TUGAS AKHIR –TE 141599

IMPLEMENTASI PERENCANAAN JALUR QBOT *MOBILE ROBOT* MENGGUNAKAN METODE *NEURO FUZZY*

Afrizal
NRP 2213106015

Dosen Pembimbing
Ir. Joko Susila MT.
Ir. Rusdhianto Effendie AK MT.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT –TE 141599

IMPLEMENTATION PATH PLANNING OF QBOT MOBILE ROBOT USING NEURO FUZZY METHOD

Afrizal
NRP 2213106015

Advisor
Ir. Joko Susila MT.
Ir. Rusdhianto Effendie AK MT.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

IMPLEMENTASI PERENCANAAN JALUR QBOT MOBILE ROBOT MENGGUNAKAN METODE *NEURO FUZZY*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Pengaturan
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Joko SusilaMT.	**Ir. Rusdhianto Effendie AK MT.**
NIP.196606061991021001	NIP. 195704241985021001

SURABAYA
JANAURI, 2016

IMPLEMENTASI PERENCANAAN JALUR QBOT *MOBILE ROBOT* MENGGUNAKAN METODE *NEURO FUZZY*

Nama : Afrizal
Dosen Pembimbing 1 : Ir. Joko Susila MT.
Dosen Pembimbing 2 : Ir. Rusdhianto Effendie AK MT.

ABSTRAK

Perencanaan jalur merupakan salah satu masalah paling penting dalam pergerakan *mobile robot*. *Mobile robot* membutuhkan sistem navigasi dan kontroler untuk menggerakkan dan mengarahkan *mobile robot* dari posisi awal menuju posisi target. Pada tugas akhir ini *mobile robot* yang digunakan adalah Qbot *mobile robot* keluaran dari Quanser. Untuk kontroler yang digunakan yaitu sistem *neuro fuzzy* agar Qbot *mobile robot* dapat mengikuti referensi jalur yang harus dilaluinya. Hasil pengujian kontroler *neuro fuzzy* pada Qbot *mobile robot* untuk *training* pola 1 waktu yang dibutuhkan untuk mencapai target adalah 19,5 detik sedangkan waktu yang dibutuhkan kontroler *neuro fuzzy* pada percobaan pertama adalah 19,2 detik, percobaan kedua 18,8 detik dan percobaan ketiga 19,0 detik. Untuk pola 2 saat *training* membutuhkan waktu 21,5 detik untuk mencapai target sedangkan kontroler *neuro fuzzy* pada percobaan pertama adalah 18,7 detik, percobaan kedua 18,9 detik dan percobaan ketiga 18,7 detik.

Kata Kunci : Perencanaan jalur**,** Neuro fuzzy, navigasi, kontroler, Qbot mobile robot, dan training

Halaman ini sengaja dikosongkan

IMPLEMENTATION PATH PLANNING OF QBOT MOBILE ROBOT USING NEURO FUZZY METHOD

Nama ***: Afrizal***
Advisor 1 ***: Ir. Joko Susila MT.***
Advisor 2 ***: Ir. Rusdhianto Effendie AK MT.***

ABSTRACT

Path planning is one of the most important problem in the movement of mobile robot. Mobile robots need a navigation system and controller to drive and steer the mobile robot from the initial position to the target position. In this final project mobile robot used Qbot mobile robot produced by Quanser. For controllers used neuro fuzzy system that Qbot mobile robot can follow a reference path that must be gone through. The test result neuro fuzzy controller on Qbot mobile robot for training pattern 1, time required to reach the target si 19,5 seconds, while the time required neuro fuzzy controllers in the first experiment l was 19,2 seconds, a second experiment was 18,8 seconds, and a third experiment was 19,0 seconds. For patterns 2 when training takes 21,5 seconds to reach the target while neuro fuzzy controller on the first experiment was 18,7 seconds, a second experiment was 18,9 seconds, and third experiment was 18,7 seconds.

Keyword : *Path planning, Neuro fuzzy, navigation, controller, Qbot mobile robot,and training*

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji dan syukur atas kehadirat Allah *Subhanahu wa ta'ala*, berkat rahmat dan hidayah-Nya, penulis dapat menyelesaikan tugas akhir dengan baik dan tepat waktu. Tugas Akhir ini disusun untuk memenuhi sebagian persyaratan guna menyelesaikan pendidikan Strata-I pada Bidang Studi Teknik Sistem Pengaturan, Jurusan Teknik Elektro, Institut Teknologi Sepuluh Nopember Surabaya.

Terima kasih penulis ucapkan kepada orang tua dan kakak-kakak penulis yang selalu memberikan doa dan dukungan. Pak Joko Susila dan Pak Rusdhianto selaku dosen pembimbing yang telah membimbing dari awal sampai terselesaikannya tugas akhir ini, teman-teman Teknik Sistem Pengaturan Lintas Jalur periode Genap 2013 serta pihak-pihak yang mendukung dan memotivasi penulis sehingga dapat menyelesaikan tugas akhir ini.

Saran dan kritik penulis harapkan guna memperbaiki tugas akhir ini. Akhir kata, semoga Tugas Akhir ini dapat bermanfaat bagi pengembangan ilmu kedepannya.

Surabaya, Januari 2016

Penulis

Halaman ini sengaja dikosongkan

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

Pada bab satu akan dibahas mengenai latar belakang, permasalahan, tujuan, metodologi, sistematika, dan relevansi tugas akhir yang dikerjakan.

1.1 Latar belakang

Mobile robot adalah sebuah mesin otomatis yang mampu bergerak pada suatu kondisi tertentu. *Mobile robot* diklasifikasikan menjadi dua, yaitu menurut lingkungan tempat robot tersebut bekerja dan alat yang digunakan untuk bergerak. Berdasarkan lingkungan tempat robot tersebut bekerja, *mobile robot* terbagi menjadi empat macam: robot yang bekerja di atas permukaan tanah (*land robot*), robot udara yang biasa disebut *unmanned aerial vehicle* (UAV), *autonomous underwater vehicles* (AUVs), dan robot yang bekerja pada lingkungan kutub, robot yang berkerja pada kondisi permukaan tanah yang dilapisi es (*polar robots*). Berdasarkan alat yang digunakan untuk bergerak, robot mobil terbagi menjadi robot berlengan atau berkaki-lengan atau kaki menyerupai manusia (*humanoid*) ataupun hewan dan robot beroda, *Wheeled Mobile robot* (WMR).[1]

Quanser Qbot merupakan robot *autonomous* yang terdiri dari *platform* iRobot yang banyak digunakan dalam aplikasi robot, dengan lima sensor inframerah, satu magnetometer,tiga sensor sonar dan kamera yang terpasang pada robot. *Quanser Controller Module* (QCM) pada Qbot memanfaatkan komputer Gumstix untuk menjalankan QuaRC, perangkat lunak Quanser's *real time* dan kartu data akuisisi Qbot.[2]

Perencanaan jalur merupakan salah satu masalah paling penting dalam pergerakan *mobile robot*. Banyak cara untuk memecahkan permasalahan ini seperti menggunakan teknik hybrid seperti *genetic algorithm-fuzzy logic, neural network-fuzzy logic, neural network-genetic algorithm*. Perpaduan tiap teknik tersebut bertujuan agar membuat proses penemuan jalur menuju target lebih efisien.[3] Dalam tugas akhir ini metode yang akan digunakan yaitu metode *neuro fuzzy*, dimana neuro fuzzy berfungsi sebagai kontroler untuk navigasi Qbot *mobile robot* bergerak dari titik awal menuju titik tujuan dan dapat menghindari *obstacles* yang berada pada jalur.

1.2 Permasalahan

Dalam pergerakan *mobile robot* diperlukan sebuah navigasi untuk mengendalikan pergerakan robot dari titik awal menuju tujuan akhir. Permasalahan yang dibahas pada tugas akhir ini adalah bagaimana *mobile robot* Qbot mengikuti jalur dari titik awal menuju titik akhir dimana pada jalur yang akan ditempuh tersebut terdapat *obstacles* dan robot harus dapat menghidari *obstacles* tersebut. Olehkarena itu dibuat sebuah kontroler neuro fuzzy untuk membantu navigasi robot menghindari *obstacles* dan mencapai tujuan akhirnya.

1.3 Tujuan

Tujuan tugas akhir ini adalah mendesain kontroler *neuro-fuzzy* yang dapat diterapkan untuk implementasi navigasi *mobile robot* Qbot untuk bergerak dari titik awal menuju tujuan akhir dan dapat menghindari *obstacles* yang terdapat pada jalur yang akan ditempuh oleh robot.

1.4 Metodologi

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

1. **Tinjauan Pustaka**
   Dilakukan dengan mengumpulkan dan mempelajari literatur yang mendukung tugas akhir melalui *internet* dan media cetak berupa buku atau jurnal.
2. **Pengambilan Data dan Identifikasi**
   Dari hasil yang diperoleh dari studi literatur, identifikasi disesuaikan dengan keadaan *plant* sesungguhnya. Pada tahap ini akan dilakukan pengambilan data dari Qbot *Mobile robot*. Setelah memperoleh data yang dibutuhkan, akan dilakukan identifikasi dari *plant* untuk memperoleh parameter yang akan digunakan untuk mendesain kontroler sesuai yang diinginkan.
3. **Perancangan Kontroler**
   Pada tahap ini dibuat struktur kontroler neuro fuzzy untuk navigasi Qbot *mobile robot* dan kontroler disimulasikan secara *real time*.
4. **Implementasi dan Uji Coba**
   Kontroler yang sudah dirancang lalu diimplementasikan pada *plant* berupa Qbot *Mobile robot* pada ruang yang sudah direncanakan jalurnya dan terdapat *obstacles*

5. Penulisan Buku Tugas Akhir

Penyusunan buku Tugas Akhir meliputi pendahuluan, teori dasar, perancangan sistem, implementasi, analisa serta penutup.

1.5 Sistematika

Pembahasan Tugas Akhir ini dibagi menjadi lima bab dengan sistematika sebagai berikut :

Bab 1 : Pendahuluan
Bab ini meliputi latar belakang, permasalahan, tujuan penelitian, dan sistematika penulisan.

Bab 2 : Teori Penunjang
Bab ini membahas tinjauan pustaka yang membantu penelitian, diantaranya teori Qbot *mobile robot*, teori identifikasi sistem kontroler *neuro fuzzy*

Bab 3 : Perancangaan Sistem
Pada bab ini dijelaskan mengenai perancangan identifikasi sistem Qbot *mobile robot* serta algoritma kontroler *neuro fuzzy*

Bab 4 : Hasil dan Analisa
Bab ini memuat hasil implementasi kontroler dan pengujiannya pada sistem

Bab 5 : Penutup
Bab ini berisi kesimpulan dan saran dari hasil penelitian yang telah dilakukan

1.6 Relevansi

Hasil yang diperoleh dari tugas akhir ini diharapkan menjadi referensi pengembangan teknologi dan perbandingan metode kontrol yang tepat untuk navigasi Qbot *mobile robot* di masa mendatang.

Halaman ini sengaja dikosongkan

BAB 2
TEORI PENUNJANG

Pada bab dua akan dibahas mengenai tinjauan pustaka dan teori-teori yang berhubungan dengan permasalahan yang dibahas pada bab sebelumnya.

2.1 Tinjauan Pustaka

Mobile robot merupakan area penelitian yang berhubungan dengan kontrol *autonomous* dan *semi-autonomous.* Teknik untuk mengembangkan navigasi *autonomous* dari *mobile robot* sekarang adalah tujuan utama dalam riset robotika. Salah satu jenis *mobile robot* yang sering dijumpai yaitu *defferentially driven mobile robot.* Robot ini menggunakan dua buah roda penggerak yang bebas, sehingga gerakan translasi maupun rotasi robot dihasilkan dari kombinasi pergerakkan dua roda. Agar robot dapat bergerak stabil maka ditambah sebuah roda bebas yang biasa disebut roda *castor.*

Pada penelitian Qbot *mobile robot* sebelumnya yang dilakukan oleh Baradiant Ivano Leotman dalam tugas akhirnya telah dibahas mengenai desain kontroler fuzzy PD untuk pelacakan objek oleh Qbot *mobile robot.*[1] Dalam penelitian Tugas Akhir kali ini, akan dibuat perencanaan jalur robot dari titik awal menuju tujuan akhir yang mampu menghindari obstacles dimana untuk perencanaan jalur akan menggunakan metode *neuro fuzzy* sebagai kontroler untuk navigasi robot.

Perencanaan jalur pada Qbot *mobile robot* dilakukan dengan cara *training* menggunakan algoritma sederhana agar robot dapat menentukan titik awal robot hingga dapat bergerak menuju titik akhirnya. Pada perencanaan jalur yang akan dilakukan oleh Qbot *mobile robot* ditentukan terlebih dahulu jarak yang akan ditempuh robot pada bidang (x,y), saat Qbot mobile robot bergerak obstacles yang berada pada jalur akan terbaca oleh sensor jarak dan robot akan menghindarinya kemudian melanjutkan pergerakan menuju titik posisi tujuan.

Untuk navigasi pergerakan Qbot *mobile robot* ini digunakan metode kontrol *neuro fuzzy* dimana kontroler ini untuk mengendalikan kecepatan pergerakan robot untuk mencapai titik tujuan akhirnya dan dapat menghindari obstacles yang berada pada jalur yang akan dilalui Qbot *mobile robot*.

2.2 Qbot *Mobile robot* [2]

Qbot *Mobile robot* merupakan sistem inovatif robot darat *autonomous* yang menggabungkan ilmu kendaraan darat dengan *Quanser Controller Module* (QCM). *Mobile robot* ini terdiri dari iRobot Create® *Robotic Platform*, sensor inframerah dan sonar serta sebuah kamera Logitech Quickcam Pro 9000 USB seperti terlihat pada Gambar 2.1. QCM yang terpasang pada Qbot yang mana menggunakan komputer Gumstix untuk menjalankan QuaRC, perangkat lunak kontrol Quanser, dan Qbot kartu data akuisisi (DAC).

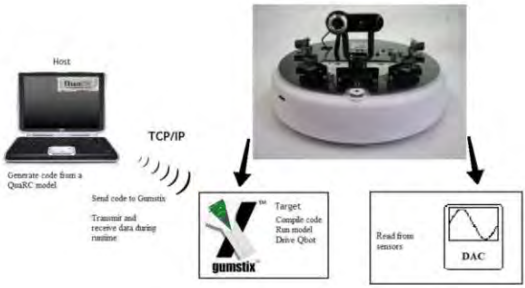
Gambar 2.1 Qbot *mobile robot*

Berikut ini spesifikasi dan model parameter Qbot *mobile robot* dapat dilihat pada table 2.1 berikut.

Tabel 2.1 Spesifikasi sistem dan parameter model Qbot *mobile robot*

Simbol	Deskripsi	Value	Unit
d	Diameter Qbot *mobile robot*	0.34	*m*
h	Tinggi Qbot *mobile robot* (dengan tambahan kamera)	0.19	*m*
vmax	Kecepatan maksimum dari Qbot *mobile robot*	0.5	*m/s*
m	Berat total Qbot *mobile robot*	2.92	*kg*

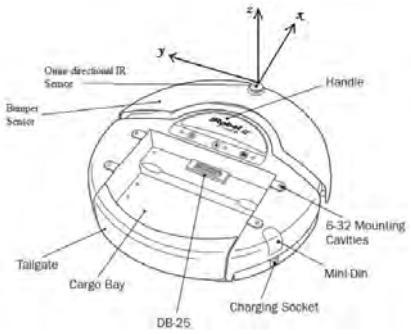
Antarmuka untuk QCM adalah MATLAB Simulink dengan QuaRC. Qbot dapat menerima perintah, melalui tiga blok yang berbeda: blok Roomba untuk menggerakkan Qbot, blok HIL untuk membaca dari sensor dan/atau menulis untuk output servo dan blok OpenCV untuk mengakses kamera. Kontroler dikembangkan pada Simulink dengan QuaRC dan model ini diunduh dan disusun pada target (Gumstix). Diagram dari konfigurasi ini ditampilkan dalam Gambar 2.2.



Gambar 2.2 Hirarki komunikasi Qbot *mobile robot*

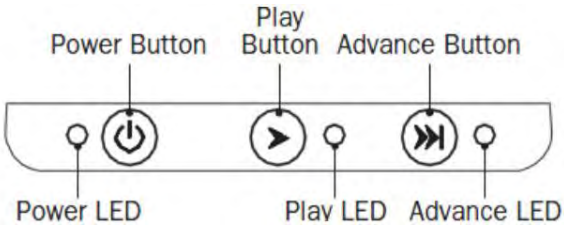
2.2.1 Sistem Perangkat Keras Qbot

Qbot menggunakan rangka iRobot Create® seperti pada Gambar 2.3. Qbot mengikuti standar Quanser untuk sumbu rangka tubuh, di mana sumbu x pada arah maju, sumbu y sebelah kiri dan sumbu z ada di atas. Diameter dari rangka ini 34 cm dan tinggi (tanpa tambahan kamera) adalah 7 cm. dan Qbot digerakkan oleh dua roda kemudi yang berbeda. iRobot Create® terdapat bumper sensor dan inframerah. QCN dapat mengakses data dari sensor ini.



Gambar 2.3 Rangka Qbot *mobile robot*

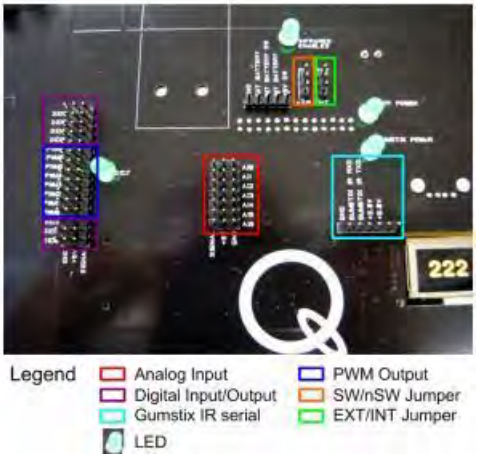
Qbot dihidupkan dengan menekan tombol *Power*. Gambar 2.4 menunjukkan tombol yang digunakan untuk mengoperasikan Qbot. Tombol *Play* dan *Advance* untuk menjalankan dalam demo untuk Qbot dan tidak diperlukan untuk tujuan Qbot.



Gambar 2.4 Tombol pada rangka Qbot

2.2.2 *Printed Circuit Board* (PCB)

Kabel dan sirkuit untuk Qbot dalam PCB yang terletak dipenutup hitam Qbot atas komputer Gumstix dan DAC. Sensor dan kamera yang juga terpasang pada PCB. Gambar 2.5 menunjukkan pin diakses bagi pengguna. Secara khusus, DIO, output PWM, dan pin input analog telah diberi label untuk lebih jelas.



Gambar 2.5 PCB Qbot

2.2.3 *Pin Digital Input/Output* (DIO)

Saluran DIO (0 samapai 6) ditetapkan sebgai input secara bawaan. Saluran DIO perlu dikonfigurasi baik sebagai input (atau ouput, tapi tidak keduanya) menggunakan blok HIL *Initialize*. Jika output harus dalam keadaan yang dikenal *power up,* dianjurkan bahwa resistor 10K diletakkan dari I/O untuk 5V atau GND sesuai kebutuhan. Ada saluran

digital akhir (7) yang merupakan output tetap, dan itu diwakili oleh LED berlabel DIO7.

2.2.4 *Pin Serial Gumstix IR*

Qbot menyediakan serial koneksi TTL ke port serial Gumstix IR (port no 2). Port serial terdiri dari *ground* (GND), menerima Gumstix IR RXD, mengirimkan Gumstix IR TXD dan pin daya (+3.3V atau +5.0V). Port serial diakses melalui blok QuaRC *Stream* atau *Stream* API.

2.2.5 SW/nSW dan INT/EXT *Jumpers*

INT/EXT *jumper switch* Qbot dari internal daya dari baterai iRobot Create (INT) dan eksternal baterai *power supply* (EXT). Tidak ada baterai eksternal yang disertakan pada Qbot, jadi jumper ini harus dibiarkan dalam posisi INT untuk daya Qbot tersebut. Saat jumper *power supply* dalam posisi INT, jumper SW/ NSW menunjukkan apakah iRobot Create harus diaktifkan (SW) untuk Qbot menerima daya atau apakah Qbot harus selalu membutuhkan daya bahkan ketika iRobot Create dalam keadaan mati (nSW).

2.2.6 Qbot *Data Acquisition Card* (DAC)

Qbot DAC adalah kartu data akuisisi, yang mampu menerima input analog dan input lainnya (untuk sensor sonar). Qbot DAC dapat juga membaca output PWM untuk servo aktuatuor. Qbot DAC terletak di bawah penutup hitam Qbot.

2.2.7 Kamera USB

Kamera USB Logitech Quickcam Pro 9000 terpasang pada bagian atas Qbot seperti pada Gambar 2.6. Blok QuaRC menggunakan *Open Source Computer Vision* yang menyediakan pengguna untuk mengambil dan menampilkan gambar secara *real time*, memproses dan menyimpannya untuk analisis.

Gambar 2.6 Kamera USB Logitech Quickcam 9000

2.2.8 Gumstix

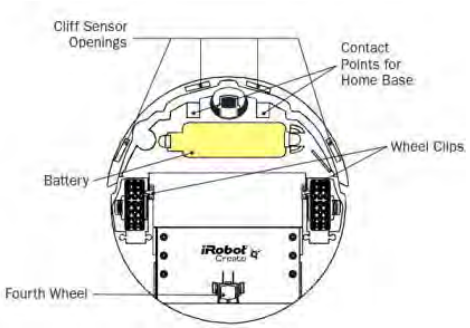
Gumstix skala kecil berfungsi penuh pada komputer *open source* dimana Simulink MATLAB secara langsung dapat diunduh, dikompilasi dan dijalankan melalui perangkat lunak QuaRC. *Motherboard* Gumstix terhubung secara langsung pada Qbot DAC. Pada Gumstix juga terdapat tambahan Wifi untuk menyediakan koneksi nirkabel antara target Gumstix dan komputer *host*.

2.2.9 Baterai Qbot

Qbot ini didukung oleh baterai *Advance Power System* (APS) yang disediakan oleh iRobot seperti pada Gambar 2.7. Baterai ini terletak di bawah Qbot dan dapat berlangsung terus menerus selam sekitar dua jam setelah terisi penuh.

Gambar 2.7 Baterai Qbot

Baterai terletah dibagian bawah seperti pada Gambar 2.8.Lampu daya Qbot mengindikasikan tingkat daya baterai. Lampu hijau menandakan baterai masih terisi penuh, kemudian secara bertahap berubah menjadi merah bila baterai akan habis. Baterai memerlukan waktu kurang dari 3 jam untuk pengisian baterai. Ketika pengisian, lampu daya akan berkedip perlahan dengan warna jingga dan akan berubah menjadi hijau ketika terisi penuh.



Gambar 2.8 Letak baterai pada Qbot

2.2.10 Sensor Inframerah dan Sonar

SHARP 2Y0A02 seperti pada Gambar 2.9 merupakan sensor inframerah dengan jarak 20-150cm. Ada lima sensor inframerah yang terpasang pada Qbot. Sensor terhubung kesaluran input analog dari Qbot DAC, yang kemudian dapat dibaca dengan menggunakn blok HIL *Read Write.*

Gambar 2.9 Sensor inframerah SHARP 2Y0A02

MaxSonar-EZ0 seperti pada Gambar 2.10 mendeteksi objek dari 0 inci sampai 254 inci dengan resolusi 1 inci. Objek antara 0 inci dan 6 inci berkisar sebagai 6 inci. Terdapat tiga sensor pada Qbot. Sensor yang terhubung ke saluran masukan lain dari Qbot DAC, yang kemudian dapat dibaca dengan menggunakan blok HIL *Read Write.*

Gambar 2.10 Sensor sonar MaxSonar-EZ0

2.3 QUARC

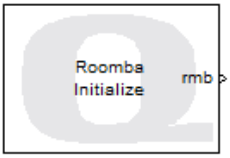
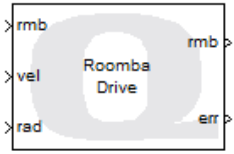
QUARC (Quanser Real-Time Control) adalah perangkat lunak yang menyediakan fasilitas untuk mengembangkan dan menguji coba rancangan kontroler pada komputer *host* dengan menggunakan perangkat lunak Simulink MATLAB, dengan model yang dirancang dapat langsung diunduh dan dieksekusi pada Gumstix dengan menggunakan komunikasi *wireless* secara *real time*. Pada waktu yang bersamaan operator dapat memantau dan mengukur nilai-nilai dari sensor dan mengatur parameter-parameter kontroler langsung dari komputer *host*. Pada MATLAB Simulink terdapat beberapa blok yang digunakan untuk melakukan komunikasi dengan Qbot seperti yang terdapat pada Tabel 2.2 dan Tabel 2.3.

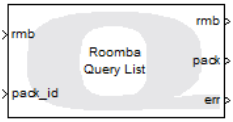
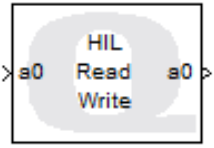
Tabel 2.2 Deskripsi Blok set Qbot yang relevan untuk Qbot di QUARC

Blok Set	Deskripsi
Interface	Blok set ini mengimplementasikan dasar Aplication Program Interfaces (API) yang disediakan oleh iRobot Create®. API dapat dikategorikan berdasarkan fungsional berikut: • Sambungkan konfigurasi serial antara kontroler tingkat tinggi(misalnya, PC atau Gumstix) dan Qbot. • Pengaturan mode operasi Qbot • Mengakses informasi sensorik Qbot • Konfigurasi hardware lainnya (misalnya,pengaturan port I/O digital dan analog, dan mengubah warna LED)
Aplikasi	Blok set ini memungkinkan pengguna untuk

	menerapkan algoritma navigasi menggunakan informasi sensorik yang tersedia. Blok ini hanya menggunakan data encoder roda dan *bump sensors* untuk navigasi dan menghindari rintangan.
Image Processing	Blok set ini mengimplementasikan akuisisi citra dan pengolahan fungsi menggunakan *Library Open CV*. Blok set pengolahan citra memungkinkan untuk menangkap gambar dari kamera USB, untuk proses online dan menyimpannya ke *disk* untuk analisis lebih lanjut.

Tabel 2.3 Penjelasan Deskripsi Blok QUARC pada Qbot

Blok	Deskripsi
	Blok ini diperlukan untuk membuat sambungan serial ke Qbot. Qbot diidentifikasi oleh *Universal Resource Identifier* (URI), seperti *serial://localhost:1baud=57600,word=8, parity=none,stop=1*, di mana port komunikasi 1 dari target terhubung ke port serial Qbot dengan parameter yang ditentukan. Catatan: Output "rmb" dari blok ini harus terhubung ke input "rmb" dari blok di blok Roomba yang ditetapkan dalam rangka untuk mengakses Roomba.
 Roomba Drive	Blok ini menggerakkan Qbot dengan dua input: kecepatan dan radius. Untuk menggerakkan lurus, input radius mengambil 32768 atau 32767.

 Roomba Query List	Blok ini mengambil berbagai data sensor dari Qbot. Input pack_id berkisar 7-42, dan setiap input akan menampilkan nilai dari output *pack*. Lihat halaman Help MATLAB untuk deskripsi lengkap untuk masing-masin paket ID. Nilai pack_id yang penting: 8 = Dinding (0 = tidak ada dinding, 1 = dinding terlihat) 19 = Jarak (Jarak yang Qbot telah melakukan perjalanan dalam milimeter) 20 = Sudut (Sudut dalam derajat Qbot berubah) 22 = Tegangan (Tegangan dari baterai Qbot dalam milivolt) 23 = Arus (Arus dalam miliampere mengalir atau keluar dari baterai Qbot)
 HIL Initialize HIL-1 (q8-0)	Blok inisialisasi yang diperlukan dalam semua model QuaRC menggunakan HIL Card. Hanya satu blok HIL *Initialize* yang dibutuhkan dalam model untuk mengatur satu kendaraan. Blok HIL *Read Write* harus referensi blok ini untuk menentukan *board* yang digunakan.
 HIL Read Write (HIL-1)	Versi saat ini dari Qbot DAC yang dapat membaca dari saluran analog (untuk sensor infra merah) dan saluran sensor sonar. Saluran PWM yang didukung untuk operasi *write.*

2.4 Logika *Fuzzy*

Manusia terbiasa mengolah sesuatu hal, baik data maupun fakta secara *fuzzy*. Bahkan saat pengambilan keputusan, didasarkan pada hal-hal yang bersifat *fuzzy*. *Fuzzy* berarti kabur atau samar atau tidak jelas.[4]

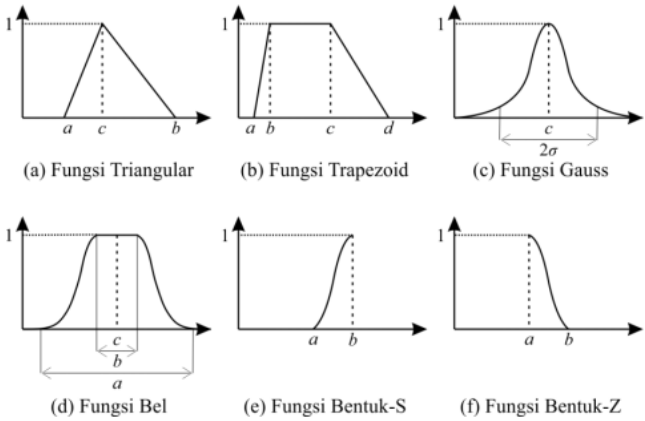
Konsep logika *fuzzy* berawal dari teori *fuzzy set* sebagai salah satu pendekatan untuk menyelesaikan permasalahan yang tidak memiliki ketentuan yang pasti. Teori tersebut dikembangkan oleh Lotfi Zadeh pada tahun 1965 di University of California – Barkeley. Dalam jurnalnya, Zadeh memperkenalkan konsep teori himpunan baru yang dinamakan *fuzzy* set. Konsep logika *fuzzy* menggantikan konsep "benar-salah" dari logika *boolean* menjadi derajat tingkat kebenaran. Teori *fuzzy* menyatakan keanggotaan suatu objek ke dalam fungsi derajat keanggotaan (*membership function*). Hal tersebut memungkinkan keanggotaan suatu objek dapat dinyatakan pada semua bilangan riil antara 0 sampai 1. Oleh karena itu, konsep *fuzzy* tersebut sesuai dengan pola pikir manusia yang cenderung menilai suatu objek secara samar.

2.4.1 Himpunan *Fuzzy* (*Fuzzy Set*)

Himpunan *Fuzzy* merupakan suatu himpunan yang beranggotakan sejumlah istilah dalam pengertian bahasa yang menyatakan level kualitatif dari semesta pembicaraan. Misalnya pengukuran kecepatan putaran motor dapat diterjemahkan ke dalam beberapa istilah bahasa yang menyatakan level kualitatif dari kecepatan putaran motor. Sehingga apabila semesta pembicaraan berupa kecepatan putaran motor, maka dapat dibuat suatu himpunan *fuzzy* yaitu "Sangat Lambat", "Lambat, "Sedang", "Cepat", "Sangat Cepat".

2.4.2 Fungsi Keanggotaan (*Membership function*)

Fungsi keanggotaan (*membership function*) merupakan kurva yang menunjukkan pemetaan *input* data ke dalam nilai keanggotaannya dengan rentang nol sampai satu. Berikut ini dapat dilihat pada Gambar 2.11 bentuk-bentuk dari fungsi keanggotaan.



Gambar 2.11 Bentuk-bentuk *membership function*

Berikut beberapa fungsi keanggotaan yang umum digundapat dilihat pada kan, yaitu:

a. Fungsi segitiga
Fungsi segitiga merupakan gabungan antara dua fungsi linear. Fungsi ini ditentukan oleh tiga parameter $\{a, b, c\}$ dengan ketentuan $a < c < b$. Pada Gambar 2.11 ditampilkan bentuk fungsi segitiga. Fungsi keanggotaan fungsi segitiga, yaitu:

$$\mu(\mathrm{x}) = \begin{cases} 0; & x \leq a \\ \frac{x-a}{c-a}; & a \leq x \leq c \\ \frac{b-x}{b-c}; & c \leq x \leq b \\ 0; & x \geq b \end{cases} \qquad 2.1$$

b. Fungsi trapesium
Fungsi trapesium pada dasarnya seperti fungsi segitiga yang memiliki beberapa titik dengan nilai keanggotaan satu. Fungsi ini ditentukan oleh empat parameter $\{a, b, c, d\}$ dengan ketentuan $a < c \leq d < b$. Pada Gambar 2.11 ditampilkan bentuk fungsi trapesium. Fungsi keanggotaan fungsi trapesium, yaitu:

$$\mu(x) = \begin{cases} 0; & x \le a \\ \frac{x-a}{c-a}; & a \le x \le c \\ 1; & c \le x \le d \\ \frac{b-x}{b-d}; & d \le x \le b \\ 0; & x \ge b \end{cases} \quad 2.2$$

c. Fungsi S

Fungsi S didefinisikan dengan menggunakan tiga parameter, yaitu nilai keanggotaan nol (α), nilai keanggotaan penuh (γ), dan titik infleksi (β). Titik infleksi yaitu titik yang memiliki domain 50% benar. Pada Gambar 2.11 ditampilkan bentuk fungsi S. Fungsi keanggotaan fungsi S, yaitu:

$$\mu(x) = \begin{cases} 0; & x \le a \\ 2\left(\frac{x-\alpha}{\gamma-\alpha}\right)^2; & \alpha \le x \le \beta \\ 1-2\left(\frac{x-\alpha}{\gamma-\alpha}\right)^2; & \beta \le x \le \gamma \\ 1; & x \ge \gamma \end{cases} \quad 2.3$$

d. Fungsi phi (π)

Fungsi phi berbentuk lonceng, menggunakan dua parameter yaitu (γ) untuk menunjukkan derajat keanggotaan bernilai satu yang terletak di pusat himpunan, dan lebar fungsi (β). Pada Gambar 2.11 ditampilkan bentuk fungsi phi. Fungsi keanggotaan fungsi phi, yaitu:

$$\mu(x) = \begin{cases} S\left(x; \gamma-\beta, \gamma-\frac{\beta}{2}, \gamma\right); & x \le \gamma \\ 1-S\left(x; \gamma+\beta, \gamma+\frac{\beta}{2}, \gamma\right); & x > \gamma \end{cases} \quad 2.4$$

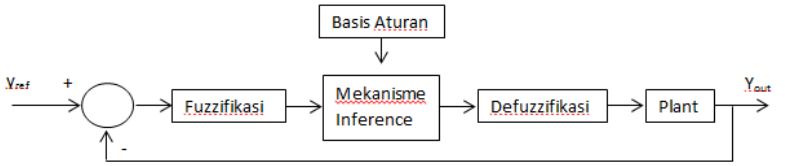
e. Fungsi gaussian

Fungsi Gaussian menggunakan dua parameter, yaitu (c) untuk menunjukkan nilai domain pada pusat kurva, dan (σ) menunjukkan lebar fungsi. Pada Gambar 2.11 ditampilkan bentuk fungsi gaussian. Fungsi keanggotaan fungsi gaussian, yaitu:

$$\mu(x) = e^{-\left(\frac{x-c}{\sigma}\right)^2} \qquad 2.5$$

2.4.3 Kontroler Logika *Fuzzy*

Kontroler logika *fuzzy* merupakan suatu kontroler yang proses perhitungan sinyal kontrolnya melalui operasi himpunan *fuzzy* meliputi proses *fuzzifikasi*, relasi *fuzzy*, inferensi *fuzzy* serta *defuzzifikasi* seperti terlihat pada Gambar 2.12.

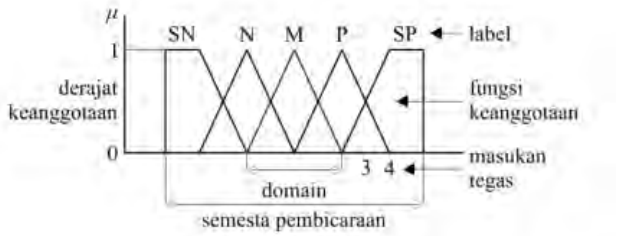


Gambar 2.12 Struktur kontroler *fuzzy*

Proses di dalam himpunan *fuzzy* pada kontroler *fuzzy* antara lain sebagai berikut :

1. *Fuzzifikasi*

Fuzzifikasi merupakan proses transformasi nilai real (nilai bukan *fuzzy*) menjadi nilai suatu himpunan *fuzzy* yang dinyatakan dalam derajat keanggotaan yang didefinisikan. Struktur fungsi keanggotaan *fuzzy* dapat dilihat pada Gambar 2.13. Derajat keanggotaan adalah derajat dari masukkan tegas pada sebuah fungsi keanggotaan, memiliki nilai 0 s/d 1. Fungsi keanggotaan mendefinisikan nilai *fuzzy* dengan melakukan pemetaan nilai tegas berdasarkan daerahnya untuk diasosiasikan dengan derajat keanggotaan.



Gambar 2.13 Struktur fungsi keanggotaan *fuzzy*

Masukkan tegas pada umumnya merupakan hasil pengukuran parameter eksternal dari sistem kontrol. Label merupakan deskripsi nama untuk menunjukkan suatu fungsi keanggotaan *fuzzy* yang memiliki domain (lebar fungsi keanggotaan *fuzzy*) tertentu. Semesta pembicaraan memiliki jarak yang mencakup seluruh masukkan tegas yang mungkin ada. Bentuk fungsi keanggotaan harus mewaliki variabel masukkan tegas, namun bentuk yang digunakan dibatasi oleh kemampuan *tool* dalam melakukan perhitungan. Bentuk fungsi yang rumit membutuhkan persamaan fungsi yang lebih kompleks.

2. Basis Aturan

Basis aturan merupakan deskripsi linguistik terhadap variabel input dan output. Penentuan basis aturan yang dipakai dalam mengontrol suatu *plant* dapat melalui metode heuristik maupun deterministik. Metode heuristik didasarkan pada pengetahuan terhadap *plant* dan perilaku dari *plant* yang akan dikontrol. Sedangkan metode deterministik diperoleh melalui identifikasi struktur dan parameter dari aturan kontrol. Pemetaan input dan output pada sistem *fuzzy* direpresentasikan dalam pernyataan berikut

IF *premis* **THEN** *konsekuen*.

Pada kontroler *fuzzy*, input kontroler direpresentasikan dalam premis dan output kontroler direpresentasikan dalam konsekuen. Jumlah basis aturan dari suatu sistem *fuzzy* ditentukan dari jumlah variabel pada input dan jumlah *membership function* pada variabel masukkan, dirumuskan dalam Persamaan 2.6.

$$\prod_{i=1}^{n} Ni = N1 \times N2 \times \ldots \times Nn \qquad 2.6$$

Dimana Ni merupakan jumlah *membership function* pada variabel input i. Sebagai contoh apabila variabel input pertama memiliki tiga *membership function* dan variabel input kedua memiliki tiga *membership function*, maka jumlah basis aturan adalah 3x3 = 9 aturan.

3. Mekanisme Inferensi

Mekanisme inferensi pada kontroler *fuzzy* merupakan mekanisme operasi matematika yang dilakukan sesuai dengan premis yang diberikan. Pada basis aturan terdapat dua operasi matematika yang dapat dilakukan yaitu *AND* dan *OR*.

Terdapat beberapa tipe mekanisme inferensi *fuzzy* antara lain Mamdani, Larsent dan Takagi sugeno. Perbedaan dari metode ini terletak pada pengambilan kesimpulan logika *fuzzy*. Pada metode Mamdani maupun Larsent, kesimpulan logika *fuzzy* berupa derajat keanggotaan sehingga dalam menyimpulkan suatu logika *fuzzy* dibutuhkan proses *defuzzifikasi*. Sedangkan pada tipe Takagi Sugeno, kesimpulan logika *fuzzy* berupa suatu persamaan sehingga tidak diperlukan proses *defuzzifikasi*. Kelebihan pada logika *fuzzy* tipe Mamdani dan Larsent lebih sederhana, akan tetapi diperlukan kemampuan untuk mengetahui karakteristik *plant* untuk menentukan batasan keluaran kontroler. Pada tipe Takagi-Sugeno tidak diperlukan
pengetahuan mengenai karakteristik dari *plant* akan tetapi diperlukan perhitungan yang lebih rumit untuk persamaan pada bagian konsekuen.

4. Defuzzifikasi

Defuzzifikasi adalah suatu proses mentransformasikan harga fuzzifikasi yang telah di inferensi kedalam harga bukan fuzzy atau harga actual. Beberapa metode dalam proses defuzzifikasi yaitu Center of Area. Mean of Maxima, weighted sum dan lainnya.

- Center of Area

Metode *center of area* digunakan untuk menentukan nilai titik tengah area yang merupakan titik pusat massa dari kombinasi fungsi-fungsi keanggotaan. Secara umum, persamaan untuk metode *Center of Area* ditunjukkan dengan persamaan 2.7

$$U_0 = \frac{\sum_{k=1}^{m} u_k(T) \cdot \mu_u(u_k(T))}{\sum_{k=1}^{m} \mu_k(u_k(T))}, \forall\, u \in U(T) \qquad 2.7$$

- Mean of Maxima

Metode *mean of maxima* mengambil semua nilai tiap fungsi keanggotaan dengan derajat keanggotaan maksimum dan menghitung rata-rata dari nilai-nilai tersebut sebagai keluaran tegas. Persamaan 2.8 menunjukkan persamaan umum metode tersebut.

$$U_0 = \frac{\sum_n \max(\mu_A^n).yn}{\sum_n \max(\mu_A^n)} \qquad 2.8$$

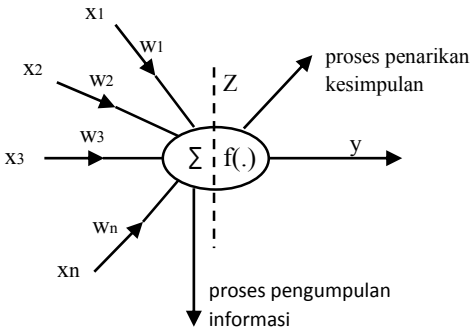
- Weighted Sum

Metode *weigthed sum* merupakan metode *defuzzifikasi* dengan fungsi keanggotaan berupa fungsi *singleton.* Persamaan 2.9 menunjukkan persamaan *defuzzifikasi* fungsi *singleton*.

$$U_0 = \frac{\sum_i \mu_A^i s_i}{\sum_i \mu_A^i} \qquad 2.9$$

2.5 Jaringan Saraf Tiruan

Jaringan saraf tiruan adalah suatu metode komputasi yang merupakan bentuk penyederhanaan dari jaringan sistem saraf biologis yang dinyatakan ke dalam formulasi matematik jaringan simbolik. Gambar 2.14 menunjukkan struktur model jaringan saraf tiruan. Jaringan saraf tiruan dapat digunakan untuk belajar dan menghasilkan aturan atau operasi dari beberapa contoh atau masukkan dan membuat prediksi tentang kemungkinan keluaran yang akan muncul atau menyimpan karakteristik masukkan yang diberikan kepada jaringan saraf tiruan.[5]



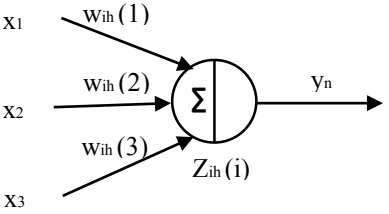
Gambar 2.14 Struktur model jaringan saraf tiruan

Pada jaringan syarat tiruan terdapat dua formulasi yaitu formulasi *forward* dan formulasi *backward.*

a. Formulasi *Forward*

Formulasi *forward* digunakan untuk menghitung keluaran dari setiap neuron pada jaringan saraf tiruan.

Gambar 2.15 menunjukkan struktur formulasi *forward* pada jaringan saraf tiruan yang terdiri dari masukkan, bobot, fungsi aktivasi dan keluaran.



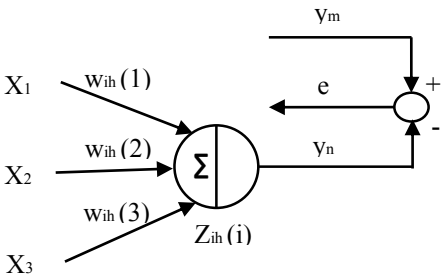
Gambar 2.15 Struktur formulasi f*orward*

Untuk formulasinya menggunakan persamaan 2.10 dan 2.11.

$$z_{ih}(i) = \sum_{i=1}^{n} w_{ih}(i,j) \times x(i) \qquad 2.10$$

$$y_{ih}(i) = \lambda \times z_{ih} \qquad 2.11$$

b. Formulasi Backward



Gambar 2.16 Struktur formulasi b*ackward*

Formulasi *backward* digunakan untuk merevisi bobot dari nilai *error* yang diperoleh dari proses adaptasi jaringan terhadap keluaran model yang diinginkan. Setiap ada *error* baru, jaringan dapat belajar dari *error* tersebut dengan merevisi nilai bobot untuk menyesuaikan karakter nilai. Gambar 2.16 menunjukkan struktur formulasi *backward* pada

jaringan saraf tiruan. Persamaan 2.12 dan Persamaan 2.13 menunjukkan formulasi revisi bobot pada proses *backward.*

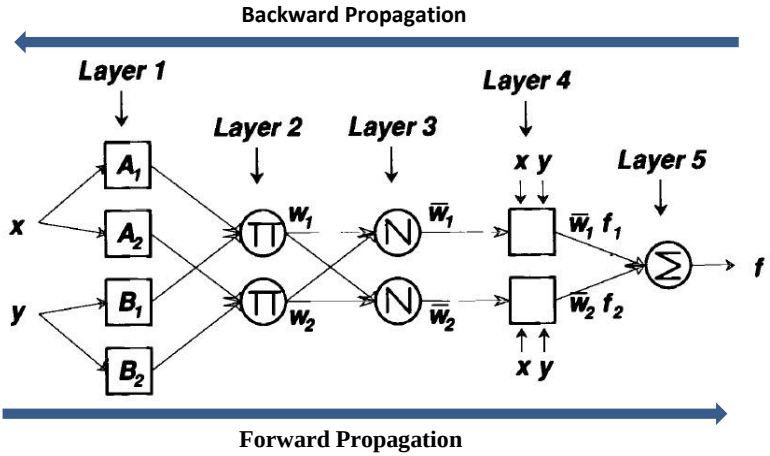
$$e = y_m - y_n \qquad 2.12$$

$$w^b = w^l + \alpha \times f'(z) \times x(i) \qquad 2.13$$

2.6 Kontroler *Neuro fuzzy*

Sistem hibrida yang menggabungkan logika *fuzzy*, jaringan saraf tiruan, algoritma genetika, dan kecerdasan buatan lainnya sudah banyak dikembangkan untuk menyelesaikan berbagai masalah yang berkaitan dengan optimalisasi. Konsep hibrida dikembangkan karena pada metode kecerdasan buatan ini memiliki kelebihan dan kekurangan masing-masing, sehingga dengan konsep hibrida akan diperolehkan suatu metode yang lebih baik dalam menyelesaikan permasalahan optimasi.

Gambar 2.17 menunjukkan skema atau model dari kontroler *Neuro fuzzy.* Terdapat beberapa skema atau struktur *Neuro fuzzy,* seperti *Neuro fuzzy* Mamdani atau Larsent, *Neuro fuzzy* TakagiSugeno (ANFIS / *Artificial Neuro fuzzy Inference System*), *Neuro fuzzy* Biner (Multi input) atau *Neuro fuzzy* formulasi Wang. Dari Gambar 2.16 dapat diketahui bahwa dalam struktur *Neuro fuzzy* terdapat 2 tahapan yaitu *forward propagation* dan *backward propagation.[5]*



Gambar 2.17 Struktur kontroler *neuro fuzzy* dengan 2 masukkan

Kontroler NF merupakan sistem hibrida yang menggabungkan konsep logika *fuzzy* dengan jaringan saraf tiruan (JST). Logika *fuzzy* memiliki kelebihan dalam pengambilan keputusan, sedangkan JST memiliki kelebihan dalam adaptasi melalui kemampuan pembelajaran yang dimilikinya. Kekurangan logika *fuzzy* dalam hal penentuan parameter yang sifatnya intuitif dapat diatasi dengan JST. Pada kontroler NF, logika *fuzzy* akan direpresentasikan dalam JST yang memungkinkan terjadinya pembelajaran di dalamnya dan akan terjadi perubahan bobot untuk mengubah parameter pada logika *fuzzy*.

2.6.1 Tahap *Training*

Pendekatan pemodelan dalam fungsi *neuro fuzzy* serupa dengan teknik-teknik identifikasi sistem pada umumnya. Pertama *neuro fuzzy* mengasumsikan adanya sebuah struktur tertentu yang menghubungkan input dengan output. Kemudian *neuro fuzzy* harus diberikan pasangan data input dan output dalam format yang kompatibel untuk *training*. Jika kedua tahap terpenuhi maka *neuro fuzzy* siap digunakan untuk melatih FIS sehingga mampu menirukan kelakuan sistem yang sedang dimodelkan. *Neuro fuzzy* melatih FIS dengan memodifikasi parameter-parameter fungsi keanggotaan sampai diperoleh selisih (error) minimal antara keluaran FIS dengan data pelatihan output.[6].

Pola *training* yang dilakukan untuk *mobile robot* ada berbagai macam cara seperti menjalankan robot menggunakan remote control kemudian data kecepatan dan pembacaan sensor disimpan untuk dijadikan data *training.* Selain itu bisa dilakukan dengan cara membuat sebuah persamaan kurva untuk *mobile robot* sehingga robot dapat membuat jalur yang menyerupai kurva kemudian data kecepatan dan pembacaan sensornya disimpan untuk data *training*. Dapat juga dilakukan dengan membuat algoritma sederhana gerakan melingkar untuk menghindari obtstacles kemudian data kecepatan dan pembacaan sensornya disimpan untuk data *training,* dan masih banyak cara lain untuk melakukan *training* terhadap *mobile robot* tersebut.

2.6.2 Tahap *Forward Propagation*

Tahapan *forward propagation* merupakan tahapan perhitungan sistem *fuzzy* yang direpresentasikan ke dalam beberapa layer seperti dalam *neural network.* Tahapan ini bertujuan untuk menentukan *output*

layer dari suatu *node*. Dalam sistem ini terdapat 3 lapis data yang dalam beberapa lapisan diantaranya input layer, hidden layer, dan output layer.

Input Layer:

Input layer merupakan *node* masukkan yang mengirim sinyal masukkan ke hidden layer. Pada lapisan ini nilai keluaran *node* merupakan hasil dari fuzzifikasi input variable sistem. Lapisan ini berfungsi sebagai *membership function* untuk mengekspresikan nilai linguistik dari variabel linguistik masukkan.

Hidden Layer :

Hidden layer digunakan untuk basis aturan (*rule-base*) *fuzzy,* setiap *node* pada lapisan mempunyai masukkan dari dua nilai pada lapisan kedua. Operasi pada lapisan ini berbeda-beda tergantung dari struktur *Neuro fuzzy* yang digunakan. Misalnya jika digunakan struktur *Neuro fuzzy* Mamdani maka digunakan operasi MIN sedangkan jika menggunakan *Neuro fuzzy* Larsent digunakan operasi *product.* Persamaan 2.14 menunjukkan persamaan basis aturan (*rule-base*) pada struktur *Neuro fuzzy* Larsent.

$$R_{1,3} = O_{n,2} \times O_{m,2} \; ; untuk \; i = 1,2, \dots, n \times m \qquad 2.14$$

Lapisan ini juga menjalankan proses inferensi sehingga keluaran pada *node* ini diperoleh dari operasi inferensi. Sama halnya pada proses sebelumnya, pada lapisan ini dilakukan operasi inferensi yang bergantung pada struktur *Neuro fuzzy* yang digunakan. Misalnya jika digunakan struktur *Neuro fuzzy* Mamdani maka digunakan operasi MAX sedangkan jika menggunakan *Neuro fuzzy* Larsent digunakan operasi jumlah. Persamaan 2.15 dan Persamaan 2.16 menunjukkan operasi jumlah pada struktur *Neuro fuzzy* Larsent.

$$r_{i,4} = \sum_{j=1}^{n \times m} w_{ij} R_{(j,3)} \; ; \; untuk \; i = 1,2, \dots, n \times m \qquad 2.15$$

$$O_{i,4} = \sum_{i=1}^{n} r_{i,4} \; ; untuk \; i = 1,2, \dots, n \qquad 2.16$$

Pada struktur *Neuro fuzzy* Mamdani/Larsent bobot wij bernilai satu karena pada struktur ini tidak terdapat revisi nilai bobot pada proses inferensi. Revisi bobot wij pada lapisan ini dilakukan apabila menggunakan struktur *Neuro fuzzy* Takagi-Sugeno, yang mana revisi bobot digunakan untuk merevisi *rule-base fuzzy.*

Output Layer :

Output layer merupakan lapisan yang menjalankan proses *defuzzikasi* untuk menghitung sinyal keluaran dari kontroler *Neuro fuzzy.* Apabila digunakan fungsi *fuzzy singleton* maka persamaan 2.17 menunjukkan persamaan dari fungsi *center of gravity.* Dengan bobot adalah wi yang merupakan nilai tengah dari fungsi keanggotaan sinyal keluaran.

$$y = \frac{\sum_{i=1}^{n} w \times Uy_i}{\sum_{i=1}^{n} Uy_i} \qquad 2.17$$

2.6.3 Tahap *Backward Propagation*

Pada kontroler *Neuro fuzzy,* proses *backward* dilakukan untuk merevisi nilai tengah dari *membership function.* Proses tersebut terdapat pada struktur *Neuro fuzzy* Mamdani atau Larsent. Proses *backward* hanya dilakukan satu kali yaitu untuk merevisi nilai tengah dari fungsi keanggotaan sinyal kontrol saja. Berbeda dengan struktur *Neuro fuzzy* Takagi-sugeno (ANFIS), proses *backward* dilakukan sampai ke proses inferensi. Jadi proses *backward* tidak hanya merevisi bobot sinyal kontrol tetapi juga merevisi *rule base* pada proses inferensi. Persamaan 2.10 menunjukkan persamaan revisi nilai tengah pada struktur *Neuro fuzzy* Mamdani.

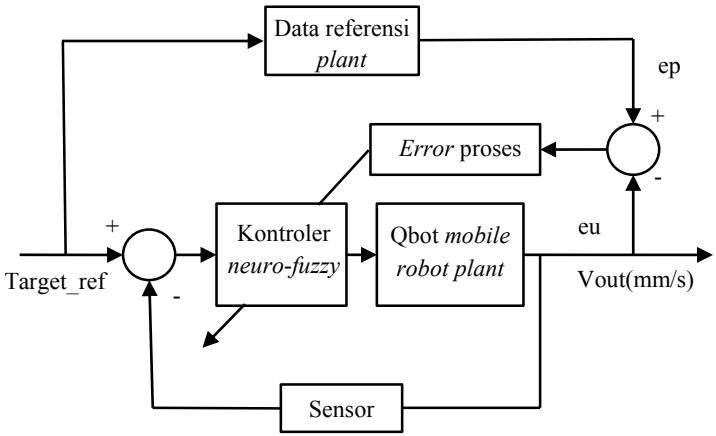
$$w_y^B = w_y^l(i) + \alpha \times e_{num} + \lambda + Uy(i) \qquad 2.18$$

BAB 3
PERANCANGAN SISTEM

Pada bab tiga ini akan dibahas mengenai proses perancangan implementasi perencanaan jalur Qbot *mobile robot* menggunakan metode neuro-fuzzy sebagai kontrolernya.

3.1 Diagram Blok Sistem

Diagram blok pada Gambar 3.1 di bawah ini menunjukkan keseluruhan dari sistem perencanaan jalur pada Qbot *mobile robot* menggunakan kontroler *neuro-fuzzy*.



Gambar 3.1 Diagram blok sistem

Pada diagram blok di atas ditunjukkan bahwa sistem ini terdiri dari beberapa subsistem atau bagian. Bagian target referensi merupakan bagian masukan berupa data jarak target, sudut target, jarak obstacles, dan sudut obstacles yang kemudian diolah kedalam blok kontroler neuro-fuzzy lalu digunakan untuk navigasi Qbot *mobile robot*.

Blok data referensi *plant* didapat dari data kecepatan robot yang sebelumnya dilakukan tahap *Forward propagation* dari data target referensi. Blok prediksi error merupakan selisih dari data referensi plant

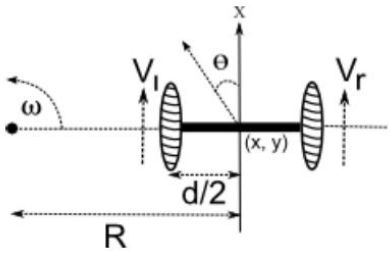
dengan data referensi plant yang diolah kembali pada kontroler *neuro-fuzzy* yang kemudian digunakan untuk proses *learning* dan revisi bobot.

3.2 Prinsip Komputasi Qbot *Mobile robot* [7]

Berikut ini catatan yang diadopsi dari Dudek dan Jenkin mengenai prinsip komputasi *mobile robot.*

3.2.1 *Differential Drive Kinematics*

Sebuah differential drive *mobile robot* terdiri dari dua roda yang terpasang pada aksis yang sama, dan tiap roda bisa dikontrol secara bebas untuk menggerakkan robot kearah depan atau ke belakang. Ketika kecepatan tiap roda dapat bervariasi untuk meraih gerakan melingkar, robot harus memutar sekitar titik yang disebut sebagai ICC (*Instantaneous Center of Curvature*), seperti yang ditunjukkan pada Gambar 3.2.



Gambar 3.2 Kinematik dari differential robot

Lintasan robot dapat dikendalikan dengan cara mengatur kecepatan pada tiap rodanya. Kecepatan rotasi ω sekitar ICC harus sama pada kedua roda. Maka, persamaan berikut ini menyusun hubungan antara parameter gerak dari *differential drive mobile robot.*

$$\omega\left(R+\frac{d}{2}\right)=V_r \qquad 3.1$$

$$\omega\left(R-\frac{d}{2}\right)=V_l \qquad 3.2$$

Dimana *d* adalah jarak antara titik pusat dari kedua roda, V_r dan V_l adalah kecepatan roda kanan dan kiri saat menyusuri lantai,dan R adalah jarak dari ICC ke titik tengah antar roda.

Persamaan 3.1 dan 3.2 dapat diselesaikan pada setiap kasus waktu untuk R dan ω sebagai berikut

$$R = \frac{d(V_r+V_l)}{2(V_r-V_l)} \quad 3.3$$

$$\omega = \frac{V_r-V_l}{d} \quad 3.4$$

3.3 Lokasi ICC

Pada Gambar 3.2 diasumsikan bahwa robot berada di sebuah posisi (x,y) dan meuju kearah yang membuat sudut θ dengan X aksis. Dengan mengetahui kecepatan Vr, V_l dan menggunakan persamaan 3.3 dan 3.4, lokasi ICC (ICC_x , ICC_y) dapat ditentukan sebagai berikut:

$$ICC_x = x - Rsin\theta$$
$$ICC_y = y + Rcos\theta \quad 3.5$$

3.4 *Forward Kinematics*

Berikut ini penjelasan permasalahan *forward kinematics* tentang bagaimana kecepatan yang diberikan pada roda dan konfigurasi posisi awal robot $(x, y, \theta)_{t=0}$ dapat menentukan posisi robot (x, y, θ).

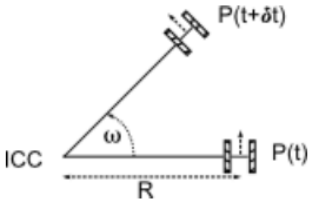
3.4.1 *Pose Relative* Robot ke Lokasi ICC

Diberikan sebuah kecepatan yang tidak bervariasi pada Vr dan Vl, lokasi ICC (ICC_x, ICC_y) akan menjadi posisi tetap. Karena itu , pada waktu t+δt pose robot akan menjadi:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos(\omega \times \delta t) & -\sin(\omega \times \delta t) & 0 \\ \sin(\omega \times \delta t) & \cos(\omega \times \delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - ICC_x \\ y - ICC_y \\ \theta \end{bmatrix} + \begin{bmatrix} ICC_x \\ ICC_y \\ \omega \times \delta t \end{bmatrix} \quad 3.6$$

Dimana (x, y, θ) dan (x', y', θ') adalah posisi robot pada waktu t dan t+δt, berturut-turut. Persamaan ini menggambarkan gerak berputar robot

sekitar ICC nya dengan sebuah radius dari lengkungan R ansebuah kecepatan sudut ω ,seperti pada Gambar 3.3.



Gambar 3.3 *Forward kinematics relative to ICC*

3.4.2 *Pose Relative* Robot pada Posisi Awal Robot

Persamaan gerak umum robot yang mampu bergerak dalam arah θ(t) tertentu pada kecepatan V(t) yang diberikan,dijelaskan sebagai berikut.

$$\begin{aligned} x(t) &= \int_0^t V(t)\cos[\theta(t)]\,dt \\ y(t) &= \int_0^t V(t)\sin[\theta(t)]\,dt \\ \theta(t) &= \int_0^t \omega(t)dt \end{aligned} \qquad 3.7$$

Untuk sebuah robot differential drive seperti Qbot,persamaan 3.7 menjadi

$$\begin{aligned} x(t) &= \frac{1}{2}\int_0^t [V_r(t) + V_l(t)]\cos[\theta(t)]\,dt \\ y(t) &= \frac{1}{2}\int_0^t V(t)\sin[\theta(t)]\,dt \\ \theta(t) &= \frac{1}{d}\int_0^t \omega(t)dt \end{aligned} \qquad 3.8$$

Persamaan 3.7 dapat disederhanakan untuk menentukan *pose* robot saat V sebagai berikut

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x + V\cos(\theta)\delta t \\ y + V\sin(\theta)\delta t \\ \theta + \omega\delta t \end{bmatrix} \qquad 3.9$$

Demikian pula pada persamaan 3.8

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x + \frac{1}{2}[V_r + V_l]\cos(\theta)\,\delta t \\ y + \frac{1}{2}[V_r + V_l]\sin(\theta)\,\delta t \\ \theta + \frac{1}{d}[V_r - V_l]\delta t \end{bmatrix} \quad 3.10$$

Dengan demikian, untuk kasus khusus dari $V_l=V_r=V$ dan $Vl=-Vr=V$, persamaan 3.9 menjadi persamaan 3.11 dan 3.12 berikut

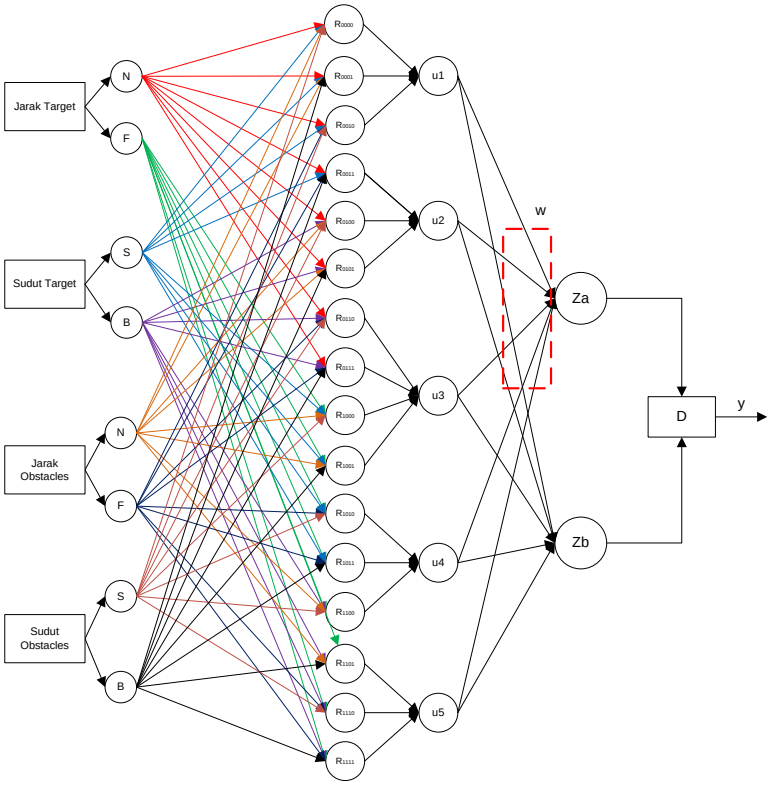
$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x + V\cos(\theta)\delta t \\ y + V\sin(\theta)\delta t \\ \theta \end{bmatrix} \quad 3.11$$

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta + \frac{2V\delta t}{d} \end{bmatrix} \quad 3.12$$

3.5 Perancangan Kontroler *Neuro-Fuzzy*

Kontroler *neuro-fuzzy* pada Qbot *mobile robot* ini digunakan untuk navigasi robot mengatur kecepatan roda dan mencapai target posisi yang diinginkan. Kontroler *neuro fuzzy* yang digunakan adalah kontroler *neuro fuzzy* mamdani karena hanya diperlukan untuk perbaikan nilai tengah dari himpunan pendukung sinyal control dan rule base yang digunakan tetap.

Pada Gambar 3.4 menunjukkan struktur kontroler *neuro fuzzy* mamdani dengan 4 input variabel dimana tiap input variabel tersebut memiliki 2 himpunan pendukung. Pada struktur tersebut terdapat 3 lapis yaitu input layer, hidden layer, dan output layer.



Gambar 3.4 Struktur *neuro fuzzy*

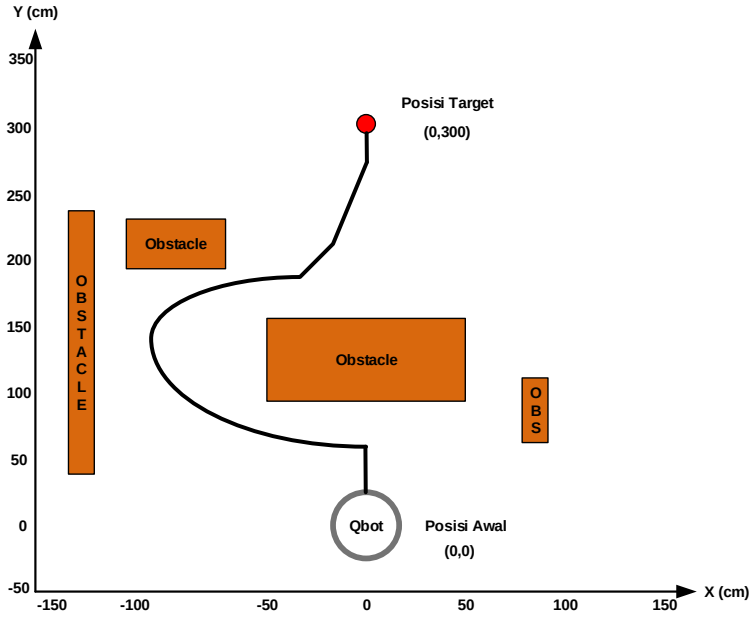
3.5.1 Tahap *Training*

Pada tahap *training* yang dilakukan pada Qbot *mobile robot* dibuat dua pola ruang yang terdapat obstacles. Objek ruang pergerakan Qbot *mobile robot* berada pada ruang dua dimensi dengan koordinat (x,y) dengan satuan centimeter. Pada tiap pola tersebut Qbot *mobile robot* bergerak dari titik awal yang telah ditentukan menuju posisi target dimana pada jalur yang dilalui tersebut terdapat obstacles. Qbot *mobile robot* ini dijalankan menggunakan algoritma sederhana untuk menghindari obstacles secara melingkar hingga didapatkan data kecepatan roda kiri,

roda kanan, sinyal kontrol, jarak ke target, sudut ke target, jarak ke obstacles, dan sudut obstacles. Dari dua pola *training* ini data *training* nanti akan digunakan untuk proses learning Qbot *mobile robot* untuk mengikuti jalur yang telah ditentukan pada proses *training*.

Pola 1

Pada pola pertama posisi awal Qbot *mobile robot* berada pada koordinat (0 , 0)cm bergerak menuju posisi target (0 , 300)cm dan sudut awal sebesar 90° dengan pola obstacles seperti pada gambar 3.5.

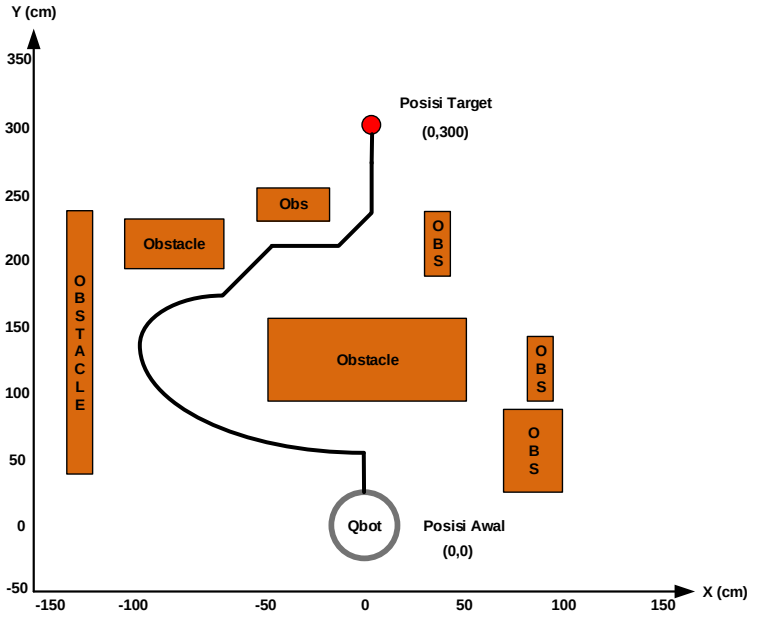


Gambar 3.5 Jalur Qbot *mobile robot* pola 1

Pola 2

Pada pola kedua psosisi awal dan target tujuan Qbot *mobile robot* sama seperti pola pertama bergerak dari koordinat (0 , 0)cm menuju koordinat (0 , 300)cm dan sudut awal sebesar 90°, yang membedakan

hanya pada pola obstaclesnya. Pola obstacles pada pola kedua seperti pada Gambar 3.6.



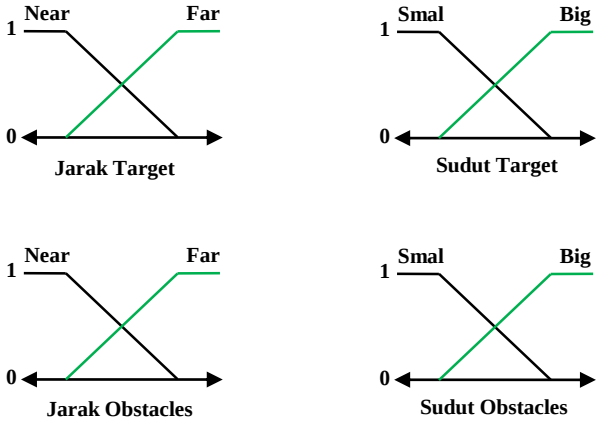
Gambar 3.6 Jalur Qbot *mobile robot* pola 2

3.5.2 Tahap *Forward propagation*

Pada tahap *forward propagation* ini dilakukan perhitungan untuk memperoleh nilai sinyal kontrol *neuro fuzzy* yang mana sinyal kontrol tersebut digunakan untuk dibandingkan dengan sinyal kontrol dari data referensi plant yang didapat dari hasil *training*, kemudian digunakan untuk perhitungan bobot baru pada tahap *backward propagation*.

Input Layer :

Pada bagian input layer terdapat 4 input variable seperti pada Gambar 3.7, diantaranya yaitu jarak target, sudut target, jarak obstacles, dan sudut obstacles. Keempat input variable tersebut difuzifikasi yang mana semua input tersebut memiliki 2 membership function. Sehingga keluaran dari input layer ini ada sebanyak 18 node.



Gambar 3.7 Empat input variable dengan 2 membership function

Hidden Layer :

Hidden layer terdiri dari 18 node yang merupakan proses *fuzzifikasi* dari jarak target, sudut target, jarak obstacles, dan sudut obstacles. Pada lapisan ini dilakukan proses menentukan basis aturan dan rules. Untuk proses perhitungannya menggunakan persamaan 2.4, metode Larsent untuk mencari nilai minimumnya. Setelah didapat nilai minumnya selanjutnya mencari nilai maksimum menggunakan persamaan 2.15 dan 2.16.

Output Layer :

Output layer merupakan hasil keluaran dari hidden layer, tahap ini melakukan proses relasi antara hidden layer dengan output layer. Pada sisitem ini output layer ada 5 node. Setelah proses ini dilanjutkan ke proses defuzzifikasi dimana output dari kelima node ini dibagi dua,ada bagian atas dan bawah, untuk perhitungannya dapat digunakan persamaan 2.17.

3.5.3 Tahap *Backward Propagation*

Pada tahap ini data output sinyal kontrol yang didapat dari proses *Forward propagation* dibandingkan dengan sinyal kontrol dari data referensi yang didapat dari proses *training* untuk mendapatan nilai error sinyal kontrol seperti pada Gambar 2.16. Setelah itu dilakukan proses

learning secara offline untuk merevisi bobot. Untuk perhitungan nilai bobot baru menggunakan persamaan 2.18. Kemudian nilai bobot baru ini dimasukan kedalam program real-time Qbot *mobile robot* dan setelah itu robot dijalankan kembali dengan nilai bobot baru dan sinyal kontrol baru sehingga pergerakan robot dapat menyerupai data referensi.

BAB 4
HASIL DAN ANALISA

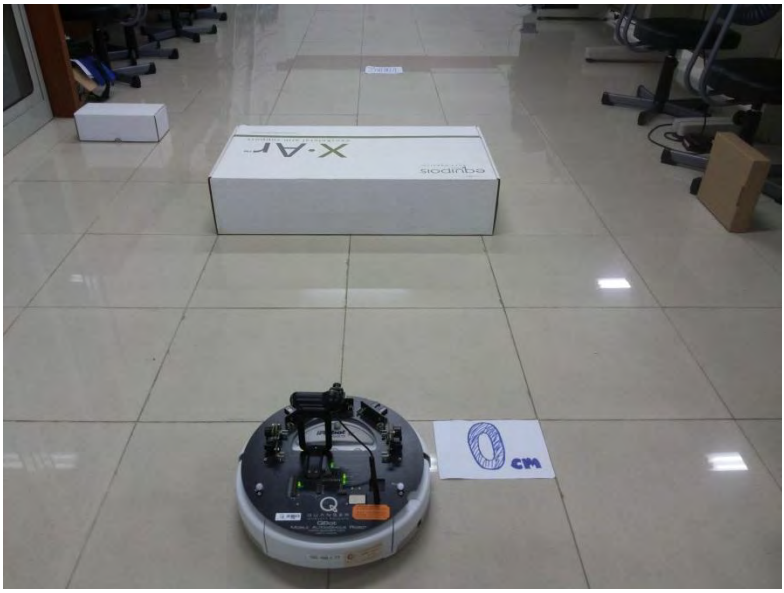
Bab ini membahas mengenai implementasi sistem kontrol *neuro fuzzy* pada Qbot *mobile robot*. Untuk implementasi kontroler *neuro fuzzy* terlebih dahulu dilakukan pengambilan data traning untuk output yang diinginkan pada plant yang kemudian akan dibandingkan dengan output *neuro fuzzy* untuk mengetahui error propagation lalu dilakukan proses *offline learning* untuk mendapatkan bobot yang baru. Pengambilan data *training* ini dilakukan dengan cara menjalankan Qbot mobil robot dari titik awal menuju target dimana pada jalurnya terdapat *obstacles*. Kemudian diambil datanya untuk data referensi plant sebagai pembanding dengan output dari kontroler *neuro fuzzy*, untuk selisih antara output data *training* dengan output *neuro fuzzy* ini disebut error model yang mana error model ini akan menjadi referensi untuk proses o*ffline learning* untuk mendapatkan bobot yang baru. Setelah mendapatkan bobot yang baru, nilai bobot baru ini dimasukan kedalam kontroler *neuro fuzzy* lalu Qbot *mobile robot* dijalankan kembali dengan pola yang sama dengan pola *training*, makan respond an sinyal kontrolnya akan mendekati sinyal kontrol data referensi hasil *training*.

4.1 Implementasi Perencanaan Jalur Pola 1

Pada implementasi perencanaan jalur pola 1 ini dilakukan beberapa tahap berikut diantaranya tahap *training*, *offline learning*, dan pengimplementasian kontroler neuro fuzzy dengan bobot baru seperti yang akan dijelaskan di bawah ini.

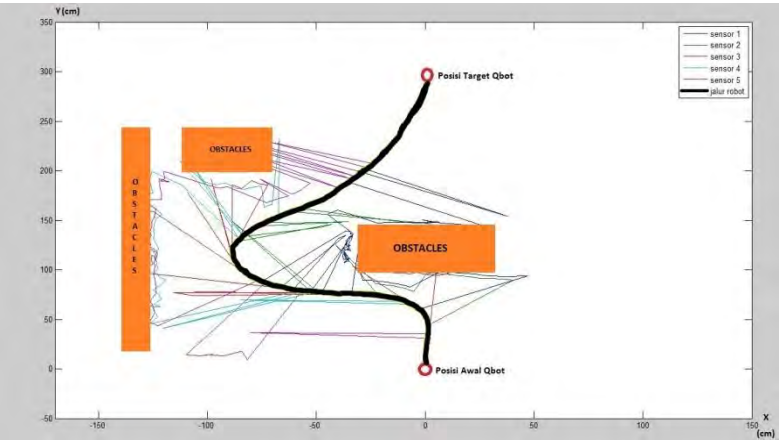
4.1.1 Tahap *Training* Pola 1

Pada tahap *training* ini Qbot *mobile robot* ditentukan terlebih dahulu posisi awal, posisi target tujuan yang akan dicapai, dan sudut awal robot seperti pada Gambar 3.4. Qbot *mobile robot* bergerak dari titik awalnya menuju posisi target yang dituju dimana pada jalur yang akan dilalui Qbot *mobile robot* ini terdapat *obstacles* seperti pada Gambar 4.1 dan Qbot mobile robot harus dapat melaui obstacles tersebut agar dapat mencapai posisi target yang dituju.



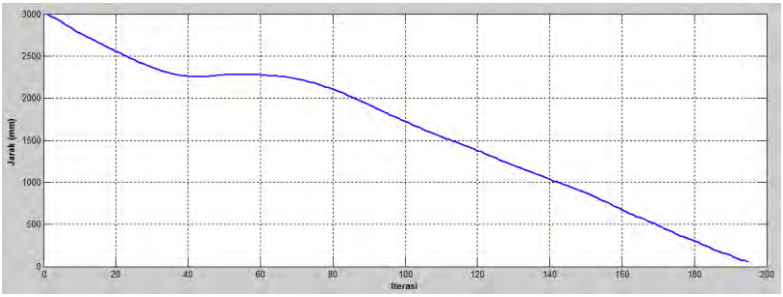
Gambar 4.1 Qbot *mobile robot* pada pola 1

Posisi awal Qbot *mobile robot* terletak pada koordinat (0,0)cm sedangkan target yang akan dituju oleh Qbot *mobile robot* terletak pada koordinat (0,300)cm dan sudut awal sebesar 90°. Kemudian Qbot mobile dijalankan menggunakan algoritma sederhana untuk gerakan melingkar menghindari obstacles, sehingga didapatlah data yang dibutuhkan untuk *offline learning* yaitu data jarak ke target, sudut ke target, jarak ke *obstacles*, sudut *obstacles*, sinyal kontrol, kecepatan motor kanan dan kecepatan motor kiri. Pada Gambar 4.2 dapat dilihat jalur yang telah dilalui oleh Qbot *mobile robot* dari titik (0,0)cm menuju titik (0,300)cm dimana pada jalur tersebut terdapat *obstacles* lalu Qbot *mobile robot* dapat menghindari *obstacles* dan dapat mencapai target tujuannya seperti pada Gambar 4.2.

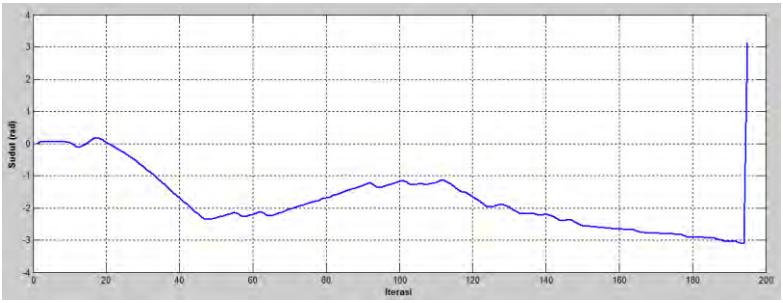


Gambar 4.2 *Training* jalur Qbot *mobile robot* pola 1

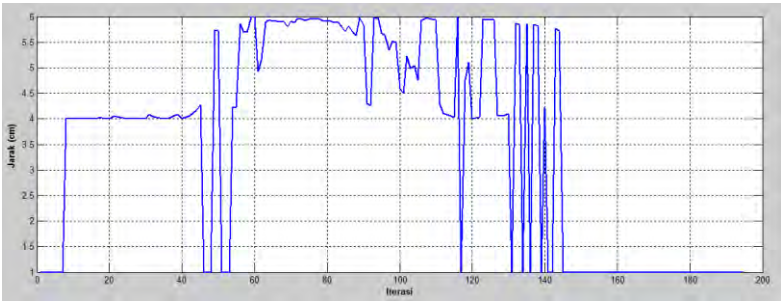
Dari hasil pergerakan Qbot *mobile robot* ini didapatkan data yang akan digunakan untuk proses learning seperti pada Gambar 4.3, Gambar 4.4, Gambar 4.5, dan Gambar 4.6.



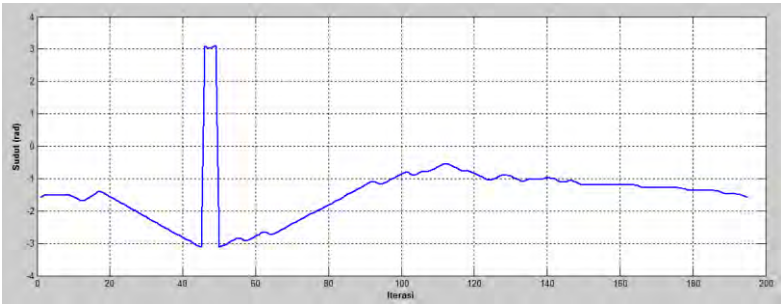
Gambar 4.3 Grafik jarak target pola 1



Gambar 4.4 Grafik Sudut Target



Gambar 4.5 Grafik Jarak *Obstacles*

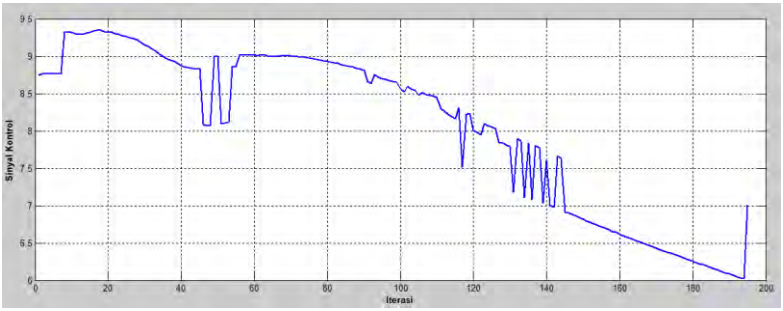


Gambar 4.6 Grafik Sudut *Obstacles*

Dari data diatas, Gambar 4.3 menunjukkan data jarak ke target yang diatur dari titik (0,0)cm menuju titik (0,300)cm. perubahan grafik jarak target ini tidak linear dikarenakan pada saat Qbot *mobile robot* menuju target terdapat *obstacles* sehingga robot harus berbelok menghindar lalu

kembali kejalur lintasan menuju target. Gambar 4.4 menunjukkan data sudut target dimana Qbot *mobile robot* mengukur sudut target yang akan dituju agar menjadi acuan robot menuju target. Gambar 4.5 menunjukkan data jarak *obstacles* dengan skala dari 1 sampai 6 cm. Data jarak obastacles ini menjadi acuan Qbot *mobile robot* untuk menghindar dari tabrakan dengan obastacles. Gambar 4.6 menunjukkan data sudut *obstacles* dimana data ini manjadi acuan Qbot *mobile robot* untuk mengetahui posisi sudut *obstacles* agar robot dapat menghindar dengan aman.

Keempat data grafik di atas merupakan data masukan yang kemudian dilakukan komputasi sehingga menghasilkan sinyal kontrol yang akan digunakan untuk mengatur navigasi Qbot mobile robot. Sinyal kontrol yang dihasilkan dari komputasi keempat data tersebut dapat dilihat pada Gambar 4.7.



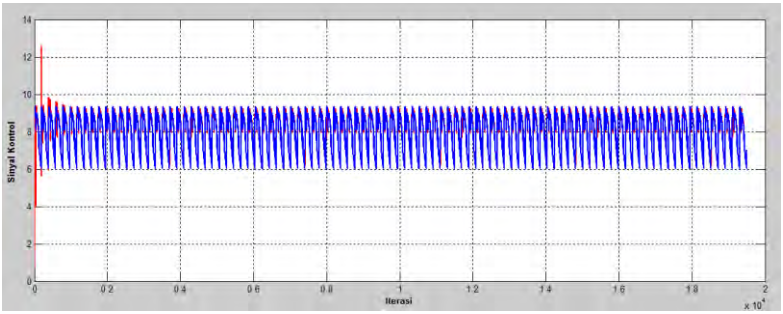
Gambar 4.7 Grafik Sinyal Kontrol

Dari keempat data jarak target, sudut target, jarak *obstacles*, dan sudut *obstacles* ini dijadikan masukan kedalam kontroler *neuro fuzzy* untuk dilakukan proses komputasi secara *forward propagation*. Kemudian keluarannya ini menjadi keluaran sinyal kontrol *neuro fuzzy* yang akan dibandingkan dengan sinyal kontrol dari data hasil *training* seperti Gambar 4.7.

4.1.2 Simulasi *Offline Learning* Pola 1

Dari data yang diperoleh dari proses *training* dilanjutkan pada tahap *offline learning* untuk dapat merevisi bobot hingga didapat sinyal kontrol neuro fuzzy. Dari Gambar 4.8 dapat dilihat sinyal kontrol dari *neuro fuzzy* yang berwarna merah melakukan proses learning dengan

perhitungan menggunakan persamaan 2.18 untuk mendekati sinyal kontrol referensi. Proses learning ini dilakukan sebanyak 100 *epoch,* semakin banyak jumlah epoch akan semakin baik.

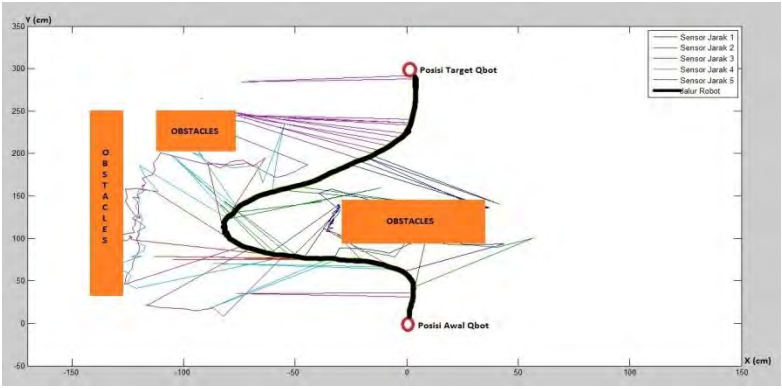


Gambar 4.8 Proses Learning Sinyal kontrol pola 1

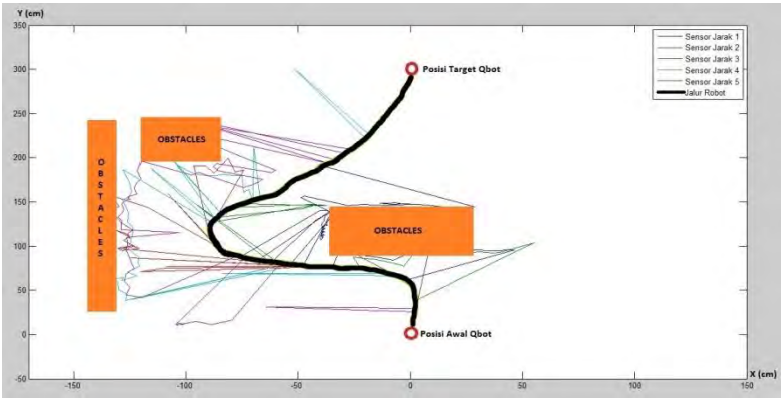
Dari proses learning tersebut didapat nilai bobot baru, w = [6.6453 6.8238 7.6222 9.7074 9.7123]. Nilai bobot baru tersebut kemudian dimasukan kedalam kontroler *neuro fuzzy* lalu Qbot *mobile robot* dijalankan dan dilihat sinyal kontrol dan pergerakannya apakah sama dengan pergerakan saat *training*.

4.1.3 Implementasi *Neuro fuzzy* Pola 1 pada Qbot *Mobile Robot*

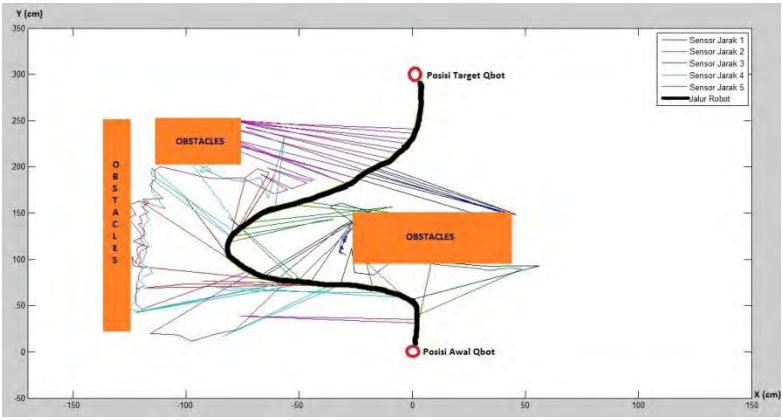
Pada tahap implementasi kontroler *neuro fuzzy* pada Qbot *mobile robot* ini nilai bobot baru sudah dimasukan kedalam kontroler neuro fuzzy. Untuk percobaan implementasi neuro fuzzy pola 1 ini dilakukan sebanyak tiga kali untuk melihat perbandingan sinyal kontrol neuro fuzzy tiap percobaan dengan sinyal kontrol referensi dari hasil *training*. Untuk ruang yang akan dilalui sama seperti pada Gambar 3.4 dan Gambar 4.1. Posisi awal robot pada titik (0,0)cm dan target yang akan dituju berada pada titik (0,300)cm. Qbot *mobile robot* dijalankan menggunakan kontroler *neuro fuzzy*. Dari Gambar 4.9, Gambar 4.10, dan Gambar 4.11 dapat dilihat Qbot *mobile robot* dapat menghindar dari *obstacles* dan mencapai target tujuannya, jalur pergerakkannya hampir menyerupai jalur pergerakan *training* yang merupakan data referensi seperti pada Gambar 4.2



Gambar 4.9 Jalur Qbot *mobile robot* pola 1 percobaan 1

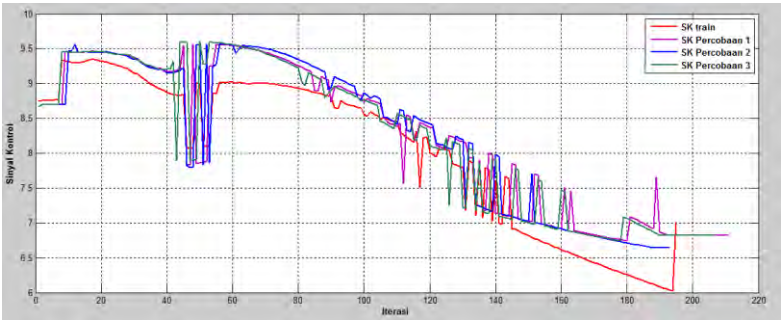


Gambar 4.10 Jalur Qbot *mobile robot* pola 1 percobaan 2



Gambar 4.11 Jalur Qbot *mobile robot* pola 1 percobaan 3

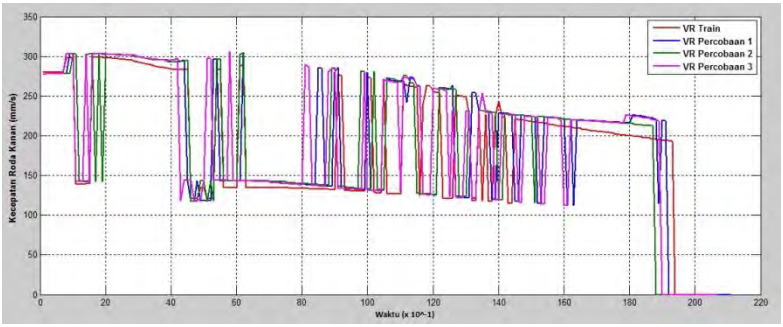
Hasil dari menjalankan Qbot *mobile robot* ini dapat dilihat pada Gambar 4.12 sinyal kontrol *neuro fuzzy* Qbot *mobile robot* dari tiga kali percobaan mendekati sinyal kontrol data referensi dan waktu mencapai targetnya kontroler *neuro fuzzy* memiliki respon lebih cepat dan besar daripada data referensinya.



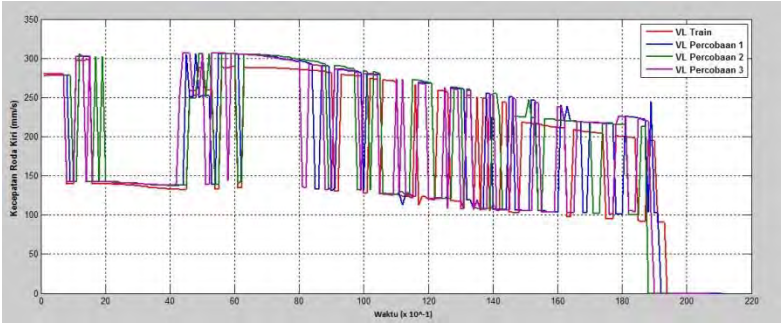
Gambar 4.12 Perbandingan sinyal kontrol *neuro fuzzy* dan referensi pola 1

Dari sinyal kontrol diperoleh kecepatan roda kanan dan roda kiri. Dari hasil perbandingan sinyal kontrol *neuro fuzzy* dan sinyal kontrol data referensi terlihat kecepatan roda kanan dan kecepatan roda kiri *neuro fuzzy* dari tiga kali percobaan mendekati kecepatan roda kiri dan

kecepatan roda kanan dari data referensi seperti pada Gambar 4.13 dan Gambar 4.14



Gambar 4.13 Perbandingan kecepatan roda kanan *neuro fuzzy* tiga percobaan dengan referensi *training* pola 1



Gambar 4.14 Perbandingan kecepatan roda kiri *neuro fuzzy* tiga percobaan dengan referensi *training* pola 1

Dari grafik kecepatan Qbot mobile robot diatas dapat diketahui perbandingan waktu yang dibutuhkan Qbot mobile robot untuk mencapai posisi target saat *training* dengan Qbot mobile robot yang menggunakan kontroler neuro fuzzy dengan dilakukaknnya tiga kali percobaan dapat dilihat pada Tabel 4.1.

Tabel 4.1 Perbandingan waktu *training* dengan kontroler *neuro fuzzy* pada pola 1

Data	Koordinat awal Qbot (cm)	Koordinat target Qbot (cm)	Waktu (detik)
Training	(0 , 0)	(0 , 300)	19,5
Percobaan 1 *neuro fuzzy*	(0 , 0)	(0 , 300)	19,2
Percobaan 2 *neuro fuzzy*	(0 , 0)	(0 , 300)	18,8
Percobaan 3 *neuro fuzzy*	(0 , 0)	(0 , 300)	19,0

Dari tabel 4.1 tersebut terlihat bahwa respon kontroler neuro fuzzy lebih cepat dari data referensi hasil *training*nya. Sehingga kontroler ini sangat baik untuk mempercepat respon dari model referensinya.

4.2 Implementasi Perencanaan Jalur Pola 2

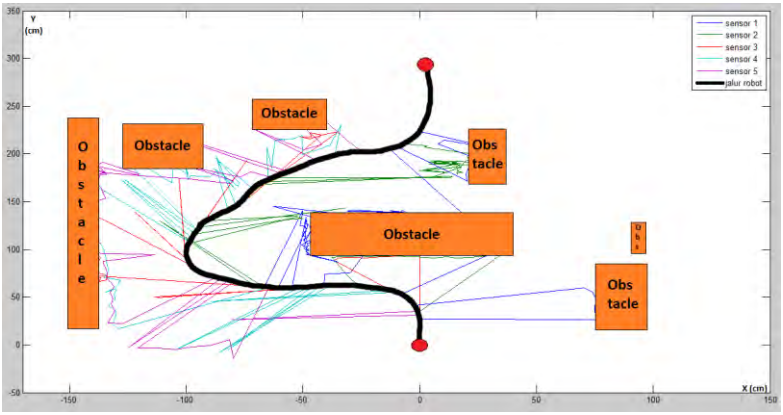
Pada implementasi perencanaan jalur pola 2 tidak jauh berbeda dengan pola 1 yaitu dengan melakukan beberapa tahap berikut diantaranya tahap *training*, *offline learning*, dan pengimplementasian kontroler neuro fuzzy dengan bobot baru, yang membedakan hanya pola obstacles yang berada pada jalur Qbot mobile robot.

4.2.1 Tahap *Training* Pola 2

Pada tahap *training* ini Qbot *mobile robot* ditentukan terlebih dahulu posisi awal, posisi target tujuan, dan sudut awal robot seperti pada Gambar 3.5. Qbot *mobile robot* bergerak dari titik awalnya menuju posisi target yang dituju dimana pada jalur yang akan dilalui Qbot *mobile robot* ini terdapat *obstacles* seperti pada Gambar 4.15.

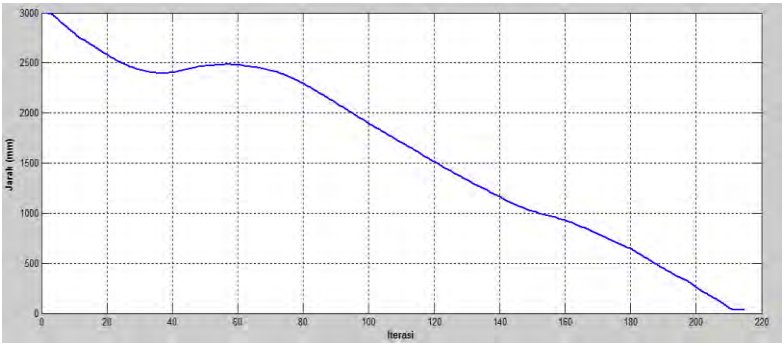
Gambar 4.15 Qbot *mobile robot* pada pola 2

Posisi awal Qbot *mobile robot* terletak pada koordinat (0,0)cm sedangkan target yang akan dituju oleh Qbot *mobile robot* terletak pada koordinat (0,300)cm dan sudut awal sebesar 90°. Kemudian Qbot mobile dijalankan menggunakan algoritma sederhana untuk gerakan melingkar menghindari obstacles, sehingga didapatlah data yang dibutuhkan untuk *offline learning* yaitu data jarak ke target, sudut ke target, jarak ke *obstacles*, sudut *obstacles*, sinyal kontrol, kecepatan motor kanan dan kecepatan motor kiri. Pada Gambar 4.2 dapat dilihat jalur yang telah dilalui oleh Qbot *mobile robot* dari titik (0,0)cm menuju titik (0,300)cm dimana pada jalur tersebut terdapat *obstacles* dengan pola 2 lalu Qbot *mobile robot* dapat menghindari *obstacles* dan dapat mencapai target tujuannya seperti pada Gambar 4.16.

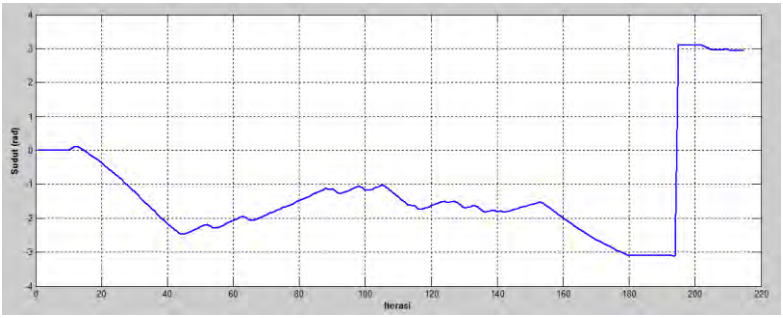


Gambar 4.16 *Training* jalur Qbot *mobile robot* pola 2

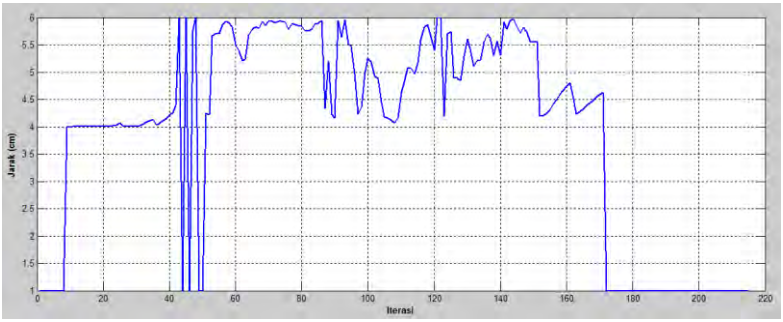
Dari hasil pergerakan Qbot *mobile robot* ini didapatkan data yang akan digunakan untuk proses learning seperti pada Gambar 4.17, Gambar 4.18, Gambar 4.19, Gambar 4.20.



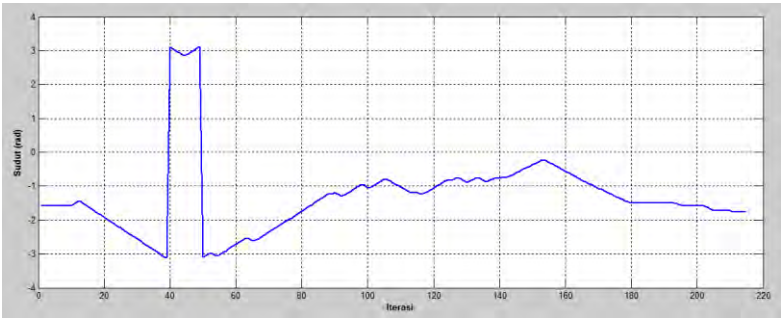
Gambar 4.17 Grafik jarak target pola 2



Gambar 4.18 Grafik sudut target pola 2



Gambar 4.19 Grafik jarak obstacles pola 2

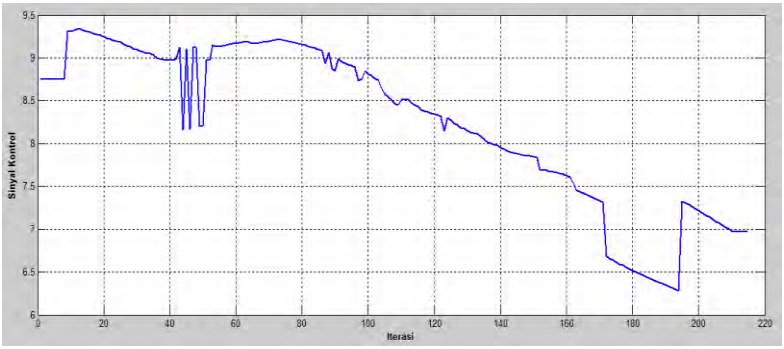


Gambar 4.20 Grafik sudut obstacles pola 2

Dari data diatas, Gambar 4.17 menunjukkan data jarak ke target yang diatur dari titik (0,0)cm menuju titik (0,300)cm. perubahan grafik jarak target ini tidak linear dikarenakan pada saat Qbot *mobile robot* menuju

target terdapat *obstacles* sehingga robot harus berbelok menghindar lalu kembali kejalur lintasan menuju target. Gambar 4.18 menunjukkan data sudut target dimana Qbot *mobile robot* mengukur sudut target yang akan dituju agar menjadi acuan robot menuju target. Gambar 4.19 menunjukkan data jarak *obstacles* dengan skala dari 1 sampai 6 cm. Data jarak obastacles ini menjadi acuan Qbot *mobile robot* untuk menghindar dari tabrakan dengan obastacles. Gambar 4.20 menunjukkan data sudut *obstacles* dimana data ini manjadi acuan Qbot *mobile robot* untuk mengetahui posisi sudut *obstacles* agar robot dapat menghindar dengan aman.

Keempat data grafik di atas merupakan data masukan yang kemudian dilakukan komputasi sehingga menghasilkan sinyal kontrol yang akan digunakan untuk mengatur navigasi Qbot mobile robot. Sinyal kontrol yang dihasilkan dari komputasi keempat data tersebut dapat dilihat pada Gambar 4.21.



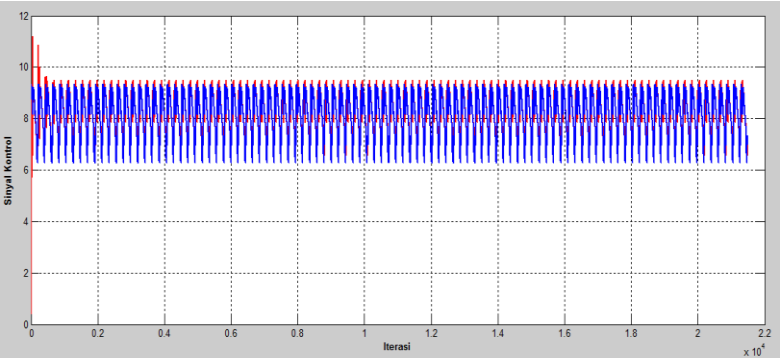
Gambar 4.21 Grafik sinyal kontrol pola 2

Dari keempat data jarak target, sudut target, jarak *obstacles*, dan sudut *obstacles* ini dijadikan masukan kedalam kontroler *neuro fuzzy* untuk dilakukan proses komputasi secara *Forward propagation*. Kemudian keluarannya ini menjadi keluaran sinyal kontrol *neuro fuzzy* yang akan dibandingkan dengan sinyal kontrol dari data hasil *training* seperti Gambar 4.21.

4.2.2 Simulasi *Offline Learning* Pola 2

Dari data yang diperoleh dari proses *training* dilanjutkan pada tahap *offline learning* untuk dapat merevisi bobot hingga didapat sinyal kontrol neuro fuzzy. Dari Gambar 4.22 dapat dilihat sinyal kontrol dari

neuro fuzzy yang berwarna merah melakukan proses learning dengan perhitungan menggunakan persamaan 2.18 untuk mendekati sinyal kontrol referensi. Proses learning ini dilakukan sebanyak 100 *epoch,* semakin banyak jumlah epoch akan semakin baik.



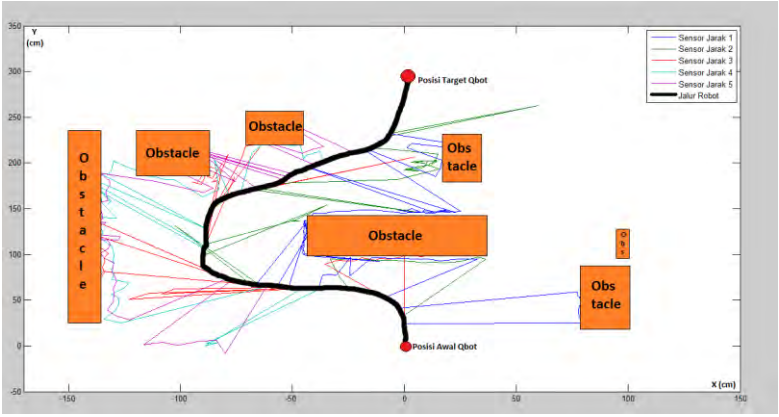
Gambar 4.22 Proses Learning Sinyal kontrol pola 2

Dari proses learning tersebut didapat nilai bobot baru, w = [7.1335 6.4287 7.2964 10.2027 9.6856]. Nilai bobot baru tersebut kemudian dimasukan kedalam kontroler *neuro fuzzy* lalu Qbot *mobile robot* dijalankan dan dilihat sinyal kontrol dan pergerakannya apakah sama dengan pergerakan saat *training*.

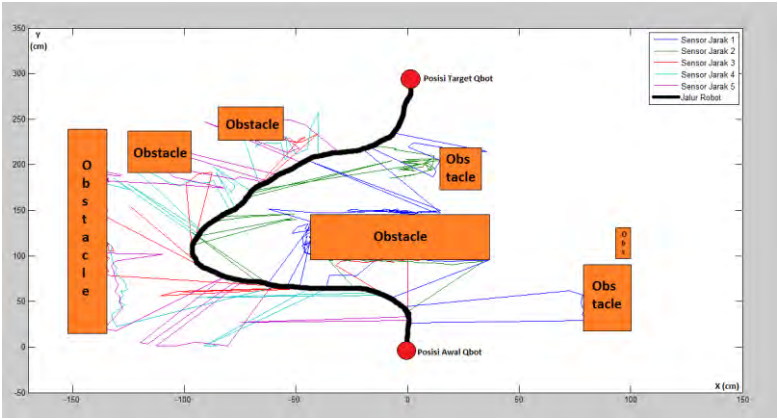
4.2.3 Implementasi *Neuro fuzzy* Pola 2 pada Qbot *Mobile Robot*

Pada tahap implementasi kontroler *neuro fuzzy* pada Qbot *mobile robot* nilai bobot baru dimasukan kedalam kontroler neuro fuzzy. Untuk percobaan implementasi neuro fuzzy pola 2 dilakukan sebanyak tiga kali untuk melihat perbandingan sinyal kontrol neuro fuzzy tiap percobaan dengan sinyal kontrol referensi dari hasil *training*. Untuk ruang yang akan dilalui sama seperti pada Gambar 3.4 dan Gambar 4.15. Posisi awal robot pada titik (0,0)cm dan target yang akan dituju berada pada titik (0,300)cm. Qbot *mobile robot* dijalankan menggunakan kontroler *neuro fuzzy*. Dari Gambar 4.23, Gambar 4.24, dan Gambar 4.25 dapat dilihat Qbot *mobile robot* dapat menghindar dari *obstacles* dan mencapai target tujuannya,

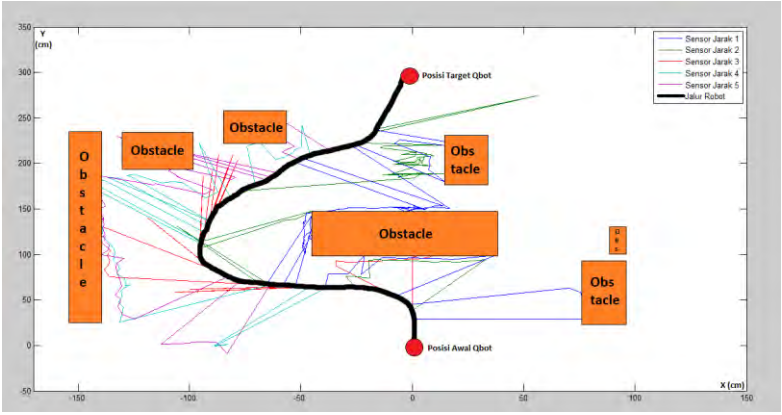
jalur pergerakkannya hampir menyerupai jalur pergerakan *training* yang merupakan data referensi seperti pada Gambar 4.16.



Gambar 4.23 Jalur Qbot *mobile robot* pola 2 percobaan 1

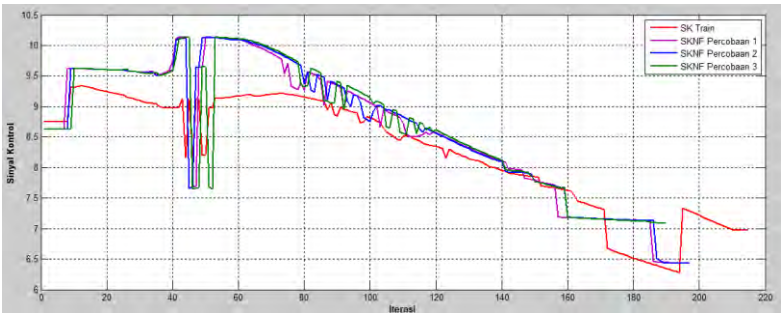


Gambar 4.24 Jalur Qbot *mobile robot* pola 2 percobaan 2



Gambar 4.25 Jalur Qbot *mobile robot* pola 2 percobaan 3

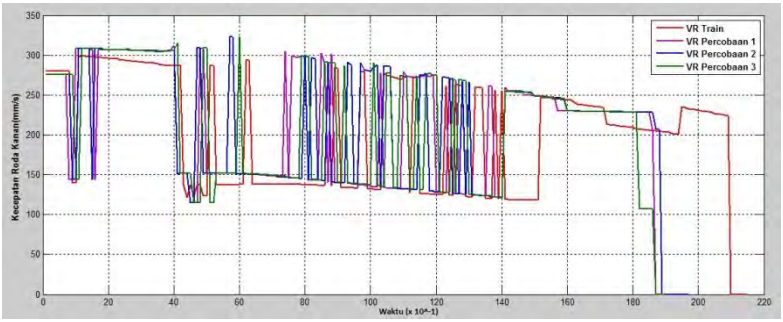
Hasil dari menjalankan Qbot *mobile robot* ini dapat dilihat pada Gambar 4.26 sinyal kontrol *neuro fuzzy* Qbot *mobile robot* dari tiga kali percobaan mendekati sinyal kontrol data referensi dan waktu mencapai targetnya kontroler *neuro fuzzy* memiliki respon lebih cepat dan besar daripada data referensinya.



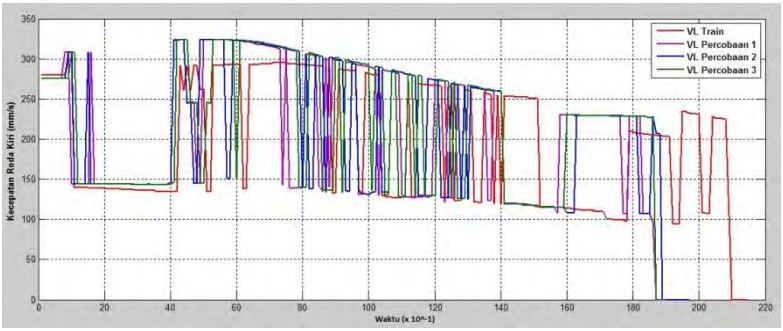
Gambar 4.26 Perbandingan sinyal kontrol *neuro fuzzy* dan referensi pola 2

Dari sinyal kontrol diperoleh kecepatan roda kanan dan roda kiri. Dari hasil perbandingan sinyal kontrol *neuro fuzzy* dan sinyal kontrol data referensi terlihat kecepatan roda kanan dan kecepatan roda kiri *neuro fuzzy* dari tiga kali percobaan mendekati kecepatan roda kiri dan

kecepatan roda kanan dari data referensi seperti pada Gambar 4.27 dan Gambar 4.28



Gambar 4.27 Perbandingan kecepatan roda kanan *neuro fuzzy* tiga percobaan dengan referensi *training* pola 2



Gambar 4.28 Perbandingan kecepatan roda kiri *neuro fuzzy* tiga percobaan dengan referensi *training* pola 2

Dari grafik kecepatan Qbot mobile robot diatas dapat diketahui perbandingan waktu yang dibutuhkan Qbot mobile robot untuk mencapai posisi target saat *training* dengan Qbot mobile robot yang telah terpasang kontroler neuro fuzzy dengan dilakukaknnya tiga kali percobaan dapat dilihat pada Tabel 4.2.

Tabel 4.2 Perbandingan waktu *training* dengan kontroler *neuro fuzzy* pada pola 2

Data	Koordinat awal Qbot (cm)	Koordinat target Qbot (cm)	Waktu (detik)
Training	(0 , 0)	(0 , 300)	21,5
Percobaan 1 *neuro fuzzy*	(0 , 0)	(0 , 300)	18,7
Percobaan 2 *neuro fuzzy*	(0 , 0)	(0 , 300)	18,9
Percobaan 3 *neuro fuzzy*	(0 , 0)	(0 , 300)	18,7

Dari tabel 4.2 tersebut terlihat bahwa respon kontroler neuro fuzzy lebih cepat dari data referensi hasil *training*nya. Sehingga kontroler ini sangat baik untuk mempercepat respon dari model referensinya.

BAB 5
PENUTUP

5.1 Kesimpulan

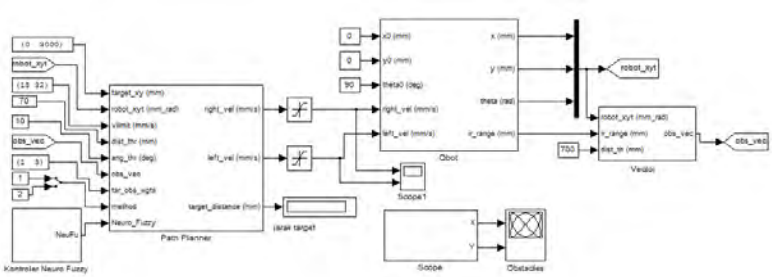
Dari hasil pengimplementasian perencanaan jalur pada Qbot *mobile robot* menggunakan metode kontroler *neuro fuzzy* dapat disimpulkan bahwa sistem kontroler *neuro fuzzy* ini dapat mengikuti data referensi dari hasil training yang telah dilakukan. Kontroler *neuro fuzzy* memiliki respon yang lebih cepat dibandingkan dengan respon saat *training*. Sebagaimana dari hasil pengujian kontroler *neuro fuzzy* pada Qbot *mobile robot* untuk *training* pola 1 waktu yang dibutuhkan untuk mencapai target adalah 19,5 detik untuk bergerak dari titik (0,0)cm menuju titik (0,300)cm, sedangkan waktu yang dibutuhkan kontroler *neuro fuzzy* pada percobaan pertama adalah 19,2 detik, percobaan kedua 18,8 detik dan percobaan ketiga 19,0 detik. Untuk pola 2 saat *training* membutuhkan waktu 21,5 detik untuk bergerak dari titik (0,0)cm menuju titik (0,300)cm, sedangkan kontroler *neuro fuzzy* pada percobaan pertama adalah 18,7 detik, percobaan kedua 18,9 detik dan percobaan ketiga 18,7 detik. Untuk perbedaan respon sinyal kontrol *neuro fuzzy* dari tiga kali kali percobaan disebabkan faktor eksternal seperti kondisi kalibrasi sensor dan lingkungan yang dapat mempengaruhi sensor.

5.2 Saran

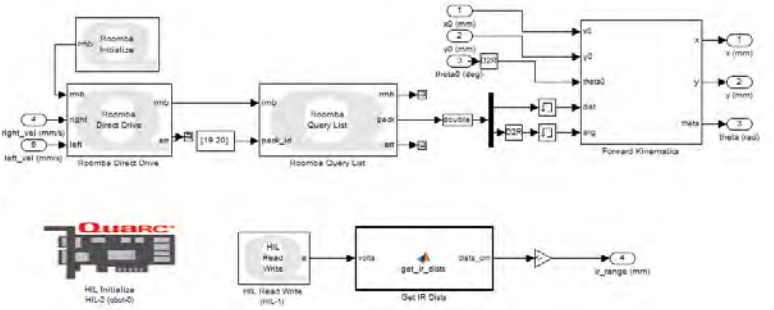
Untuk pengembangan penelitian mengenai Qbot *mobile robot* diharapkan pada penelitian selanjutnya dapat menggunakan metode kurva untuk proses learning *neuro fuzzy*. Selain itu diharapkan dapat mengembangkan pergerakan Qbot *mobile robot* ini menggunakan kontroler lain dengan dilengkapi komponen lain yang mendukung kinerja robot seperti menggunakan kamera opti track untuk proses trayektori.

Halaman ini sengaja dikosongkan

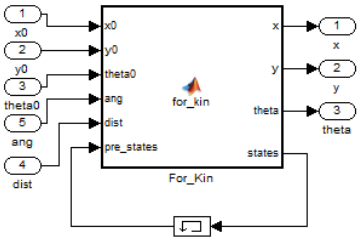
LAMPIRAN A



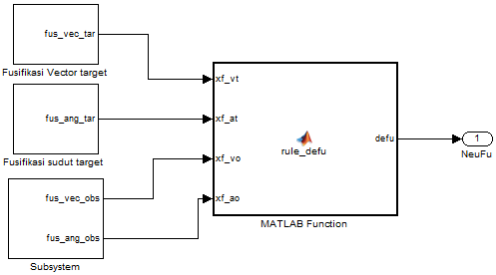
Gambar A1. Blok keseluruhan program



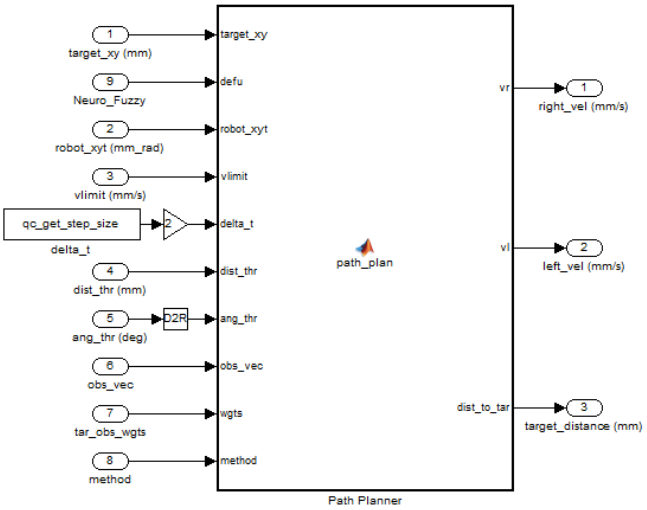
Gambar A2. Blok Qbot



Gambar A3. Blok forward kinematik



Gambar A4. Blok kontroler *neuro fuzzy*



Gambar A5. Blok path planner

LAMPIRAN B

B.1 Program *Forward Kinematics*

```
function [x, y, theta, states] = for_kin(x0, y0, theta0, ang, dist,
pre_states)
% This block supports an embeddable subset of the MATLAB language.
% See the help menu for details.

states = pre_states;

if states(1) == 0
   x = x0;
   y = y0;
   theta = check_angle(theta0);
   states(1) = 1;
else
   theta = check_angle(check_angle(states(4)) + check_angle(ang));
   x = states(2) + dist*cos(theta);
   y = states(3) + dist*sin(theta);
end

states(2) = x;
states(3) = y;
states(4) = theta;

return;

function y = check_angle(x)
y = x;
if y > pi
   y = x - 2*pi;
elseif y < -pi
   y = x + 2*pi;
end
return;
```

B.2 Program Path Planner

```
function [vr, vl, dist_to_tar] = inv_kin(target_xy,defu, robot_xyt, ...
   vlimit, delta_t, dist_thr, ang_thr, obs_vec, wgts, method)
% This block supports an embeddable subset of the MATLAB language.
% See the help menu for details.

rx = robot_xyt(1);
ry = robot_xyt(2);
rtheta = robot_xyt(3);
tx = target_xy(1);
ty = target_xy(2);

dist_to_tar = find_dist(rx, ry, tx, ty);
if dist_to_tar < dist_thr
   vr = int16(0);
   vl = int16(0);
else
   ang_to_tar = find_theta(rx, ry, rtheta, tx, ty);
   obs_vec_abs = abs(obs_vec);
   if obs_vec_abs ~= 0
      tar_vec = complex((tx-rx), (ty-ry));
      tar_vec_abs = abs(tar_vec);
      if tar_vec_abs ~= 0
         tar_vec = tar_vec/tar_vec_abs;
      end
      vec = abs(wgts(1))*tar_vec - abs(wgts(2))*obs_vec;
      ang_to_tar = check_angle(atan2(imag(vec), real(vec)) - rtheta);
   end
   if method == 2
      [vr, vl] = solve_inv_kin(dist_to_tar, ang_to_tar, vlimit,
delta_t,defu);
   else
      [vr, vl] = rotate_and_go(ang_to_tar,defu,vlimit, ang_thr);
%method 1
      if obs_vec_abs ~= 0
         if vr < 0
            vr = int16(20);
         elseif vl < 0
```

```
            vl = int16(20);
        end
    end

  end
end

return;

function dist = find_dist(rx, ry, tx, ty)
dist = sqrt((rx-tx)^2 + (ry-ty)^2);
return;

function theta = find_theta(rx, ry, rtheta, tx, ty)
X = tx - rx;
Y = ty - ry;
theta = atan2(Y, X);
theta = check_angle(theta - check_angle(rtheta));
return;

function [vr, vl] = rotate_and_go(theta,defu,vlimit, ang_thr)

vhn=vlimit(2)*defu;
vln=vlimit(1)*defu;
if abs(theta) > ang_thr
  if theta >= 0
    vr = int16(vhn);
    vl = int16(vln);
  else
    vr = int16(vln);
    vl = int16(vhn);
  end
else
  vr = int16(vhn);
  vl = int16(vhn);
end

return;
```

```
function [vr, vl] = solve_inv_kin(dist,theta, vlimit, delta_t,defu)
d = 252.5;
vmax = vlimit(2);
wmax = (2*vmax)/d;
w = theta/delta_t;
w_sign = sign(w);
if abs(w) > wmax
    w = w_sign*wmax;
end
v = dist/delta_t;
if v > vmax
    v = vmax;
end
vr_tmp = (2*v + d*w)/2;
vl_tmp = 2*v - vr_tmp;

max_of_vrvl = abs(max(vr_tmp, vl_tmp));
if max_of_vrvl > vmax
    vr_tmp = (vr_tmp/max_of_vrvl)*vmax;
    vl_tmp = (vl_tmp/max_of_vrvl)*vmax;
end
vr = int16(vr_tmp*defu);
vl = int16(vl_tmp*defu);


return;


function y = check_angle(x)
y = x;
if y > pi
    y = x - 2*pi;
elseif y < -pi
    y = x + 2*pi;
end
return;
```

B.3 Program Kontroler *Neuro Fuzzy*

```
function defu = rule_defu(xf_vt,xf_at,xf_vo,xf_ao)

Q=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]';

m=0;
for i=1:2
   for j=1:2
      for k=1:2
         for l=1:2
            m=m+1;
            Q(m)=xf_vt(i)*xf_at(j)*xf_vo(k)*xf_ao(l);  %Larsent
            %p=min(xf_vt,xf_at);
            %q=min(xf_vo,xf_ao);
            %Q(m)=min(p,q);              %mamdani
         end
      end
   end
end
%rule
u=[0 0 0 0 0]';
u(1)=Q(1)+Q(2)+Q(3);
u(2)=Q(4)+Q(5)+Q(6);
u(3)=Q(7)+Q(8)+Q(9)+Q(10);
u(4)=Q(11)+Q(12)+Q(13);
u(5)=Q(14)+Q(15)+Q(16);

%normalisasi
su=u(1)+u(2)+u(3)+u(4)+u(5);
for i=1:5
   u(i)=u(i)/su;
end

w=6:10;
atas=w(1)*u(1)+w(2)*u(2)+w(3)*u(3)+w(4)*u(4)+w(5)*u(5);
bawah=u(1)+u(2)+u(3)+u(4)+u(5);
defu=atas/bawah;
```

DAFTAR PUSTAKA

[1] Leotman, Baradiant Ivano (2014). *Desain Kontroler Fuzzy PD untuk Pelacakan Objek pada Qbot Mobile Robot*. Surabaya, Indonesia: Institut Teknologi Sepuluh Nopember.

[2] Inovative Educate, Quanser (2012). *USer Manual Quanser Qbot*. Document number 830.Revision 7.

[3] Khelchandra Thongam, Huang Jie, and Somen Debnath, "Path Planning of Mobile Robot with Neuro Fuzzy", *IEEE/ASME TRANSACTIONS on Mechatronics,* September 2012.

[4] Son Kuswadi, *Kendali Cerdas Teori dan Aplikasi Praktisnya*, edisi pertama. Yogyakarta, Indonesia: Penerbit Andi, 2007.

[5] Pertiwi, Mentari Madi (2015). *Perancangan Kontroler Neuro Fuzzy Untuk Pengaturan Kecepatan Motor Spindle Pada CNC Jenis Milling*. Surabaya, Indonesia: Institut Teknologi Sepuluh Nopember.

[6] Naba, Agus (2009). *Tutorial Cepat dan Mudah Fuzzy Logic dengan MATLAB*. Malang, Indonesia: Deli Publishing.

[7] Inovative Educate, Quanser (2012). *Kinematics Modeling* Document number 830.Revision 7.

RIWAYAT PENULIS

Afrizal adalah nama penulis dan akrab dipanggil bang hungkul. Penulis dilahirkan di Bandung tanggal 27 April 1989, merupakan putra bungsu dari empat bersaudara dari pasangan Nasrul dan Yusnimar. Penulis memulai pendidikan dari TK Ria, SD Negeri Lengkong 105/1 Bandung, SMP Negeri 43 Bandung, dan SMA PGII 1 Bandung. Setelah menyelesaikan pendidikan SMA pada tahun 2008, penulis melanjutkan pendidikan tinggi di Politeknik Manufaktur Negeri Bandung tepatnya pada program studi Teknik Mekatronika dan lulus pada tahun 2011.Stelah lulus penulis sempat bekerja di PT.United Tractors Pandu Engineering selama hampir 3 tahun. Selanjutnya pada tahun 2014 penulis meneruskan pendidikan sarjana di Teknik Elektro ITS, tepatnya pada bidang studi Teknik Sistem Pengaturan. Pada bulan Januari 2016 penulis mengikuti seminar dan ujian Tugas Akhir sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Elektro dari Institut Teknologi Sepuluh Nopember Surabaya.