot yang berpengaruh kepada keluaran. Pada pengujian *forward kinematic* pada sistem didapatkan *error* yang cukup besar antara keluaran grafik dengan bentuk lintasan aslinya. Beberapa *variable* yang mempengaruhi adalah kondisi slip pada kedua roda ketika berbelok, kondisi lantai yang tidak rata. Tetapi pembacaan nilai posisi x,y serta theta sudah cukup baik.

5.2 Saran

Saran yang dapat diberikan dan dilakukan untuk guna melanjutkan serta melakukan pengembangan pada Tugas Akhir ini adalah sebagai berikut.

1. Pengembangan robot AUMR dapat dilakukan dengan memperhatikan mekanik para robot seperti distribusi beban yang baik agar tidak terjadi slip pada roda.
2. Menggunakan sensor lain seperti IMU, LiDAR sebagai tambahan referensi untuk memperbaiki posisi pada algoritma *forward kinematic* agar lintasan dan keluaran posisi dapat sesuai.
3. Menambahkan sensor yang dapat mendeteksi keberadaan manusia jikalau tidak sadar sedang dioprasikannya robot AUMR dan dapat menonaktifkan lampu UVC tersebut, karena sinar UVC berbahaya untuk makhluk hidup.
4. Sensor encoder diletakan diposisi *master* karena ketika banyaknya data dan parameter yang dikirimkan dari *slave* ke *master* akan mengakibatkan terjadinya *delay* saat pengiriman data.
5. Dapat menggunakan mikrokontroller selain arduino yang memiliki bit dan *clock* lebih besar sehingga pengiriman data, dan penggunaan metode-metode yang banyak memakan memori dapat berjalan lebih baik.

DAFTAR PUSTAKA

- [1] R. Takasawa, H. Nakamura, T. Mori, and S. Tanuma, "Differential apoptotic pathways in human keratinocyte HaCaT cells exposed to UVB and UVC," *2005 Spring Science + Business Media, Inc. Manufactured in The Netherlands*, pp. 1121-1130, 2005.
- [2] Anthony J Dixon, and Brian F Dixon, "Ultraviolet radiation from welding and possible risk of skin ocular malignancy." vol. 181, no. 3, 2004.
- [3] R.S. Nugroho, "Rekap Kasus Corona Indonesia Selama Maret dan Prediksi di Bulan April," [Online]. Available: <https://www.kompas.com/tren/read/2020/03/31/213418865/rekap-kasus-corona-indonesia-selama-maret-dan-prediksi-di-bulan-april?page=all>. [Accessed 6 October 2020].
- [4] R. Satriatama, "Sistem Kontrol Troli Rotari Sebagai Tempat Penitipan Barang Otomatis Berbasis RFID Menggunakan Fuzzy Logic," Telkom University, Bandung, 2019.
- [5] Roboteq, "MGS1600GY Magnetic Sensor Datasheet," ver. 1.3, 2018
- [6] P. Setiyopamuji, P. Pangaribuan, A.S. Wibowo, "Perancangan dan Implementasi Anti Sway Gantry Crane Berbasis Fuzzy Logic Controller," Telkom University, Bandung, 2019.
- [7] M.A. Irawan, "Kendali Motor DC Encoder Dengan Metode PID Pada Robot MiroSot Menggunakan Komunikasi Wireless," Telkom University, Bandung, 2017.
- [8] M. H. Goodarzi, A. Amiri, "Evaluating Student' Learning Progress Using Fuzzy Inference System," *2009 Sixth International Conference on Fuzzy System and Knowledge Discovery*, pp 561, 2009.
- [9] V. Cherkassky, "Fuzzy Inference Systems: A Critical Review," *Department of Electrical Engineering, University of Minnesota, Minneapolis, Minnesota 55455, USA*, pp 179, 1998.
- [10] Arduino, "Arduino Mega 2560 Rev3," [Online]. Available: <https://store.arduino.cc/usa/mega-2560-r3>. [Accessed 16 Desember 2020].

- [11] G. Dudek, M. Jenkin, “Computational Principles of Mobile Robotics,” *Cambridge University Press*, Cambridge, 2010.
- [12] C. Burbridge, “Robot Kinematics” *University of Birmingham*, England, 2015.

LAMPIRAN

❖ Lampiran 1 (Source Code)

```
#include <Arduino.h>

#define master_code true
// #define slave_code true
// #define loadTesting true
// #define loadFuzzy true
// #define loadEncoder true
// #define loadKinematic true
// #define loadMaster true
#define uvMode true
// #define loadSlave
#define printToPC true

#ifndef master_code
#include <Wire.h>

// ----- Master -----
// Open The Command if want to upload on Slave

// void prosedur
//Master To Slave

#define Master Serial3
// Motor DC Pin Out
#define MRF 5 // Motor Right Front Direction
#define MRB 4 // Motor Right Back Off Direction
#define MLF 7 // Motor Left Front Direction
#define MLB 6 // Motor Left Back Off Direction
int RpwmOne = 100,
```



```
LpwmOne = 100,  
RpwmTwo = 50,  
LpwmTwo = 50,  
Rpwm,  
Lpwm;
```

```
int leftSpeed,  
    rightSpeed,  
    leftSpeedVal,  
    rightSpeedVal;
```

```
bool check = false;
```

```
// Variable Sensor  
int pointerValue,  
    trackDetect,  
    AnalogDetect,  
    RightMarkerDetect,  
    LeftMarkerDetect,  
    value;
```

```
long xPositionInCM,  
     yPositionInCM,  
     thetaPositionInDegree,
```

```
leftLinearSpeed,  
rightLinearSpeed;
```

```
float voltageDividerOne,  
     voltageDividerTwo,  
     outputOne,
```

```
outputTwo;
```

```
// Receiver Control
```

```
const int CH1 = 32,
```

```
      CH2 = 33,
```

```
      CH3 = 34,
```

```
      CH4 = 35,
```

```
      CH5 = 36,
```

```
      CH6 = 37,
```

```
      CH7 = 38,
```

```
      CH8 = 39,
```

```
      CH9 = 40,
```

```
      CH10 = 41;
```

```
int ch1, //variable constrain
```

```
      ch2,
```

```
      ch3,
```

```
      ch4,
```

```
      ch5,
```

```
      ch6;
```

```
//UVC Activation
```

```
#define UB 29      // UV Behind
```

```
#define UF 28      // UV Front
```

```
#define BUZZ 42    // Buzzer Relay PIN Before Turn the Lamp ON
```

```
#define ACTIVE_RL 30 // Activation Relay for Inverter
```

```
int n = 1,
```

```
      resultant;
```

```
long xPostKuadrat,
```

```
      yPostKuadrat,
```

```

    kuadrat;

const int rellyOn = HIGH,
      rellyOff = LOW,
      relayOn = LOW,
      relayOff = HIGH,
      BuzzerOn = HIGH,
      BuzzerOff = LOW,
      buzzerTime = 1000,
      buzzerCycle = 3;

void relaySteady(){
    digitalWrite(UB, relayOff);
    digitalWrite(UF, relayOff);
}

// DHT and FAN
#define DHT 42    // DHT11 RELAY
#define FAN_RLY 31 // FAN RELAY

// Ultrasonic
int reading = 0;

// Constrain Variable
int max = 1988, //Stick Constrain
    min = 988;

int analogMax = 500, // Output Sensor Constrain
    analogMin = 120;

#define errorRules 5
#define deltaErrorRules 5

```

String state;

float fuzzyError[errorRules],
deltaError[deltaErrorRules],
errorPosition,
deltaErrorPosition,
previousError;

float rule[10][10],
rule00,
rule01,
rule02,
rule03,
rule04,

rule10,
rule11,
rule12,
rule13,
rule14,

rule20,
rule21,
rule22,
rule23,
rule24,

rule30,
rule31,
rule32,
rule33,
rule34,

```
rule40,  
rule41,  
rule42,  
rule43,  
rule44;  
  
//Membership function Error Position  
float midLeft = -100, //left  
rightLeft = -50,  
  
leftMostLeft = -100, //Most Left  
midMostLeft = -50,  
rightMostLeft = 0,  
  
leftCenter = -50, // Center  
midCenter = 0,  
rightCenter = 50,  
  
leftMostRight = 0, // Most Right  
midMostRight = 50,  
rightMostRight = 100,  
  
leftRight = 50, // Right  
midRight = 100;  
  
float midNegative = -20,  
rightNegative = -10,  
  
leftMostNegative = -20,  
midMostNegative = -10,  
rightMostNegative = 0,
```

```
leftZero = -10,  
midZero = 0,  
rightZero = 10,
```

```
leftMostPositive = 0,  
midMostPositive = 10,  
rightMostPositive = 20,
```

```
leftPositive = 10,  
midPositive = 20;
```

```
float LF = -30,  
      LM = -20,  
      LS = -10,  
      FW = 0,  
      RS = 10,  
      RM = 20,  
      RF = 30;
```

```
float speedMotor,  
      decision,  
      divider,  
      yValue,  
      yFirst,  
      ySecond;
```

```
//Variable PID
```

```
float error,  
      RsetPointPwm = 80,  
      LsetPointPwm = 80,  
      setPointSensor = 320,  
      pValue,
```

```
Kp = 0.4,  
Ki,  
Kd = 0.2,  
Ts = 0.16,  
lastError;
```

```
//Variabel Derivatif
```

```
float derivatifOne,  
      derivatifTwo,  
      derivatifThre,  
      derivatifValue,  
      proporsionalDerivatif;
```

```
unsigned long startTimeMillis,  
              endTimeMillis,  
              loopTimer;
```

```
void checkLoopTimer(){  
    endTimeMillis = millis();  
    loopTimer = (endTimeMillis - startTimeMillis);  
    startTimeMillis = millis();
```

```
    Serial.println(loopTimer);  
    // Serial.print("\t");  
}
```

```
void goForward(){  
    if(Lpwm>0){  
        analogWrite(MLF, Lpwm);  
        analogWrite(MLB, 0);  
    }  
    else if(Lpwm<0){
```

```

    analogWrite(MLF, 0);
    analogWrite(MLB, -Lpwm);
}
else if(Lpwm==0){
    analogWrite(MLF, 0);
    analogWrite(MLB, 0);
}

if(Rpwm>0){
    analogWrite(MRF, Rpwm);
    analogWrite(MRB, 0);
}
else if(Rpwm<0){
    analogWrite(MRF, 0);
    analogWrite(MRB, -Rpwm);
}
else if(Rpwm==0){
    analogWrite(MRF, 0);
    analogWrite(MRB, 0);
}
}

void Buzzer(){
    for(int i=0; i<=2; i++){
        digitalWrite(BUZZ, BuzzerOn);
        delay(100);
        digitalWrite(BUZZ, BuzzerOff);
        delay(50);
    }
}

void buzzerUV(){
    for(int i=0; i<=2; i++){

```



```

        digitalWrite(BUZZ, BuzzerOn);
        delay(500);
        digitalWrite(BUZZ, BuzzerOff);
        delay(2000);
    }
}

void uvActivation()
{
    buzzerUV();
    digitalWrite(ACTIVE_RL, relayOn);
    digitalWrite(UB, relayOn);
    digitalWrite(UF, relayOn);
}

void uvDeActivation(){
    digitalWrite(ACTIVE_RL, relayOff);
    digitalWrite(UB, relayOff);
    digitalWrite(UF, relayOff);
}

void getSensor(){
    if(Master.available())
    {
        char c = Master.peek();
        if (c == '!'){
            Master.read();
            pointerValue = 1;
        }
        else if(c == '@'){
            Master.read();

```

```
    pointerValue = 2;
}
else if (c == '&'){
    Master.read();
    pointerValue = 3;
}
else if (c == '='){
    Master.read();
    pointerValue = 4;
}
else if (c == ' '){
    Master.read();
    pointerValue = 5;
}
// else if (c == '^'){
//  Master.read();
//  pointerValue = 6;
// }
// else if (c == '&')
// {
//  Master.read();
//  pointerValue = 7;
// }
// else if (c == '*')
// {
//  Master.read();
//  pointerValue = 8;
// }
// else if (c == '-')
// {
//  Master.read();
//  pointerValue = 9;
```

```

// }
else {
    value = Master.parseInt();
    if (pointerValue == 1){
        trackDetect=value;
        trackDetect=constrain(trackDetect, 0, 1);
    }
    else if (pointerValue == 2){
        AnalogDetect=value;
        AnalogDetect= constrain(AnalogDetect, analogMin, analogMax);
    }
    else if (pointerValue == 3){
        xPositionInCM=value;
    }
    else if (pointerValue == 4){
        yPositionInCM=value;
        check = true;
    }
    else if (pointerValue == 5){
        thetaPositionInDegree=value;
        check = true;
    }
}
}
}

```

```

void getMagneticData(){
    Serial.print(" TP: ");
    Serial.print(trackDetect);
    Serial.print(" AO: ");
    Serial.print(AnalogDetect);
    // Serial.print(" R_Encoder: ");
}

```

```

// Serial.print(R_encoder_position);
// Serial.print(" L_Encoder: ");
// Serial.println(encoder_position);
}

```

```

void getKinematicData(){
    Serial.print(" xPostCM: ");
    Serial.print(xPositionInCM);
    Serial.print(" yPostCM: ");
    Serial.print(yPositionInCM);
    Serial.print(" thtPostDeg: ");
    Serial.print(thetaPositionInDegree);
}

```

```

void getUVActivationData(){
    Serial.print(" xPostKuadrat: ");
    Serial.print(xPostKuadrat);
    Serial.print(" yPostKuadrat: ");
    Serial.print(yPostKuadrat);
    Serial.print(" resultant: ");
    Serial.println(resultant);
}

```

```

float callErrorPosition(float x, float xOne, float Xtwo, String highLow){
    if (highLow == "High"){
        yFirst=0;
        ySecond=1;
    }
    else if(highLow == "Low"){
        yFirst=1;
        ySecond=0;
    }
}

```

```

yValue=((x-xOne)*(ySecond-yFirst))/(Xtwo-xOne))+yFirst;
return yValue;
}

void fuzzyfication(){
    errorPosition = setPointSensor - AnalogDetect;

    if(errorPosition <= midLeft){
        fuzzyError[0] = 1;
    }
    else if(errorPosition > midLeft && errorPosition <= rightLeft){
        fuzzyError[0]=callErrorPosition(errorPosition, midLeft, rightLeft,
"Low");
    }
    else if(errorPosition > rightLeft){
        fuzzyError[0]=0;
    }

    if(errorPosition <= leftMostLeft){
        fuzzyError[1]=0;
    }
    else if(errorPosition > leftMostLeft && errorPosition <=
midMostLeft){
        fuzzyError[1]=callErrorPosition(errorPosition, leftMostLeft,
midMostLeft, "High");
    }
    else if(errorPosition > midMostLeft && errorPosition <=
rightMostLeft){
        fuzzyError[1]=callErrorPosition(errorPosition, midMostLeft,
rightMostLeft, "Low");
    }
    else if(errorPosition > rightMostLeft){

```

```

    fuzzyError[1]=0;
}

if(errorPosition <= leftCenter){
    fuzzyError[2]=0;
}
else if(errorPosition > leftCenter && errorPosition <= midCenter){
    fuzzyError[2]=callErrorPosition(errorPosition, leftCenter, midCenter,
"High");
}
else if(errorPosition > midCenter && errorPosition <= rightCenter){
    fuzzyError[2]=callErrorPosition(errorPosition, midCenter, rightCenter,
"Low");
}
else if(errorPosition > rightCenter){
    fuzzyError[2]=0;
}

if (errorPosition <= leftMostRight){
    fuzzyError[3]=0;
}
else if (errorPosition > leftMostRight && errorPosition <=
midMostRight){
    fuzzyError[3]=callErrorPosition(errorPosition, leftMostRight,
midMostRight, "High");
}
else if(errorPosition > midMostRight && errorPosition <=
rightMostRight){
    fuzzyError[3]=callErrorPosition(errorPosition, midMostRight,
rightMostRight, "Low");
}
else if(errorPosition > rightMostRight){

```

```

    fuzzyError[3]=0;
}

if (errorPosition <= leftRight){
    fuzzyError[4]=0;
}
else if(errorPosition > leftRight && midRight){
    fuzzyError[4]=callErrorPosition(errorPosition, leftRight, midRight,
"High");
}
else if(errorPosition > midRight){
    fuzzyError[4]=1;
}

//Delta Error Calculation

deltaErrorPosition = errorPosition - previousError;

if(deltaErrorPosition <= midNegative){
    deltaError[0]=1;
}
else if(deltaErrorPosition > midNegative && deltaErrorPosition <=
rightNegative){
    deltaError[0]=callErrorPosition(deltaErrorPosition, midNegative,
rightNegative, "Low");
}
else if(deltaErrorPosition > rightNegative){
    deltaError[0]=0;
}

if(deltaErrorPosition <= leftMostNegative){
    deltaError[1]=0;

```

```

    }
    else if(deltaErrorPosition > leftMostNegative && deltaErrorPosition <=
midMostNegative){
        deltaError[1]=callErrorPosition(deltaErrorPosition, leftMostNegative,
midMostNegative, "High");
    }
    else if(deltaErrorPosition > midMostNegative && deltaErrorPosition
<= rightMostNegative){
        deltaError[1]=callErrorPosition(deltaErrorPosition, midMostNegative,
rightMostNegative, "Low");
    }
    else if(deltaErrorPosition > rightMostNegative){
        deltaError[1]=0;
    }

    if(deltaErrorPosition <= leftZero){
        deltaError[2]=0;
    }
    else if(deltaErrorPosition > leftZero && deltaErrorPosition <=
midZero){
        deltaError[2]=callErrorPosition(deltaErrorPosition, leftZero, midZero,
"High");
    }
    else if(deltaErrorPosition > midZero && deltaErrorPosition <=
rightZero){
        deltaError[2]=callErrorPosition(deltaErrorPosition, midZero,
rightZero, "Low");
    }
    else if(deltaErrorPosition > rightZero){
        deltaError[2]=0;
    }

```



```

    if(deltaErrorPosition <= leftMostPositive){
        deltaError[3]=0;
    }
    else if(deltaErrorPosition > leftMostPositive && deltaErrorPosition <=
midMostPositive){
        deltaError[3]=callErrorPosition(deltaErrorPosition, leftMostPositive,
midMostPositive, "High");
    }
    else if(deltaErrorPosition > midMostPositive && deltaErrorPosition <=
rightMostPositive){
        deltaError[3]=callErrorPosition(deltaErrorPosition, midMostPositive,
rightMostPositive, "Low");
    }
    else if(deltaErrorPosition > rightMostPositive){
        deltaError[3]=0;
    }

    if(deltaErrorPosition <= leftPositive){
        deltaError[4]=0;
    }
    else if(deltaErrorPosition > leftPositive && deltaErrorPosition <=
midPositive){
        deltaError[4]=callErrorPosition(deltaErrorPosition, leftPositive,
midPositive, "High");
    }
    else if(deltaErrorPosition >= midPositive){
        deltaError[4]=1;
    }

    previousError = errorPosition;
}

```

```
void rules(){
    divider = 0;
    for(int i=0; i<=4; i++){
        for(int j=0; j<=4; j++){
            rule[i][j] = min(deltaError[i], fuzzyError[j]);
            divider = divider + rule[i][j];}}}
```

```
rule00 = rule[0][0];
rule01 = rule[0][1];
rule02 = rule[0][2];
rule03 = rule[0][3];
rule04 = rule[0][4];
```

```
rule10 = rule[1][0];
rule11 = rule[1][1];
rule12 = rule[1][2];
rule13 = rule[1][3];
rule14 = rule[1][4];
```

```
rule20 = rule[2][0];
rule21 = rule[2][1];
rule22 = rule[2][2];
rule23 = rule[2][3];
rule24 = rule[2][4];
```

```
rule30 = rule[3][0];
rule31 = rule[3][1];
rule32 = rule[3][2];
rule33 = rule[3][3];
rule34 = rule[3][4];
```

```
rule40 = rule[4][0];
```

```

rule41 = rule[4][1];
rule42 = rule[4][2];
rule43 = rule[4][3];
rule44 = rule[4][4];

decision = rule00 * RM + rule01 * RS + rule02 * LS + rule03 * LS +
rule04 * LM +
rule10 * LM + rule11 * LS + rule12 * LS + rule13 * RS + rule14
* RM +
rule20 * LF + rule21 * LM + rule22 * FW + rule23 * RM +
rule24 * RF +
rule30 * LM + rule31 * LS + rule32 * RS + rule33 * RS + rule34
* RM +
rule40 * RM + rule41 * RS + rule42 * RS + rule43 * LS + rule44
* LM;
}

```

```

void defuzzification(){
    speedMotor = decision/devider;
    rightSpeedVal = RpwmTwo + speedMotor;
    leftSpeedVal = LpwmTwo - speedMotor;
}

```

```

void goFuzzy(){
    if(leftSpeedVal>0){
        analogWrite(MLF, leftSpeedVal);
        analogWrite(MLB, 0);
    }
    else if(leftSpeedVal<0){
        analogWrite(MLF, 0);
        analogWrite(MLB, -leftSpeedVal);
    }
}

```

```

    }
    else if(leftSpeedVal==0){
        analogWrite(MLF, 0);
        analogWrite(MLB, 0);
    }

    if(rightSpeedVal>0){
        analogWrite(MRF, rightSpeedVal);
        analogWrite(MRB, 0);
    }
    else if(rightSpeedVal<0){
        analogWrite(MRF, 0);
        analogWrite(MRB, -rightSpeedVal);
    }
    else if(rightSpeedVal==0){
        analogWrite(MRF, 0);
        analogWrite(MRB, 0);
    }
}

void stop(){
    analogWrite(MLF, 0);
    analogWrite(MLB, 0);
    analogWrite(MRF, 0);
    analogWrite(MRB, 0);
}

void fuzzy(){
    fuzzyfication();
    rules();
    defuzzification();
}

```

```

void uvActivationKinematic(){
    xPostKuadrat = xPositionInCM * xPositionInCM;
    yPostKuadrat = yPositionInCM * yPositionInCM;
    resultant = sqrt(xPostKuadrat + yPostKuadrat);
    if (resultant >= 300 * n)
    {
        n++;
        stop();
        uvActivation();
        delay(120000);
        uvDeActivation();
    }
}

```

```

void loadFuzzyPWM(){
    Serial.print("DATA,TIME,");
    Serial.print(errorPosition);
    Serial.print(", ");
    Serial.print(deltaErrorPosition);
    Serial.print(", ");
    Serial.print(rightSpeedVal);
    Serial.print(", ");
    Serial.println(leftSpeedVal);
}

```

```

void loadKinematics(){
    Serial.print("DATA,TIME,");
    Serial.print(xPositionInCM);
    Serial.print(", ");
    Serial.println(yPositionInCM);
}

```

```

// void loadVariationFuzzy(){

// }

void testing(){
    Serial.print("DATA,TIME,");
    Serial.print(errorPosition);
    Serial.print(", ");
    Serial.print(deltaErrorPosition);
    Serial.print(", ");
    Serial.print(xPositionInCM);
    Serial.print(", ");
    Serial.print(yPositionInCM);
    Serial.print(", ");
    Serial.print(rightSpeedVal);
    Serial.print(", ");
    Serial.println(leftSpeedVal);

}

void getFuzzy(){
    getSensor();
    if(check == true){
        // getMagneticData();
        // getKinematicData();
        fuzzy();
        goFuzzy();
        //uvActivationKinematic();
        #ifdef uvMode
            getKinematicData();
            uvActivationKinematic();
            getUVActivationData();

```

```
#endif
```

```
#ifdef loadFuzzy  
    loadFuzzyPWM();  
#endif
```

```
#ifdef loadKinematic  
    loadKinematics();  
#endif
```

```
#ifdef loadTesting  
    testing();  
#endif
```

```
    //forwardKinematic();  
    check = false;  
}  
}
```

```
void uvActivationReset()  
{  
    digitalWrite(ACTIVE_RL, relayOn);  
}
```

```
// Variable Pergerakan
```

```
void voltageDivider(){  
    float aref = 5;  
    float rOne = 10000;  
    float rTwo = 5800;
```

```

float analogValue = 1023;
float rOne_two = 45000;
float rTwo_two = 6800;

voltageDividerOne = analogRead(A8);
Serial.print("Output Analog 12V: ");
Serial.print(voltageDividerOne);

voltageDividerTwo = analogRead(A9);
Serial.print(" Output Analog 24V: ");
Serial.print(voltageDividerTwo);

outputOne = (voltageDividerOne*aref*(rOne +
rTwo))/(analogValue*rTwo);
Serial.print(" Output 12V: ");
Serial.print(outputOne);

outputTwo =
(voltageDividerTwo*aref*(rOne_two+rTwo_two))/(analogValue*rTwo_t
wo);
Serial.print(" Output 24V: ");
Serial.println(outputTwo);
}

void outputRemote(){
  ch1 = pulseIn(CH1, HIGH); //
  // Serial.print(" ch1: ");
  // Serial.print(ch1);
  ch2 = pulseIn(CH2, HIGH); //
  // Serial.print(" ch2: ");
  // Serial.print(ch2);
  ch3 = pulseIn(CH3, HIGH); //

```



```

// Serial.print(" ch3: ");
// Serial.print(ch3);
ch4 = pulseIn(CH4, HIGH); //
// Serial.print(" ch4: ");
// Serial.print(ch4);
ch5 = pulseIn(CH5, HIGH); //
// Serial.print(" ch5: ");
// Serial.print(ch5);
ch6 = pulseIn(CH6, HIGH); //
// Serial.print(" ch6: ");
// Serial.println(ch6);
}

void rcMode(){
  int deadzone = 15;
  outputRemote();
  int throt = map(ch3, min, max, 0, 255);
  // Serial.print("Throt: ");
  // Serial.print(throt);

  int x = map(ch1, min, max, -throt, throt);
  if(x>-deadzone && x<deadzone) {
    x = 0;
  }
  // Serial.print(" x: ");
  // Serial.print(x);

  int y = map(ch2, min, max, -throt, throt);
  if(y>-deadzone && y<deadzone) {
    y = 0;
  }
  // Serial.print(" y: ");

```

```

// Serial.print(y);

int leftSpeed = y + x;
int rightSpeed = y - x;
leftSpeed = constrain(leftSpeed, -255, 255);
rightSpeed = constrain(rightSpeed, -255, 255);
// Serial.print(" LS: ");
// Serial.print(leftSpeed);
// Serial.print(" RS: ");
// Serial.print(rightSpeed);

if (leftSpeed == 0){
    analogWrite(MLF, 0);
    analogWrite(MLB, 0);
}
else if (leftSpeed > 0){
    analogWrite(MLF, leftSpeed);
    analogWrite(MLB, 0);
}
else {
    analogWrite(MLF, 0);
    analogWrite(MLB, -leftSpeed);
}

// right Motor Setup
if (rightSpeed == 0){
    analogWrite(MRF, 0);
    analogWrite(MRB, 0);
}
else if (rightSpeed > 0){
    analogWrite(MRF, rightSpeed);
    analogWrite(MRB, 0);
}

```

```

    }
    else {
        analogWrite(MRF, 0);
        analogWrite(MRB, -rightSpeed);
    }
}

```

```

void PID(){
    // PID

    getSensor();
    if(trackDetect == 1){
        error = setPointSensor - AnalogDetect;
        pValue = Kp * error;
        derivatifValue = ((Kd/Ts)*(error-lastError));
        lastError=error;
        proporsionalDerivatif = pValue + derivatifValue;
        Rpwm = RsetPointPwm + proporsionalDerivatif; // Proporsional -
Derivatif (PD)
        Lpwm = LsetPointPwm - proporsionalDerivatif;

        goForward();
        // Rpwm = RsetPointPwm + pValue; // Proporsional
        // Lpwm = LsetPointPwm - pValue;
        // Serial.print("PD: ");
        // Serial.print(proporsionalDerivatif);
        // Serial.print(" P: ");
        // Serial.print(pValue);
        // Serial.print(" D: ");
        // Serial.print(derivatifValue);
        // Serial.print(" Error: ");
    }
}

```

```

    // Serial.print(error);
    // Serial.print(" trackDetect: ");
    // Serial.print(trackDetect);
    // Serial.print(" AnalogOutput: ");
    // Serial.print(AnalogDetect);
    // Serial.print(" Rpwm: ");
    // Serial.print(Rpwm);
    // Serial.print(" Lpwm: ");
    // Serial.println(Lpwm);
}
else if (trackDetect==0){
    stop();
}
}
void ultraSonic(){

```

```

Wire.beginTransmission(113);
Wire.write(byte(0x00));
Wire.write(byte(0x50));
Wire.endTransmission();

```

```

delay(65);

```

```

Wire.beginTransmission(113);
Wire.write(byte(0x02));
Wire.endTransmission();
Wire.requestFrom(113, 2);

```

```

if (2 <= Wire.available()){
    reading=Wire.read();
    reading=reading << 8;
    reading |= Wire.read();
}

```

```
    Serial.print("Ultrasonic: ");  
    Serial.println(reading);  
}  
delay(250);  
}
```

```
void setup() {  
    Wire.beginTransmission(113);  
    Master.begin(115200);  
  
    #ifdef printToPC  
    Serial.begin(115200);  
    #endif  
  
    #ifdef loadMaster  
    Serial.begin(128000);  
    Serial.println("CLEARDATA");  
    #endif  
  
    #ifdef loadFuzzy  
    Serial.println("LABEL,CLOCK,DeltaError,Error,Output Fuzzy Right  
PWM,Output Fuzzy Left PWM");  
    #endif  
  
    #ifdef loadKinematic  
    Serial.println("LABEL,CLOCK,xPositionInCM,yPositionInCM");  
    #endif  
  
    #ifdef loadTesting
```

```
Serial.println("LABEL,CLOCK,DeltaError,Error,xPositionInCM,yPositionInCM,LeftEncoder,RightEncoder,Output Fuzzy Right PWM,Output Fuzzy Left PWM");
```

```
#endif
```

```
pinMode(MRF, OUTPUT);
```

```
pinMode(MRB, OUTPUT);
```

```
pinMode(MLF, OUTPUT);
```

```
pinMode(MLB, OUTPUT);
```

```
pinMode(UB, OUTPUT);
```

```
pinMode(UF, OUTPUT);
```

```
pinMode(BUZZ, OUTPUT);
```

```
pinMode(ACTIVE_RL, OUTPUT);
```

```
pinMode(CH2, INPUT);
```

```
pinMode(CH3, INPUT);
```

```
pinMode(CH4, INPUT);
```

```
pinMode(CH5, INPUT);
```

```
pinMode(CH6, INPUT);
```

```
//Buzzer();
```

```
Buzzer();
```

```
relaySteady();
```

```
uvActivationReset();
```

```
}
```

```
void loop() {
```

```
    // checkLoopTimer();
```

```
    getFuzzy();
```

```
    //testing();
```

```

    //uvActivation();
    //ultraSonic();
    //PID();
    //getSensor();
    //voltageDivider();
    // rcMode();
    //kinematic();
}

#elif slave_code
// ----- Slave -----

// Open The Command if want to upload on Slave
#include <Arduino.h>
#include <Wire.h>
#include <kinematics_RS.h>

// Slave to Master
#define Slave Serial3

// Slave Read Magnetic Sensor
#define ForkRight 12
#define ForkLeft 13
#define RightMarker 7
#define LeftMarker 6
#define TrackPresent A2
#define AnalogOut A0
// Encoder
// enum {ENC_STOP, ENC_CLOCKWISE_ROTATION,
ENC_COUNTERCLOCKWISE_ROTATION};
// const byte SINPin = 4;
// const byte COSPin = 2;

```

```

// const byte R_SINPin = 5;
// const byte R_COSPin = 3;
// volatile byte encoder_state = ENC_STOP;
// volatile int encoder_position = 0;
// volatile int encoder_oldpos = 0;
// volatile byte R_encoder_state = ENC_STOP;
// volatile int R_encoder_position = 0;
// volatile int R_encoder_oldpos = 0;
unsigned long startTimeMillis,
           endTimeMillis,
           loopTimer;

#define phi 3.14285714286
#define rangeBetweenWheels 34.5
// #define wheelRadius 7.45 //6" to cm
#define wheelRadius 15 //6" to cm
#define encoderPPR 600
#define encoderCPR (4 * encoderPPR)
#define wheelCircumference ((wheelRadius * phi) / encoderPPR)
#define xEncoder 2.108012
#define encoderLeftA 2
#define encoderLeftB 4
#define encoderRightA 5
#define encoderRightB 3
kinematicAUMR_RS kinematics(encoderLeftA, encoderLeftB,
encoderRightA, encoderRightB, wheelRadius, rangeBetweenWheels,
encoderPPR);
long xPositionInCM,
     yPositionInCM,
     thetaPositionInDegree,

     leftLinearSpeed,

```



```

        rightLinearSpeed;

float leftPosition,
        rightPosition;

int leftCounter,
        rightCounter;

void checkLoopTimer()
{
    endTimeMillis = millis();
    loopTimer = (endTimeMillis - startTimeMillis);
    startTimeMillis = millis();
    Serial.print(loopTimer);
    Serial.print(" \t");
}

void ForkRightDetect(){
    digitalWrite(ForkRight, HIGH);
    digitalWrite(ForkRight, LOW);
}

void ForkLeftDetect(){
    digitalWrite(ForkLeft, HIGH);
    digitalWrite(ForkLeft, LOW);
}

int trackDetect(){
    int TrackPresentValue = digitalRead(TrackPresent);
    Serial.print(" TP: ");
    Serial.print(TrackPresentValue);
    return TrackPresentValue;
}

int AnalogOutDetect(){
    int AnalogOutValue = analogRead(AnalogOut);

```

```

    Serial.print(" AO: ");
    Serial.print(AnalogOutValue);

    return AnalogOutValue;
}

int RightMarkerDetect() {
    int RightMarkerValue = digitalRead(RightMarker);
    Serial.print(" RM: ");
    Serial.print(RightMarkerValue);
    return RightMarkerValue;
}

int LeftMarkerDetect(){
    int LeftMarkerValue = digitalRead(LeftMarker);
    Serial.print(" LM: ");
    Serial.print(LeftMarkerValue);
    return LeftMarkerValue;
}

void sendData(){
    int delaySending = 9;
    Slave.print("!");
    delay(delaySending);
    Slave.print(trackDetect());
    delay(delaySending);
    Slave.print("@");
    delay(delaySending);
    Slave.print(AnalogOutDetect());
    delay(delaySending);
    Slave.print("&");
    delay(delaySending);
    Slave.print((long)xPositionInCM);
    delay(delaySending);
    Slave.print("=");
}

```

```

delay(delaySending);
Slave.print((long)yPositionInCM);
delay(delaySending);
Slave.print("");
delay(delaySending);
Slave.print((long)thetaPositionInDegree);
delay(delaySending);

// Slave.print("#");
// delay(delaySending);
// Slave.print(RightMarkerDetect());
// delay(delaySending);

// Slave.print("$");
// delay(delaySending);
// Slave.print(LeftMarkerDetect());
// delay(delaySending);
// Slave.print("%");
// delay(delaySending);
// Slave.print((int)leftLinearSpeed);
// delay(delaySending);
// Slave.print("^");
// delay(delaySending);
// Slave.print((int)rightLinearSpeed);
// delay(delaySending);

// Encoder
// Slave.print("%");
// Slave.print(R_encoder_position);
// Slave.print("^");
// Slave.print(encoder_position);
}

```

```

// void kinematicToMaster()
// {
// // Delay to adjust speed of the master microcontroller

// // Print to master

// }
void getKinematicData(){
    Serial.print("xPostCM: ");
    Serial.print((long)xPositionInCM);
    Serial.print(" yPostCM: ");
    Serial.print((long)yPositionInCM);
    Serial.print(" ThetPostDeg: ");
    Serial.print((long)thetaPositionInDegree);
    Serial.print(" LeftLinSpd: ");
    Serial.print(leftLinearSpeed);
    Serial.print(" RghtLnrSpd: ");
    Serial.println(rightLinearSpeed);
}
void processLeftForward()
{
    kinematics.processLeftForward();
}
void processLeftBackward()
{
    kinematics.processLeftBackward();
}
void processRightForward()
{
    kinematics.processRightForward();
}
void processRightBackward()

```

```

{
    kinematics.processRightBackward();
}

void setup() {
    Slave.begin(115200);
    Serial.begin(115200);
    pinMode(ForkRight, OUTPUT);
    pinMode(ForkLeft, OUTPUT);
    pinMode(RightMarker, INPUT);
    pinMode(LeftMarker, INPUT);
    pinMode(TrackPresent, INPUT);
    pinMode(AnalogOut, INPUT);
    // pinMode(COSPin, INPUT_PULLUP);
    // pinMode(SINPin, INPUT);
    // pinMode(R_COSPin, INPUT_PULLUP);
    // pinMode(R_SINPin, INPUT);
    // attachInterrupt(digitalPinToInterrupt(COSPin), encoder_isr,
    RISING);
    // attachInterrupt(digitalPinToInterrupt(R_COSPin), R_encoder_isr,
    RISING);
    pinMode(encoderLeftA, INPUT_PULLUP);
    pinMode(encoderLeftB, INPUT_PULLUP);
    pinMode(encoderRightA, INPUT_PULLUP);
    pinMode(encoderRightB, INPUT_PULLUP);
    attachInterrupt(digitalPinToInterrupt(encoderLeftA),
    processLeftForward, CHANGE);
    attachInterrupt(digitalPinToInterrupt(encoderLeftB),
    processLeftBackward, CHANGE);
    attachInterrupt(digitalPinToInterrupt(encoderRightA),
    processRightForward, CHANGE);

```

```

        attachInterrupt(digitalPinToInterrupt(encoderRightB),
processRightBackward, CHANGE);
    }
    void loop() {
        //encoder()
        checkLoopTimer();
        kinematics.calculate();
        yPositionInCM = kinematics.getYPositionInCM();
        xPositionInCM = kinematics.getXPositionInCM();
        thetaPositionInDegree = kinematics.getThetaInDegree();
        leftLinearSpeed = kinematics.getLeftSpeed();
        rightLinearSpeed = kinematics.getRightSpeed();
        rightPosition = kinematics.getRightPositionInCM();
        leftPosition = kinematics.getLeftPositionInCM();

        Serial.print(" rightPost: ");
        Serial.print(rightPosition);
        Serial.print(" leftPost: ");
        Serial.print(leftPosition);

        //print kinematics data
        getKinematicData();

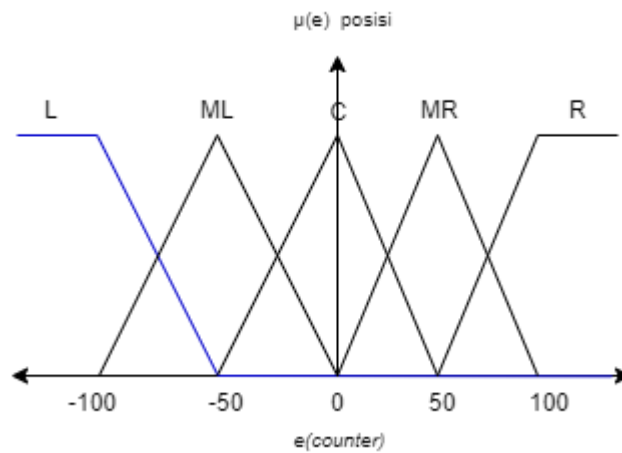
        //send Magnetic data to master
        sendData();
    }
#endif

```

❖ Lampiran 2 Penurunan Rumus FLC

- A. Himpunan fungsi keanggotaan dari *error* (e) posisi yang dimana dapat dilihat pada Gambar III-18. Untuk menurunkan fungsi keanggotaan digunakan persamaan sebagai berikut :

1. Keanggotaan *Left*



$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

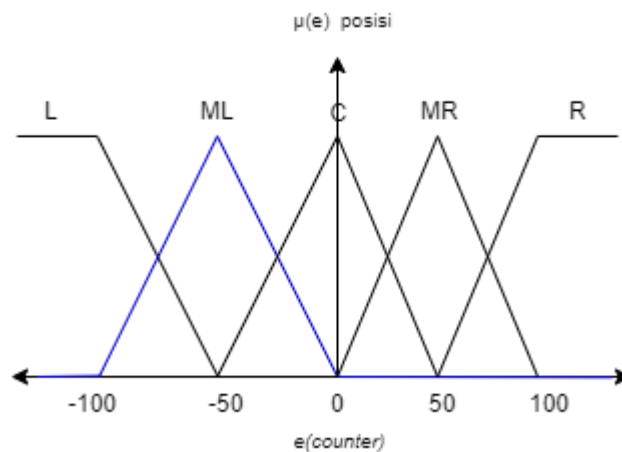
$$\frac{y - 1}{0 - 1} = \frac{-x - (-100)}{-50 - (-100)}$$

$$y = \frac{x - (-50)}{50}$$

Mendapatkan hasil persamaan fungsi keanggotaan *left* sebagai berikut.

$$\mu_L(x) = \begin{cases} 1 & , x \leq -100 \\ \frac{x - (-50)}{50} & , -100 \leq x \leq -50 \\ 0 & , x \geq -50 \end{cases}$$

2. Keanggotaan *Most Left*



$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

$$\frac{y - 1}{0 - 1} = \frac{x - (-100)}{-50 - (-100)}$$

$$y = \frac{x + 100}{50}$$

Penurunan bagian garis kedua.

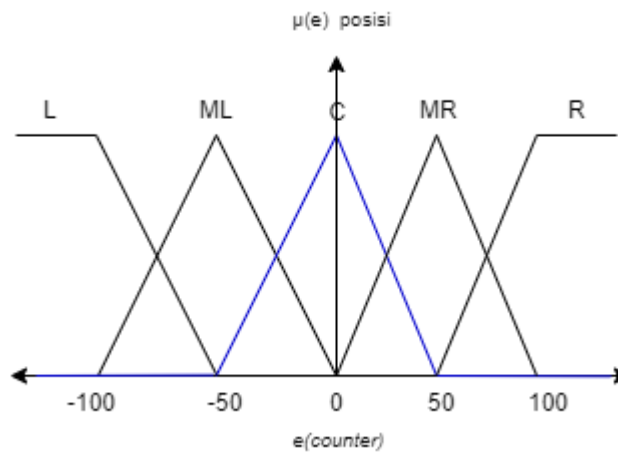
$$\frac{y - 1}{0 - 1} = \frac{x - (0)}{0 - (-50)}$$

$$y = \frac{-x}{50}$$

Mendapatkan hasil persamaan fungsi keanggotaan *Most Left* sebagai berikut.

$$\mu_{ML}(x) = \begin{cases} 0 & , \quad x \leq -100 \\ \frac{x+100}{50} & , \quad -100 < x \leq -50 \\ \frac{-x}{50} & , \quad -50 < x \leq 0 \\ 0 & , \quad x > 0 \end{cases}$$

3. Keanggotaan *Center*



$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

$$\frac{y - 1}{0 - 1} = \frac{x - (-50)}{0 - (-50)}$$

$$y = \frac{x + 50}{50}$$

Penurunan bagian garis kedua.

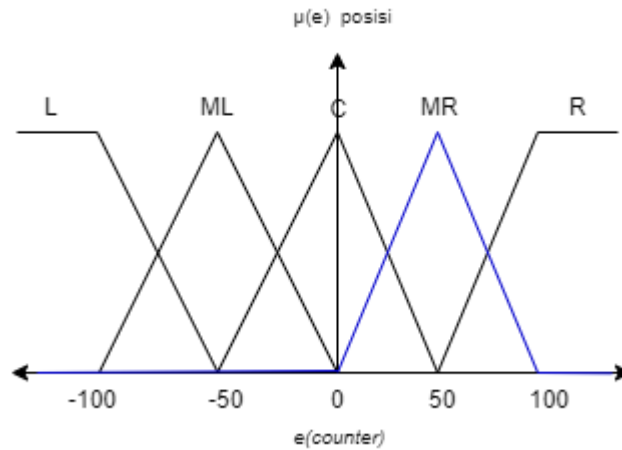
$$\frac{y - 1}{0 - 1} = \frac{x - (50)}{50 - (0)}$$

$$y = \frac{-x + 50}{50}$$

Persamaan fungsi keanggotaan *Center* adalah sebagai berikut.

$$\mu_c(x) = \begin{cases} 0 & , \quad x \leq -50 \\ \frac{x+50}{50} & , \quad -50 < x \leq 0 \\ \frac{-x+50}{50} & , \quad 0 < x \leq 50 \\ 0 & , \quad x > 50 \end{cases}$$

4. Fungsi Keanggotaan *Most Right*



$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

$$\frac{y - 1}{0 - 1} = \frac{x - (0)}{50 - (0)}$$

$$y = \frac{x}{50}$$

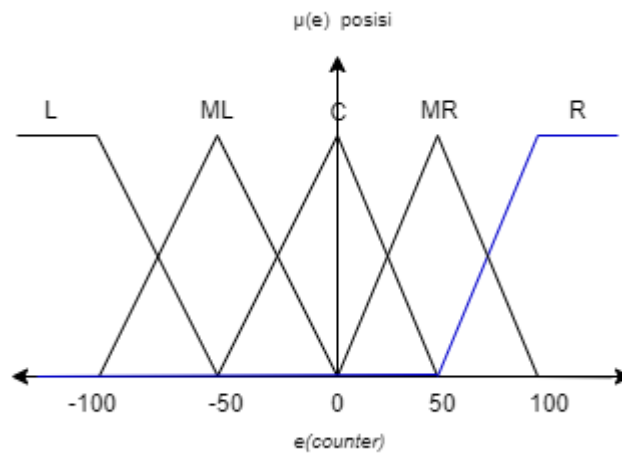
Penurunan bagian garis kedua.

$$\frac{y - 1}{0 - 1} = \frac{x - (100)}{100 - (50)}$$

$$y = \frac{-x + 100}{50}$$

$$\mu_{MR}(x) = \begin{cases} 0 & , \quad x \leq 0 \\ \frac{x}{50} & , \quad 0 < x \leq 50 \\ \frac{-x + 100}{50} & , \quad 50 < x \leq 100 \\ 0 & , \quad x > 100 \end{cases}$$

5. Fungsi Keanggotaan *Right*



$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

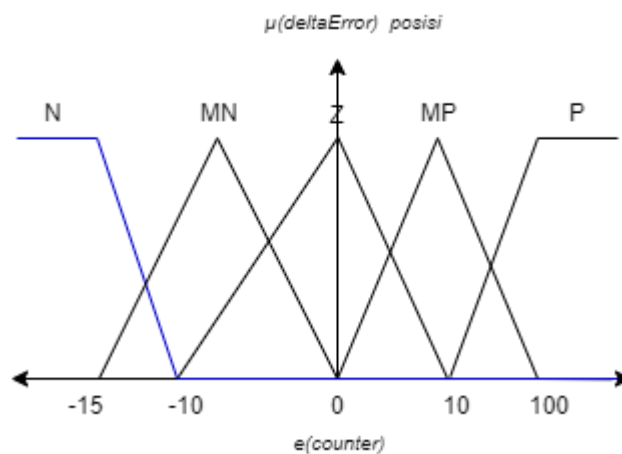
$$\frac{y - 1}{0 - 1} = \frac{x - (50)}{100 - (50)}$$

$$y = \frac{x - 50}{50}$$

$$\mu_R(x) = \begin{cases} 0 & , x \leq 50 \\ \frac{x - 50}{50} & , 50 < x \leq 100 \\ 1 & , x > 100 \end{cases}$$

B. Himpunan fungsi keanggotaan *delta error* (Δe) dapat dilihat pada Gambar III-19. Berikut persamaan dan penurunan rumus keanggotaannya.

1. Fungsi keanggotaan Negative



$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

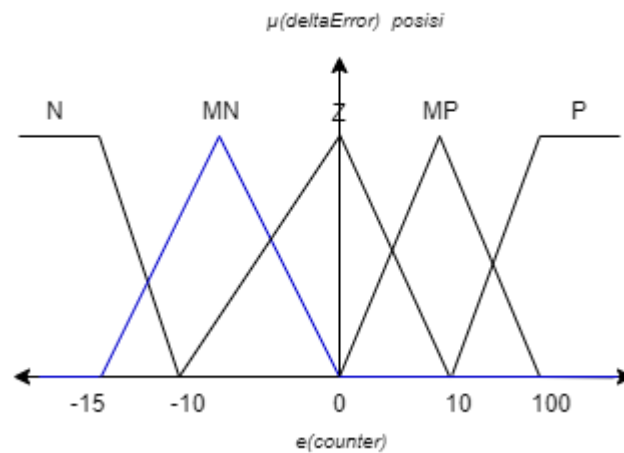
$$\frac{y - 1}{0 - 1} = \frac{x - (-15)}{-10 - (-15)}$$

$$y = \frac{-x - 10}{5}$$

Didapatkan hasil dari persamaan fungsi *Negative* pada *delta error* adalah sebagai berikut.

$$\mu_N(x) = \begin{cases} 1 & , x \leq -15 \\ \frac{-x - 10}{5} & , -15 < x \leq -10 \\ 0 & , x > -10 \end{cases}$$

2. Fungsi keanggotaan *Most Negative*



$$\begin{aligned} \frac{y - y_1}{y_2 - y_1} &= \frac{x - x_1}{x_2 - x_1} \\ \frac{y - 0}{1 - 0} &= \frac{x - (-15)}{-10 - (-15)} \\ y &= \frac{x + 15}{5} \end{aligned}$$

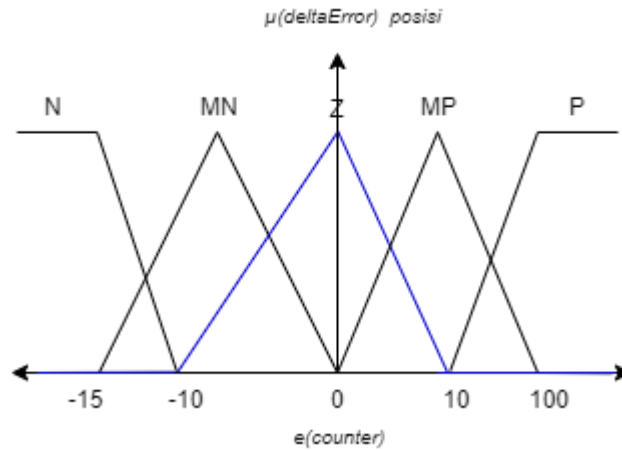
Penurunan bagian garis kedua.

$$\begin{aligned} \frac{y - y_1}{y_2 - y_1} &= \frac{x - x_1}{x_2 - x_1} \\ \frac{y - 1}{0 - 1} &= \frac{x - (-10)}{0 - (-10)} \\ y &= \frac{-x}{10} \end{aligned}$$

Didapatkan hasil dari persamaan fungsi *Most Negative* pada *delta error* adalah sebagai berikut.

$$\mu_{MN}(x) = \begin{cases} 0 & , \quad x \leq -15 \\ \frac{x+15}{5} & , \quad -15 < x \leq -10 \\ \frac{-x}{10} & , \quad -10 < x \leq 0 \\ 0 & , \quad x > 0 \end{cases}$$

3. Fungsi keanggotaan Zero



$$\begin{aligned} \frac{y - y_1}{y_2 - y_1} &= \frac{x - x_1}{x_2 - x_1} \\ \frac{y - 0}{1 - 0} &= \frac{x - (-10)}{0 - (-10)} \\ y &= \frac{x + 10}{10} \end{aligned}$$

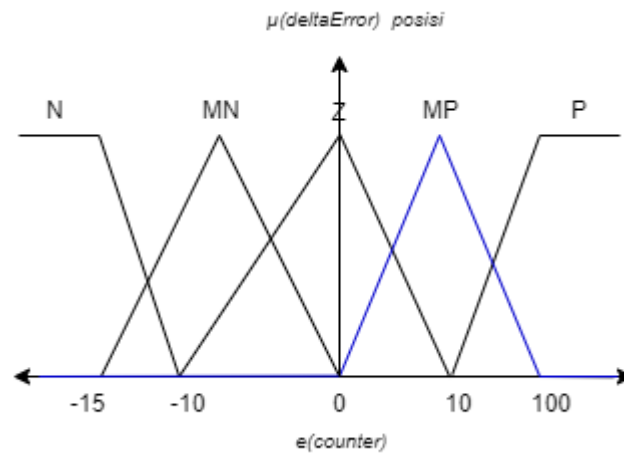
Penurunan bagian garis kedua.

$$\begin{aligned} \frac{y - y_1}{y_2 - y_1} &= \frac{x - x_1}{x_2 - x_1} \\ \frac{y - 1}{0 - 1} &= \frac{x - 0}{10 - 0} \\ y &= \frac{-x + 10}{10} \end{aligned}$$

Didapatkan hasil dari persamaan fungsi *Zero* pada *delta error* adalah sebagai berikut.

$$\mu_z(x) = \begin{cases} 0 & ; \quad x \leq -10 \\ \frac{x+10}{10} & ; \quad -10 < x \leq 0 \\ \frac{-x+10}{10} & ; \quad 0 < x \leq 10 \\ 0 & ; \quad x > 10 \end{cases}$$

4. Fungsi keanggotaan *Most Positive*



$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

$$\frac{y - 0}{1 - 0} = \frac{x - 0}{10 - 0}$$

$$y = \frac{x}{10}$$

Penurunan bagian garis kedua

$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

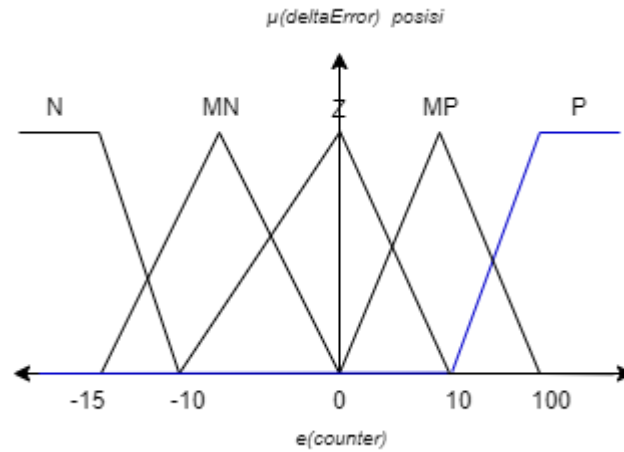
$$\frac{y - 1}{0 - 1} = \frac{x - 10}{15 - 10}$$

$$y = \frac{-x + 15}{5}$$

Didapatkan hasil dari persamaan fungsi *Most Positive* pada *delta error* adalah sebagai berikut.

$$\mu_{MP}(x) = \begin{cases} 0 & ; \quad x \leq 0 \\ \frac{x}{10} & ; \quad 0 < x \leq 10 \\ \frac{-x + 15}{5} & ; \quad 10 < x \leq 15 \\ 0 & ; \quad x > 15 \end{cases}$$

5. Fungsi Keanggotaan *Positive*



$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$

$$\frac{y - 0}{1 - 0} = \frac{x - (10)}{15 - (10)}$$

$$y = \frac{x - 10}{5}$$

Didapatkan hasil dari persamaan fungsi *Most Positive* pada *delta error* adalah sebagai berikut.

$$\mu_P(x) = \begin{cases} 0 & ; x \leq 10 \\ \frac{x - 10}{15} & ; 10 < x \leq 15 \\ 1 & ; x > 15 \end{cases}$$

❖ Lampiran 3 Penurunan persamaan *Forward Kinematics of Differential Drive*

Saat ICC sudah diketahui pada Persamaan II-3 dan perubahan posisi terhadap (X', Y', θ') maka dapat diturunkan dengan sistem yang dibuat.

2. Persamaan X'

$$\begin{aligned} X' &= (\cos(\omega\delta t) * (x - ICCx)) - (\sin(\omega\delta t) * (y - ICCy)) + 0 \\ &= \cos(\omega\delta t) * (x - x - R\sin(\theta)) - (\sin(\omega\delta t) * (y - y + R\cos(\theta)) + 0 \\ &= \cos(\omega\delta t) * (-R\sin(\theta)) - (\sin(\omega\delta t) * (-R\cos(\theta)) + 0 \\ &= R[-\cos(\omega\delta t) * (\sin(\theta))] + R[(\sin(\omega\delta t) * (\cos(\theta)))] + 0 \\ &= R[-\cos(\omega\delta t) * \sin(\theta) + \sin(\omega\delta t) * \cos(\theta)] + 0 \\ &= R[\sin(\theta + \omega\delta t)] + 0 \end{aligned}$$

3. Persamaan Y'

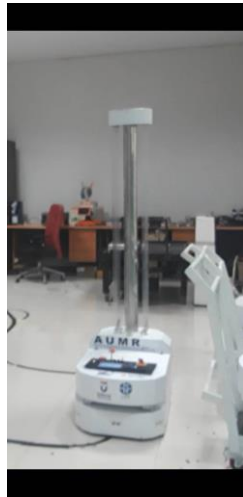
$$\begin{aligned}
Y' &= (\sin(\omega\delta t) * (x - ICCx)) - (\cos(\omega\delta t) * (y - ICCy)) + 0 \\
&= (\sin(\omega\delta t) * (x - x - R\sin(\theta))) + (\cos(\omega\delta t) * (y - y + R\cos(\theta))) + 0 \\
&= (\sin(\omega\delta t) * (-R\sin(\theta))) + (\cos(\omega\delta t) * (-R\cos(\theta))) + 0 \\
&= R[\sin(\omega\delta t) * \sin(\theta) + \cos(\omega\delta t) * \cos(\theta)] + 0 \\
&= R[\cos(\theta - \omega\delta t)] + 0
\end{aligned}$$

4. Persamaan θ'

$$\theta' = (0 * 0 * 0) + (\omega\delta t)$$

$$\theta' = (\omega\delta t)$$

❖ Lampiran 4 Gambar alat



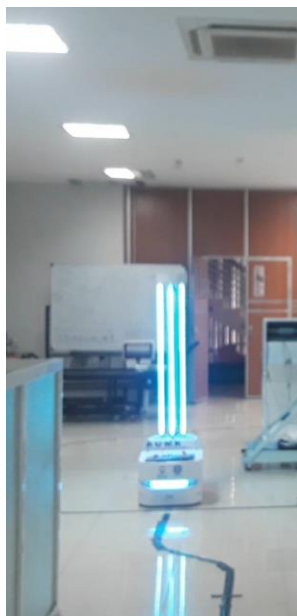
Gambar 1. Tampak Sisi Depan Robot



Gambar 2. Tampak Sisi Samping Robot



Gambar 3. Tampak Sisi Belakang Robot



Gambar 4. Robot Menghidupkan Lampu UV