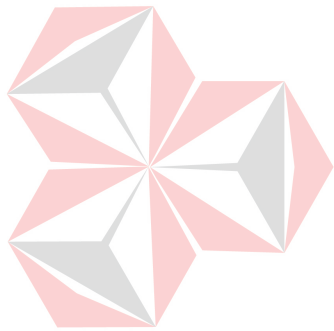


**SIMULASI *TRAJECTORY PLANNING* DAN PEMBENTUKAN
FORMASI PADA ROBOT *OBSTACLE AVOIDANCE***

TUGAS AKHIR



Nama : Aditya Prima Suparno

Nim : 10.41020.0096

Program : S1 (Strata Satu)

Jurusan : Sistem Komputer

**SEKOLAH TINGGI
MANAJEMEN INFORMATIKA & TEKNIK KOMPUTER
SURABAYA**

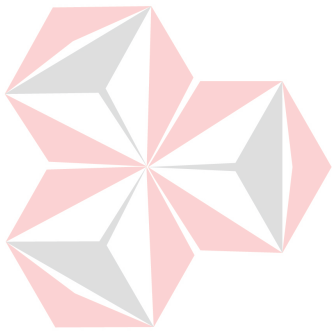
2014

**SIMULASI *TRAJECTORY PLANNING* DAN PEMBENTUKAN
FORMASI PADA ROBOT *OBSTACLE AVOIDANCE***

TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk menyelesaikan

Program Sarjana Komputer



Oleh:

Nama : Aditya Prima Suparno

Nim : 10.41020.0096

Program : S1 (Strata Satu)

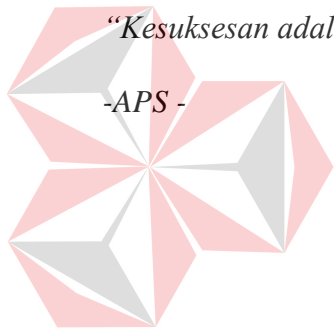
Jurusan : Sistem Komputer

SEKOLAH TINGGI

MANAJEMEN INFORMATIKA DAN TEKNIK KOMPUTER

SURABAYA

2014



“Kesuksesan adalah buah dari kerja keras, kesabaran, dan doa”

-APS-

UNIVERSITAS
Dinamika

Bismillahirrohmanirrohim, dengan mengucapkan basmalah dan syukur

Alhamdulillah, dengan bangga saya ingin mempersembahkan skripsi ini untuk:

Ayah, Ibu, dan semua keluarga tercinta,

Yang selalu mendukung, memotivasi dan mendoakan saya dalam doa-doanya.

Untuk teman-teman seperjuangan,
yang selalu mendukung dan memberikan inspirasi.



UNIVERSITAS
Dinamika

Tugas Akhir

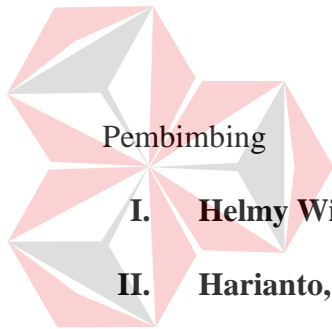
SIMULASI *TRAJECTORY PLANNING* DAN PEMBENTUKAN FORMASI PADA ROBOT *OBSTACLE AVOIDANCE*

dipersiapkan dan disusun oleh

Aditya Prima Suparno

NIM : 10.41020.0096

Telah diperiksa, diuji dan disetujui oleh Dewan Penguji
pada : 29 Agustus 2014



Susunan Dewan Penguji

Pembimbing

I. Helmy Widyantara, S.Kom., M.Eng. _____

II. Harianto, S.Kom., M.Eng. _____

Penguji

I. Dr. Jusak _____

II. Susijanto Tri Rasmana, S.Kom., M.T. _____

Tugas Akhir ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Sarjana

Pantjawati Sudarmaningtyas, S.Kom., M.Eng.
Pembantu Ketua Bidang Akademik

PERNYATAAN

Dengan ini menyatakan dengan benar, bahwa Tugas Akhir ini adalah asli karya saya, bukan plagiat baik sebagian maupun apalagi keseluruhan. Karya atau pendapat orang lain yang ada dalam tugas akhir ini adalah semata hanya rujukan yang dicantumkan dalam daftar pustaka saya. Apabila dikemudian hari ditemukan adanya tindakan plagiat pada karya tugas akhir ini, maka saya bersedia untuk dilakukan pencabutan terhadap gelar kesarjanaan yang telah diberikan kepada saya.



UNIVERSITAS
Dinamika

Surabaya, 29 Agustus 2014

Aditya Prima Suparno
NIM : 10.41020.0096

ABSTRAK

Sistem pertahanan negara telah mengembangkan teknologi yang memiliki kecerdasan dan keakuratan yang cukup tinggi dalam membantu mempertahankan negara agar lebih optimal. Sebagai contoh kendaraan tempur Batalyon Kavaleri yang digunakan untuk menyerang dan menghancurkan markas musuh dengan membentuk formasi khusus dalam pergerakan menuju posisi penyerangan.

Berdasarkan perkembangan teknologi yang telah ada, dalam penelitian ini dibuat simulasi pergerakan koloni robot sebelum tahap pengimplementasiannya secara nyata. Dalam simulasi, pergerakan koloni robot dibentuk menyerupai formasi Batalyon Kavaleri ketika menentukan jalur menuju posisi penyerangan. Koloni ini dibuat menggunakan 3 robot simulasi. Metode yang digunakan meliputi fuzzy Sugeno, formasi ketetapan robot dan *obstacle avoidance*.

Pembentukan formasi dapat dilakukan berdasarkan pengiriman koordinat tujuan yang dilakukan oleh pemimpin robot, koordinasi kecepatan yang dilakukan berdasarkan jarak dari masing-masing robot terhadap robot lain dan kemampuan robot untuk memberikan keputusan pergerakan terhadap dirinya sendiri ketika melewati *obstacle*. Rata-rata kesalahan pergerakan *follower* A dalam perjalanannya mengikuti koordinat target yang dikirimkan oleh robot pemimpin adalah 4,778% terhadap sumbu X, dan 0,807% terhadap sumbu Y. Rata-rata kesalahan pergerakan *follower* B dalam perjalanannya mengikuti koordinat target yang dikirimkan oleh robot pemimpin adalah 4,893% terhadap sumbu X, dan 1,314% terhadap sumbu Y.

KATA PENGANTAR

Pertama-tama penulis panjatkan puji syukur kehadirat Alloh SWT, karena berkat limpahan Rahmat dan Karunia-Nya penulis dapat menyelesaikan Tugas Akhir yang merupakan persyaratan dalam menyelesaikan Program Studi Strata Satu Jurusan Sistem Komputer di Sekolah Tinggi Manajemen Informatika dan Teknik Komputer (STIKOM) Surabaya. Tugas Akhir ini berjudul “Simulasi *Trajectory Planning* dan Pembentukan Formasi pada Robot *Obstacle Avoidance*”.

Pada kesempatan ini, penulis juga mengucapkan terima kasih kepada :

1. Bapak dan Ibu, yang telah memberikan segalanya demi cita-cita penulis.
2. Sodara - Sodara tercinta yang telah membantu memberi dukungan moril dan doanya, serta telah membantu dalam menjalani hidup di kota Surabaya.
3. Bpk. Prof. Dr. Budi Jatmiko, M.Pd selaku ketua STIKOM Surabaya yang peduli terhadap program jurusan Sistem Komputer.
4. Helmy Widyantara, S.Kom., M.Eng. selaku Dosen Pembimbing I, dan Hariyanto, S.Kom., M.Eng. selaku Dosen Pembimbing II, yang telah membimbing penulis dalam menyelesaikan Tugas Akhir ini.
5. Dr. Jusak selaku Dosen Penguji I, dan Susijanto Tri Rasmana, S.Kom., M.T. selaku Dosen Penguji II, yang telah memberikan masukan dalam penyusunan buku Tugas Akhir.
6. Segenap Dosen Pengajar program studi S-1 Sistem Komputer.
7. Perpustakaan STIKOM Surabaya, yang telah membantu menyediakan buku-buku bagi keperluan penulis.

8. Sahabat – sahabatku yang telah senantiasa saling menopang penulis dalam keadaan suka maupun duka.
9. Teman - teman penulis di Jurusan Sistem Komputer yang telah membantu dalam perkuliahan selama di STIKOM Surabaya.
10. Teman – teman seperjuangan angkatan 2010 Jurusan S1 Sistem Komputer.
11. Teman – teman tim robot STIKOM Surabaya.
12. Teman – teman di non Jurusan S1 Sistem Komputer.

Banyak hal dalam laporan Tugas Akhir ini yang masih perlu diperbaiki lagi. Oleh karena itu penulis mengharapkan saran dan kritik yang dapat membangun dari semua pihak agar dapat menyempurnakan penulisan ini kedepannya. Penulis juga memohon maaf yang sebesar-besarnya jika terdapat kata-kata yang salah serta menyinggung perasaan pembaca. Akhir kata penulis ucapkan banyak-banyak terima kasih yang sebesar-besarnya kepada para pembaca, semoga tulisan ini dapat bermanfaat bagi para pembaca.

Surabaya, 29 Agustus 2014

Penulis

DAFTAR ISI

ABSTRAK	vii
KATA PENGANTAR	viii
DAFTAR ISI	x
DAFTAR GAMBAR	xiii
DAFTAR TABEL	xvi
BAB I	
PENDAHULUAN	1
1.1 Latar Belakang Masalah	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	2
1.4 Tujuan	3
1.5 Sistematika Penulisan	3
BAB II	
LANDASAN TEORI	5
2.1 Visual Basic.Net	5
2.1.1 Variabel	5
2.1.2 Tipe Data	6
2.1.3 Operator Matematika	7
2.1.4 Koordinat Form	8
2.2 Grafis	8
2.2.1 GDI+	8
2.2.2 Koordinat Tampilan Layar	9

2.2.3 Gambar Grafis Vektor 2 Dimensi	10
2.3 Penampilan Gambar	10
2.4 Formasi Ketetapan Robot	11
2.5 Posisi Awal Robot	12
2.6 Kesalahan Arah	14
2.7 <i>Obstacle Avoidance</i>	15
2.7.1 Metode Analisa Ujung Tepi <i>Object</i>	15
2.7.2 Kepastian Gerak Terhadap <i>Obstacle</i>	16
2.7.3 Metode <i>Support Relevansi Obstacle Avoidance</i>	18
2.8 <i>Fuzzy Logic</i>	19
2.8.1 Himpunan Fuzzy	20
2.8.2 Fungsi Keanggotaan	20
2.8.3 Operator Dasar	20
2.8.4 Penalaran Monoton	21
2.8.5 Fungsi Implikasi	22
2.8.6 Metode Sugeno	22

BAB III

METODE PENELITIAN	24
3.1. Alat dan Bahan Penelitian	27
3.1.1 Alat Penelitian	27
3.1.2 Bahan Penelitian	27
3.2. Jalan Penelitian	28
3.2.1 Pengumpulan Literatur	28
3.2.2 Perancangan Perangkat Lunak	28

BAB IV

PENGUJIAN SISTEM	55
4.1. Pengujian Perangkat Lunak	55
4.1.1. Pengujian Target Robot	55
4.1.2. Pengujian Derajat Sensor	57
4.1.3. Pengujian Pembentukan Formasi Robot	59
4.1.4. Pengujian Pergerakan Kemudi	61
4.1.5. Pengujian Komunikasi USART	62
4.1.6. Pengujian Metode Fuzzy	64
4.1.7. Pengujian Pembentukan Formasi Robot ke Formasi Awal	72
4.2. Pengujian Sistem Secara Keseluruhan	76
4.2.1 Gerak Robot <i>Follower</i> Terhadap Koordinat Target	76
BAB V	
PENUTUP	95
5.1. Kesimpulan	95
5.2. Saran	96
DAFTAR PUSTAKA	98
LAMPIRAN	100
BIODATA PENULIS	113

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Sistem pertahanan negara telah mengembangkan teknologi yang memiliki kecerdasan dan keakuratan yang cukup tinggi dalam membantu mempertahankan negara agar lebih optimal. Sebagai contoh kendaraan tempur Batalyon Kavaleri yang digunakan untuk menyerang dan menghancurkan markas musuh, robot penjinak bom yang dirancang untuk menjamin keselamatan para petugas dalam penjinakan bom, dan teknologi yang dapat membantu tugas-tugas yang cukup beresiko lainnya.

Perkembangan teknologi seperti robot, dalam pembuatannya telah melewati beberapa tahapan tertentu, terutama proses pergerakan yang dilakukan dengan menggunakan simulasi. Simulasi sendiri merupakan proses peniruan dari sesuatu yang nyata beserta keadaan sekelilingnya (*state of affairs*). Hal ini bertujuan untuk mengetahui pergerakan yang diinginkan sebelum tahap pengimplementasiannya. Dalam Perkembangan teknologi di dunia militer tentunya dibutuhkan robot yang mampu berjalan secara berkoloni dan membentuk formasi pertahanan seperti Batalyon Kavaleri, serta dapat beradaptasi dengan cepat terhadap lingkungan yang belum diketahui sehingga apabila terdapat halangan yang mendadak, robot akan dapat menghindari dan dapat menentukan jalur yang sesuai terhadap dirinya. Dalam hal tersebut telah dikembangkan sistem penghindar halangan yang disebut *obstacle avoidance*.

Dari perkembangan-perkembangan yang telah ada, peneliti mencoba mengembangkan sistem simulasi *trajectory planning* dan pembentukan formasi pada robot *obstacle avoidance* dengan menggunakan program visual basic.net. Simulasi ini bertujuan untuk mengetahui respon formasi robot, sehingga robot dapat kembali ke formasi awal setelah melewati *obstacle* (halangan), formasi robot dapat melakukan penyesuaian terhadap *obstacle* dan robot dapat berjalan menuju target yang telah ditentukan (*trajectory planning*). Dalam mengembangkan sistem simulasi pergerakan robot, peneliti menggunakan *Fuzzy Logic Control*. Fuzzy digunakan sebagai proses pergerakan robot, sehingga pergerakan robot akan lebih halus (*smooth*) dalam menghindari *obstacle*.

1.2 Rumusan Masalah

Adapun permasalahan yang akan dihadapi oleh peneliti ke depannya dalam proses pengerjaan tugas akhir ini adalah:

1. Bagaimana menentukan dan mendesain metode *fuzzy logic* yang digunakan untuk pengendalian kecepatan serta sudut belok robot saat terdapat *obstacle* (halangan)?
2. Bagaimana cara untuk mengatur formasi robot agar dapat berjalan secara teratur sesuai intruksi robot utama dan dapat membentuk formasi awal setelah melewati *obstacle*?

1.3 Batasan Masalah

Dalam perancangan dan pembuatan alat ini, terdapat beberapa batasan masalah, antara lain :

1. Menggunakan 3 robot simulasi.
2. Dimensi minimal objek penghalang 60 x 60 *pixel*.

3. Dimensi robot 15 x 15 *pixel*.
4. Jarak baca sensor *ultrasonic* 60 *pixel*.

1.4 Tujuan

Adapun tujuan dari pembuatan simulasi ini yaitu:

1. Menentukan dan mendesain metode *fuzzy logic* yang digunakan untuk pengendalian kecepatan serta sudut belok robot saat terdapat *obstacle*.
2. Mengatur formasi robot agar dapat berjalan secara teratur sesuai intruksi robot utama dan dapat membentuk formasi awal setelah melewati *obstacle*.

1.5 Sistematika Penulisan

Laporan Tugas Akhir ini ditulis dengan sistematika penulisan sebagai

berikut:

BAB I : PENDAHULUAN

Bab ini membahas tentang latar belakang masalah, perumusan masalah, pembatasan masalah, tujuan penulisan laporan tugas akhir, dan sistematika penulisan tugas akhir.

BAB II : LANDASAN TEORI

Bab ini membahas tentang berbagai teori yang mendukung tugas akhir ini. Hal tersebut meliputi *visual basic.net*, *obstacle avoidance*, *odometry*, formasi ketetapan robot dan *fuzzy logic*.

BAB III : METODE PENELITIAN

Dalam bab ini dijelaskan tentang metode penelitian serta alasan penggunaan metode tersebut dalam penelitian. Pada bab ini dijelaskan pula

tentang pembuatan perangkat lunak (software) dengan menggunakan *visual basic.net* sebagai pengontrol grafis pada simulasi.

BAB IV : PENGUJIAN DAN ANALISIS SISTEM

Bab ini berisi tentang pengujian simulasi secara keseluruhan. Pengujian yang dilakukan meliputi sudut peletakan sensor, koordinat tujuan, kecepatan dan arah belok robot, pembentukan formasi, komunikasi antar robot, dan penentuan parameter fuzzy untuk pergerakan robot.

BAB V : PENUTUP

Bab ini berisi tentang kesimpulan penelitian serta saran untuk pengembangan penelitian berikutnya.



UNIVERSITAS
Dinamika

BAB II

LANDASAN TEORI

2.1 Visual Basic.Net

Visual Basic.Net 2008 (VBNet 2008) adalah salah satu program berorientasi objek, selain itu ada pula program *Java* dan *C++* yang juga berbasis objek. Program *Visual Basic Net 2008* adalah produksi Microsoft Corp. Program ini biasanya dipaketkan bersama-sama dengan *Visual C# 2008* dan *Visual C++ 2008* dalam paket *Visual Studio 2008*. Dalam VBNet 2008 teknik pemrograman dapat dibuat lebih terstruktur dan lebih banyak bantuan dalam pemrograman dibandingkan dengan versinya yang terdahulu, yaitu *Visual Basic 6* (VB6) (Wardana, 2008:11).

2.1.1 Variabel

Variabel berguna untuk menyimpan nilai sementara untuk dapat dipergunakan kembali. Dikatakan sementara waktu karena nilai sebuah variabel akan disimpan dalam memori komputer yang bersifat tidak permanen. Menurut Leong M, (2004:51) *Visual basic.net* memiliki beberapa aturan dalam memmmberikan nama sebuah variabel, diantaranya:

1. Nama variabel harus diawali dengan karakter huruf, dan dalam penamaannya tidak boleh menggunakan tambahan karakter khusus lain, kecuali dengan menggunakan *underscore* (_).
2. Panjang karakter maksimal 255.
3. Sifatnya unik sehingga tidak boleh ada dua deklarasi variabel yang sama dalam prosedur.

4. Tidak mengandung perintah eksekusi yang dikenal oleh *Visual Basic.Net*.

Dalam situasi tertentu, ada kalanya kita melakukan konversi variabel dengan tipe data tertentu ke tipe data lain. Dalam mempermudah mengonversi tipe data tersebut dapat menggunakan fungsi konversi yang ada dalam pustaka *visual basic*. Beberapa fungsi pustaka *visual basic* adalah:

Tabel 2.1. Fungsi Pustaka *Visual Basic*.

Fungsi	Hasil Konversi
CBool	Boolean
CByte	Byte
CChar	Char
CDate	Date
CDBl	Double
CInt	Integer
CLng	Long
CObj	Object
CShort	Short
CSng	Single
CStr	String

2.1.2 Tipe Data

Tipe data adalah jenis nilai yang tersimpan dalam variabel, bisa berupa huruf, angka, ataupun tanggal. Tipe data diperlukan agar *visual basic* dapat langsung mengenal jenis data yang tersimpan dalam variabel. Menurut Leong M, (2004:48) berikut beberapa jenis tipe data yang didukung oleh *visual basic.Net*, berikut jangkauan nilai yang didukungnya Tabel 2.2.

Tabel 2.2. Nilai dari Tipe Data.

Tipe Data	Range
<i>Boolean</i>	Hanya dapat diisi dengan TRUE (benar) atau FALSE (salah)
<i>Byte</i>	0 – 255
<i>Char</i>	0 – 65535

<i>Date</i>	Merupakan nilai sebuah tanggal dan waktu 1 januari 0001 – 31 desember 9999, contoh: <i>Dim Tgl as Date</i> <i>Tgl = #9/16/2008 19:20:20#</i>
<i>Decimal</i>	0 - +/- 79.228.162.514.264.337.593.543.950. 335 (tanpa bilangan decimal di belakang koma) atau 0 - +/- 7,9228162514.264337593543950 335 (dengan bilangan decimal di belakang koma), contoh: <i>Dim Nilai as Decimal</i> <i>Nilai = 100,5</i>
<i>Double</i>	-1,79769313486231570E+308 - 1,7976931348 6231570E+308.
<i>Integer</i>	-2.147.483.648 - 2.147.483.648.
<i>Long</i>	-9.223.372.036.854.775.808 - 9.223.372.036.854.775.807
<i>Sbyte</i>	-128 – 127.
<i>Short</i>	-32.768 - 32.767.
<i>Single</i>	-3,4028235E+38 - -1,401298E-45 (untuk bilangan negatif) 1,401298E-45 - 3,4028235E+38 (untuk bilangan positif).
<i>String</i>	0 – 2 milyar karakter.
<i>Object</i>	Tipe data umum (sama seperti varian) yang dapat menampung berbagai tipe data lainnya.

2.1.3 Operator Matematika

Menurut Leong M, (2004:163) *Visual Basic.Net* telah merangkum fungsi matematika secara lengkap dalam *class* khusus, yaitu *class Math*. Di dalam *class Math* dapat ditemukan banyak fungsi matematika yang berguna, misalnya, fungsi trigonometri, logaritma, dan lain-lain. Beberapa fungsi pada *class Math* dapat dilihat pada Tabel 2.3.

Tabel 2.3. Fungsi pada *Class Math*.

E	Bilangan natural atau $e = 2,7182818284590452354$
PI	Konstanta diameter lingkaran yaitu $\phi = 3,14159265358979323846$
Abs	Fungsi absolut dari bilangan
Acoc	Fungsi sudut dari cosinus bilangan
Asin	Fungsi sudut dari sinus bilangan
Atan	Fungsi sudut dari tangent bilangan
Atan2	Fungsi sudut dari tangen yang ditetapkan dari 2 bilangan spesifik
Ceiling	Fungsi mencari bilangan terkecil dari angka yang lebih besar atau sama dengan angka yang ditentukan
Cos	Fungsi cosines

Cosh	Fungsi cosinus hiperbola dari suatu sudut
Exp	Fungsi eksponensial
Floor	Fungsi mencari bilangan terbesar dari angka yang lebih besar atau sama dengan angka yang ditentukan
Log	Fungsi log
Log10	Fungsi log10
Max	Mencari nilai maksimum atau terbesar
Min	Mencari nilai minimum atau terkecil
Round	Fungsi pembulatan
Sign	Mencari tanda dari bilangan
Sin	Fungsi sinus
Sinh	Fungsi sinus hiperbola dari sudut
Sqrt	Akar kuadrat dari bilangan
Tan	Fungsi tangent
Tanh	Fungsi tangent hiperbola dari sudut

2.1.4 Koordinat Form

Seluruh *form* pada VB memiliki sebuah koordinat. Koordinat ini berguna untuk menentukan posisi gambar atau *graphic* pada *form*. Koordinat sebuah *form* terdiri atas dua bagian yaitu koordinat horizontal dan koordinat vertikal biasa ditandai dengan x dan y. Satuan yang diperlukan untuk keperluan koordinat *form* adalah *pixel*, seperti halnya sebuah gambar. Koordinat dimulai dari bagian kiri atas *form* yang merupakan koordinat dasar dari *form*. Koordinat ini memiliki nilai 0 pada koordinat x dan koordinat y (Priyanto R, 2009:226).

2.2 Grafis

Grafis pada VB.Net membutuhkan media yang dimana media tersebut dapat menjadi kanvas yang dapat digunakan untuk menampilkan atau membuat suatu gambar, sehingga dibutuhkannya sistem operasi yaitu GDI+ (Graphic Device Interface).

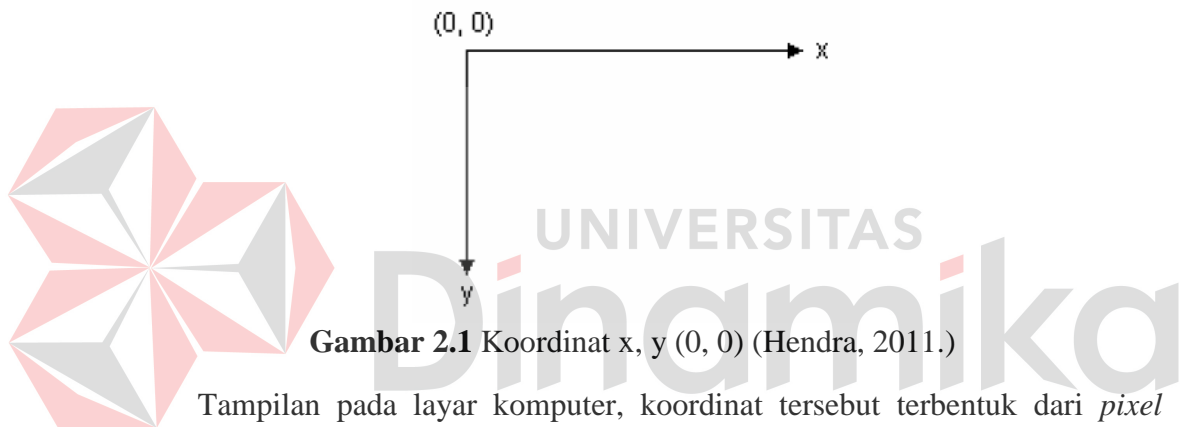
2.2.1 GDI+

GDI+ (Graphic Device Interface) adalah Suatu *class*, yang difungsikan untuk memproses sebuah grafis pada VB.Net. GDI+ sendiri memiliki kemampuan

untuk mengolah grafis vektor dua dimensi (garis), *imaging* (bitmap) dan *typography* (*font*) pada layar maupun Printer. Menurut Hendra dengan GDI, programmer tidak perlu mengetahui detail dari masing-masing peralatan untuk menampilkan grafis di atasnya, tetapi cukup menggunakan fasilitas yang telah disediakan oleh GDI+ class.

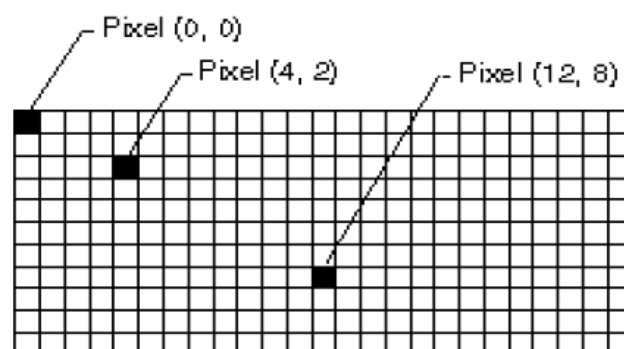
2.2.2 Koordinat Tampilan Layar

GDI+ sendiri menggunakan sistem koordinat, sehingga koordinat tersebut akan ditampilkan pada layar (x, y), mulai dari titik koordinat 0, 0.

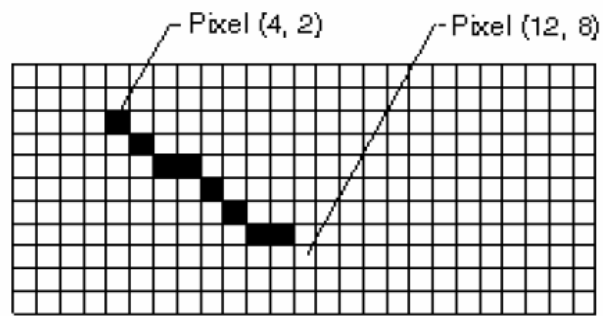


Gambar 2.1 Koordinat x, y (0, 0) (Hendra, 2011.)

Tampilan pada layar komputer, koordinat tersebut terbentuk dari *pixel* yang terdapat pada layar komputer.



Gambar 2.2 *Pixel* pada Tampilan Layar Komputer.



Gambar 2.3 Proses Pembuatan Garis Koordinat 4,2 sampai 12, 8.

2.2.3 Gambar Grafis Vektor 2 Dimensi

Grafis vektor 2D merupakan proses penggambaran bentuk-bentuk seperti garis, kurva dan bentuk berdasarkan sekumpulan titik tertentu (*pixel*). Penggambaran objek 2D pada VB.net dapat menggunakan *Object Pen* di mana *pen* dibuat untuk proses pembentukan garis.

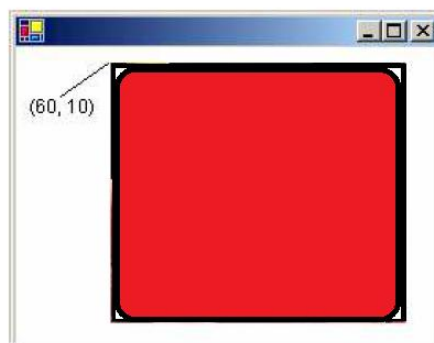
2.3 Penampilan Gambar

Proses penampilan gambar pada VB.Net dari file gambar ke *layer*, dapat menggunakan objek yang berada pada VB.net, objek tersebut berupa objek *bitmap* dan objek *graphics*. Objek *Bitmap* (*class bitmap*) hanya mendukung beberapa format seperti BMP, GIF, JPEG, PNG dan TIFF.

Contoh pemrograman:

```
Dim bitmap As New Bitmap ("Grapes.jpg")
```

```
e.Graphics.DrawImage (bitmap, 60, 10)
```



Gambar 2.4 Tampilan Contoh Program (Hendra, 2011).

2.4 Formasi Ketetapan Robot

Pada bagian ini didesain sebuah metode algoritma pengontrol, untuk sekelompok *mobile robot* yang saling mengikuti antar robot lainnya, di mana robot akan berjalan sesuai pergerakan dari pemimpin robot (robot utama). Tujuan dari metode kontrol ini adalah untuk membuat koloni robot yang dapat bergerak dari sisi ke sisi dengan jarak, kecepatan dan sudut orientasi yang telah ditentukan dan disesuaikan terhadap pemimpin robot. (Gustavi, 2013)

Dari penjabaran di atas telah didesain sebuah rumus *nonholonomic* model untuk setiap robot yaitu:

$$x = v \cos \phi \dots\dots\dots(2.1)$$

$$y = v \sin \phi \dots\dots\dots(2.2)$$

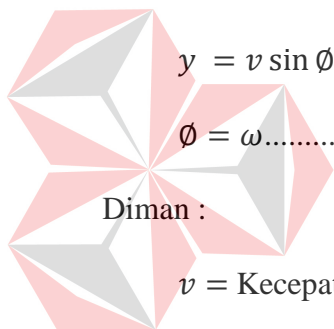
$$\phi = \omega \dots\dots\dots(2.3)$$

Diman :

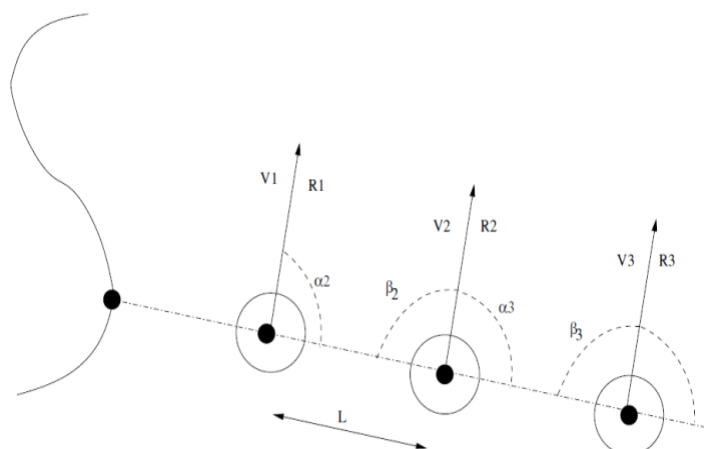
v = Kecepatan

ϕ = Rotasi

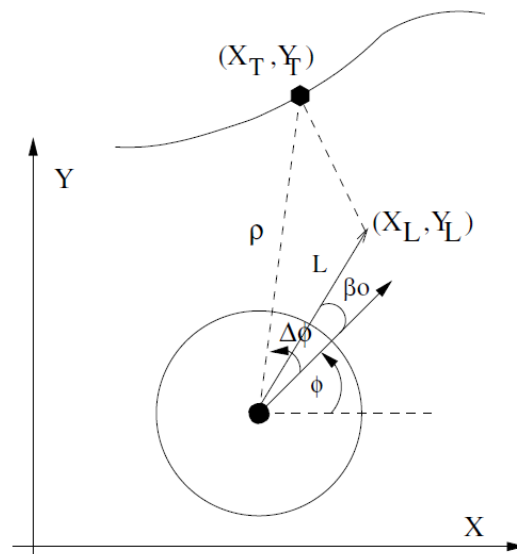
ω = Kecepatan sudut



UNIVERSITAS
Dinamika



Gambar 2.5 Pelacakan Horisontal dan Pembentukan Ketetapan Formasi .



Gambar 2.6 Penelusuran Sudut Robot.

Sumbu XL dan YL seperti gambar di atas titik pusat dialokasikan pada garis yang memiliki sudut β_0 dari sumbu orientasi robot, dan memiliki jarak L dari titik senter robot. Jika robot memiliki titik pusat x dan y maka dapat dituliskan seperti persamaan di bawah ini:

$$xL = x + L \cos(\phi + \beta_0) \quad (2.4)$$

$$yL = y + L \sin(\phi + \beta_0) \quad (2.5)$$

Turunan dari rumus di atas dikombinasikan dengan rumus 5.1 akan memberikan hubungan antara kecepatan dan sudut dari robot (v, ω), terhadap pergerakan dari titik XL dan YL. Dari penulisan ini didapatkan persamaan sebagai berikut:

$$\begin{bmatrix} xL \\ yL \end{bmatrix} = \begin{bmatrix} \cos \phi - L \sin(\phi + \beta_0) \\ \sin \phi - L \cos(\phi + \beta_0) \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \quad (2.6)$$

2.5 Posisi Awal Robot

Posisi awal atau proses memperkirakan posisi robot, pada dasarnya menggunakan perhitungan dengan mengakumulasi data jarak tempuh robot,

yang dihasilkan dari sensor robot berupa *rotary encoder* di mana pulsa dari sensor tersebut akan digunakan untuk memperhitungkan jarak tempuh robot. Permisalan dengan menggunakan satuan millimeter. Dalam mendapatkan jumlah pulsa setiap satu kali putaran roda, digunakan rumus sebagai berikut (Ardilla, 2011):

$$K_{roda} = 2 \times \pi \times r \dots\dots\dots (2.7)$$

$$pulsa_per_mm = resolusi\ encoder / K\ roda \dots\dots\dots (2.8)$$

Pada sistem penggerak *differential* terdapat dua roda yaitu roda kanan dan roda kiri dan dimisalkan jumlah *pulsa_per_mm* untuk roda kanan adalah *right_encoder* dan roda kiri adalah *left_encoder* dan jarak antara dua roda adalah *wheel_base*, maka didapatkan jarak tempuh (*distance*) dan sudut orientasi (θ).

Rumusnya adalah sebagai berikut.

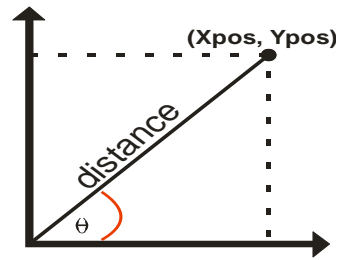
$$distance = (left_encoder + right_encoder) / 2 \dots\dots\dots (2.9)$$

$$\theta = (left_encoder - right_encoder) / wheel_base \dots\dots\dots (2.10)$$

Karena θ adalah sudut dalam radian maka untuk mengetahui sudut dalam derajat (heading) digunakan rumus sebagai berikut :

$$heading = \theta \times \frac{180}{\pi} \dots\dots\dots (2.11)$$

Dari ketentuan di atas didapatkan bahwa nilai *heading* akan bernilai negatif (-) ketika robot berputar melawan arah jarum jam dan akan bernilai positif (+) ketika robot berputar searah dengan jarum jam. Dengan mengetahui jarak dan sudut (*distance* dan θ) maka kita dapat mengetahui koordinat x dan koordinat y dengan persamaan trigonometri sebagai berikut :



Gambar 2.7 Panjang Jarak Koordinat.

Dari ilustrasi di atas maka koordinat dari robot dapat kita ketahui dengan rumus:

$$X_{pos} = distance \times \sin(\theta) \dots\dots\dots (2.12)$$

$$Y_{pos} = distance \times \cos(\theta) \dots\dots\dots (2.13)$$

2.6 Kesalahan Arah

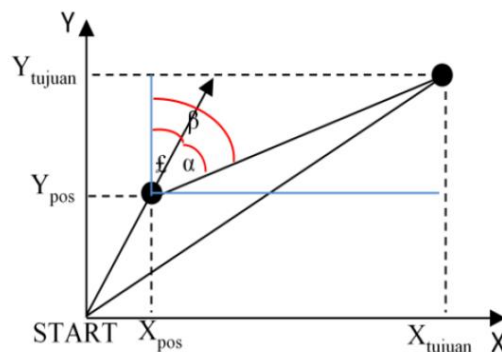
Dalam menentukan *error* arah hadap dari robot terhadap titik tujuan maka digunakan *theorema pythagoras* yang akan menghasilkan posisi (*path*) saat ini dan jarak terhadap titik tujuan (Ardilla, 2011). berikut perhitungannya:

$$X = X_{tujuan} - X_{pos} \dots\dots\dots (2.14)$$

$$Y = Y_{tujuan} - Y_{pos} \dots\dots\dots (2.15)$$

$$Target_distance = \sqrt{x^2 + y^2} \dots\dots\dots (2.16)$$

Heading dari robot yang telah diketahui sehingga kita dapat menghitung *error* arah hadap (*heading error*) robot terhadap titik tujuan.



Gambar 2.8 Sudut α , β , dan ϵ (Ardilla, 2011).

Gambar 2.8 menunjukkan ilustrasi untuk mencari *heading error* (α) di mana β adalah *target bearing* yaitu sudut antara posisi robot saat ini terhadap titik tujuan. Sedangkan garis berwarna biru adalah garis bantu yang masing-masing sejajar dengan sumbu x dan sumbu y. Dalam mendapat nilai dari β , maka digunakan rumus sebagai berikut :

$$\beta = \arctan \frac{(Y_{tujuan} - Y_{pos})}{(X_{tujuan} - X_{pos})} \dots\dots\dots (2.17)$$

Sehingga

$$\alpha = \beta - \epsilon \dots\dots\dots (2.18)$$

2.7 Obstacle Avoidance

Pada bagian *Obstacle Avoidance* menjelaskan tentang metode yang digunakan untuk menghindari halangan dengan cukup relevan yaitu menggunakan deteksi ujung tepi *object*, tahap kepastian pergerakan terhadap *obstacle*, metode yang *support* terhadap relevansi *obstacle avoidance*.

2.7.1 Metode Analisa Ujung Tepi Object

Metode *obstacle avoidance* yang lebih umum dan sering digunakan untuk menghindari rintangan adalah deteksi ujung tepi. Dalam metode ini, algoritma dicoba untuk menentukan posisi dari sisi kendala vertikal dan kemudian mengarahkan robot pada salah satu sisi ujung tepi *obstacle*. Metode ini digunakan dalam penelitian, serta beberapa proyek lainnya semua menggunakan sensor jarak meliputi sensor shap dan sensor *ultrasonic* untuk mendeteksi *obstacle*. Kelemahan dalam menghindari rintangan yang berbasis pendeteksian ujung tepi adalah kebutuhan robot untuk berhenti di depan sebuah halangan dalam rangka untuk memungkinkan pengukuran yang lebih akurat. Dalam rangka menanggulangi

kelemahan dari pendeteksian tersebut kedepannya akan digunakan komputer yang lebih cepat dalam pengekseskuan program agar robot tidak berhenti terlebih dahulu ketika pengambilan data.

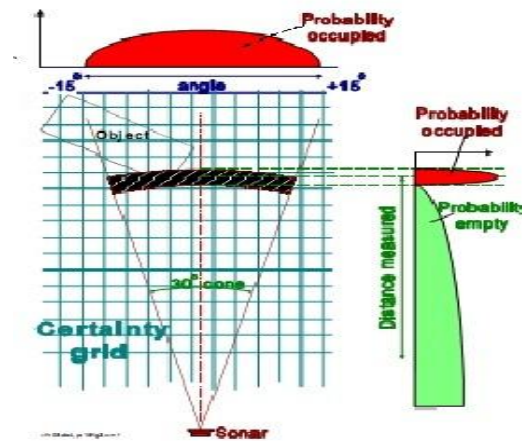
Kelemahan lebih lanjut dari metode deteksi ujung tepi sensitivitas untuk sensor akurasi seperti sensor *ultrasonic*, yang sebagian besar digunakan dalam aplikasi *mobile robot* memberikan banyak kekurangan. Dimana posisi minimal untuk batas benda dengan sensor pada posisi spasial adalah 10-50 cm, tergantung pada jarak kendala dan sudut antara permukaan benda.

2.7.2 Kepastian Gerak Terhadap *Obstacle*

Sebuah metode untuk merepresentasikan probabilitas halangan dalam jenis model jaringan telah dikembangkan di Carnegie-Mellon University (CMU). Model metode ini yang sering disebut dengan kepastian jaringan (*certainty grid*). Secara khusus metode ini sangat cocok untuk akomodasi data sensor yang kurang akurat seperti jarak pengukuran dari sensor *ultrasonic*.

Dalam metode jaringan, area kerja robot diwakili oleh array dua dimensi, yang dilambangkan sebagai sel. setiap sel berisi nilai yang pasti (CV) nilai tersebut menunjukkan bahwa halangan berada dalam wilayah sel tersebut. dalam perkembangannya metode ini dikembangkan oleh CMU, dengan memperbarui nilai kepastian CV dengan fungsi probabilitas yang memperhitungkan karakteristik sensor yang diberikan. Sebagai contoh sensor *ultrasonic*, yang memiliki pandangan seperti bidang yang berbentuk kerucut. Sebuah sensor *ultrasonic* hanya akan memantulkan gelombang untuk mendapatkan jarak ke objek dalam bidang baca berbentuk kerucut, dan sensor tidak dapat menentukan

lokasi sudut objek. (Gambar menunjukkan area A di mana sebuah *obstacle* harus berada dalam range untuk menghasilkan pengukuran jarak d).



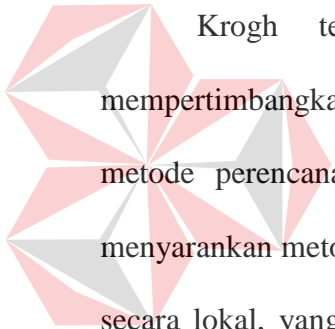
Gambar 2.9 Sensor Mengenali Benda.

Jika benda terdeteksi oleh sensor *ultrasonic*, hal ini dimungkinkan bahwa objek ini lebih dekat dengan sumbu akustik (gelombang) sensor, daripada bidang kerucut pembacaan sensor *ultrasonic*. Dengan alasan tersebut CMU membuat nilai fungsi Cx dengan meningkatkan nilai CV pada sel yang terdekat pada sumbu akustik lebih dari nilai CV pada sel-sel yang terdapat di pinggir bidang kerucut (pembacaan sensor).

Dalam aplikasi metode CMU ini, mobile robot akan diam saat pembacaan panorama lingkungannya dengan 24 sensor *ultrasonic*. Selanjutnya, fungsi Cx diterapkan pada masing-masing sensor *ultrasonic* dalam upayanya memperbarui informasi jaringan pergerakan. Sehingga, robot dapat bergerak menuju lokasi baru, dan robot akan berhenti kembali untuk mengulangi prosedur tersebut. Setelah robot melintasi ruangan dengan cara ini, kepastian jaringan yang dihasilkan merupakan jalur atau peta yang cukup akurat dari ruangan.

2.7.3 Metode *Support Relevansi Obstacle Avoidance*

Metode ini sangat relevan dengan mempertimbangkan *imaginer* yang bekerja pada sebuah robot telah disarankan oleh Khatib. Dalam metode *Relevansi Obstacle avoidance*, mengenai relevansi gaya menghindari rintangan, sedangkan target dari sensor yang terdapat pada robot menentukan gerak robot yang atraktif. Resultan A memiliki gaya vektor R , yang terdiri dari jumlah gaya atraktif yang diarahkan terhadap target dan daya repulsif dari hambatan, hal ini digunakan untuk menghitung letak posisi robot. Dengan R sebagai gaya percepatan yang bekerja pada posisi baru robot untuk menentukan interval waktu tertentu yang telah dihitung.



Krogh telah meningkatkan konsep ini lebih lanjut dengan mempertimbangkan *velocit* robot di sekitar hambatan. Thorpe telah menerapkan metode perencanaan benda di lapangan secara *off-line*. Krogh dan Thorpe menyarankan metode gabungan untuk perencanaan secara global dan perencanaan secara lokal, yang menggunakan metode *Generalized Potential Field*. Newman dan Hogan memperkenalkan fungsi gaya potensial dengan melalui penggabungan kendala individu fungsi dengan operasi logis. Umumnya metode ini diasumsikan pada model *a known and prescribed world*, yang sederhana, bentuk geometris yang telah ditetapkan merupakan hambatan dari jalan robot secara *gen-erated off-line*.

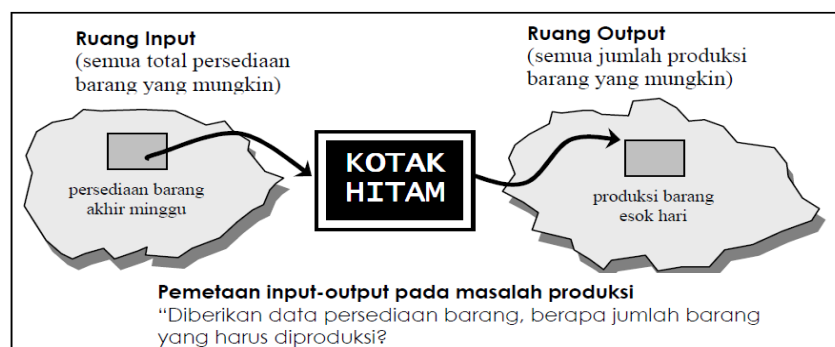
Metode *Potential field* telah diimplementasikan pada robot mobile dengan data nyata oleh Brooks (1986), dan oleh Arkin (1989). Namun, robot karya Arkin sangat lambat, dengan pergerakan melintasi rintangan sebesar 0.12 cm / detik (0,4 kaki /detik).

2.8 Fuzzy Logic

Logika fuzzy adalah cabang dari sistem kecerdasan buatan (*Artificial Intelegant*) yang memanipulasi kemampuan manusia dalam berpikir ke dalam bentuk algoritma yang kemudian dijalankan oleh mesin. Logika fuzzy merupakan cara yang tepat untuk memetakan ruang *input* ke dalam ruang *output*. Sebagai contoh:

- Manajer pergudangan mengatakan pada manajer produksi, seberapa banyak persediaan barang pada akhir minggu ini, kemudian manajer produksi akan menetapkan jumlah barang yang harus diproduksi esok hari.
- Pelayan restoran memberikan pelayanan terhadap tamu, kemudian tamu akan memberikan tip yang sesuai atas baik tidaknya pelayan yang diberikan.
- Anda mengatakan pada saya seberapa sejuk ruangan yang anda inginkan, saya akan mengatur putaran kipas yang ada pada ruangan ini.
- Penumpang taksi berkata pada sopir taksi seberapa cepat laju kendaraan yang diinginkan, sopir taksi akan mengatur pijakan gas taksinya.

Contoh pemetaan *input-output* dalam bentuk grafis terlihat pada Gambar 2.10.



Gambar 2.10 Pemetaan *Input Output* .

Ada beberapa alasan mengapa orang menggunakan logika fuzzy, antara lain:

- Konsep logika fuzzy mudah dimengerti. Konsep matematis yang mendasari penalaran fuzzy sangat sederhana dan mudah dimengerti.
- Logika fuzzy sangat fleksibel.

- c. Logika fuzzy memiliki toleransi terhadap data-data yang tidak tepat.
- d. Logika fuzzy mampu memodelkan fungsi-fungsi nonlinear yang sangat kompleks.
- e. Logika fuzzy dapat membangun dan mengaplikasikan pengalaman para pakar secara langsung tanpa harus melalui proses pelatihan.
- f. Logika fuzzy dapat bekerjasama dengan teknik-teknik kendali secara konvensional.

2.8.1 Himpunan Fuzzy

Pada himpunan tegas (*crisp*), nilai keanggotaan suatu *item* x dalam himpunan A yang sering ditulis dengan $\mu_A[x]$, memiliki 2 kemungkinan yaitu:

1. Satu (1), yang berarti bahwa suatu *item* menjadi anggota dalam suatu himpunan.
2. Nol (0), yang berarti bahwa suatu *item* tidak menjadi anggota dalam suatu himpunan.

2.8.2 Fungsi Keanggotaan

Fungsi keanggotaan adalah kurva yang menunjukkan pemetaan titik – titik *input* data kedalam nilai keanggotaannya (derajat keanggotaan) yang memiliki interval antara 0 sampai 1.

2.8.3 Operator Dasar

Seperti halnya himpunan konvensional, ada beberapa operasi yang didefinisikan secara khusus untuk mengkombinasi dan memodifikasi himpunan fuzzy.

a. Operator AND

Operator ini berhubungan dengan operasi interseksi pada himpunan. α -predikat sebagai hasil operasi dengan operator AND diperoleh dengan mengambil nilai keanggotaan terkecil antar elemen pada himpunan – himpunan yang bersangkutan.

$$\mu_{A \cap B} = \min(\mu_A[x], \mu_B[y]) \dots\dots\dots(2.19)$$

b. Operator OR

Operator ini berhubungan dengan operasi union pada himpunan. α -predikat sebagai hasil operasi dengan operator OR diperoleh dengan mengambil nilai keanggotaan terkecil antar elemen pada himpunan – himpunan yang bersangkutan.

$$\mu_{A \cup B} = \max(\mu_A[x], \mu_B[y]) \dots\dots\dots(2.20)$$

c. Operator NOT

Operator ini berhubungan dengan operasi komplemen pada himpunan. α -predikat sebagai hasil operasi dengan operator NOT diperoleh dengan mengambil nilai keanggotaan terkecil antar elemen pada himpunan – himpunan yang bersangkutan.

$$\mu_{A'} = 1 - \mu_A[x] \dots\dots\dots(2.21)$$

2.8.4 Penalaran Monoton

Metode ini digunakan sebagai dasar untuk teknik implikasi fuzzy. Jika 2 daerah fuzzy direalisasikan dengan implikasi sederhana sebagai berikut:

IF x is A THEN y is B

transfer fungsi:

$$Y = f((x, A), B)$$

Maka *system fuzzy* dapat berjalan tanpa harus melalui komposisi dan dekomposisi fuzzy. Nilai output dapat diestimasi secara langsung dari nilai keanggotaan yang berhubungan dengan antesedennya.

2.8.5 Fungsi Implikasi

Bentuk umum aturan yang digunakan dalam fungsi implikasi:

IF x is A THEN y is B

Dengan x dan y adalah skalar, A dan B adalah himpunan fuzzy. Proposisi yang mengikuti IF disebut anteseden, sedangkan proposisi yang mengikuti THEN disebut konsekuen. Secara umum, ada dua fungsi implikasi, yaitu:

1. Min (minimum), fungsi ini akan memotong output himpunan fuzzy
2. Dot (*product*), fungsi ini akan menskala output himpunan fuzzy.

2.8.6 Metode Sugeno

Penalaran dengan metode SUGENO hampir sama dengan penalaran MAMDANI, hanya saja output (konsekuen) sistem tidak berupa himpunan fuzzy, melainkan berupa konstanta atau persamaan linear. Metode ini diperkenalkan oleh Takagi-Sugeno Kang pada tahun 1985.

1. Model Fuzzy Sugeno Orde-Nol

Secara umum bentuk model *fuzzy* SUGENO Orde-Nol adalah:

IF (x_1 is A_1) • (x_2 is A_2) • (x_3 is A_3) • • (x_N is A_N) THEN $z=k$

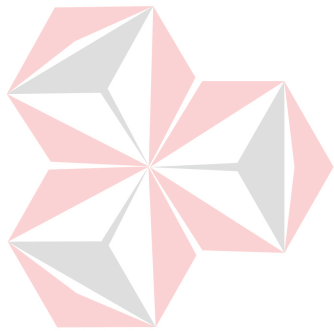
dengan A_i adalah himpunan fuzzy ke- i sebagai anteseden, dan k adalah suatu konstanta (tegas) sebagai konsekuen.

2. Model Fuzzy Sugeno Orde-Satu

Secara umum bentuk model fuzzy SUGENO Orde- Satu adalah:

$$\text{IF } (x_1 \text{ is } A_1) \bullet \dots \bullet (x_N \text{ is } A_N) \text{ THEN } z = p_1 * x_1 + \dots + p_N * x_N + q$$

dengan A_i adalah himpunan fuzzy ke- i sebagai anteseden, dan p_i adalah suatu konstanta (tegas) ke- i dan q juga merupakan konstanta dalam konsekuen. Apabila komposisi aturan menggunakan metode SUGENO, maka defuzzifikasi dilakukan dengan cara mencari nilai rata-ratanya.



UNIVERSITAS
Dinamika

BAB IV

PENGUJIAN SISTEM

4.1. Pengujian Perangkat Lunak

Dalam mengetahui simulasi ini dapat berjalan atau tidak maka diperlukan tahap pengujian. Alat dan bahan yang digunakan sebelum pengujian:

1. *Software Visual Basic.Net 2008.*
2. *Personal Computer (PC).*

Berikut merupakan hasil pengujian dari setiap modul yang telah dibuat:

4.1.1. Pengujian Target Robot

A. Tujuan

Pengujian target robot adalah pengujian yang dilakukan untuk mengetahui apakah robot dapat menuju target tujuan yang telah ditentukan oleh *user* atau tidak.

B. Prosedur Pengujian

1. *RUN* program (*user/manual*).
2. Arahkan *mouse* pada koordinat yang diinginkan dalam koordinat arena (*user/manual*).
3. Penekanan tombol kanan *mouse* (menghasilkan titik koordinat merah).
4. Penekanan *push button start* pada simulasi (*user/manual*).
5. Robot utama akan membuat garis bantu kanan dan kiri menuju koordinat tujuan yang telah diberikan oleh *user* untuk acuan arah belok dengan menggunakan rumus *theorema phytagoras (auto)*.

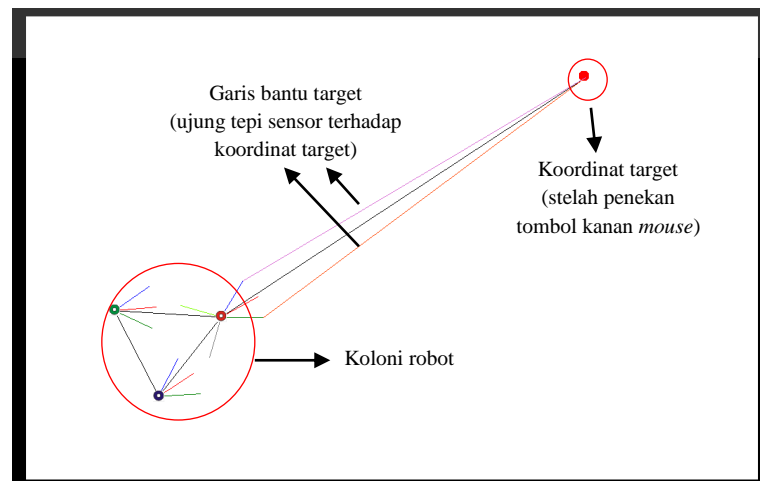
C. Hasil Pengujian

Sesuai dengan langkah pengujian yang dilakukan, didapatkan hasil dari penarikan garis bantu target kanan dan kiri yang digunakan untuk menentukan arah koordinat target, dengan cara membandingkan panjang dari masing-masing garis bantu tersebut. Tabel 4.1 merupakan hasil pergerakan berdasarkan target.

Tabel 4.1. Pergerakan Arah Robot Berdasarkan Jarak Garis Bantu Target.

Pengujian	Jarak Garis Bantu (Pixel)		Error Garis Bantu	Sudut Tujuan Target Terhadap Robot (Derajat)	Sudut Pergerakan (Derajat)	Error Arah (Derajat)
	Kanan	Kiri				
1	762	762	0	-30	-30	0
2	758	758	0	-30	-30	0
3	712	668	-44	-30	13	43
4	709	666	-43	-30	11	41
5	706	664	-42	-30	10	40
6	703	663	-40	-30	8	38
7	421	444	23	-30	-51	-21
8	418	439	21	-30	-50	-20
9	414	434	20	-30	-48	-18
10	411	429	18	-30	-46	-16

Berikut merupakan tampilan grafis pembentukan garis bantu Gambar 4.1.



Gambar 4.1. Tampilan Simulasi Target Robot.

D. Pembahasan

Sesuai hasil pengujian yang telah dilakukan. Logika peneliti untuk mencapai koordinat tujuan dengan menggunakan garis bantu yang dibentuk dari

ujung tepi sensor kanan dan kiri terhadap koordinat target, menggunakan rumus *theorema pythagoras* telah membantu robot untuk mengoreksi kesalahan arah tujuan. Nilai jarak garis bantu kanan dan kiri tersebut akan digunakan untuk mendapatkan hasil *error* target yang akan dikelola oleh fuzzy untuk mendapatkan pergerakan arah robot yang sesuai.

4.1.2. Pengujian Derajat Sensor

A. Tujuan

Pengujian derajat sensor adalah pengujian yang dilakukan untuk mengetahui sudut dari sensor kanan dan kiri terhadap sensor tengah dalam pendeteksian letak *obstacle*. Sehingga dapat menghasilkan pergerakan arah belok yang sesuai terhadap *obstacle*.

B. Prosedur Pengujian

1. *Run* program (*user/manual*).
2. Pemilihan target tujuan robot (*user/manual*).
3. Penekanan *push button start* (*user/manual*).
4. Arahkan *mouse* pada koordinat yang diinginkan dalam koordinat arena (*user/manual*).
5. Penekanan tombol kanan *mouse* (menghasilkan titik koordinat hitam (*obstacle*)).
6. Amati pergerakan robot (arah belok) dan posisi sudut sensor kanan dan kiri terhadap sensor tengah apakah sudah sesuai yang diharapkan atau belum.
7. Lakukan percobaan beberapa kali (*trial and error*) untuk mengetahui hasil dari sudut sensor kanan dan kiri yang sesuai.

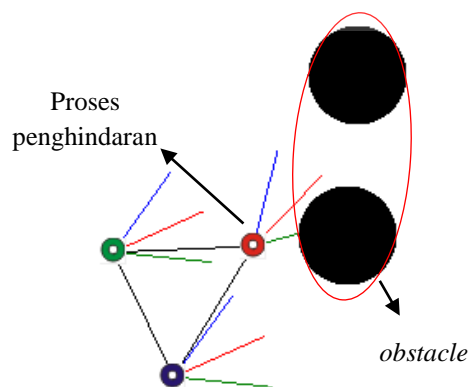
C. Hasil Pengujian

Sesuai dengan langkah pengujian yang dilakukan. Proses ini telah didesain dengan objek Penghalang (*obstacle*) berukuran 60×60 pixel dan panjang maksimal pembacaan sensor *ultrasonic* sebesar 60 pixel. Tabel 4.2 merupakan penentuan peletakan sudut sensor agar dapat mendapatkan peletakan yang ideal.

Tabel 4.2. Sudut Sensor Kanan dan Kiri.

Pengujian	Jarak Baca Sensor	Sudut Sensor (Derajat)		Menyentuh Benda	Kestabilan Pergerakan Robot
		Kanan	Kiri		
1	60	10°	-10°	Menyentuh	Tidak
2	60	20°	-20°	Menyentuh	Tidak
3	60	30°	-30°	Menyentuh	Stabil
4	60	40°	-40°	Tidak	Stabil
5	60	50°	-50°	Tidak	Stabil
6	60	60°	-60°	Tidak	Stabil
7	60	70°	-70°	Tidak	Stabil
8	60	80°	-80°	Tidak	Stabil
9	60	90°	-90°	Tidak	Stabil

Berdasarkan hasil Tabel 4.2 diperoleh tampilan grafis pada simulasi seperti Gambar 4.2.



Gambar 4.2. Tampilan Simulasi Objek Penghalang.

D. Pembahasan

Dari hasil pengujian ini didapatkan peletakan sudut sensor yang ideal yaitu sensor kanan 30° dan sensor kiri -30° , karena sudut sensor tersebut dapat

mendeteksi *obstacle* dengan cukup baik. Sehingga robot dapat bergerak secara stabil dalam menentukan arah pergerakannya terhadap *obstacle*. Dengan demikian, kemungkinan robot untuk menabrak *obstacle* sangat kecil.

4.1.3. Pengujian Pembentukan Formasi Robot

A. Tujuan

Pengujian yang merupakan proses pembentukan formasi dari koloni robot. Pembentukan formasi ini didapatkan ketika robot utama mengirimkan koordinat tujuan ke robot *follower* dengan acuan jarak dan sudut dari posisi robot utama (pemimpin).

B. Prosedur Pengujian

1. *RUN* program (*user/manual*).
2. Pemilihan target tujuan robot (*user/manual*).
3. Penekanan *push button start* (*user/manual*).
4. Robot utama mengirimkan dua koordinat tujuan (*auto*).
5. Robot *follower* A dan B membuat garis bantu menuju koordinat yang telah diberikan robot utama, setiap robot memiliki koordinat yang berbeda (*auto*).
6. Amati pergerakan robot *follower* terhadap koordinat target yang dikirimkan robot utama dan posisi sudut pengiriman koordinat tujuan untuk pembentukan formasi apakah sudah sesuai yang diharapkan atau belum.
7. Lakukan percobaan beberapa kali (*trial and error*) untuk mengetahui hasil yang sesuai dari sudut koordinat tujuan yang dikirimkan robot utama.

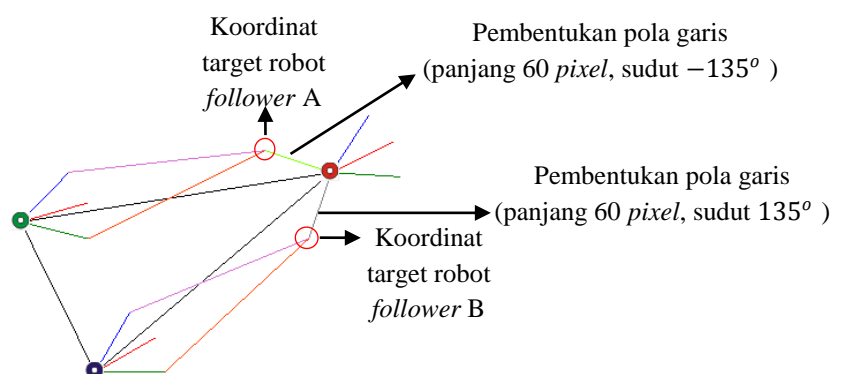
C. Hasil Pengujian

Sesuai dengan langkah pengujian yang dilakukan Pembentukan formasi, terbentuk ketika robot utama mengirimkan koordinat target ke setiap robot *follower* A dan B dengan sudut dan jarak yang telah ditentukan oleh robot utama. Hasil pengujian koordinat berdasarkan *trial and error* didapatkan hasil pergerakan terhadap formasi Tabel 4.3.

Tabel 4.3. Koordinat Target Berdasarkan Posisi Sudut Robot Utama.

Pengujian	Sudut Target (Derajat)		Panjang Koordinat (Pixel)	Bentuk Formasi	Kestabilan Formasi Robot
	<i>Follower A</i>	<i>Follower B</i>			
1	180°	-180°	60	Horizontal	Stabil
2	160°	-160°	60	Segitiga	Tidak
3	145°	-145°	60	Segitiga	Stabil
4	135°	-135°	60	Segitiga	Stabil
5	125°	-125°	60	Segitiga	Stabil
6	90°	-90°	60	Vertikal	Tidak

Sesuai dengan langkah pengujian yang dilakukan, peneliti menggunakan titik koordinat pengiriman dengan sudut 135° dan -135° dengan pembentukan formasi segitiga, karena sudut tersebut merupakan sudut yang cukup ideal. Proses dari pembentukan formasi robot seperti Gambar 4.3.



Gambar 4.3. Tampilan Simulasi Target Robot *Follower*.

D. Pembahasan

Sesuai hasil pengujian, pembentukan formasi robot sebenarnya sama seperti pembuatan logika menggunakan garis ujung tepi sensor yang dilakukan

robot utama menuju koordinat target. Dalam hasil ini robot utama telah berhasil mengirimkan titik koordinat yang akan dituju oleh setiap robot *follower*. Robot utama akan membentuk dua pola garis yang koordinat awalnya berada di titik koordinat posisi, garis tersebut dibentuk dengan jarak 60 *pixel* dari posisi robot utama. Kedua garis kemudian diberi sudut yang berbeda yaitu 135° dan -135° berdasarkan sudut dari robot utama. Ujung tepi setiap garis itulah yang nantinya akan digunakan sebagai target dari setiap robot *follower*.

4.1.4. Pengujian Pergerakan Kemudi

A. Tujuan

Pengujian pergerakan kemudi adalah pengujian yang dilakukan untuk mengetahui pergerakan yang dilakukan oleh setiap robot dalam menghindari *obstacle*, Dengan menggunakan sensor *ultrasonic* (sensor kanan, tengah, dan kiri) sebagai pendeteksi *obstacle*.

B. Prosedur Pengujian

1. *RUN* program (*user/manual*).
2. Arahkan *mouse* pada koordinat yang diinginkan dalam koordinat arena (*user/manual*).
3. Penekanan tombol kanan *mouse* (target tujuan titik koordinat merah).
4. Penekanan *push button start* pada simulasi (*user/manual*).
5. Pembentukan sensor *ultrasonic* (*auto*).
6. Letakkan *obstacle* pada setiap jalur yang akan dilalui robot untuk mengetahui sensor *ultrasonic* dapat membaca *obstacle* yang berada didepannya atau tidak.

C. Hasil Pengujian

Sesuai dengan langkah pengujian yang dilakukan, didapatkan hasil dari pergerakan robot Tabel 4.4.

Tabel 4.4. Pergerakan Kemudi.

Sensor Tengah (<i>pixel</i>)	Sensor Kanan (<i>pixel</i>)	Sensor Kiri (<i>pixel</i>)	Halangan (<i>Obstacle</i>)	Kemudi
60	60	60	Tidak ada halangan	Lurus
60	40	60	Halangan di kanan	Belok Kiri
40	40	60	Halangan depan kanan	Belok Kiri
60	60	40	Halangan di kiri	Belok kanan
40	60	40	Halangan depan kiri	Belok kanan
10	30	30	Halangan di Depan	Berhenti

D. Pembahasan

Dari hasil yang didapatkan terlihat kepekaan sensor menyentuh *obstacle*.

Robot akan memberikan respon pergerakan ke kiri ketika sensor kanan menyentuh *obstacle* begitu pula sebaliknya robot akan bergerak ke arah kanan ketika sensor kiri telah menyentuh *obstacle*. Robot akan berhenti ketika jarak baca sensor tengah berada di *range* 10-30 *pixel* dan *obstacle* berada diantara sensor

kanan dan kiri. Dengan hasil tersebut dapat disimpulkan bahwa kemudi dapat berjalan sesuai dengan yang diharapkan.

4.1.5. Pengujian Komunikasi USART

A. Tujuan

Pengujian komunikasi USART adalah pengujian yang dilakukan untuk mengkoordinasikan pergerakan kecepatan robot dengan mengirimkan koordinat posisi dari setiap robot ke robot lainnya. Setelah mendapatkan informasi koordinat, setiap robot akan dapat memperhitungkan jarak dari setiap robot dengan menggunakan rumus *theorema pythagoras*.

B. Prosedur Pengujian

1. *RUN* program (*user/manual*).
2. Arahkan *mouse* pada koordinat yang diinginkan dalam koordinat arena (*user/manual*).
3. Penekanan tombol kanan *mouse* (target tujuan titik koordinat merah).
4. Penekanan *push button start* pada simulasi (*user/manual*).
5. Robot utama mengirimkan koordinat robot *follower* (*auto*).
6. Penentuan koordinat dari setiap robot (*auto*).
7. Pembentukan pola garis yang saling menyambung antar robot, dibentuk berdasarkan koordinat posisi (*auto*).
8. Perhitungan jarak dari setiap robot terhadap robot lainnya dengan menggunakan rumus *theorema pythagoras* (*auto*).

C. Hasil Pengujian

Sesuai dengan langkah pengujian yang dilakukan, didapatkan hasil dari koordinasi robot dengan menggunakan komunikasi USART Tabel 4.5.

Tabel 4.5. Kecepatan Berdasarkan Jarak Robot.

Jarak Antar Robot (<i>Pixel</i>)			Pergerakan Kecepatan		
Robot Utama		Antar <i>Follower</i>	Robot Utama	Robot <i>Follower</i> A	Robot <i>Follower</i> B
<i>Follower A</i>	<i>Follower B</i>				
100	150	30	Pelan	Pelan	Stop
150	100	30	Pelan	Stop	Pelan
100	100	30	Pelan	Stop	Stop
100	150	60	Pelan	Pelan	Cepat
150	100	60	Pelan	Cepat	Pelan
100	100	60	Cepat	Cepat	Cepat
100	150	100	Pelan	Pelan	Cepat
150	100	100	Pelan	Cepat	Pelan
100	100	100	Cepat	Cepat	Cepat

D. Pembahasan

Dari hasil yang didapatkan komunikasi USART membantu dalam memberikan struktur koordinasi kecepatan, dengan menggunakan jarak setiap robot (robot utama terhadap robot *follower* A, robot utama terhadap *follower* B, dan robot antar *follower*). Jarak tersebut didapatkan dari hasil komunikasi titik koordinat posisi yang telah dikirimkan oleh masing-masing robot terhadap robot lainnya (bertukar informasi mengenai posisi). Robot dapat menentukan kecepatan yang di butuhkan dalam kondisi tertentu. Sehingga dapat membentuk kombinasi pergerakan yang tidak mengakibatkan tabrakan dan tidak saling mendahului satu dengan yang lainnya. Dengan hasil tersebut disimpulkan bahwa komunikasi USART telah membantu dalam mengkoordinasi kecepatan dari setiap robot untuk mengurangi resiko tabrakan ketika pembentukan formasi.

4.1.6. Pengujian Metode Fuzzy

A. Tujuan

Pengujian ini dilakukan untuk menguji apakah program fuzzy yang dibuat mampu berjalan sesuai dengan keinginan dan kebutuhan tugas akhir ini.

B. Prosedur Pengujian

1. *RUN* program (*user/manual*).
2. Arahkan *mouse* pada koordinat yang diinginkan dalam koordinat arena (*user/manual*).
3. Penekanan tombol kanan *mouse* (target tujuan titik koordinat merah).
4. Penekanan *push button start* pada simulasi (*user/manual*).
5. Pembentukan sensor *ultrasonic* (*auto*).

6. Letakkan *obstacle* pada setiap jalur yang akan dilalui robot sehingga sensor *ultrasonic* dapat membaca *obstacle* yang ada di depannya untuk memberikan *input* data fuzzy.
7. Amati pergerakan robot (kecepatan dan arah belok) dan parameter data fuzzy apakah sudah sesuai yang diharapkan atau belum.
8. Lakukan percobaan beberapa kali untuk mengetahui hasil dari parameter fuzzy yang sesuai.

C. Hasil Pengujian

Sesuai dengan langkah pengujian, didapatkan hasil dari parameter fuzzy yang dapat mengatur kecepatan dan arah belok robot. Berikut merupakan parameter yang digunakan:

a. Kecepatan Robot

Dalam proses pengaturan kecepatan terdapat lima nilai *output* fuzzy yang berbeda-beda. Setiap nilai *output* tersebut akan digunakan untuk mengontrol kecepatan robot dalam pembentukan formasi. Berikut merupakan penetapan nilai *output* fuzzy berdasarkan jarak antara robot satu dengan yang lain Tabel 4.6, 4.7 dan 4.8.

Tabel 4.6. *Output Fuzzy Robot Utama Berdasarkan Jarak Follower.*

No	Jarak Robot (<i>pixel</i>)			Output Fuzzy		
	Robot Utama		Antar Follower	Stop	Pelan	Cepat
	Follower A	Follower B				
1	50	50	60	0	1	2
2	60	60	60	0	1	2
3	80	80	60	0	1	2
4	120	100	30	0	1	2
5	100	120	30	0	1	2
6	120	100	40	0	1	2
7	120	100	60	0	2	4
8	100	120	60	0	2	4
9	100	120	80	0	2	4
10	80	100	20	0	1	2

11	70	100	30	0	1	2
12	80	100	40	0	1	2
13	80	100	50	0	1	2
14	70	100	50	0	1	2
15	80	100	60	0	1	2
16	70	100	60	0	1	2
17	100	60	20	0	1	2
18	100	80	30	0	1	2
19	100	60	40	0	1	2
20	100	60	50	0	1	2

Tabel 4.7. Output Fuzzy Follower A Berdasarkan Jarak Follower B dan Robot Utama.

No	Jarak Robot (pixel)			Output Fuzzy		
	Robot Utama		Antar Follower	Stop	Pelan	Cepat
	Follower A	Follower B				
1	50	50	60	0	0	0
2	60	60	60	0	0,25	0,5
3	80	80	60	0	0,5	1
4	120	100	30	0	0	0
5	100	120	30	0	0,25	1
6	120	100	40	0	0,5	1
7	120	100	60	0	1	2
8	100	120	60	0	2	4
9	100	120	80	0	2	4
10	80	100	20	0	1	2
11	70	100	30	0	1	2
12	80	100	40	0	1	2
13	80	100	50	0	1	2
14	70	100	50	0	1	2
15	80	100	60	0	1	2
16	70	100	60	0	1	2
17	100	60	20	0	0	0
18	100	80	30	0	0,25	0,5
19	100	60	40	0	0,5	1
20	100	60	50	0	1	2

Tabel 4.8. Output Fuzzy Follower B Berdasarkan Jarak Follower A dan Robot Utama.

No	Jarak Robot (pixel)			Output Fuzzy		
	Robot Utama		Antar Follower	Stop	Pelan	Cepat
	Follower A	Follower B				
1	50	50	60	0	0	0
2	60	60	60	0	0,25	0,5
3	80	80	60	0	0,5	1
4	120	100	30	0	0,25	1
5	100	120	30	0	0	0

6	120	100	40	0	1	2
7	120	100	60	0	2	4
8	100	120	60	0	1	2
9	100	120	80	0	2	4
10	80	100	20	0	0	0
11	70	100	30	0	0,25	0,5
12	80	100	40	0	0,5	1
13	80	100	50	0	1	2
14	70	100	50	0	1	2
15	80	100	60	0	2	4
16	70	100	60	0	2	4
17	100	60	20	0	1	2
18	100	80	30	0	1	2
19	100	60	40	0	1	2
20	100	60	50	0	1	2

Dari tabel di atas di dapatkan penentuan hasil *output* fuzzy kecepatan, hasil tersebut didapatkan dari jarak robot satu dengan robot yang lain menggunakan penggambaran komunikasi USART pada simulasi. Berikut merupakan hasil pergerakan kecepatan robot terhadap *obstacle* dengan sampel *output* fungsi keanggotaan 0 sebagai nilai pergerakan *stop*, 2 sebagai nilai pergerakan sedang, dan 4 sebagai nilai pergerakan cepat (Tabel 4.9).

Tabel 4.9. *Output* Kecepatan Berdasarkan Jarak Sensor Tengah dan Jarak Target.

waktu	<i>Input Fuzzy</i>		<i>Output</i>	
	Sensor Tengah (<i>Pixel</i>)	Jarak Target (<i>Pixel</i>)	Kecepatan Robot (<i>Pixel</i>)	Kecepatan Robot (%)
0,63	60	539	4	100
0,64	59	535	3,773	94,319
0,65	55	531	3,307	82,667
0,66	52	528	2,88	72
0,67	49	525	2,493	62,316
0,68	46	523	2,196	54,903
0,69	44	520	1,834	45,854
0,7	42	519	1,596	39,911
0,71	41	517	1,413	35,333
0,72	39	516	1,227	30,667
0,73	38	514	1,04	26

0,74	37	513	0,907	22,667
0,75	36	512	0,8	20
0,76	35	512	0,693	17,333
0,78	34	510	0,508	12,706
0,8	33	509	0,401	10,027
0,84	32	508	0,25	6,262
0,9	31	507	0,102	2,5456
0,98	30	507	0,035	0,8781
0,99	30	506	0	0

b. Arah Belok

Dalam pergerakan arah robot berdasarkan *output* fuzzy, diperoleh hasil ketika sensor kanan dan kiri menyentuh *obstacle* Tabel 4.10 dan 4.11.

Tabel 4.10. *Output* Pergerakan Berdasarkan Sensor Kiri dan Arah Target.

No	Waktu	Error Sensor	Error Target	<i>Output Pergerakan</i> (derajat)
1	1,05	0	0	0
2	1,06	0	0	0
3	1,07	0	0	0
4	1,08	2,8	2,29	1,48
5	1,09	7,4	5,21	2,72
6	1,1	13	8,37	4,01
7	1,11	16	8,93	4,85
8	1,12	19	9,14	5,02
9	1,13	22	8,82	5,06
10	1,14	25	8,74	5,13
11	1,15	27	8,09	5,1
12	1,16	30	7,83	5,13
13	1,17	32	7,73	5,18
14	1,18	34	6,81	5,12
15	1,19	36	5,7	5,1
16	1,2	37	3,38	4,93
17	1,21	37	1,11	4,69
18	1,22	32	-7,2	4,28
19	1,41	0	-19	-3
20	1,42	0	-18	-3
21	1,43	0	-16	-3
22	1,44	0	-15	-2,9
23	1,45	0	-13	-2,6
24	1,46	0	-11	-2,3

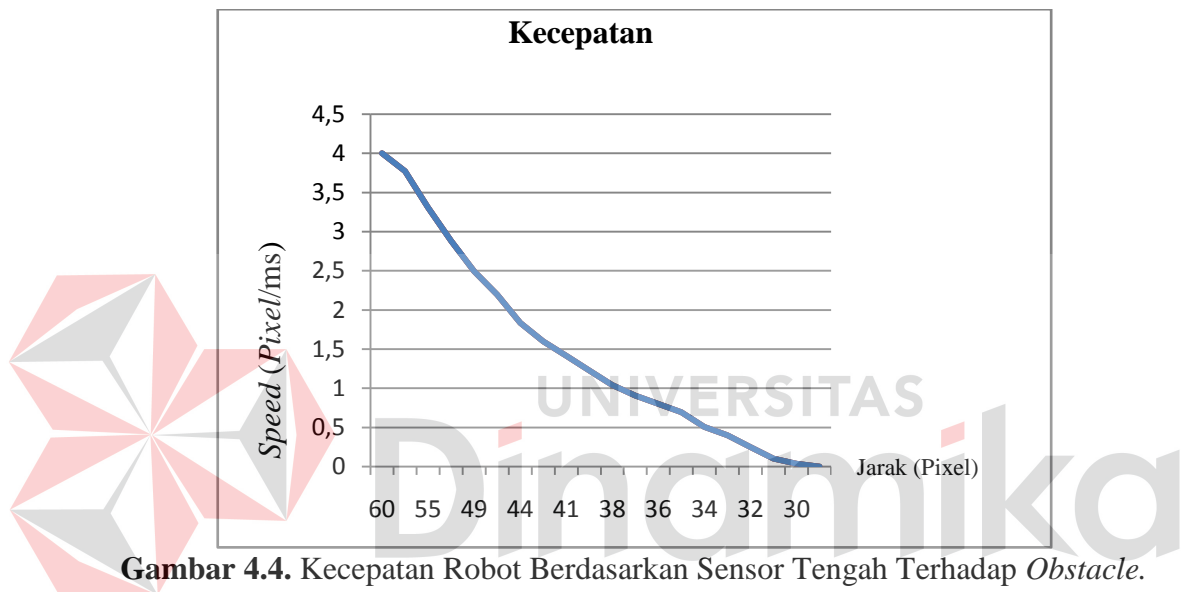
25	1,47	0	-10	-2
26	1,48	0	-9	-1,8
27	1,49	0	-7,9	-1,6
28	1,5	0	-7	-1,4
29	1,51	0	-6,2	-1,2
30	1,52	0	-5,5	-1,1
31	1,53	0	-4,8	-1
32	1,54	0	-4,3	-0,9
33	1,55	0	-3,8	-0,8
34	1,56	0	-3,3	-0,7
35	1,57	0	-2,9	-0,6
36	1,58	0	-2,6	-0,5
37	1,59	0	-2,3	-0,5
38	1,6	0	-2	-0,4
39	1,61	0	-1,8	-0,4
40	1,62	0	-1,5	-0,3
41	1,63	0	-1,4	-0,3
42	1,64	0	-1,2	-0,2
43	1,65	0	-1,1	-0,2
44	1,67	0	-0,8	-0,2
45	1,68	0	-0,7	-0,1
46	1,69	0	-0,6	-0,1
47	1,7	0	-0,5	-0,1
48	1,71	0	0	0
49	1,72	0	0	0
50	1,73	0	0	0

Tabel 4.11. *Output Pergerakan Berdasarkan Sensor Kanan dan Arah Target.*

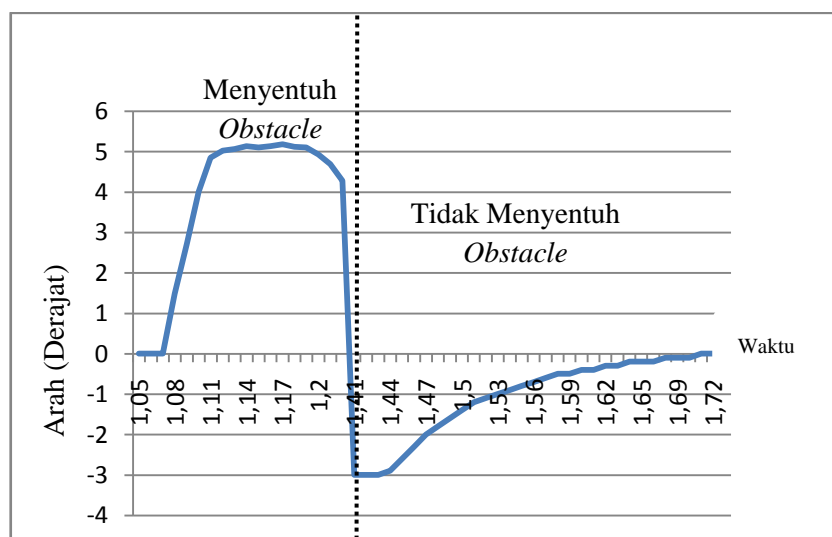
No	Waktu	Error Sensor	Error Target	<i>Output Pergerakan (derajat)</i>
1	1,14	0	0	0
2	1,15	0	0	0
3	1,16	0	0	0
4	1,17	-1,4	-1	-0,8
5	1,18	-6,4	-5	-2,5
6	1,19	-11,4	-7	-3,5
7	1,2	-15	-9	-4,7
8	1,21	-18,4	-9	-4,9
9	1,22	-21,6	-9	-5,1
10	1,23	-25	-10	-5,2
11	1,24	-28	-10	-5,3

12	1,25	-30,2	-9	-5,3
13	1,26	-32,2	-8	-5,2
14	1,27	-34,2	-7	-5,1
15	1,28	-35	-5	-5
16	1,29	-35,6	-2	-4,8
17	1,3	-30	6	-4,2
18	1,47	0	18	3
19	1,48	0	16	3
20	1,49	0	15	2,97
21	1,5	0	13	2,63
22	1,51	0	12	2,33
23	1,52	0	10	2,07
24	1,53	0	9,1	1,83
25	1,54	0	8,1	1,62
26	1,55	0	7,1	1,43
27	1,56	0	6,3	1,26
28	1,57	0	5,6	1,11
29	1,58	0	4,9	0,98
30	1,59	0	4,3	0,87
31	1,6	0	3,8	0,77
32	1,61	0	3,4	0,67
33	1,62	0	3	0,59
34	1,63	0	2,6	0,52
35	1,64	0	2,3	0,46
36	1,65	0	2	0,41
37	1,7	0	1,1	0,21
38	1,75	0	0,6	0,11
39	1,76	0	0,5	0,1
40	1,77	0	0,4	0,09
41	1,78	0	0,4	0,07
42	1,79	0	0,3	0,07
43	1,8	0	0,3	0,06
44	1,81	0	0,2	0,05
45	1,88	0	0,1	0,02
46	1,96	0	0	0
47	1,97	0	0	0
48	1,98	0	0	0
49	1,99	0	0	0
50	2	0	0	0

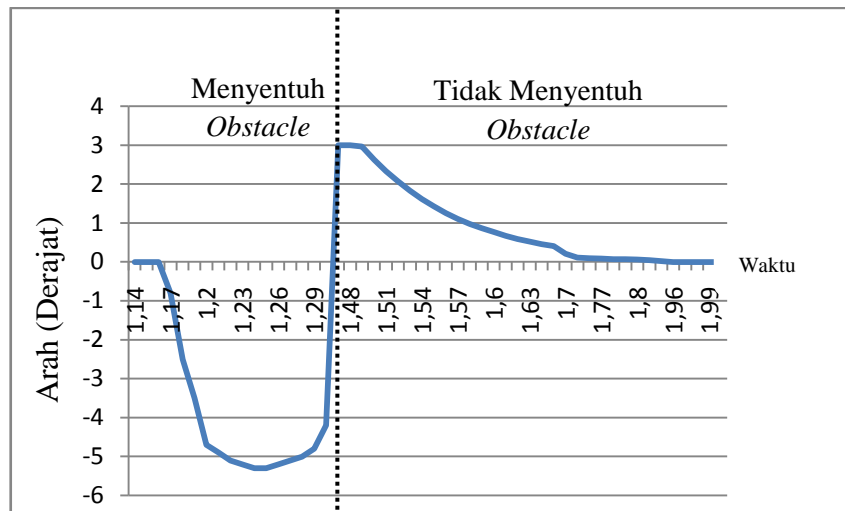
Dari tabel 4.9 sampai 4.11 didapatkan hasil parameter fuzzy terhadap respon robot melewati *obstacle* dan menuju koordinat tujuannya. Respon ini didapatkan dari nilai sensor ketika menyentuh *obstacle* sehingga dapat memberikan hasil kecepatan dan arah belok robot yang telah diharapkan. Gambar 4.4 sampai 4.6 merupakan penggambaran grafik berdasarkan data yang diperoleh dari tabel 4.9 sampai 4.11.



Gambar 4.4. Kecepatan Robot Berdasarkan Sensor Tengah Terhadap *Obstacle*.



Gambar 4.5. Arah Belok Robot Berdasarkan Sensor Kiri Terhadap *Obstacle*.



Gambar 4.6. Arah Belok Robot Berdasarkan Sensor Kanan Terhadap *Obstacle*.

D. Pembahasan

Dengan diperolehnya data-data tersebut telah memberikan parameter fuzzy yang sesuai dalam pergerakan simulasi. Dimana Pergerakan kecepatan robot dapat diatur menggunakan nilai jarak yang dihasilkan oleh sensor tengah dan jarak robot terhadap target. Semakin kecil nilai jarak baca sensor, semakin pelan pergerakan dari robot (Gambar 4.4). Nilai *error* target dan *error* sensor yang didapatkan dari sensor kanan dan kiri dapat memberikan pergerakan dalam menentukan kesalahan arah target dan proses melewati *obstacle* (Gambar.4.5 dan 4.6).

4.1.7. Pengujian Pembentukan Formasi Robot ke Formasi Awal

A. Tujuan

Pengujian pembentukan formasi robot ke formasi awal adalah pengujian yang dilakukan untuk mengetahui respon robot *follower* terhadap robot utama (pemimpin) dalam upayanya membentuk formasi awal setelah melewati *obstacle*.

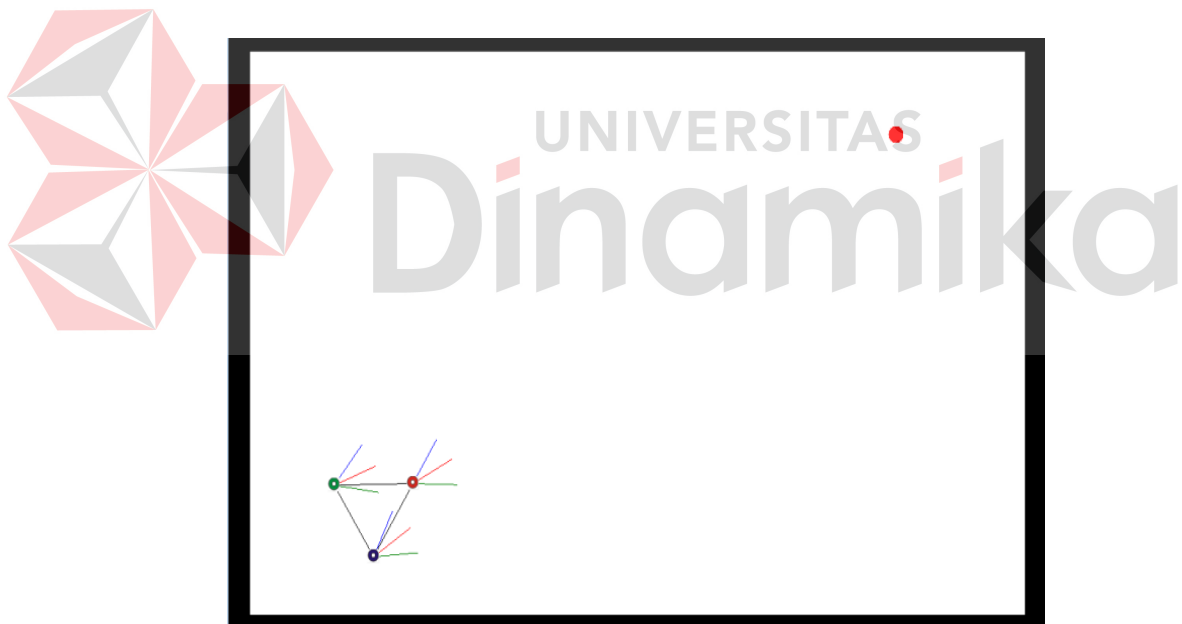
B. Prosedur Pengujian

1. Run program (*user/manual*).
2. Pemilihan target tujuan robot (*user/manual*).

3. Penekanan *push button start (user/manual)*.
4. Letakkan *obstacle* pada area pergerakan robot, usahakan agar peletakan *obstacle* dapat menghambat pergerakan masing-masing robot, sehingga formasi robot akan terpecah. Setelah melewati *obstacle* diharapkan robot dapat kembali ke formasi semula (*auto*).

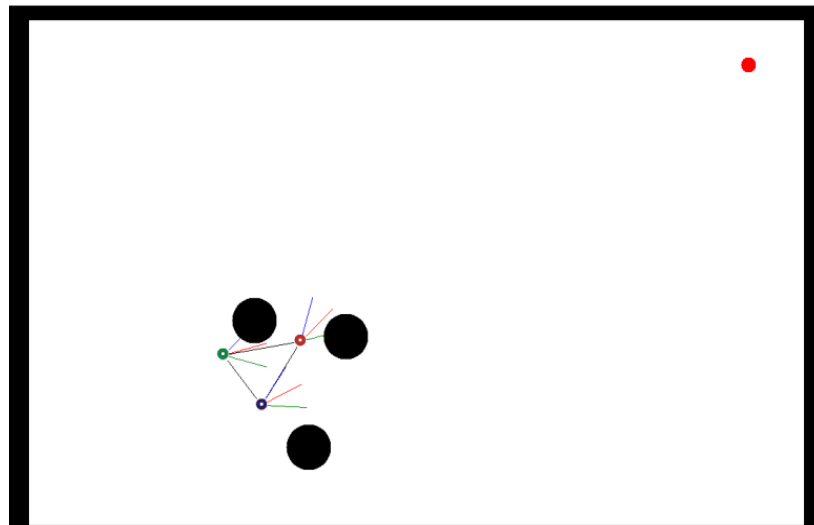
C. Hasil Pengujian

Sesuai dengan langkah pengujian yang dilakukan, didapatkan hasil dari pergerakan koloni dalam pembentukan formasi robot setelah melewati *obstacle*. Berikut merupakan hasil dari pergerakan yang dilakukan koloni robot ketika melewati 3 *obstacle* seperti Gambar 4.7 sampai 4.10 berikut ini.



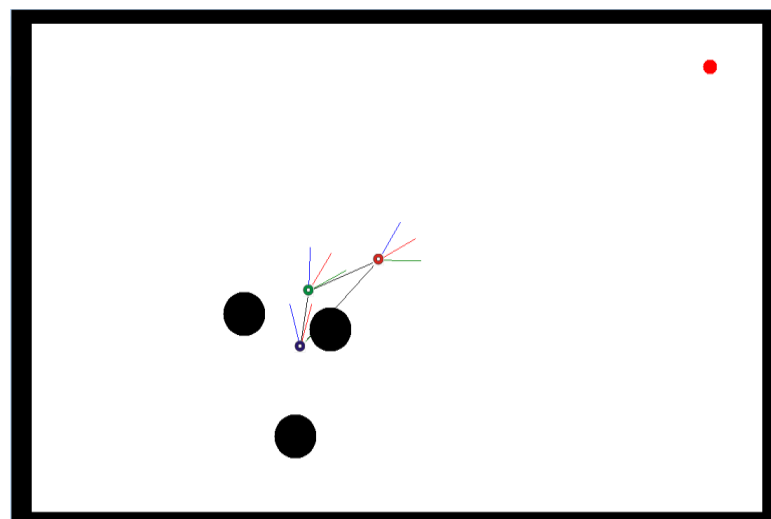
Gambar 4.7. Formasi Awal Koloni Robot.

Gambar 4.7 merupakan proses pembentukan formasi awal di mana robot terdepan merupakan robot pemimpin (merah) sedangkan robot A (biru) dan B (hijau) merupakan robot *follower*. Formasi terbentuk ketika robot pemimpin mengirimkan koordinat target terhadap robot *follower*. Koordinat tersebut berfungsi Sebagai intruksi dalam pembentukan formasi.



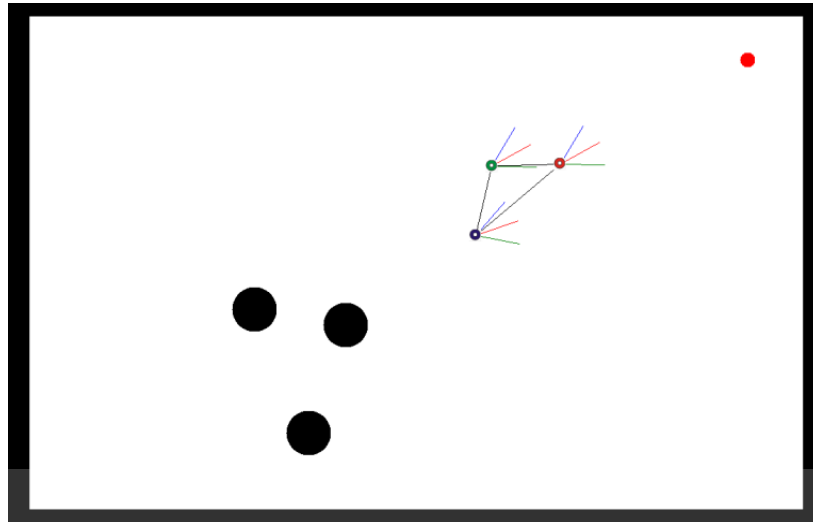
Gambar 4.8. Formasi Ketika Terdapat *Obstacle*.

Gambar 4.8 merupakan proses pergerakan robot terhadap *obstacle*. Robot pemimpin (utama) mencoba menghindari *obstacle* dan bergerak ke arah kiri sedangkan robot *follower* B (hijau) menghindari *obstacle* dengan bergerak ke arah kanan. Pergerakan dilakukan berdasarkan keputusan dari sensor yang menyentuh *obstacle*. Dari sinilah formasi robot akan terpecah karena masing-masing robot akan menentukan jalur yang sesuai terhadap dirinya dalam upaya menghindari *obstacle*.



Gambar 4.9. Formasi Melewati *Obstacle*.

Gambar 4.9 merupakan proses formasi robot setelah melewati *obstacle*. Robot *follower* yang telah melewati *obstacle* akan menentukan arah pergerakannya kembali berdasarkan koordinat target yang telah dikirimkan oleh robot pemimpin.



Gambar 4.10. Pembentukan Formasi Awal.

Gambar 4.10 merupakan proses pembentukan formasi robot ke formasi awal. setelah semua robot *follower* berhasil melewati *obstacle*. robot *follower* akan berusaha untuk menuju target yang telah diberikan oleh robot pemimpin sehingga formasi akan dapat terbentuk kembali.

D. Pembahasan

Sesuai hasil pengujian, Robot *follower* memiliki kemampuan untuk memberikan keputusan pergerakan terhadap dirinya sendiri dengan menggunakan sensor *ultrasonic*. Keputusan tersebut digunakan untuk menentukan jalur pergerakan yang sesuai dalam proses menghindari *obstacle* dan robot *follower* dapat dengan segera mengoreksi kesalahan arah terhadap titik koordinat yang telah diberikan oleh robot utama setelah melewati *obstacle*. Sehingga robot *follower* dapat membentuk formasi awal yang diharapkan oleh robot utama.

4.2. Pengujian Sistem Secara Keseluruhan

Hasil pengujian keseluruhan dari sistem ini terbagi atas beberapa pergerakan diantaranya respon robot *follower* terhadap koordinat target yang diberikan oleh robot utama. Dalam proses ini dilakukan pengambilan data koordinat selama 60 detik saat program dijalankan. Data tersebut diperoleh saat robot pemimpin bergerak menuju koordinat tujuan yang telah ditetapkan. Pada percobaan ini jarak pergerakan robot pemimpin terhadap koordinat tujuannya adalah 738 pixel. Sehingga pengiriman koordinat tujuan robot *follower* akan stabil ketika robot pemimpin mencapai titik tujuannya dan pengambilan data koordinat akan berhenti selama batas waktu 60 detik. Berikut merupakan hasil dari pengujian sistem secara keseluruhan.

4.2.1. Gerak Robot *Follower* Terhadap Koordinat Target

A. Hasil Pengujian

Dari hasil pengujian ini peneliti mengambil data pergerakan robot *follower* terhadap koordinat target, saat melewati *obstacle* dan tidak melewati *obstacle*. Dalam proses melewati *obstacle* peneliti mengambil dua data yaitu respon robot *follower* terhadap koordinat target ketika melewati satu *obstacle* dan respon robot *follower* ketika melewati dua *obstacle*. Setiap pengujian, data diperoleh dari *input* koordinat yang dikirimkan oleh robot utama. Dikarenakan koordinat ini merupakan koordinat bergerak, dengan demikian data tersebut memiliki nilai koordinat target yang berubah-ubah. Berikut merupakan hasil dari pengujian:

a. Tidak Melewati *Obstacle*.Tabel 4.12. Percobaan *Follower A* Tidak Melewati *Obstacle*.

PERCOBAAN FOLLOWER A								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	135	431	112	431	23	0	17	0
5,1	239	431	216	432	23	1	9,62	0,23
7,7	343	430	320	430	23	0	6,71	0
10,3	447	430	424	430	23	0	5,15	0
12,9	551	429	528	429	23	0	4,17	0
15,5	655	429	632	429	23	0	3,51	0
18,1	759	428	736	428	23	0	3,03	0
20,7	805	428	780	428	25	0	3,11	0
23,3	813	428	792	428	21	0	2,58	0
25,9	816	428	800	428	16	0	1,96	0
28,5	818	428	806	428	12	0	1,47	0
31,1	818	428	811	428	7	0	0,86	0
33,7	819	428	815	428	4	0	0,49	0
36,3	819	428	819	428	0	0	0	0
38,9	819	428	819	428	0	0	0	0
41,5	819	428	819	428	0	0	0	0
44,1	819	428	819	428	0	0	0	0
46,7	819	428	819	428	0	0	0	0
49,3	819	428	819	428	0	0	0	0
51,9	819	428	819	428	0	0	0	0
54,5	819	428	819	428	0	0	0	0
57,1	819	428	819	428	0	0	0	0
59,7	819	428	819	428	0	0	0	0
Rata-Rata					10,696	0,043	2,594	0,01

Tabel 4.13. Percobaan *Follower B* Tidak Melewati *Obstacle*.

PERCOBAAN FOLLOWER B								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	134	347	110	346	24	1	17,9	0,288
5,1	238	346	214	347	24	1	10,1	0,29
7,7	342	345	318	345	24	0	7,02	0
10,3	446	345	422	345	24	0	5,38	0
12,9	550	344	526	344	24	0	4,36	0
15,5	654	344	630	344	24	0	3,67	0
18,1	758	343	734	343	24	0	3,17	0

20,7	804	343	778	343	26	0	3,23	0
23,3	813	343	790	343	23	0	2,83	0
25,9	816	343	799	343	17	0	2,08	0
28,5	817	343	805	343	12	0	1,47	0
31,1	818	343	810	343	8	0	0,98	0
33,7	818	343	814	343	4	0	0,49	0
36,3	818	343	818	343	0	0	0	0
38,9	818	343	818	343	0	0	0	0
41,5	818	343	818	343	0	0	0	0
44,1	818	343	818	343	0	0	0	0
46,7	818	343	818	343	0	0	0	0
49,3	818	343	818	343	0	0	0	0
51,9	818	343	818	343	0	0	0	0
54,5	818	343	818	343	0	0	0	0
57,1	818	343	818	343	0	0	0	0
59,7	818	343	818	343	0	0	0	0
Rata-Rata					11,217	0,087	2,725	0,025

b. Melewati Satu Obstacle

Tabel 4.14. Percobaan ke 1 *Follower A* Melewati Satu Obstacle.

PERCOBAAN KE 1: FOLLOWER A								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	135	431	112	431	23	0	17	0
5,1	230	424	186	434	44	10	19,1	2,36
7,7	302	396	241	397	61	1	20,2	0,25
10,3	348	410	285	357	63	53	18,1	12,93
12,9	400	412	332	358	68	54	17	13,11
15,5	458	414	409	400	49	14	10,7	3,382
18,1	554	417	512	416	42	1	7,58	0,24
20,7	658	421	616	419	42	2	6,38	0,475
23,3	760	424	715	423	45	1	5,92	0,236
25,9	804	426	765	424	39	2	4,85	0,469
28,5	813	426	790	425	23	1	2,83	0,235
31,1	816	426	799	425	17	1	2,08	0,235
33,7	817	426	805	426	12	0	1,47	0
36,3	817	426	810	426	7	0	0,86	0
38,9	817	426	814	426	3	0	0,37	0
41,5	817	426	817	426	0	0	0	0
44,1	817	426	817	426	0	0	0	0
46,7	817	426	817	426	0	0	0	0

49,3	817	426	817	426	0	0	0	0
51,9	817	426	817	426	0	0	0	0
54,5	817	426	817	426	0	0	0	0
57,1	817	426	817	426	0	0	0	0
59,7	817	426	817	426	0	0	0	0
Rata-Rata					23,391	6,087	5,845	1,475

Tabel 4.15. Percobaan ke 2 *Follower A* Melewati Satu *Obstacle*.

PERCOBAAN KE 2: FOLLOWER A								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	135	431	112	431	23	0	17	0
5,1	237	431	216	432	21	1	8,86	0,23
7,7	261	420	243	424	18	4	6,9	0,95
10,3	339	465	302	418	37	47	10,9	10,11
12,9	400	464	352	448	48	16	12	3,448
15,5	452	460	405	458	47	2	10,4	0,435
18,1	504	456	459	458	45	2	8,93	0,44
20,7	577	450	534	453	43	3	7,45	0,67
23,3	679	442	636	445	43	3	6,33	0,68
25,9	773	435	728	438	45	3	5,82	0,69
28,5	812	431	777	434	35	3	4,31	0,7
31,1	819	431	798	433	21	2	2,56	0,46
33,7	821	431	806	432	15	1	1,83	0,23
36,3	822	431	811	431	11	0	1,34	0
38,9	822	431	816	431	6	0	0,73	0
41,5	822	431	820	431	2	0	0,24	0
44,1	822	431	822	431	0	0	0	0
46,7	822	431	822	431	0	0	0	0
49,3	822	431	822	431	0	0	0	0
51,9	822	431	822	431	0	0	0	0
54,5	822	431	822	431	0	0	0	0
57,1	822	431	822	431	0	0	0	0
59,7	822	431	822	431	0	0	0	0
Rata-Rata					20	3,783	4,591	0,828

Tabel 4.16. Percobaan ke 1 *Follower B* Melewati Satu *Obstacle*.

PERCOBAAN KE 1: FOLLOWER B								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	134	347	110	346	24	1	17,9	0,288
5,1	208	343	183	353	25	10	12	2,92

7,7	290	324	253	336	37	12	12,8	3,7
10,3	351	326	310	325	41	1	11,7	0,307
12,9	403	327	362	326	41	1	10,2	0,306
15,5	461	329	422	328	39	1	8,46	0,304
18,1	556	332	518	331	38	1	6,83	0,301
20,7	660	336	622	335	38	1	5,76	0,298
23,3	762	339	721	338	41	1	5,38	0,295
25,9	807	341	771	340	36	1	4,46	0,293
28,5	816	341	795	340	21	1	2,57	0,293
31,1	818	341	803	341	15	0	1,83	0
33,7	819	341	809	341	10	0	1,22	0
36,3	820	341	813	341	7	0	0,85	0
38,9	820	341	817	341	3	0	0,37	0
41,5	820	341	820	341	0	0	0	0
44,1	820	341	820	341	0	0	0	0
46,7	820	341	820	341	0	0	0	0
49,3	820	341	820	341	0	0	0	0
51,9	820	341	820	341	0	0	0	0
54,5	820	341	820	341	0	0	0	0
57,1	820	341	820	341	0	0	0	0
59,7	820	341	820	341	0	0	0	0
Rata-Rata					18,087	1,348	4,449	0,405

Tabel 4.17. Percobaan ke 2 *Follower B Melewati Satu Obstacle.*

PERCOBAAN KE 2: FOLLOWER B								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	134	347	110	346	24	1	17,9	0,288
5,1	236	346	214	347	22	1	9,32	0,29
7,7	307	349	243	344	64	5	20,8	1,433
10,3	364	405	298	360	66	45	18,1	11,11
12,9	393	379	330	411	63	32	16	8,44
15,5	445	375	374	430	71	55	16	14,7
18,1	497	371	432	412	65	41	13,1	11,1
20,7	571	365	527	371	44	6	7,71	1,64
23,3	672	357	630	360	42	3	6,25	0,84
25,9	766	350	722	353	44	3	5,74	0,86
28,5	806	347	771	350	35	3	4,34	0,86
31,1	812	346	792	348	20	2	2,46	0,58
33,7	814	346	800	347	14	1	1,72	0,29
36,3	815	346	805	347	10	1	1,23	0,29
38,9	815	346	809	347	6	1	0,74	0,29

41,5	815	346	813	346	2	0	0,25	0
44,1	815	346	815	346	0	0	0	0
46,7	815	346	815	346	0	0	0	0
49,3	815	346	815	346	0	0	0	0
51,9	815	346	815	346	0	0	0	0
54,5	815	346	815	346	0	0	0	0
57,1	815	346	815	346	0	0	0	0
59,7	815	346	815	346	0	0	0	0
Rata-Rata					25,739	8,696	6,159	2,305

c. Melewati dua *Obstacle*

Tabel 4.18. Percobaan ke 1 *Follower A Melewati Dua Obstacle.*

PERCOBAAN KE 1: FOLLOWER A								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	135	431	112	431	23	0	17	0
5,1	245	438	198	434	47	4	19,2	0,913
7,7	316	379	252	425	64	46	20,3	12,1
10,3	350	410	299	369	51	41	14,6	10
12,9	409	422	332	356	77	66	18,8	15,64
15,5	461	423	365	370	96	53	20,8	12,53
18,1	513	424	413	392	100	32	19,5	7,547
20,7	565	424	494	412	71	12	12,6	2,83
23,3	637	425	593	424	44	1	6,91	0,235
25,9	739	426	697	426	42	0	5,68	0
28,5	799	427	754	426	45	1	5,63	0,234
31,1	814	427	791	427	23	0	2,83	0
33,7	816	427	800	427	16	0	1,96	0
36,3	817	427	806	427	11	0	1,35	0
38,9	818	427	810	427	8	0	0,98	0
41,5	818	427	815	427	3	0	0,37	0
44,1	818	427	818	427	0	0	0	0
46,7	818	427	818	427	0	0	0	0
49,3	818	427	818	427	0	0	0	0
51,9	818	427	818	427	0	0	0	0
54,5	818	427	818	427	0	0	0	0
57,1	818	427	818	427	0	0	0	0
59,7	818	427	818	427	0	0	0	0
Rata-Rata					31,348	11,130	7,326	2,696

Tabel 4.19. Percobaan ke 2 *Follower A* Melewati Dua *Obstacle*.

PERCOBAAN KE 2: FOLLOWER A								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	135	431	112	431	23	0	17	0
5,1	192	417	180	432	12	15	6,25	3,6
7,7	246	447	220	418	26	29	10,6	6,488
10,3	336	464	283	435	53	29	15,8	6,25
12,9	400	424	327	449	73	25	18,3	5,9
15,5	429	445	372	421	57	24	13,3	5,393
18,1	479	445	423	407	56	38	11,7	8,539
20,7	531	442	478	426	53	16	9,98	3,62
23,3	595	439	551	439	44	0	7,39	0
25,9	695	435	652	437	43	2	6,19	0,46
28,5	781	431	736	433	45	2	5,76	0,46
31,1	814	430	786	431	28	1	3,44	0,23
33,7	818	429	798	430	20	1	2,44	0,23
36,3	819	429	806	430	13	1	1,59	0,23
38,9	820	429	811	430	9	1	1,1	0,23
41,5	820	429	815	430	5	1	0,61	0,23
44,1	820	429	820	429	0	0	0	0
46,7	820	429	820	429	0	0	0	0
49,3	820	429	820	429	0	0	0	0
51,9	820	429	820	429	0	0	0	0
54,5	820	429	820	429	0	0	0	0
57,1	820	429	820	429	0	0	0	0
59,7	820	429	820	429	0	0	0	0
Rata-Rata					24,348	8,043	5,715	1,82

Tabel 4.20. Percobaan ke 1 *Follower B* Melewati Dua *Obstacle*.

PERCOBAAN KE 1: FOLLOWER B								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	134	347	110	346	24	1	17,9	0,288
5,1	210	361	196	349	14	12	6,67	3,324
7,7	286	322	252	353	34	31	11,9	9,63
10,3	368	327	319	327	49	0	13,3	0
12,9	435	362	365	337	70	25	16,1	6,906
15,5	462	338	407	373	55	35	11,9	10,4
18,1	514	339	459	379	55	40	10,7	11,8
20,7	566	339	514	355	52	16	9,19	4,72
23,3	638	340	595	341	43	1	6,74	0,29

25,9	740	341	697	341	43	0	5,81	0
28,5	800	342	754	341	46	1	5,75	0,292
31,1	815	342	791	342	24	0	2,94	0
33,7	817	342	800	342	17	0	2,08	0
36,3	818	342	806	342	12	0	1,47	0
38,9	819	342	811	342	8	0	0,98	0
41,5	819	342	815	342	4	0	0,49	0
44,1	819	342	819	342	0	0	0	0
46,7	819	342	819	342	0	0	0	0
49,3	819	342	819	342	0	0	0	0
51,9	819	342	819	342	0	0	0	0
54,5	819	342	819	342	0	0	0	0
57,1	819	342	819	342	0	0	0	0
59,7	819	342	819	342	0	0	0	0
Rata-Rata					23,913	7,043	5,388	2,072

Tabel 4.21. Percobaan ke 2 *Follower B* Melewati Dua *Obstacle*.

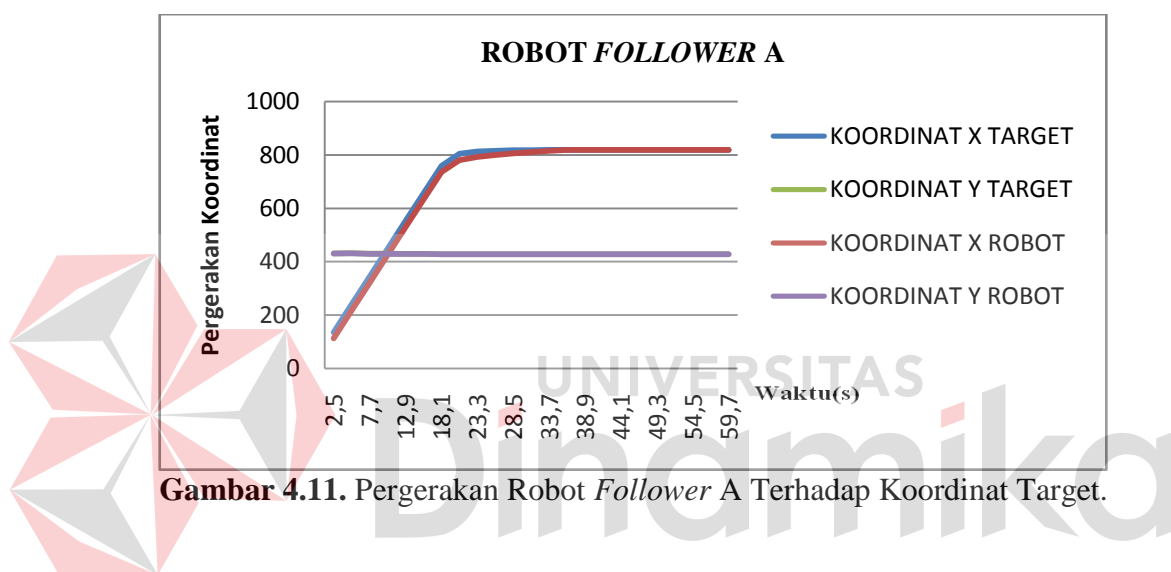
PERCOBAAN KE 2: FOLLOWER B								
Waktu (s)	Koordinat Target		Koordinat Robot		Kesalahan (Pixel)		Kesalahan (%)	
	X	Y	X	Y	X	Y	X	Y
2,5	134	347	110	346	24	1	17,9	0,288
5,1	219	336	178	347	41	11	18,7	3,27
7,7	288	399	230	350	58	49	20,1	12,28
10,3	333	395	268	388	65	7	19,5	1,772
12,9	371	365	297	423	74	58	19,9	15,9
15,5	424	362	332	425	92	63	21,7	17,4
18,1	476	360	377	401	99	41	20,8	11,4
20,7	527	358	449	377	78	19	14,8	5,31
23,3	591	355	547	358	44	3	7,45	0,85
25,9	691	350	649	352	42	2	6,08	0,57
28,5	777	346	733	348	44	2	5,66	0,58
31,1	810	345	782	346	28	1	3,46	0,29
33,7	814	345	795	346	19	1	2,33	0,29
36,3	816	345	802	345	14	0	1,72	0
38,9	816	345	807	345	9	0	1,1	0
41,5	817	345	811	345	6	0	0,73	0
44,1	817	345	816	345	1	0	0,12	0
46,7	817	345	817	345	0	0	0	0
49,3	817	345	817	345	0	0	0	0
51,9	817	345	817	345	0	0	0	0
54,5	817	345	817	345	0	0	0	0

57,1	817	345	817	345	0	0	0	0
59,7	817	345	817	345	0	0	0	0
Rata-Rata					32,087	11,217	7,915	3,052

B. Pembahasan

Dari Tabel 4.12 sampai 4.21 didapatkan pergerakan robot *follower* terhadap koordinat target yang dikirimkan robot utama sebagai berikut.

a. Tidak Melewati *Obstacle*

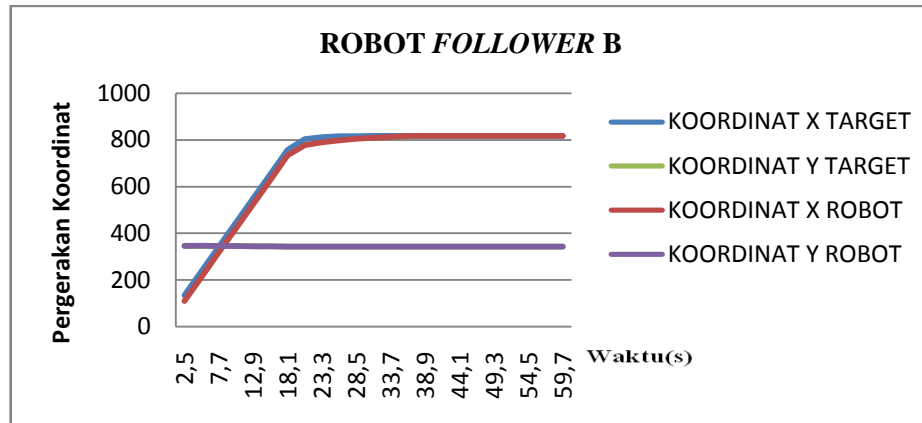


Gambar 4.11. Pergerakan Robot *Follower A* Terhadap Koordinat Target.

Gambar 4.11 menjelaskan proses pergerakan robot *follower* mengikuti koordinat target yang telah dikirimkan oleh robot utama. Hasil rata-rata dari kesalahan sumbu robot *follower A* saat mengikuti koordinat target berdasarkan data dari Tabel 4.12 adalah sebagai berikut:

$$\text{Rata-rata kesalahan sumbu X} = \frac{246}{23} = 10,696 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{1}{23} = 0,043 \text{ pixel}$$



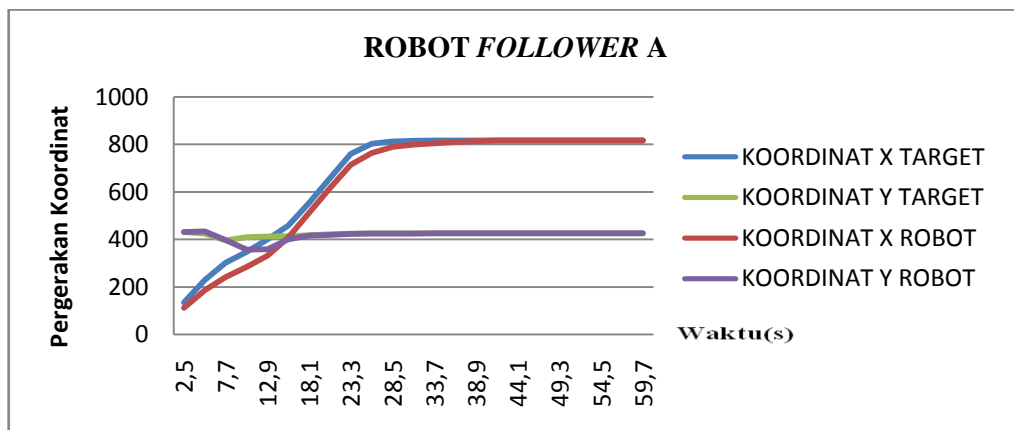
Gambar 4.12. Pergerakan Robot *Follower B* Terhadap Koordinat Target.

Gambar 4.12 menjelaskan proses pergerakan robot *follower* mengikuti koordinat target yang telah dikirimkan oleh robot utama. Hasil rata-rata dari kesalahan sumbu robot *follower A* saat mengikuti koordinat target berdasarkan data dari Tabel 4.13 adalah sebagai berikut:

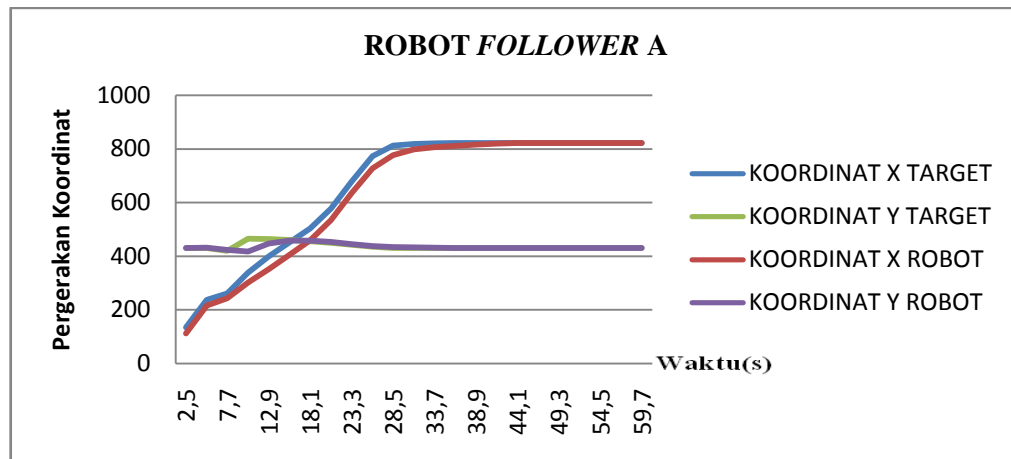
$$\text{Rata-rata kesalahan sumbu X} = \frac{258}{23} = 11,217 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{2}{23} = 0,086 \text{ pixel}$$

b. Melewati Satu *Obstacle*



Gambar 4.13. Percobaan ke 1 Pergerakan Robot *Follower A* Terhadap Koordinat Target Melewati 1 *Obstacle*.



Gambar 4.14. Percobaan ke 2 Pergerakan Robot *Follower A* Terhadap Koordinat Target Melewati 1 *Obstacle*.

Gambar 4.13 dan 4.14 menjelaskan proses pergerakan robot *follower* mengikuti koordinat target yang telah dikirimkan oleh robot utama. Hasil rata-rata dari kesalahan sumbu robot *follower A* saat mengikuti koordinat target berdasarkan data dari Tabel 4.14 dan 4.15 adalah sebagai berikut:

Percobaan ke 1:

$$\text{Rata-rata kesalahan sumbu X} = \frac{538}{23} = 23,391 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{140}{23} = 6,087 \text{ pixel}$$

Percobaan ke 2:

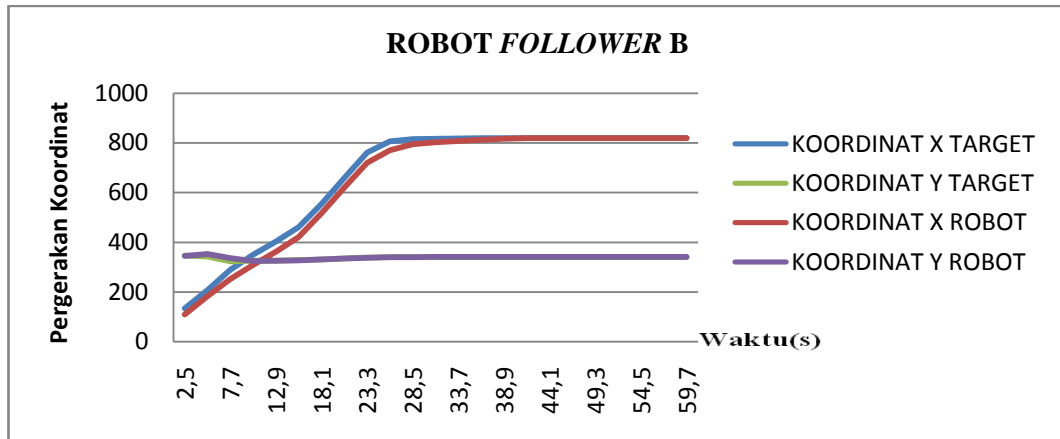
$$\text{Rata-rata kesalahan sumbu X} = \frac{460}{23} = 20 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{87}{23} = 3,783 \text{ pixel}$$

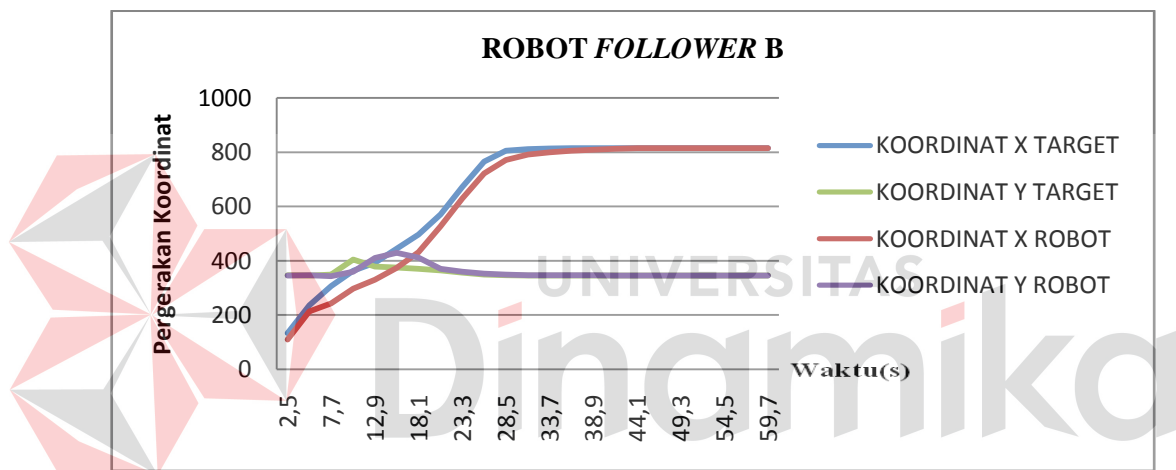
Rata – rata keseluruhan dari percobaan ke 1 dan percobaan ke 2:

$$\text{Rata-rata kesalahan sumbu X} = \frac{23,391+20}{2} = 21,696 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{6,087+3,783}{2} = 4,935 \text{ pixel}$$



Gambar 4.15. Percobaan ke 1 Pergerakan Robot *Follower B* Terhadap Koordinat Target Melewati 1 *Obstacle*.



Gambar 4.16. Percobaan ke 2 Pergerakan Robot *Follower B* Terhadap Koordinat Target Melewati 1 *Obstacle*.

Gambar 4.15 dan 4.16 menjelaskan proses pergerakan robot *follower* mengikuti koordinat target yang telah dikirimkan oleh robot utama. Hasil rata-rata dari kesalahan sumbu robot *follower B* saat mengikuti koordinat target berdasarkan data dari Tabel 4.16 dan 4.17 adalah sebagai berikut:

Percobaan ke 1:

$$\text{Rata-rata kesalahan sumbu X} = \frac{416}{23} = 18,087 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{31}{23} = 1,348 \text{ pixel}$$

Percobaan ke 2:

$$\text{Rata-rata kesalahan sumbu X} = \frac{592}{23} = 25,739 \text{ pixel}$$

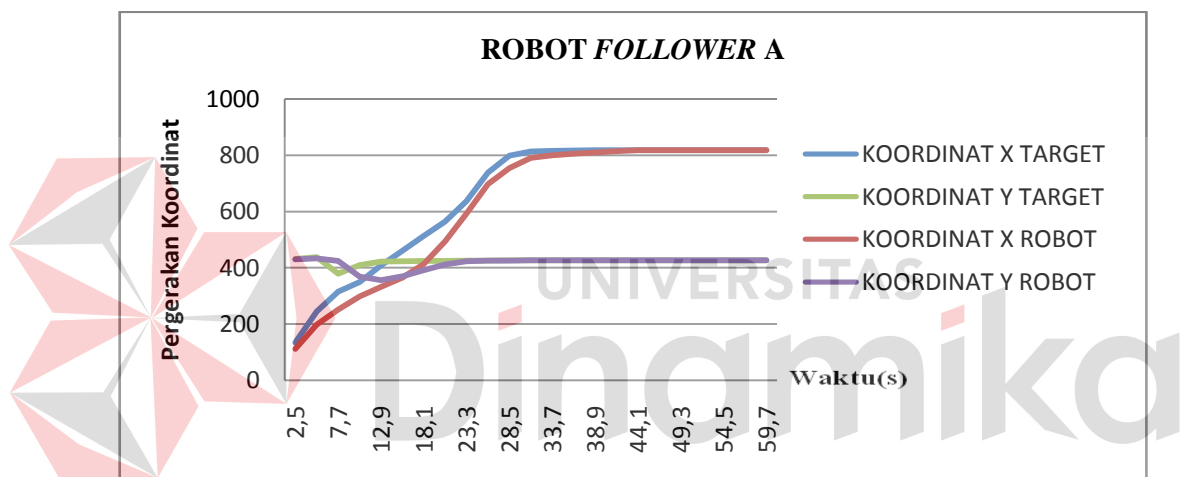
$$\text{Rata-rata kesalahan sumbu Y} = \frac{200}{23} = 8,696 \text{ pixel}$$

Rata – rata keseluruhan dari percobaan ke 1 dan percobaan ke 2:

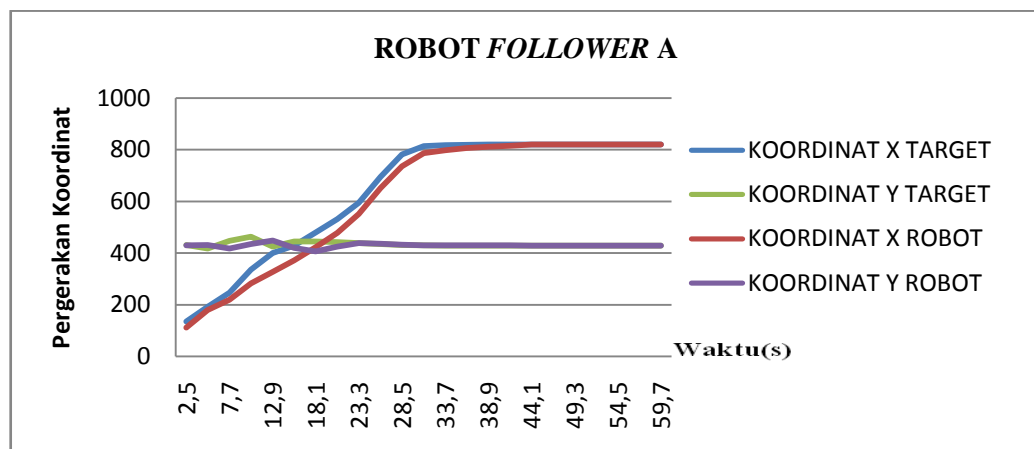
$$\text{Rata-rata kesalahan sumbu X} = \frac{18,087 + 25,739}{2} = 21,912 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{1,348 + 8,696}{2} = 10,044 \text{ pixel}$$

c. Melewati Dua *Obstacle*



Gambar 4.17. Percobaan ke 1 Pergerakan Robot *Follower A* Terhadap Koordinat Target Melewati 2 *Obstacle*.



Gambar 4.18. Percobaan ke 2 Pergerakan Robot *Follower A* Terhadap Koordinat Target Melewati 2 *Obstacle*.

Gambar 4.17 dan 4.18 menjelaskan proses pergerakan robot *follower* mengikuti koordinat target yang telah dikirimkan oleh robot utama. Hasil rata-rata dari kesalahan sumbu robot *follower* A saat mengikuti koordinat target berdasarkan data dari Tabel 4.18 dan 4.19 adalah sebagai berikut:

Percobaan ke 1:

$$\text{Rata-rata kesalahan sumbu X} = \frac{720}{23} = 31,348 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{256}{23} = 11,13 \text{ pixel}$$

Percobaan ke 2:

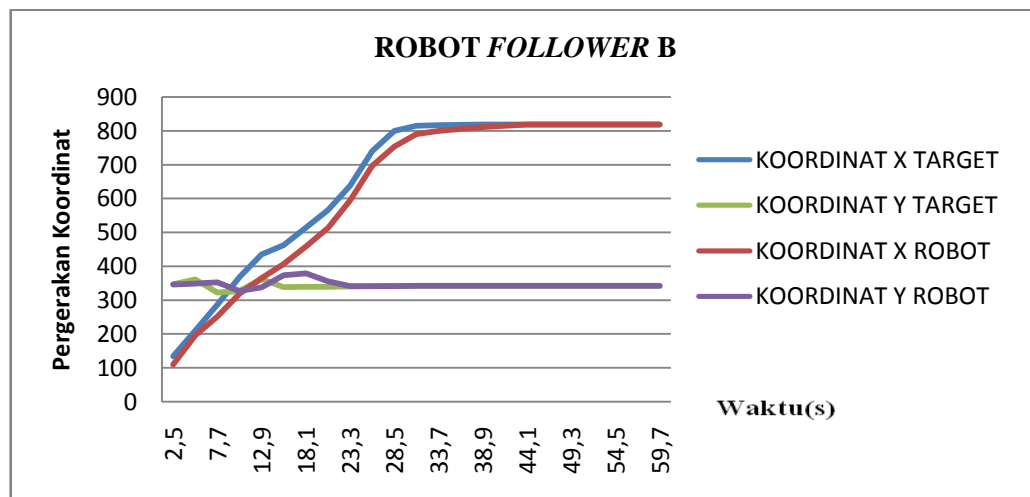
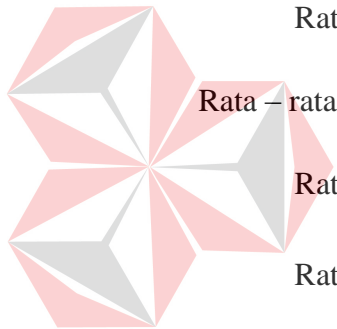
$$\text{Rata-rata kesalahan sumbu X} = \frac{560}{23} = 24,348 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{185}{23} = 8,044 \text{ pixel}$$

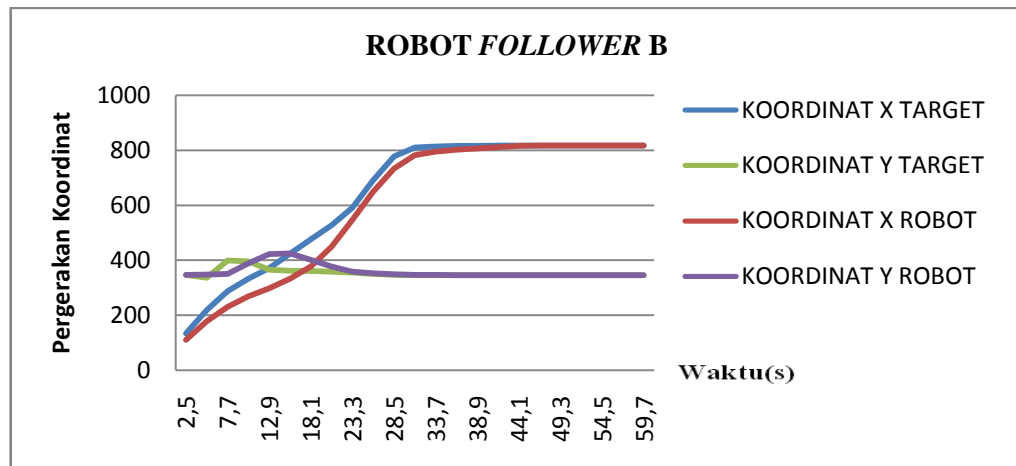
Rata-rata keseluruhan dari percobaan ke 1 dan percobaan ke 2:

$$\text{Rata-rata kesalahan sumbu X} = \frac{31,348 + 24,348}{2} = 27,848 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{11,13 + 8,044}{2} = 9,587 \text{ pixel}$$



Gambar 4.19. Percobaan ke 1 Pergerakan Robot *Follower* B Terhadap Koordinat Target Melewati 2 *Obstacle*.



Gambar 4.20. Percobaan ke 2 Pergerakan Robot *Follower B* Terhadap Koordinat Target Melewati 2 *Obstacle*.

Gambar 4.19 dan 4.20 menjelaskan proses pergerakan robot *follower* mengikuti koordinat target yang telah dikirimkan oleh robot utama. Hasil rata-rata dari kesalahan sumbu robot *follower B* saat mengikuti koordinat target berdasarkan data dari Tabel 4.20 dan 4.21 adalah sebagai berikut:

Percobaan ke 1:

$$\text{Rata-rata kesalahan sumbu X} = \frac{550}{23} = 23,913 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{162}{23} = 7,044 \text{ pixel}$$

Percobaan ke 2:

$$\text{Rata-rata kesalahan sumbu X} = \frac{738}{23} = 32,087 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{258}{23} = 11,217 \text{ pixel}$$

Rata – rata keseluruhan dari percobaan ke 1 dan percobaan ke 2:

$$\text{Rata-rata kesalahan sumbu X} = \frac{23,913 + 32,087}{2} = 28 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{7,044 + 11,217}{2} = 9,131 \text{ pixel}$$

Berikut merupakan persentase rata-rata kesalahan yang terjadi dari setiap percobaan pada robot *follower A* dan *B* dapat dihitung sebagai berikut.

a. Tidak Melewati *Obstacle**Follower A:*

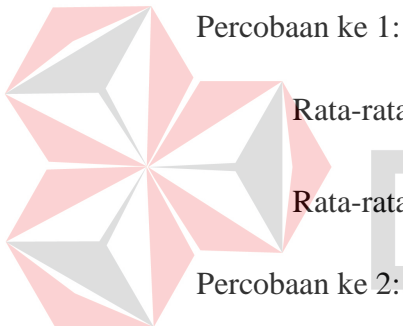
$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{59,66}{23} = 2,594 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{0,23}{23} = 0,01 \%$$

Follower B:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{62,68}{23} = 2,725 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{0,578}{23} = 0,025 \%$$

b. Melewati Satu *Obstacle**Follower A:*

Percobaan ke 1:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{134,44}{23} = 5,845 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{33,922}{23} = 1,475 \%$$

Percobaan ke 2:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{105,6}{23} = 4,591 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{19,043}{23} = 0,828 \%$$

Rata – rata keseluruhan dari percobaan ke 1 dan percobaan ke 2:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{5,845+4,591}{2} = 5,218 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{1,475+0,828}{2} = 1,152 \%$$

Follower B:

Percobaan ke 1:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{102,33}{23} = 4,449 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{9,305}{23} = 0,405 \%$$

Percobaan ke 2:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{141,66}{23} = 6,159 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{53,011}{23} = 2,305 \%$$

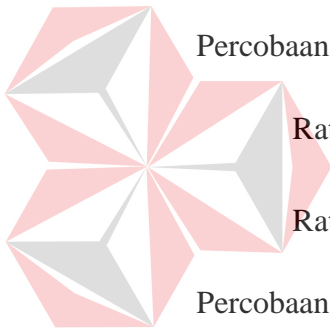
Rata – rata keseluruhan dari percobaan ke 1 dan percobaan ke 2:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{4,449+6,159}{2} = 5,304 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{0,405+2,305}{2} = 1,355 \%$$

c. Melewati Dua *Obstacle*

Follower A:



Percobaan ke 1:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{168,51}{23} = 7,327 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{62,029}{23} = 2,697 \%$$

Percobaan ke 2:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{131,45}{23} = 5,715 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{41,86}{23} = 1,82 \%$$

Rata – rata keseluruhan dari percobaan ke 1 dan percobaan ke 2:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{7,327+5,715}{2} = 6,521 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{2,697+1,82}{2} = 2,258 \%$$

Follower B:

Percobaan ke 1:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{123,92}{23} = 5,388 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{47,65}{23} = 2,072 \%$$

Percobaan ke 2:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{182,05}{23} = 7,915 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{70,2}{23} = 3,052 \%$$

Rata – rata keseluruhan dari percobaan ke 1 dan percobaan ke 2:

$$\text{Rata-rata persentase kesalahan sumbu X} = \frac{5,388+7,915}{2} = 6,651 \%$$

$$\text{Rata-rata persentase kesalahan sumbu Y} = \frac{2,072+3,052}{2} = 2,562 \%$$

4.3. Pembahasan Keseluruhan

Dari hasil pengujian seperti pada Tabel 4.12 sampai 4.21 didapatkan hasil

pergerakan kesalahan secara keseluruhan:

a. Pergerakan robot *follower* A:

$$\text{Rata-rata kesalahan sumbu X} = \frac{10,696+21,696+27,848}{3} = 20,171 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{0,043+4,935+9,587}{3} = 4,855 \text{ pixel}$$

b. Pergerakan robot *follower* B:

$$\text{Rata-rata kesalahan sumbu X} = \frac{11,217+21,912+28}{3} = 20,376 \text{ pixel}$$

$$\text{Rata-rata kesalahan sumbu Y} = \frac{0,086+10,044+9,131}{3} = 6,42 \text{ pixel}$$

Berikut merupakan hasil dari bentuk persentase kesalahan secara keseluruhan yang dilakukan oleh robot *follower*:

a. Pergerakan robot *follower* A :

$$\text{Persentase kesalahan sumbu X} = \frac{2,594+5,218+6,521}{3} = 4,778 \%$$

$$\text{Persentase kesalahan sumbu Y} = \frac{0,01+1,152+2,258}{3} = 0,807 \%$$

b. Pergerakan robot *follower* B :

$$\text{Persentase kesalahan sumbu X} = \frac{2,725+5,304+6,651}{3} = 4,893 \%$$

$$\text{Persentase kesalahan sumbu Y} = \frac{0,025+1,355+2,562}{3} = 1,314 \%$$

Pada hasil pengujian, robot utama berhasil mengirimkan titik koordinat yang akan dituju oleh robot *follower* dalam proses pembentukan formasi. Robot *follower* memiliki kemampuan untuk memberikan keputusan pergerakan terhadap dirinya sendiri dengan menggunakan sensor *ultrasonic* sebagai *input* fuzzy. Keputusan tersebut digunakan untuk menentukan jalur pergerakan yang sesuai dalam proses menghindari *obstacle* dan dapat dengan segera membentuk formasi awal yang diharapkan oleh robot utama setelah melewati *obstacle*. Dengan demikian robot *follower* tidak selalu terpacu oleh pergerakan dari koordinat tujuan yang diberikan oleh robot utama ketika melewati *obstacle*. Pergerakan koloni ini akan selalu berbeda tergantung dengan sudut target yang diberikan oleh *user*. Selain itu robot *follower* juga dapat menentukan jalur yang lebih sesuai dengan pergerakannya ketika melewati *obstacle* yang peletakannya telah ditentukan oleh *user*.

BAB V

PENUTUP

Berdasarkan hasil pengujian pada perangkat lunak yang dipergunakan dalam perancangan simulasi *trajectory planning* dan pembentukan formasi pada robot *obstacle avoidance* ini, maka dapat diambil beberapa kesimpulan dan saran sebagai berikut.

5.1. Kesimpulan

Adapun kesimpulan yang dapat dituliskan setelah melakukan analisa dari hasil sistem yang telah dibuat antara lain sebagai berikut :

1. Pengujian dalam pergerakan robot terhadap *obstacle* dapat dilakukan berdasarkan peletakan ketiga sensor *ultrasonic*. Peletakan sensor yang ideal berdasarkan jarak bacanya (60 *pixel*) yaitu peletakan sensor kanan, tengah, dan kiri yang masing-masing memiliki sudut baca 30° , 0° , dan -30° , karena sudut sensor tersebut dapat mendeteksi *obstacle* dengan cukup baik sehingga robot dapat bergerak secara stabil dalam menentukan arah pergerakannya terhadap *obstacle*. Algoritma fuzzy dengan pengelolaan hasil jarak banca sensor menggunakan metode sugeno dapat mengatur kecepatan dan arah belok robot dengan baik. Pergerakan kecepatan robot diatur menggunakan nilai jarak yang dihasilkan oleh sensor tengah dan jarak robot terhadap target. Nilai *error* target dan *error* sensor yang didapatkan dari sensor kanan dan kiri, dapat memberikan pergerakan dalam menentukan kesalahan arah target dan proses melewati *obstacle*.

2. Pengujian dari pembentukan formasi sesuai instruksi robot utama sebelum melewati *obstacle* dan pembentukan formasi setelah melewati *obstacle* dapat dilakukan berdasarkan pengiriman koordinat tujuan oleh robot utama, koordinasi kecepatan yang dilakukan berdasarkan jarak dari masing-masing robot terhadap robot lain, dan kemampuan robot *follower* untuk memberikan keputusan pergerakan terhadap dirinya sendiri dengan menggunakan sensor *ultrasonic*. Koordinat tujuan dari percobaan ini memiliki jarak 60 *pixel* dari titik posisi robot dan setiap koordinat memiliki sudut 135 dan -135 dari sudut robot utama. Rata-rata kesalahan pergerakan *follower* A dalam perjalanannya mengikuti koordinat target yang dikirimkan oleh robot pemimpin adalah 4,778% terhadap sumbu X, dan 0,807% terhadap sumbu Y. Rata-rata kesalahan pergerakan *follower* B dalam perjalanannya mengikuti koordinat target yang dikirimkan oleh robot pemimpin adalah 4,893% terhadap sumbu X, dan 1,314% terhadap sumbu Y.
3. Kekurangan dari hasil pembentukan formasi robot adalah masih adanya kelemahan dalam komunikasi robot satu dengan yang lain sehingga masih memungkinkan robot saling bertabrakan dan masih kurang optimalnya pergerakan robot saat menuju target tujuan.

5.2 Saran

Saran yang dapat diberikan untuk mengembangkan sistem ini sebagai berikut :

1. Menggunakan cara lain yang lebih efisien dalam menentukan titik koordinat tujuan dan menentukan koordinasi robot satu dengan yang lain sehingga tidak terjadi tabrakan.

2. Setiap robot dapat berperan sebagai robot utama (pemimpin) dengan penentuan letak posisi target tujuan awal yang terdekat dari salah satu robot.
3. Simulasi dapat disinkronisasikan dengan robot nyata sehingga simulasi dapat berfungsi sebagai alat pengontrol robot.



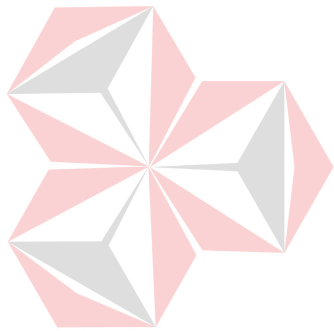
UNIVERSITAS
Dinamika

DAFTAR PUSTAKA

- Ardilla, Fernando, Bayu Sandi M., dan A. R. Anom B. 2011. *Path Tracking Pada Mobile Robot Dengan Umpan Balik Odometry*. Electronic Engineering Polytechnic Institute of Surabaya (EEPIS): Surabaya.
- Borenstein, J. dan Y. Koren. 1989. *Real-time Obstacle Avoidance for Fast Mobile Robots*. IEEE Transactions on Systems, Man, and Cybernetics.
- Borenstein, J. dan Y. Koren. 1991. *The Vector Field Histogram - Fast Obstacle Avoidance For Mobile Robots*. IEEE Journal of Robotics and Automation.
- Gustavi, Tove, Xiaoming H., dan Maja K. 2005. *Multi-Robot Formation Control and Terrain Servoing With Limited Sensor Information*. Institute of Technology: Sweden.
- Hendra. 2011. *Pemrograman GDI+ pada VB.Net*. http://ocw.stikom.edu/course/download/2013/05/07-12-2011.10.38.15_970230_410113007_Pemrograman-Visual-I-SI-KA_P1_Pert12_1.pdf. Diakses tanggal: 10 januari 2014.
- Kariyanto, Jusuf Dwi, Ali H. A., Fernando A., dan Nofria H. 2011. *Navigasi Mobile Robot Berbasis Trajektori dan Odometry dengan Pemulihan Jalur Secara Otomatis*. Politeknik Elektronika Negri Surabaya: Surabaya.
- Kusumadewi, Sri dan Hari Purnomo, 2010. *“Aplikasi Logika Fuzzy Untuk Pendukung Keputusan, edisi 2”*. Yogyakarta, Graha Ilmu.
- Leong, Marlon. 2004. *Pemrograman Dasar Microsoft Visual Basic.NET*. Yogyakarta: ANDI.
- Olson, Edwin. 2004. *A Primer on Odometry and Motor Control*. <http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-186-mobile-autonomous-systems-laboratory-january-iap-2005/study-materials/odomtutorial.pdf>. Diakses tanggal: 12 November 2013.
- Priyanto, Rahmat. 2009. *Langsung Bisa Visual Basic.Net 2008*. Yogyakarta: ANDI.
- R.L. Williams II, B. Carter, P. Gallina, dan G. Rosati, 2002, *Dynamic Model with Slip for Wheeled Omnidirectional Robots*. IEEE Transactions on Robotics and Automation, 18(3): 285-293.
- Supardi, Yuniar. 2011. *Semua Bisa Menjadi Programmer VB6 Hingga VB 2008 BASIC*. Jakarta: Gramedia.

Wardana. 2008. *Membuat Aplikasi Berbasis Pendekatan Sistem dengan Visual Basic .Net 2008*. Jakarta: Gramedia.

Yuswanto. 2002. *Visual Basic Pemograman Grafis dan Multimedia*. Surabaya: Prestasi Pustaka.



UNIVERSITAS
Dinamika