APLIKASI LOGIKA FUZZY

UNTUK SISTEM KONTROL ROBOTIK

SITI NURMAINI
BAMBANG TUTUKO

UNIVERSITAS SRIWIJAYA

APLIKASI LOGIKA FUZZY PADA SISTEM KONTROL ROBOTIK

Cetakan 1, Oktober 2018

Penulis : Siti Nurmaini, Bambang Tutuko
Desain sampul & Tata letak : Firdaus
Diterbitkan oleh: Unsri Press

Jln. Srijaya Negara, Kampus Unsri
Bukit Besar, Palembang
Telp/Fax
Email

ISBN. 978-979-587-781-2

Pengantar

Assalamu'alaikum Wr. Wb

Bismillahirahmannirrohim,

Atas rahmat dan karunia Allah SWT penulis dapat menyelesaikan penyusunan buku **Aplikasi Logika Fuzzy Pada Sistem Kontrol Robotik.** Materi yang di kaji secara komprehensif dalam buku ini adalah perlakuan analisis dan desain sistem kontrol gerak robot bergerak otonom. Kerangka buku ini ditulis dalam 5 bab. Bab pertama membahas konsep-konsep dari sistem kontrol untuk diaplikasikan pada sistem robotik. Bab kedua membahas robot *mobile* berpenggerak differensial. Bab ketiga membahas tentang pemodelan sistem logika fuzzy dan jenis-jenis teknik logika fuzzy. Bab keempat membahas tentang sistem logika fuzzy pada robot *mobile*. Bab ke lima membahas tentang kinerja pengendali fuzzy dalam penggunaan pada sistem robotik. Keseluruhan dari buku ini menggunakan contoh-contoh yang berasal dari hasil beberapa penelitian yang telah dilakukan yaitu hibah bersaing, hibah PUPT dan hibah unggulan universitas sriwijaya. Diharapkan dapat menjadi referensi dan memberi pengertian kepada pembaca khususnya dosen, peneliti dan mahasiswa tentang materi yang dibahas secara komprehensif.

Buku ini adalah Edisi Pertama, sehingga masih banyak kekurangan di sana-sini. Kritik dan saran serta masukan yang bersifat konstruktif sangat penulis harapkan untuk meningkatkan mutu buku ini, khususnya tentang model sistem robotik ditinjau dari analisis kinematik maupun analisis dinamik yang telah dikembangkan sebelumnya. Tak lupa, penulis sampaikan ucapan terima kasih dan penghargaan yang setinggi-tingginya kepada rekan sejawat di Fakultas Ilmu Komputer, Universitas Sriwijaya atas segala dukungan yang telah diberikan. Selain itu kepada mahasiswa ku yang telah membantu dalam diskusi yaitu, Kemala Dewi, Velia, Kadek, Maylena, Maya dan Chusniah Semoga buku ini memberikan manfaat.

Waalaikumussalam Wr. Wb.

Inderalaya, Universitas Sriwijaya
November, 2018
Tim Penulis

Daftar Isi

Daftar Gambar

Daftar Tabel

BAB 1

Pendahuluan

Beragam penelitian dan pengembangan sebuah sistem kontrol telah menghasilkan banyak produk yang berguna bagi masyarkat luas. Khususnya pengembangan sistem kontrol pada sistem robotik akan sangat berguna bagi pengembangan teknologi di masa depan. Bab ini akan memberikan pandangan secara singkat mengenai pengertian, kegunaaan, dan aplikasi dalam suatu sistem kontrol agar dapat dipahami secara benar sebelum membahas hal-hal yang lebih detail.

1.1. Sistem Kontrol

Sistem kontrol dibutuhkan untuk memperbaiki tanggapan sistem dinamik agar didapat sinyal keluaran seperti yang diinginkan. Sistem kontrol yang baik mempunyai tanggapan yang baik terhadap terhadap sinyal masukan yang beragam. Dalam perancangan sistem kontrol ini diperlukan gambaran tanggapan sistem dengan sinyal masukan dan aksi pengontrolan yang meliputi:

1. Tanggapan sistem terhadap masukan yang dapat berupa fungsi langkah, fungsi undak, fungsi implus atau fungsi lainnya.
2. Kesetabilan sistem yang dirancang.
3. Tanggapan sistem terhadap berbagai jenis aksi pengontrolan.

Dalam sebuah rancangan sistem kontrol harus didapatkan persamaan fungsi alih dari sistem tersebut. Dua jenis sistem kontrol yang sering digunakan adalah sistem kontrol lup tertutup dan sistem kontrol lup terbuka. Masing-masing memiliki kelebihan dan kekurangan tersendiri, dan secara spesifik akan mengendalikan kecepatan sudut dari dua roda pada *mobile* robot.

1.1.1. Sistem Kontrol Lup Terbuka

Suatu sistem yang keluarannya tidak mempunyai pengaruh terhadap aksi kontrol disebut sistem kontrol lup terbuka. Dengan kata lain, sistem kontrol lup terbuka keluarannya tidak dapat dipergunakan sebagai perbandingan umpan balik dengan masukan. Suatu contoh sederhana adalah mesin cuci. Perendaman, pencucian, dan pembilasan dalam mesin cuci dilakukan atas basis waktu. Mesin ini tidak mengatur sinyal keluaran yaitu tingkat kebersihan pakaian.

Dalam suatu sistem kontrol lup terbuka, keluaran tidak dapat dibandingkan dengan masukan acuan. Blok diagram sistem kontrol lup terbuka dapat dilihat pada Gambar 1.1. Sehingga untuk tiap masukan acuan berhubungan dengan kondisi operasi tertentu; sebagai akibat, ketetapan dari sistem tergantung pada kalibrasi. Dengan adanya gangguan, sistem kontrol lup terbuka tidak dapat melaksanakan tugas seperti yang diharapkan. Sistem kontrol lup terbuka dapat digunakan, hanya jika hubungan antara masukan dan keluaran diketahui dan tidak terdapat gangguan internal maupun eksternal.

Gambar 1.1. Blok Diagram Sistem Kontrol Lup Terbuka

Sedangkan kelebihan sistem kontrol Lup terbuka adalah:

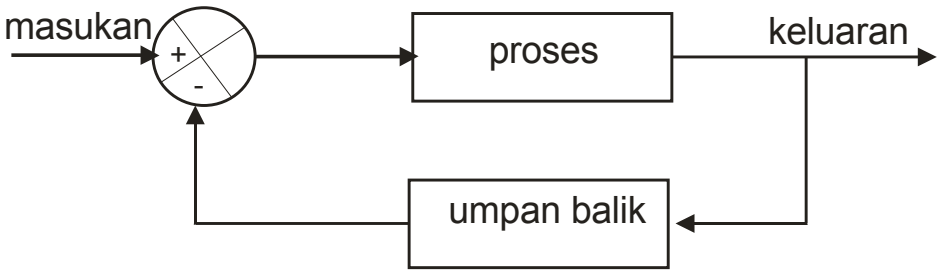
1. Konstruksinya sederhana dan perawatannya mudah.
2. Lebih murah dari pada sistem kontrol lup tertutup.
3. Tidak ada persoalan kestabilan.

Kelemahan sistem kontrol lup terbuka adalah :

1. Gangguan dan perubahan kalibrasi akan menimbulkan kesalahan, sehingga keluaran mungkin berbeda dengan yang diinginkan.
2. Untuk menjaga kualitas yang diperlukan pada keluaran diperlukan kalibrasi ulang dari waktu ke waktu.

1.1.2. Sistem Kontrol Lup Tertutup

Sistem kontrol umpan balik sering kali disebut sebagai sistem kontrol lup tertutup. Praktisnya, istilah kontrol umpan balik dan kontrol lup tertutup dapat saling dipertukarkan penggunaannya. Pada sistem kontrol lup tertutup, sinyal kesalahan yang bekerja, yaitu antara sinyal masukan dan sinyal umpan balik (yang mungkin sinyal keluarannya sendiri atau fungsi dari sinyal keluaran dan turunannya), disajikan ke kontroler sedemikian rupa untuk mengurangi kesalahan dan membawa keluaran sistem ke nilai yang dikehendaki. Istilah kontrol lup tertutup selalu berarti penggunaan aksi kontrol umpan balik untuk mengurangi kesalahan sistem seperti pada Gambar 1.2.



Gambar 1.2. Blok Diagram Sistem Kontrol Lup tertutup

Kelebihan sistem kontrol lup tertutup adalah :

1. Tidak memerlukan kalibrasi ulang dari waktu ke waktu.
2. Dapat digunakan untuk komponen-komponen yang relatif kurang teliti dan murah untuk mendapatkan pengontrolan "plant" yang teliti.
3. Dapat digunakan pada sistem jika terdapat gangguan yang tidak dapat diramalkan pada komponen sistem.

Kelemahan sistem kontrol lup tertutup adalah :

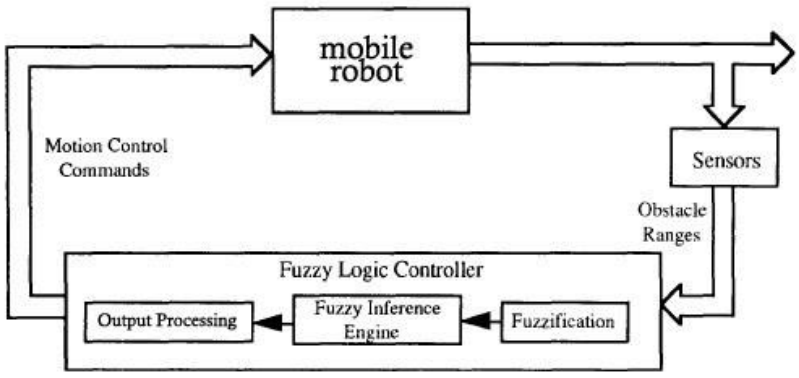
1. Kestabilan selalu merupakan persoalan utama karena cenderung terjadi akibat kesalahan koreksi berlebih yang dapat menimbulkan osilasi pada amplitude konstan maupun berubah.
2. Harga lebih mahal dari pada sistem kontrol lup terbuka.

1.2. Teknik Logika Fuzzy

Sistem logika fuzzy menjadi pendekatan atau metode yang sangat popular untuk pengendali reaksi robot dalam beberapa tahun belakangan ini (Saffiotti, 1997). Logika fuzzy diperkenalkan oleh Lukasiewicz sekitar tahun 1920, dianalisa oleh Max Blanc sekitar tahun 1930, dan Lotfi Zadeh sekitar tahun 1960 menemukan dan menambahkan logika fuzzy pada sistem logika matematis formal. Aturan fuzzy mempersiapkan metode menarik pada memetakan data dari *sensor* kepada pengendali gerakan robot yang berasal dari informasi yang tidak jelas dan tidak lengkap tentang lingkungan sekitarnya.

Kesuksesan dari pengendali dengan logika fuzzy adalah berkontribusi besar pada kemampuan teknologi untuk mengkonversikan deskripsi linguistik secara kuantitatif menjadi fungsi matematis yang kompleks (H. Hagras,2001). Metode dari sistem logika fuzzy dirasakan sangat berguna ketika proses yang dilakukan akan sulit bila menggunakan teknik konvensional (if … then …) atau informasi yang tersedia tidak jelas atau tidak tepat, yang menjadi masalah pada robot pintar bergerak.

Kaidah-kaidah yang digunakan pada sebuah sistem kontrol robotik dengan logika fuzzy bergantung pada perilaku (*behaviour*) dari sistem yang dikendalikan. Penghindar halangan adalah salah satu jenis dari perilaku lokal dari *robot*. Tujuan utamanya untuk menghindari objek-objek lain yang terletak disekitar robot. Secara sederhana dapat direpresentasikan seperti Gambar 1.3, jika teknik logika fuzzy akan digunakan.



Gambar 1.3. Arsitektur SLF Pada Robot Bergerak Otonom

Sebuah sistem kontrol robotik dengan teknik logika fuzzy, tahapan proses nya terdiri atas blok fuzzifikasi, blok basis kaidah dan mekanisme inferensi, dan blok pemrosesan output. Keluaran dari sebuah SLF adalah perintah berupa kontrol gerakan atau yang lainnya, sedangkan masukannya didapat dari berbagai sensor.

1.3. Sistem Kontrol Berbasis Prilaku

Sebuah robot untuk bergerak kearah tujuan yang telah ditetapkan memerlukan suatu pengendali, dengan kinerja yang telah ditetapkan. Sehingga untuk mencapai kinerja tersebut dengan hasil yang memuaskan semua sistem pendukung harus bekerja secara optimal. Sistem sensor harus mendeteksi dengan baik, sistem pemrosesan harus dengan cepat mengatasi setiap perubahan sistem sensor dengan memberikan sinyal keluaran yang baik dan yang terakhir sistem penggerak harus mampu menghasilkan gerak dari hasil pengontrolan secara teliti. Jika hal tersebut dilakukan secara benar dan sesuai dengan mekanisme pengendalian maka kinerja pergerakan yang dihasilkan akan baik, dan begitu juga sebaliknya.

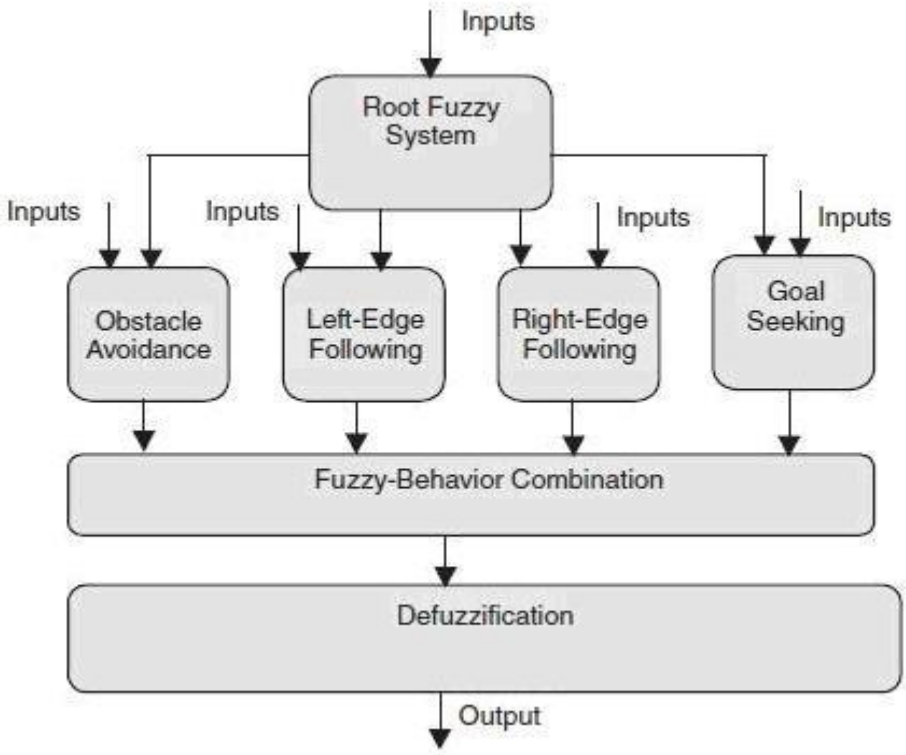
Banyaknya derau pengukuran akan secara teknis akan mempengaruhi sistem control akibatnya keluaran dari sistem penggerak pun menjadi tidak teliti. Metode logika fuzzy sangat baik untuk mengatasi hal tersebut, karena sistem ini bekerja sangat baik pada lingkungan pengendalian yang di penuhi oleh derau pengukuran. Agar sistem kontrol dapat bekerja pada sistem robotic dengan baik, maka sebuah robot harus dibagi terhadap prilaku nya secara modular. Hal tersebut untuk memudah proses pengendalian. Sistem kendali robot berbasis prilaku dapat dijelaskan sebagai berikut:

1.3.1. Sistem Kontrol Berbasis Prilaku

Robot pintar dapat bergerak pada suatu lingkungan harus memiliki 3 perilaku ini yaitu: menghindar halangan, berjalan mengikuti dinding, dan pencari tujuan. Setiap perilaku memiliki fungsi untuk menganalisa lingkungan yang tidak terstruktur dan memberikan masukan kepada sistem pengendali. Beberapa perilaku biasanya sederhana dan mandiri yang terfokus pada tujuan tunggal ketika beroperasi dalam mode reaktif (non-

deliberatif) atau refleksif (memori lebih sedikit). Contohnya termasuk menghindari rintangan yang sederhana dan gerak menuju tujuan sub yang diperintahkan.

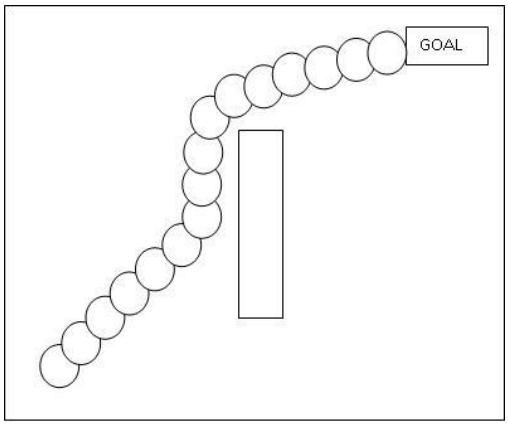
Perilaku primitif melakukan pemetaan dari berbagai sensor yang tersedia dan menjadi masukan bagi aktuator. Saat beroperasi sendirian, masing-masing prilaku tidak efektif untuk melaksanakan tugas-tugas navigasi yang kompleks. Perilaku primitif adalah pembentuk dari perilaku koordinasi yang lebih, disebut sebagai perilaku komposit, seperti perilaku pencari tujuan. Yaitu kemampuan mereka dapat dikombinasikan melalui koordinasi sinergis untuk menghasilkan perilaku-perilaku komposit yang cocok untuk navigasi pergerakan menuju tujuan. Gambar 1.4 di bawah ini secara umum menjelaskan bagaimana pengendali fuzzy menyelesaikan permasalahan pada sistem robotik secara hirarki dengan banyak perilaku (menghindar halangan, mengikuti dinding dan dan pencari target).



Gambar 1.4. Sistem Koordinasi Perilaku

1.3.2. Menghindar Halangan

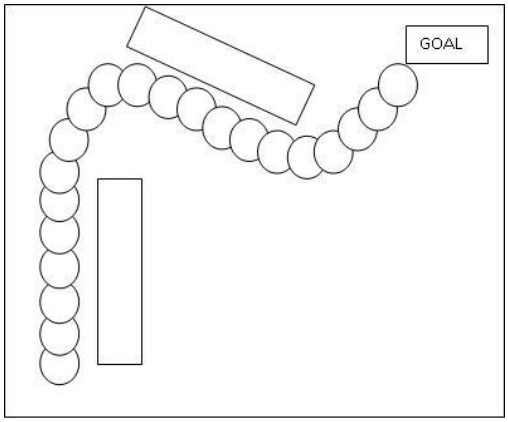
Di dunia nyata, lingkungan yang tidak terprediksi dan memiliki kemungkinan untuk berubah memastikan robot akan menghadapi halangan sebelum mencapai tujuannya. Fakta ini memastikan robot memerlukan perilaku yang dikhususkan untuk membantu menghindari halangan atau rintangan di depan. Perilaku ini akan menganalisa lingkungan di depan dan menghasilkan perintah kepada robot untuk bergerak. Tujuan utama dari perilaku ini adalah untuk menghindari semua penghalang dan menggapai target yang telah diketahui. Gambar 1.5 adalah contoh dari simulasi visual dengan perilaku menghindari halangan.



Gambar 1.5. Simulasi Visual Dengan Perilaku Menghindari Halangan

1.3.3. Berjalan Mengikuti Dinding

Selain halangan, robot harus mengambil keuntungan dari dinding-dinding yang mengelilinginya untuk mencapai target. Terkadang target yang akan dicapai oleh robot berada di ujung dinding sehingga jikalau robot mengikuti dinding tersebut, robot akan mendekati targetnya.

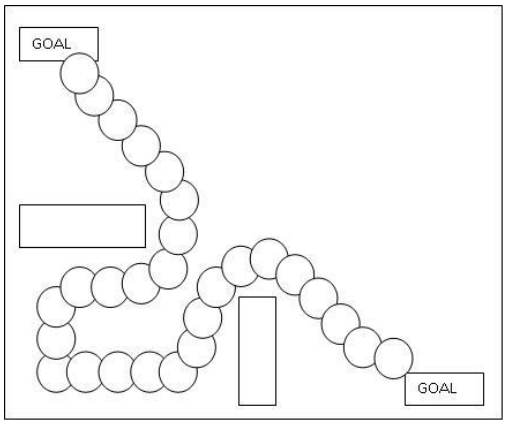


Gambar 1.6. Simulasi Visual Dengan Perilaku Berjalan Mengikuti Dinding

Perilaku berjalan di samping dinding terbagi menjadi 2, yang pertama adalah berjalan mengikuti dinding kiri atau kanan dan yang kedua adalah berjalan di koridor. Tujuan utama perilaku ini adalah mengikuti dinding di samping kanan, kiri, atau dikedua sisi. Gambar 1.6 di atas adalah contoh dari simulasi visual dengan perilaku berjalan mengikuti dinding.

1.3.4. Pencari Tujuan

Robot yang dikembangkan memiliki tujuan atau target yang harus diraih. Akan tetapi masalahnya adalah ketika robot melakukan pergerakan, itu bisa diartikan 2 hal: pergerakan robot tersebut membuatnya semakin dekat dengan target atau sebaliknya. Sehingga, robot harus memiliki perilaku yang menangani masalah ini. Pencari tujuan adalah perilaku robot untuk mencari target yang berada di depan, kiri, kanan, atau dimanapun. Gambar 1.7 berikut adalah contoh simulasi visual dengan perilaku pencari tujuan.



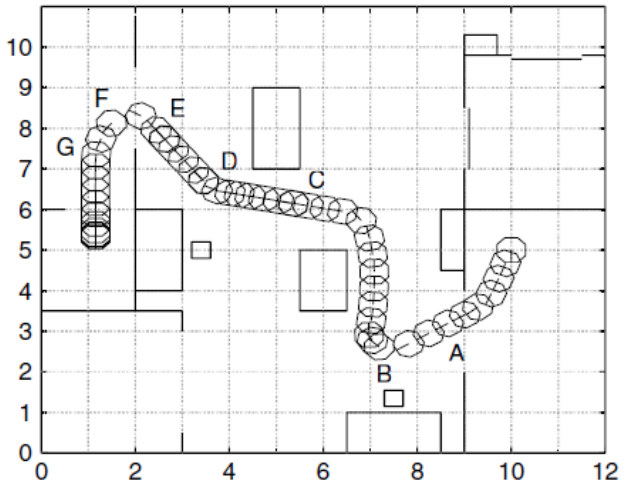
Gambar 1.7. Simulasi visual dengan perilaku pencari tujuan.

1.3.5. Koordinasi Dari Beberapa Perilaku

Merancang robot serba guna, yang mahir berkomunikasi dengan lingkungan yang tidak terstruktur, masih menjadi tantangan di beberapa pusat penelitian Hirai, 1998). Kecerdasaan buatan dan dunia nyata telah diusahakan untuk berjalan bersama selama beberapa tahun belakangan. Robot akan diberikan perkejaan yang memudahkan manusia dan memberikan reaksi terhadap keadaan. Namun ketidakjelasan dan dinamisme yang dihadapi oleh robot di dunia nyata adalah tantangan yang harus diselesaikan. Masalah utama yang harus diselesaikan adalah, koordinasi prilaku dan ketidakpastian lingkungan pergerakan.

Perilaku pada robot dan koordinasinya sangat berkaitan dengan lingkungan disekitarnya dikarenakan alasan robot melakukan setiap pergerakan adalah lingkungan. Perilaku pada robot dapat dibagi menjadi 2 bagian: perilaku prioritas dan perilaku berorientasi hasil. Perilaku berorientasi hasil termasuk perilaku berjalan mengikuti dinding dan perilaku pencari tujuan. Perilaku prioritas dapat pula disebut sebagai perilaku mendesak, misalnya perilaku menghindari halangan (Siripun Thongchai, 2000). Dengan koordinasi, robot akan memilih perilaku berdasarkan apa yang ada dan terjadi disekitarnya. Dapat dicontohkan bila robot berada pada koridor, maka robot akan menggunakan perilaku berjalan di samping dinding untuk menggapai target. Permasalahan lain adalah ketika robot menghadapi lingkungan yang kompleks dan tidak diketahui maka robot harus mengambil keputusan apa perilaku yang akan digunakan untuk mencapai target atau tujuan utama.

Teknik yang dipakai untuk menyelesaikan permasalahan ini adalah logika fuzzy dengan sistem hierarki atau bertingkat. Logika fuzzy adalah teknik penyelesaian matematis untuk menghitung data yang tidak jelas dan masalah yang memiliki solusi yang beragam (Zadeh, 1994) dan perilaku hirarki fuzzy adalah ilmu yang dipakai untuk mengurai masalah-masalah sistem menjadi modular sehingga lebih mudah dikelola dalam bentuk subsistem (Tunstel, 2006). Hasil dari proses ini adalah informasi terbaik untuk membimbing robot mendekati targetnya.



Gambar 1.8. Simulasi dengan perilaku menghindar halangan, berjalan mengikuti dinding, dan pencari tujuan

Simulasi visual dengan lingkungan yang kompleks menyediakan komponen yang beragam berupa halangan, dinding, atau koridor yang berhubungan satu sama lain. Dalam simulasi ini, diharapkan terjadi koordinasi dari seluruh prilaku, yaitu menghindar halangan, berjalan mengikuti dinding, dan pencari tujuan yang direpresentasikan pada Gambar 1.8.

BAB 2

MODEL SISTEM ROBOTIK

2.1. Pendahuluan

Teknologi robot telah diaplikasikan di berbagai aspek seperti industri, perkebunan, pelayan rumah, dan sebagainya (Lin Rui, 2009). Perkembangan yang terbaru saat ini adalah robot dapat menentukan pilihan untuk bergerak atau melakukan tindakan yang lain yang disebut sebagai robot cerdas. Secara natural manusia memiliki batasan, bahwa tidak setiap pekerjaan yang diberikan dapat diselesaikan sempurna seperti menyusun bagian-bagian kecil dari *handphone*, diselesaikan dengan cepat dan melakukannya dengan baik secara konstan. Untuk penjelajahan lingkungan yang berbahaya seperti kutub utara. Hal tersebut dapat dilakukan oleh sebuah robot pintar dengan baik.

Untuk mencapai tujuan yang telah ditetapkan dengan kinerja yang baik, sistem kontrol sangat dibutuhkan oleh robot cerdas. Sistem ini akan memandu robot menyelesaikan pekerjaannya tanpa diganggu oleh lingkungan disekitarnya. Robot akan mengetahui apa yang harus mereka lakukan seperti harus berbelok ke kanan atau ke kiri, berjalan di sebelah dinding, atau apapun untuk membawa robot mendekati tujuannya. Ini dapat pula disebut sebagai sistem pengendali. Sistem pengendali harus mendapatkan masukan berupa data-data dari lingkungan sekitar yang tidak pasti dan memprosesnya untuk mendapatkan informasi yang digunakan untuk memandu robot tersebut. Data-data dari lingkungan didapat dari beberapa pendeteksi (sensor) yang ditanam di dalam robot. Informasi yang didapat oleh robot memungkinkannya untuk memilih langkah apa yang paling tepat menuju target.

Setiap data yang diterima oleh robot pasti diproses oleh sebuah perilaku atau sifat yang digunakan untuk mengendalikan robot. Ini akan menjadi masalah ketika robot berhadapan dengan lingkungan yang tidak jelas dan sensor memberi informasi ke dua atau lebih perilaku. Untuk permasalahan seperti ini, teknik baru untuk koordinasi subsistem dan

sistem pengendali secara keseluruhan sangat dibutuhkan (Tunstel, 2002). Koordinasi perilaku-perilaku tersebut membutuhkan teknik untuk mengeliminasi perilaku yang tidak dibutuhkan dan memilih perilaku yang terbaik untuk memandu robot melakukan pergerakan selanjutnya. Semua informasi yang mungkin akan diproses oleh suatu sistem kontrol berdasarkan perilaku bersama.

Karakteristik dari dunia nyata adalah lingkungan yang bergerak dan tidak terstruktur sehingga setiap tindakan yang robot lakukan adalah untuk mempertahankan pergerakan yang benar. Menyediakan dunia nyata yang fiktif untuk robot yang mendekati dunia nyata dibutuhkan untuk memastikan pergerakan robot ditetapkan oleh lingkungan yang tidak terstruktur. Simulasi visual dapat membantu pengembangan robot pintar bergerak, sehingga robot secara fisik diharapkan dapat bergerak lebih halus setelah menyentuh dunia nyata.

2.2. Sistem Sensor

Sistem sensor adalah suatu peralatan yang digunakan untuk mengukur suatu nilai fisik dan kemudian dijadikan suatu sinyal yang bisa terbaca oleh sistem yang akan dikendalikan. Dalam hubungannya dengan sistem robotik, ada beberapa jenis sensor yang dapat digunakan untuk mengetahui berbagai macam parameter parameter penting yaitu kecepatan, jarak, posisi, dan lain-lain. Sistem sensor yang dapat digunakan bermacam-macam antara lain sensor jarak dan sensor kamera, yang akan dijelaskan berikut ini.

2.2.1. Sensor Jarak

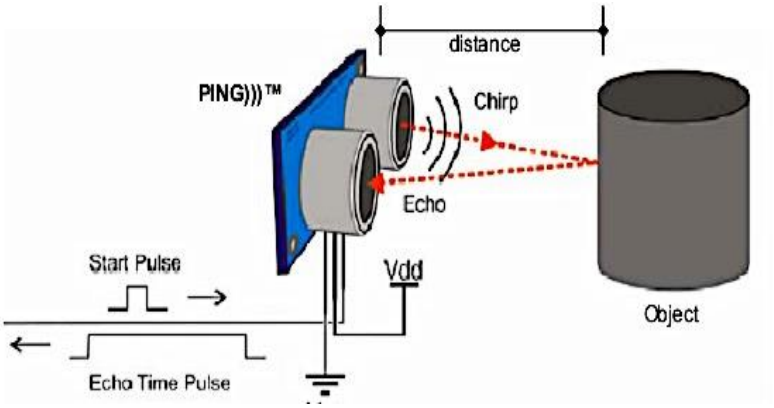
Sensor jarak merupakan sebuah perangkat yang berfungsi untuk menentukan jarak ke suatu objek. Pada robot, sensor jarak biasa digunakan untuk mengetahui keberadaan suatu objek tanpa adanya kontak fisik. Penggunaan perangkat sensor jarak pada sebuah robot mampu menghindarkan robot tersebut dari benturan objek, memandu robot melewati serangkaian objek, dan bahkan melakukan pencitraan terhap lingkungan sekitarnya. Untuk mendeteksi objek, perangkat ini dapat berkerja baik dengan memanfaatkan suara atau cahaya. Sensor dengan metode suara bekerja dengan cara menembakkan gelombang suara

dan kemudian menangkap pantulan suara tersebut untuk dianalisa. Contoh sensor dengan suara adalah sensor ultrasonik dan sonar.

Sensor ultrasonik adalah sebuah sensor yang mengubah besaran fisis (suara) menjadi besaran listrik". Rangkaian sensor ultrasoni terdiri dari Transmitter, Rieceiver, dan komparator. Ultrasonik modul umunya berbentuk papan elektronik ukuran kecil dengan beberapa rangkaian elektronik dan 2 buah transducer. Dari 2 buah transducer ini, salah satu berfungsi sebagai transmitter dan satu lagi sebagai receiver. Ada juga modul yang hanya mempunyai 1 buah transducer, berfungsi sebagai transmitter dan receiver sekaligus.

Sensor ultrasonik bekerja dengan cara menghasilkan gelombang suara pada frekuensi tinggi, yang kemudian dipancarkan oleh bagian transmitter. Pantulan gelombang suara yang mengenai benda di depannya akan ditangkap oleh bagian receiver. Dengan mengetahui lamanya waktu antara dipancarkannya gelombang suara sampai ditangkap kembali, kita dapat menghitung jarak benda yang ada di depan modul tersebut. Lamanya waktu tempuh gelombang suara dikalikan kecepatan suara, kemudian dibagi 2 akan menghasilkan jarak antara ultrasonic modul dengan benda di depannya.

Sensor ultrasonik mendeteksi jarak objek dengan cara memancarkan gelombang ultrasonik (40 KHz) selama t_{BURST} (200 us). Sensor memancarkan gelombang setelah mikrokontroler memberikan pulsa *trigger* dengan t_{OUT} min 2 us. Gelombang ini melalui udara dengan kecepatan 344 m/s, selanjutnya mengenai objek dan memantul kembali ke sensor. Sensor ultrasonik mengeluarkan pulsa *output high* pada pin SIG setelah memancarkan gelombang. Setelah gelombang pantulan terdeteksi sensor, mikrokontroler memberikan *output low* pada pin SIG. Lebar pulsa *high* yang akan merepresentasikan jarak antara sensor dengan objek. Lebar pulsa *high* (t_{IN}) akan sesuai dengan lama waktu tempuh gelombang ultrasonik untuk 2x jarak ukur dengan objek. Maka jarak yang diukur ialah [(t_{IN} s x 344 m/s) : 2] meter. Gambar 2.1. menunjukkan cara kerja sensor ultasonik.



Gambar 2.1. Ilustrasi Cara Kerja Sensor Ultrasonik

Sistem sensor lainnya yang dapat dianggap sensor jarak yaitu sensor Laser. Sistem sensor ini bekerja mengunakan sinyal cahaya. Sensor ini bekerja dengan menembakkan denyut cahaya dan menangkap kembali pantulan cahayanya untuk kemudian dianalisa. Dilihat dari kecepatan rambatannya, sensor cahaya lebih unggul dibanding dengan sensor suara karena cahaya bergerak lebih cepat dan jarak rambatan cahaya lebih jauh dibandingkan dengan sensor berbasis suara. Sensor dengan gelombang cahaya juga memberikan respon yang lebih cepat, resolusi tinggi dan error yang lebih sedikit dibanding dengan sensor berbasis suara. Keunggulan inilah yang membuat sensor dengan gelombang cahaya lebih mahal dibandingkan dengan sensor dengan gelombang suara.

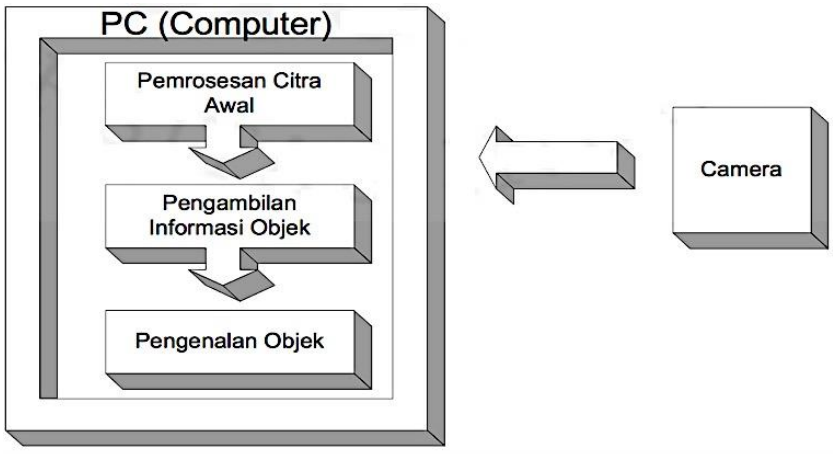
Pengaplikasian sensor laser cukup luas dibanding sensor suara, antara lain dalam militer, pengintai memanfaatkan pemindai jarak yang biasanya dipasang pada senjata atau teropong untuk menentukan jarak terhadap target. Pemindai jarak laser juga digunakan secara luas untuk pengenalan objek 3-D, pemodelan 3-D, dan berbagai area di studi computer visi. Sementara itu, dalam ranah kehutanan (*forestry*) perangkat ini dilengkapi dengan filter anti-daun dan bekerja dengan berbagai pemantul agar pemindaian lebih akurat.

2.2.2. Sensor Kamera

Kamera adalah alat untuk mengakuisisi citra digital. Citra digital yang ditangkap oleh kamera tersebut dapat diproses sehingga didapatkan banyak parameter yang dibutuhkan oleh sistem pengendalian robot, diantaranya parameter posisi, parameter

bentuk, bahkan parameter kecepatan. Kamera adalah alat untuk menangkap gelombang cahaya yang dipantulkan oleh obyek, sehingga obyek tersebut dapat dipetakan dalam dalam bentuk citra. Pada kamera digital citra yang dihasilkan merupakan citra dalam bentuk digital. Secara umum kamera digital terdiri dari lensa, sensor cahaya, dan digitizer seperti yang diperlihatkan pada Gambar 2.2.

Lensa pada kamera berguna untuk memfokuskan cahaya yang diterima sehingga tepat mengenai bidang sensor cahaya. Sensor cahaya yang terdapat pada kamera merupakan kumpulan dari banyak sensor cahaya (phototransistor / photodiode) yang disusun dalam bentuk matrik. Sinyal analog yang dihasilkan oleh sensor lalu dikonversikan ke dalam bentuk digital oleh digitizer sehingga sinyal digital inilah yang dihasilkan oleh kamera digital.



Gambar 2.2 Blok Diagram Sistem Sensor Kamera

Sistem sensor kamera akan menangkap citra yang didalamnya terdapat beberapa objek yang berbeda, yaitu objek mobile robot dan objek penghalang, jika ada. Tujuan sistem ini adalah mengenali masing-masing objek yang ditangkap kamera, selain itu juga untuk mendapatkan informasi koordinat dan sudut kemiringan masing-masing objek terhadap garis acuan. Karena itu, untuk mendapatkan informasi yang diinginkan dari sensor kamera, perlu dilakukan pemrosesan citra. Proses yang dilakukan oleh komputer yang didapat dari kamera dibagi menjadi urutan berikut:

1. Pemrosesan citra awal

2. Pengambilan informasi sudut dan posisi objek, dan
3. Pengenalan objek

Namun, sebelum pemrosesan yang dilakukan oleh komputer dimulai, perlu dilakukan beberapa penyesuaian awal yaitu melakukan perancangan pola dari objek yang akan dideteksi, dan juga perlu dilakukan kalibrasi ukuran antara keadaan sebenarnya dengan citra yang diakuisisi komputer.

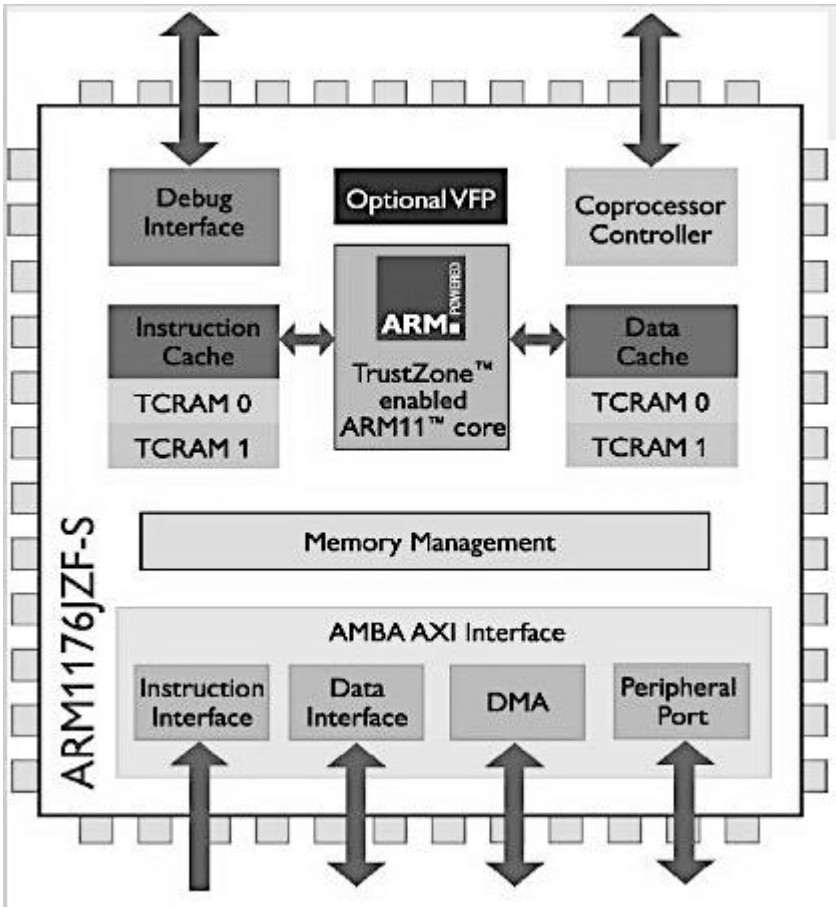
2.3. Sistem Pemrosesan

Di dalam dunia riset, sistem akuisisi data merupakan ujung terdepan dari proses pengumpulan data secara mentah langsung dari sumbernya dimana sistem ini mengkonversi sinyal fisik menjadi sinyal elektronik dan kemudian mendigitalisasi sinyal tersebut sehingga dapat disimpan, ditransmisikan atau disajikan pada display atau komputer. Perkembangan teknologi mikroelektronik yang mendorong berkembangnya mikrokontroler/mikroprosesor saat ini semakin membuka peluang untuk mewujudkan perangkat keras sistem akuisisi yang dapat mengakuisisi lebih dari satu macam besaran. Dalam melakukan pemrosesan sinyal akuisi dapat disimpan serta ditransmisikan melalui komputer dengan menggunakan fasilitas komunikasi serial yang terdapat dalam sebuah chip mikrokontroler. Sistem Akuisisi data ini terdiri dari sensor (yang mengubah besaran fisik menjadi besaran listrik) dan sistem mikrokontroler yang mengolah besaran listrik menjadi kuantitas yang terukur yang berbentuk data digital yang siap diolah atau dianalisis.

Mikroprosessor/mikrokontroler dapat melakukan berbagai macam pekerjaan, yaitu penulisan program sederhana, rangkaian kontrol otomatis. Sistem ini memiliki dua bagian, yaitu perangkat keras dan perangkat lunak. Perangkat keras membahas mulai dari digital dasar, memori CPU, *input-output peripheral* dan rangkaian pendukung lainnya. Sedangkan pada perangkat lunak membahas bahasa komputer, sistem bilangan, dan bahasa mesin *assembly.*

2.3.1. Mikroprosessor ARM

Mikroprosessor ARM ialah komputer mini yang memiliki prosessor berarsitektur ARM dengan sirkuit papan tunggal (*single-board circuit*) dengan ukuran 56 x 85 milimeter yang menyerupai kartu kredit bisa dipergunakan untuk menjalankan beberapa hal seperti permainan komputer, program perkantoran dan juga dapat memutar media. Gambar 2.3 berikut ini merupakan arsitektur berbasis ARM.



Gambar 2.3. Arsitektur Prosessor Broadcom BCM2835 ARM11

2.3.2. Memori

Ada dua jenis memori yang digunakan dalam Raspi yaitu: *dynamic random access memory* (DRAM) dan *Secure Digital* (SD). Versi asli, Model A, dari Raspi memiliki 256 MB RAM yang telah tertanam, sedangkan terbaru, Model B, memiliki 512 MB. DRAM dirancang untuk aplikasi mobile. Ini berarti bahwa walaupun menggunakan tegangan

rendah tetapi tetap menjaga kecepatan clock yang wajar. Memiliki 512 MB DRAM berarti sistem operasi akan berfungsi sangat efisien dan program juga dapat berjalan lancar asalkan mereka dibuat dengan tepat.

Memori *flash* SD digunakan untuk menyimpan sistem operasi, semua program, dan semua data lainnya yang perlu ketelitian. Dengan kata lain, tidak ada yang akan hancur ketika listrik dimatikan. Raspi menggunakan memori *flash* SD dengan cara yang sama seperti halnya PC menggunakan hard drive untuk secara permanen menyimpan data dan program.

2.3.3. Konektor

Raspi memiliki beberapa konektor diantaranya adalah penghubung listrik, *High-Definition Multimedia Interface* (HDMI), video analog komposit, audio, *ethernet*, *Universal Serial Bus* (USB), dan *General Purpose Input / Output* (GPIO) *interface*. Setiap konektor memiliki fungsi khusus masing-masing pada setiap bagiannya. Berikut penjelasan dari pin yang ada pada konektor GPIO dapat dilihat pada Gambar 2.4.

Pin#	NAME		NAME	Pin#
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I2C)		DC Power 5v	04
05	GPIO03 (SCL1 , I2C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I2C ID EEPROM)		(I2C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Gambar 2.4. Datasheet GPIO pin

2.4. Sistem Penggerak

Sebuah robot dapat bergerak ke segala arah untuk mencapai target mengunakan suatu sistem penggerak yang disebut aktuator. Aktuator adalah elemen yang mengkonversikan besaran listrik analog menjadi besaran lainnya misalnya kecepatan putaran dan merupakan perangkat elektromagnetik yang menghasilkan daya gerakan sehingga dapat menghasilkan gerakan pada robot. Untuk meningkatkan tenaga mekanik aktuator ini dapat dipasang sistem *gearbox*. Aktuator dapat melakukan hal tertentu setelah mendapat perintah dari pengendali. Misalnya pada suatu robot pencari cahaya, jika terdapat cahaya, maka sensor akan memberikan informasi pada pengendali yang kemudian akan memerintah pada aktuator untuk bergerak mendekati arah sumber cahaya.

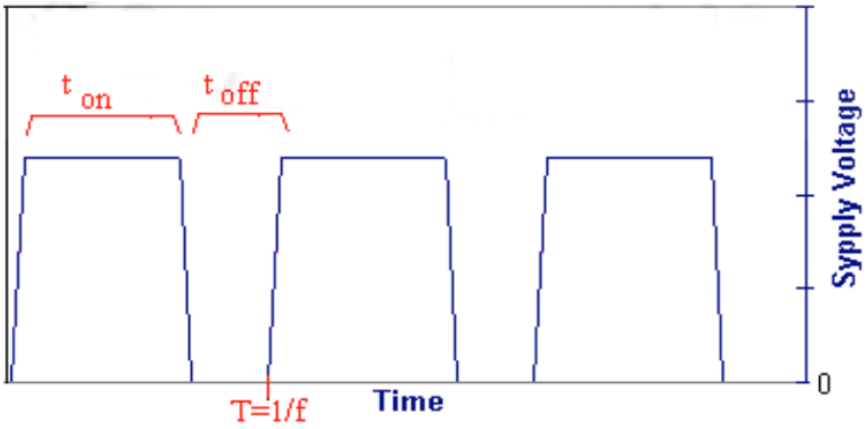
Sebuah aktuator dapat berbentuk rangkaian motor yang menghasilkan energi mekanik, yaitu motor DC, motor DC dengan gear, motor stepper, motor stepper dengan egar dan motor servo. Motor merupakan komponen yang berotasi dan membangkitkan

gerakan. Motor dirancang untuk mengkonversi energi listrik mwnjadi energi mekanik untuk membentuk beberapa pekerjaan fisik. Motor stepper merupakan perangkat elektro-mekanik yang mengkonversi daya listrik kw dalam torque dengan posisi untuk kontrol. Putaran motor stepper dilakukan sesuai namanya, dalam satu putaran atau *revolution* dilakukan dalam beberapa step. Sebagai contoh untuk motor 1.8 derajat mempunyai 200 step/putaran, motor 7.5 derajat memerlukan 48 step/putaran.

Motor DC mempunyai karakteristik yang berbeda dengan motor stepper, dimana putaran yang dihasilkan oleh motor DC dapat beberapa kali putaran dengan catuan power DC, biasanya 10 s/d 24 Vdc sesuai dengan tenaga yang dihasilkan. Biasanya motor DC digunakan sebagai penggerak robot untuk sistem roda (mobil). Sedangkan motor stepper digunakan untuk penggerak robot yang tidak memerlukan putaran penuh, seperti untuk kontrol robot lengan atau robot kaki. Namun dapat diprogram untuk melakukan revolusi sesuai dengan kebutuhan sehingga akan lebih fleksibel dibandingkan dengan motor DC.

Dalam prakteknya diperlukan mekanisme untuk mengendalikan putaran motor, baik ke depan atau ke belakang, cepat atau lambat, lama atau sebentar. Sistem kendali motor berfungsi menerima sinyal dari mikrokontroler dan mengeluarkan sinyal yang digunakan langsung untuk mengerakkan motor. Untuk melakukan hal tersebut, dilakukan pengendalian kecepatan motor menggunakan PWM (Phase Width Modulation) yang kemudian menjadi masukan rangkaian kendali motor.

Metode PWM ini merupakan metode yang menyerupai konsep switching on-off (Gambar 2.5). Jika switching on-off ini dilakukan dengan frekuensi yang lebih cepat dari reaksi motor, maka akan didapatkan kecepatan motor yang stabil pada suatu nilai. Jadi, dengan mengatur perbandingan sinyal on dan off tersebut, bisa didapatkan rata-rata daya yang diinginkan untuk mengatur kecepatan motor. Dengan menambah lamanya sinyal on, akan didapatkan kecepatan motor yang bertambah dikarenakan rata-rata daya yang disuplai bertambah.



Gambar 2.5. Bentuk Sinyal PWM

Untuk mengukur besar PWM, digunakan perhitungan duty cycle. Duty cycle merupakan perbandingan antara waktu menyala (t_{on}) dengan periode PWM tersebut (T), biasa direpresentasikan dalam bentuk persen.

$$Duty\ Cycle = \frac{t_{on}}{T}\ x\ 100\% \qquad (2.1)$$

BAB 3

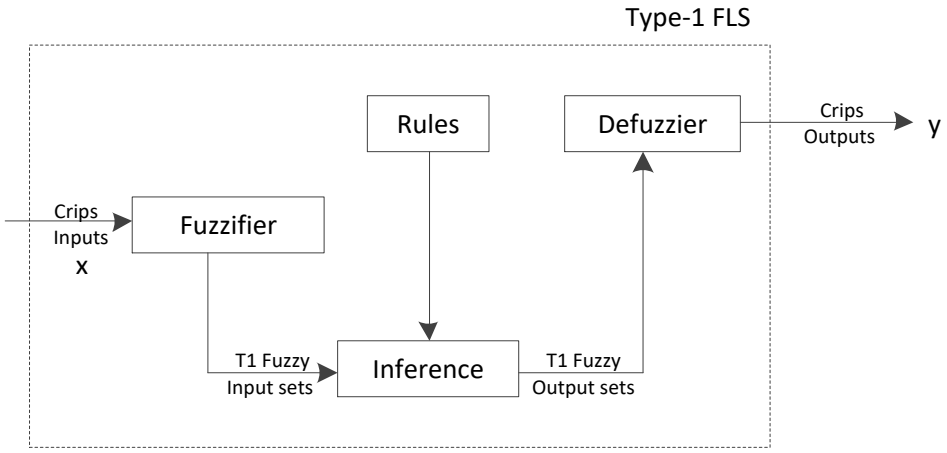
SISTEM LOGIKA FUZZY

3.1. Pendahuluan

Dalam bab ini akan dibahas mengenai sistem logika fuzzy. Pembahasan diawali dari jenis macam-macam teknik logika fuzzy, proses dan yang dilakukan dalam pembangunan perangkat lunak.

3.2. Sistem Logika Fuzzy Tipe-1

Sistem logika *fuzzy* tipe-1 (SLF tipe-1) pertama kali diperkenalkan oleh Lotfi A. Zadeh. Berdasarkan Gambar 3.1, SLF tipe-1 terdiri atas tiga proses utama, fuzzifikasi, penarikan kesimpulan (*inferencing*), dan defuzzifikasi. Setiap proses-proses ini akan dijelaskan kemudian, bersama dengan dasar dari himpunan *fuzzy* tipe-1 dan operator-operatornya.



Gambar 3.1. Sistem Logika *Fuzzy* Tipe-1 (Mendel, 2009)

3.2.1. Himpunan Fuzzy Tipe 1

Pada suatu himpunan *fuzzy*, elemen-elemennya memiliki nilai derajat keanggotaan tertentu, sehingga nilainya didalam himpunan tidak sepenuhnya benar atau salah, tetapi mungkin sebagian benar atau salah dalam derajat tertentu. Derajat keanggotaan ini bernilai antara nol sampai dengan satu. Pemetaan elemen-elemen dari suatu himpunan *fuzzy* ke derajat keanggotaannya dinyatakan dalam fungsi keanggotaan, oleh karena itu, sebuah himpunan *fuzzy* dinyatakan dalam bentuk fungsi keanggotaannya. Sebuah himpunan *fuzzy* A pada semesta pembicaraan X didefinisikan sebagai,

$$\mu_A(x): X \rightarrow [0,1] \tag{3.1}$$

dimana μA melambangkan fungsi keanggotaan dari himpunan fuzzy tersebut.

Domain dari suatu himpunan fuzzy dapat bernilai diskrit atau kontinu. Adanya perbedaan domain membuat representasi dari sebuah himpunan fuzzy berbeda-beda. Apabila X adalah domain kontinu, maka A dinyatakan sebagai,

$$A = \int_x \mu_A(x)/x \tag{3.2}$$

sedangkan apabila X adalah domain diskrit, maka:

$$A = \sum_{x_i} \in x\, \mu_A(x_i)/x_i \tag{3.3}$$

Representasi diskrit dari suatu himpunan fuzzy digunakan apabila pemrosesan dilakukan oleh komputer (Coupland, 2003).

3.2.2. Operasi Himpunan Fuzz*y* Tipe-1

Himpunan Bagian

Sebuah himpunan *fuzzy* A dapat dinyatakan sebagai himpunan bagian dari himpunan *fuzzy* B jika dan hanya jika $\mu_A(x) \leq \mu_B(x)$ untuk setiap nilai x. Merujuk pada (Castilo dan Melin, 2008) dapat dinyatakan sebagai,

$$A \in B \leftrightarrow \mu_A(x) \leq \mu_B(B) \tag{3.4}$$

Gabungan (*union*)

Himpunan *fuzzy* C adalah gabungan dari himpunan *fuzzy* A dan B dituliskan sebagai C = A ∪ B atau C = A OR B. C memiliki fungsi keanggotaan yang berkorelasi dengan A dan B, yang didefinisikan sebagai berikut merujuk pada (Castilo dan Melin, 2008):

$$\mu_C(x) = \max\big(\mu_A(x), \mu_B(x)\big) = \mu_A(x) \cup \mu_B(x) \tag{3.5}$$

Irisan (intersection)

Himpunan *fuzzy* C adalah irisan dari himpunan *fuzzy* A dan B dituliskan sebagai C = A ∩ B atau C = A AND B. C memiliki fungsi keanggotaan yang berkorelasi dengan A dan B, yang didefinisikan sebagai berikut merujuk pada (Castilo dan Melin, 2008):

$$\mu_C(x) = \min\big(\mu_A(x), \mu_B(x)\big) = \mu_A(x) \cap \mu_B(x) \tag{3.6}$$

Komplemen

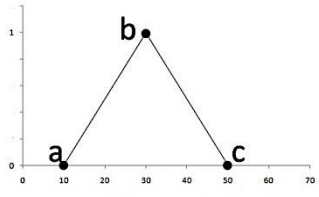
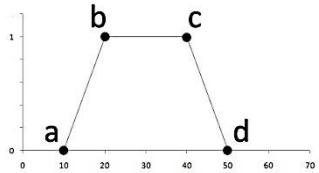
Komplemen dari himpunan *fuzzy* A yang dituliskan sebagai $\overline{A}$(¬A, NOT A), didefinisikan sebagai berikut merujuk pada (Castilo dan Melin, 2008):

$$\mu_{\overline{A}}(x) = 1 - \mu_A(x) \tag{3.7}$$

3.2.3. Proses Fuzzifikasi

Fuzzifikasi adalah proses pemetaan nilai masukkan (*crisp input*) kedalam himpunan fungsi keanggotaan *fuzzy*. Tujuan dari proses fuzzifikasi adalah untuk menerjemahkan nilai *crisp input* kedalam rules. Proses ini melibatkan penggunaan himpunan *fuzzy* tipe-1, dan proses pemetaan nilai tegas dari masukan ke dalam himpunan *fuzzy* menggunakan fungsi keanggotaan. Fungsi keanggotaan tipe-1 adalah fungsi yang merepresentasikan himpunan *fuzzy* tipe-1. Nilai masukan dipetakan kedalam derajat keanggotaan menggunakan fungsi ini. Fungsi keanggotaan SLF tipe-1 yang telah digunakan pada banyak penelitian dijelaskan pada Tabel 3.1 berikut ini,

Tabel 3.1 Pernyataan Matematis dan Ilustrasi Model *Fuzzy*

Fungsi Keanggotaan	**Pernyataan matematis**	**Ilustrasi**	***Pseudocode***
Fungsi keanggotaan segitiga	$$\text{triangle}(x; a, b, c) \begin{cases} 0, x \le a \\ \frac{x-a}{b-a}, a < x \le b \\ \frac{c-x}{c-b}, b < x \le c \\ 0, c < x \end{cases}$$		`if x ≤ a then return 0 else if x > a and x ≤ b return (x-a) /(b-a) else if x > b and x ≤ c return (c-x) /(c-b) else if x > c return 0`
Fungsi keanggotaan trapezium	$$\text{trapezoid}(x; a, b, c, d) \begin{cases} 0, x \le a \\ \frac{x-a}{b-a}, a < x \le b \\ 1, b < x \le c \\ \frac{d-x}{d-c}, c < x \le d \\ 0, d < x \end{cases}$$		`if x ≤ a then return 0 else if x > a and x ≤ b return (x-a) /(b-a) else if x > b and x ≤ c return 1 else if x > c and x ≤ d return (d-x) /(d-c) else if x > d return 0`

3.2.4. Fuzzy Inferensi Sistem Tipe-1

Sebuah *Fuzzy Inference System* (FIS), adalah sistem pengambilan keputusan pada konsep logika fuzzy, dimana derajat keanggotaan yang dihasilkan dari proses fuzzifikasi digabungkan berdasarkan aturan tertentu dan kemudian, kaidah-kaidah yang aktif dipotongkan ke himpunan kesimpulan (*rule firing*). Terdapat lebih dari satu metode yang dapat digunakan dalam sebuah FIS antara lain Sugeno, Mamdani, Takagi-Sugeno, tetapi yang paling banyak digunakan adalah metode inferensi Mamdani (Negnevitsky, 2005 :106). Selanjutnya yang akan dijelaskan adalah hanya metode *inference* Mamdani.

3.2.5. Kaidah Fuzzy

Kaidah atau *rule*, adalah pernyataan linguistik yang menyatakan keputusan apa yang harus diambil dalam keadaan tertentu. Sebuah kaidah berbentuk pernyataan IF-THEN seperti pada persamaan (3.8) berikut ini,

$$IF\ x_i\ is\ A_i\ THEN\ y\ is\ B_i \tag{3.8}$$

Dimana x dan y adalah variabel linguistik, dan A dan B adalah nilai linguistik. Bagian awal dari kaidah (bagian IF) disebuat sebagai *antecedent* atau *premise*, sedangkan bagian lainnya (bagian THEN) disebut sebagai *consequent* atau *conclusion*.

3.2.6. Komposisi Relasi *Fuzzy*

Komposisi relasi *fuzzy* adalah sebuah operator khusus yang digunakan untuk menggabungkan dua relasi *fuzzy* menjadi sebuah himpunan *fuzzy* yang baru.

a. Komposisi Max-Min

Dimisalkan R_1 dan R_2 adalah relasi *fuzzy* pada X x Y dan Y x Z. Komposisi max-min dari R1 ∘ R2 didefinisikan sebagai merujuk pada (Castilo dan Melin, 2008):

$$R_1.R_2 \max = \left\{\left[(x,y), \underset{y}{max} \min(\mu_{R_1}(x,y), \mu_{R_2}(y,z))\right] | x \in X, y \in Y, z \in Z\right\} \tag{3.9}$$

Merujuk pada (Castilo dan Melin, 2008) dapat pula didefinisikan sebagai,

$$\mu_{R_1 \circ R_2}(x,y) = \underset{y}{max} \min\left[(\mu_{R_1}(x,y), \mu_{R_2}(y,z))\right] \cup_y \left[(\mu_{R_1}(x,y) \cap \mu_{R_2}(y,z))\right] \tag{3.10}$$

Dengan simbol ∪ melambangkan Max dan simbol ∩ melambangkan Min.

b. Komposisi Max-Product

Dimisalkan R_1 dan R_2 adalah relasi *fuzzy* pada X x Y dan Y x Z. Komposisi max-min dari R1 ∘ R2 didefinisikan sebagai:

$$R_1 \circ R_2 = \{[(x,z), \max_y [\mu_{R_1}(x,y)\mu_{R_2}(y,z)] \quad (3.11)$$

3.2.7. Penalaran

Proses penalaran adalah proses untuk mendapatkan kesimpulan dari pernyataan-pernyataan yang ada. Teorema dasar yang digunakan dalam mendapatkan kesimpulan dari dua pernyataan adalah modus ponens. Secara umum, merujuk pada (Castilo dan Melin, 2008) modus ponens dituliskan sebagai berikut,

Premise 1	: x is A
Premise 2	: if x is A then y is B
Consequence	: y is B

Dapat dilihat bahwa *consequence* diambil berdasarkan nilai kebenaran pada *premise* 1. Pada proses penalaran, modus ponens digunakan secara pendekatan, yang artinya bahwa apabila terdapat suatu kaidah yang mengimplikasikan *premise* yang sama, consequence yang didapatkan dari kaidah tersebut akan mengimplikasikan hal yang sama dari aturan yang ada. Merujuk pada (Castilo dan Melin, 2008) contohnya adalah,

Premise 1	: x is A’
Premise 2	: if x is A then y is B
Consequence	: y is B’

Sebuah *consequence* yang didapat dengan penalaran dengan cara pendekatan disebut sebagai Generalized Modus Ponens (GMP).

Untuk mendefinisikan penalaran *fuzzy*, dimisalkan A, A’ dan B adalah himpunan *fuzzy* pada X, X dan Y, dengan A → B adalah relasi R pada X x Y. Himpunan *fuzzy* B’ yang digambarkan pada contoh sebelumnya didefinisikan sebagai:

$$\mu_{B'}(y) = \max_{x} min[\mu_{A'}(x), \mu_R(x,y)] \quad (3.12)$$
$$= \cup_x [\mu_{A'}(x) \cap \mu_R(x,y)]$$

Dimana $\mu_{A'}(x)$ dinyatakan di dalam *fuzzy* sebagai *antecedent* dan $\mu_{B'}(y)$ dinyatakan di dalam *fuzzy* sebagai *consequence.* Sedangkan $\mu_R(x,y)$ adalah relasi dari x dan y. Ataupun dapat didefinisikan sama dengan,

$$B' = A' \circ R = A'(A \rightarrow B) \quad (3.13)$$

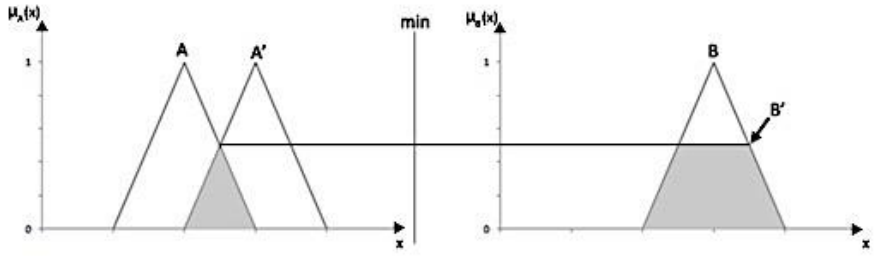
Terdapat beberapa komposisi antara banyaknya kaidah dan *antecedent* dalam suatu sistem inferensi. Kaidah tersebut meliputi kaidah tunggal dengan antecedent tunggal, kaidah jamak dengan antecedent tunggal dan kaidah jamak dengan antecedent jamak.

3.2.7.1. Kaidah Tunggal Dengan *Antecedent* Tunggal

Mendapatkan kesimpulan menggunakan GMP dalam sistem dengan sebuah kaidah dan sebuah *antecedent* dapat dilakukan dengan menggunakan penyederhanaan dari persamaan (3.10), yaitu:

$$\mu_{B'}(y) = \left[\cup_x [\mu_{A'}(x) \cap \mu_A(y)]\right] \cap \mu_B \quad (3.14)$$

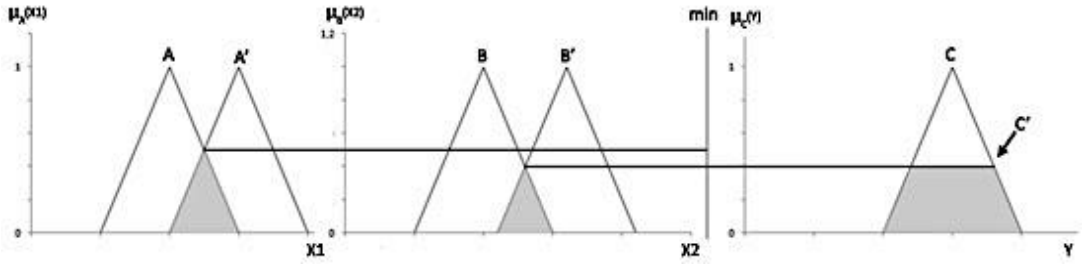
Dari persamaan (3.14), dapat dilihat bahwa proses penalaran dilakukan dengan menghitung nilai maksimum dari $\mu_{A'}(x) \cap \mu_A(x)$. Fungsi keanggotaan dari himpunan *consequent* B' dapat didefinisikan dengan memotong nilai dari B menggunakan nilai yang telah dihitung.



Gambar 3.2 Penalaran GMP pada kaidah tunggal dan *antecedent* tunggal
(Castilo dan Melin, 2008)

3.2.7.2. Kaidah Jamak Dengan *Antecedent* Tunggal

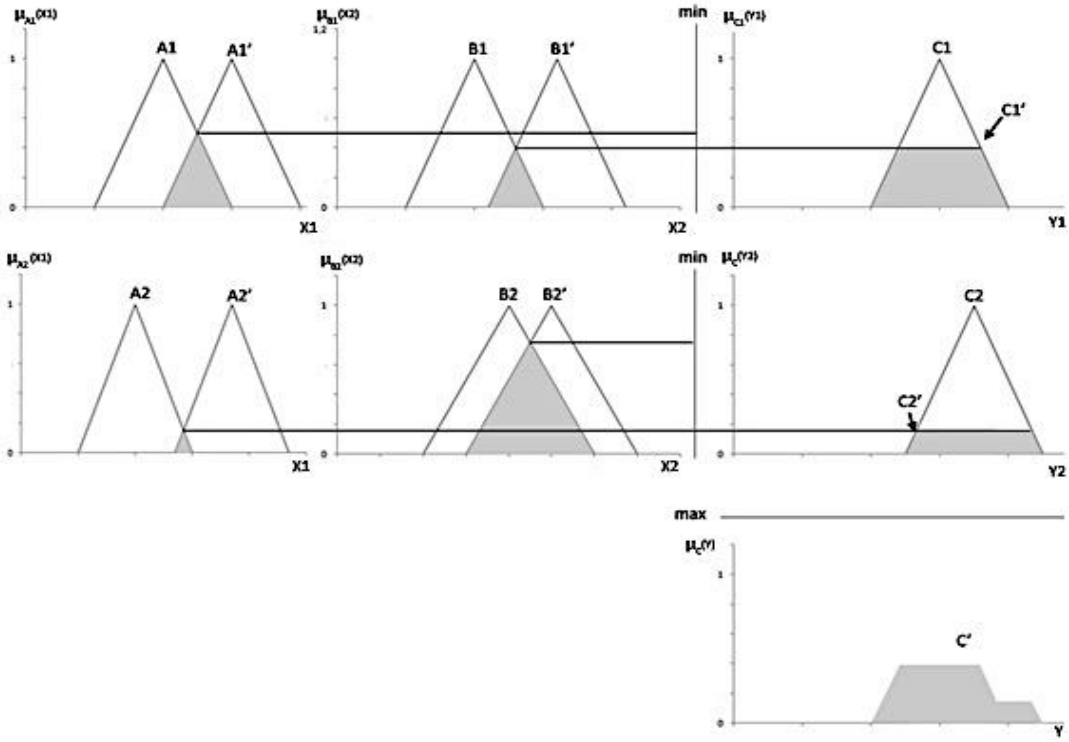
Mendapatkan kesimpulan dari sistem yang memiliki sebuah rule dengan *antecedent* jamak hampir sama dengan pengambilan kesimpulan pada sistem dengan kaidah tunggal dengan *antecedent* tunggal (Gambar 3.3). Perbedaannya adalah, pada saat pengambilan keputusan, nilai minimum dari *antecedent* dihitung terlebih dahulu. Nilai inilah yang kemudian memotong himpunan kesimpulan pada kaidah tersebut.



Gambar 3.3 Penalaran GMP Pada Kaidah Tunggal Dengan *Antecedent* Jamak

3.2.7.3. Kaidah Jamak Dengan *Antecedent* Jamak

Pengambilan keputusan pada sistem dengan kaidah jamak dengan *antecedent* jamak dilakukan dengan menghitung nilai terkecil dari tiap-tiap *antecedent* dan kemudian dipotongkan pada himpunan kesimpulan pada kaidah yang terkait (Gambar 3.4.). Himpunan-himpunan kesimpulan yang didapatkan ini kemudian digabungkan menjadi suatu himpunan kesimpulan dengan menggunakan operator max.



Gambar 3.4 Penalaran GMP Pada Kaidah Jamak Dengan *Antecedent* Jamak

3.2.8. Defuzzifikasi

Proses defuzzifikasi adalah proses terakhir pada SLF tipe-1. Ini merupakan proses mengubah nilai dari himpunan kesimpulan yang dihasilkan pada proses sebelumnya ke dalam nilai tegas yang kemudian menjadi keluaran dari SLF tipe-1. Defuzzifikasi dapat dilakukan dalam berbagai cara, tetapi teknik defuzzifikasi centroid adalah teknik yang paling banyak digunakan (Negnevitsky, 2005:111). Ide dasar dari teknik ini adalah untuk menemukan pusat gravitasi dari daerah *consequent* yang dihasilkan pada proses penalaran. Pusat gravitasi adalah titik dimana apabila daerah tersebut dibagi dua berdasarkan titik itu, hasil pembagiannya memiliki luasan yang sama. Untuk menghitung pusat gravitasi dari sebuah daerah digunakan persamaan (3.15) berikut,

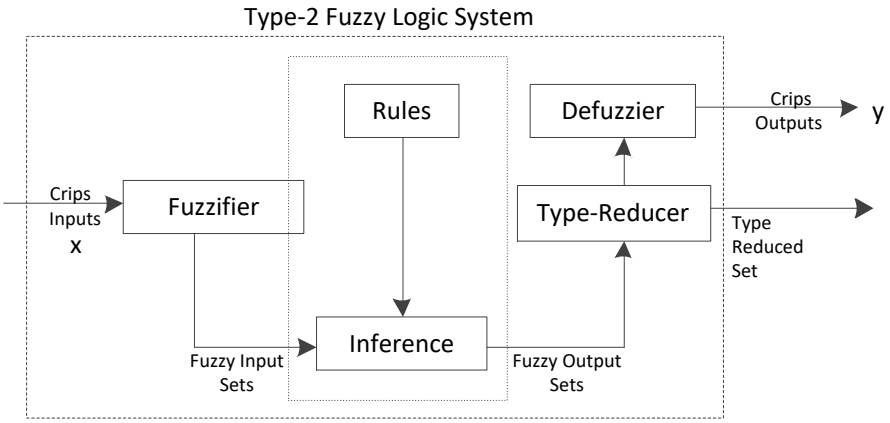
$$COG = \frac{\int_b^a \mu_A(x) x dx}{\int_b^a \mu_A(x) dx} \tag{3.15}$$

dimana A adalah suatu himpunan fuzzy dan *a b* adalah interval yang dicakupi oleh A. Nilai perkiraan dari persamaan (3.15) juga dapat diperoleh dengan membagi daerah A menjadi beberapa titik sampling. Menggunakan metode diskritisasi ini, pusat gravitasi dari A dapat dihitung berdasarkan persamaan (3.16) berikut,

$$COG = \frac{\sum_{x=a}^{b} \mu_A(x)x}{\sum_{x=a}^{b} \mu_A(x)} \tag{3.16}$$

3.3. Sistem Logika Fuzzy Tipe-2

SLF tipe-2 adalah penyempurnaan perhitungan ketidakpastian dalam proses logika fuzzy secara keseluruhan. SLF tipe-2 memiliki kemampuan untuk menangani ketidakpastian yang tidak dimiliki oleh SLF tipe-1. Blok pemrosesan SLF tipe-2 secara umum digambarkan pada Gambar 3.5.



Gambar 3.5. Blok Pemrosesan SLF Tipe-2 (Karnik et. al., 1999)

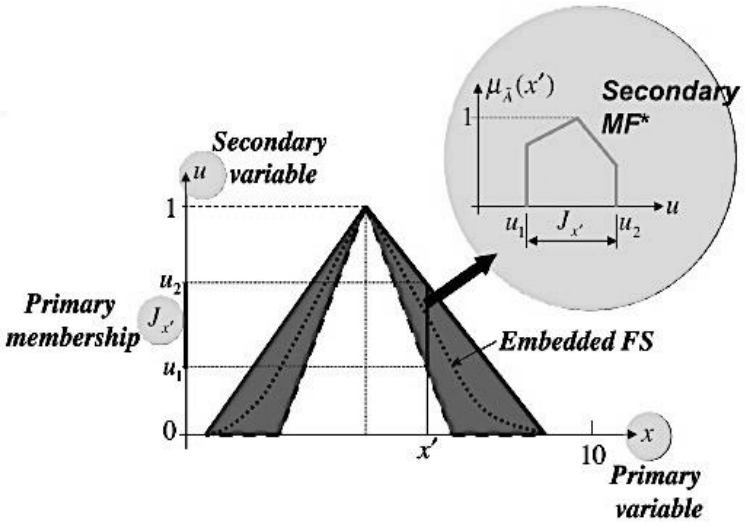
Blok pemrosesan dari SLF tipe-2 mirip dengan SLF tipe-1, pembeda utama dari kedua sistem ini adalah pada blok pemrosesan keluaran (*output processing*), dimana SLF tipe-2 memiliki proses reduksi tipe (*type reducer*) sebelum defuzzifikasi. Blok ini akan mengubah nilai fuzzy tipe-2 ke tipe-1 dan selanjutnya ke nilai crisp.

3.3.1. Himpunan *Fuzzy* Interval Tipe-2

Sebuah himpunan fuzzy interval tipe-2 adalah himpunan fuzzy tipe-2 yang memiliki ciri khas khusus. Sebuah himpunan fuzzy tipe-2 adalah himpunan fuzzy yang

fungsi keanggotaannya adalah daerah ketidakpastian. Nilai-nilai ketidakpastian ini didapatkan dengan tidak hanya mengambil satu nilai tetap sebagai titik mula dan akhir (*endpoint*) dari fungsi keanggotaan, tetapi juga mengambil nilai-nilai lain yang dimungkinkan untuk mewakili linguistik yang sama. Memiliki titik mula dan akhir yang lebih dari satu, sebuah input x dapat memiliki beberapa derajat keanggotaan berbeda yang berasal dari fungsi keanggotaan yang berbeda-beda pula.

Pada himpunan fuzzy tipe-2, derajat keanggotaan ini disebut sebagai derajat keanggotaan primer. Pada tiap-tiap fungsi keanggotaan, diberikan sebuah nilai bobot (*weight*). Nilai bobot ini diberikan berdasarkan nilai kemungkinan pada tiap-tiap fungsi keanggotaan. Gabungan dari tiap-tiap bobot ini kemudian membentuk derajat keanggotaan sekunder dari himpunan. Derajat keanggotaaan sekunder menambah sebuah dimensi pada himpunan fuzzy tipe-2, menjadi fungsi keanggotaan berdimensi 3. Reprensentasi himpunan fuzzy tipe-2 dalam SLF dapat dilihat seperti Gambar 3.6.



Gambar 3.6 Himpunan *Fuzzy* Tipe-2 (Mendel, 2009)

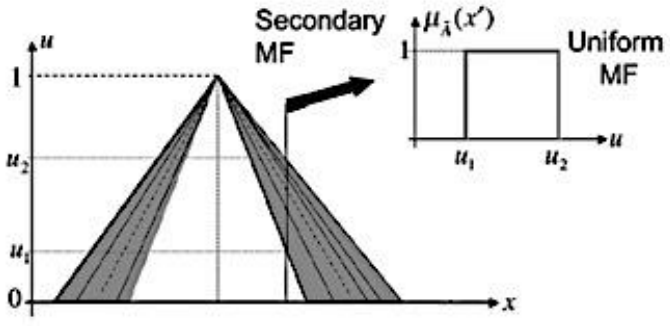
Untuk tiap-tiap *x* bagian dari semesta pembicaraan *X*, Himpunan fuzzy tipe-2 $\tilde{A}$ didefinisikan sebagai berikut,

$$\tilde{A} = \int_{x \in X} \frac{\mu_{\tilde{A}}(x)}{x} \qquad (3.17)$$

dimana:

$$\mu_{\tilde{A}}(x) = \int_u f_x(u)/u \, , u \in J \subseteq [0,1] \qquad (3.18)$$

Apabila nilai-nilai ketidakpastian dari fungsi keanggotaan himpunan *fuzzy* tipe-2 dinyatakan sebagai sebuah daerah yang dibatasi dua buah garis, sebuah di tepatkan pada nilai-nilai tertinggi, dan yang lain pada nilai-nilai terendah. Daerah ini disebut sebagai *Footprint of Uncertainties* (FOU), garis yang berada pada nilai-nilai tertinggi disebut sebagai *Upper Membership Function* (UMF), dan yang berada pada nilai-nilai terendah disebut sebagai *Lower Membership Function* (LMF). Sebuah himpunan *fuzzy* interval tipe-2 adalah himpunan *fuzzy* yang ditandai dengan FOU. Nilai FOU dibatasi dengan LMF dan UMF, dan nilai derajat keanggotaan sekundernya sama dengan 1 (*uniform*), seperti yang direpresentasikan pada Gambar 3.7.



Gambar 3.7 Himpunan Fuzz*y* Interval Tipe-2 (Karnik &Mendel, 2009)

Karena nilai derajat keanggotaan sekunder pada himpunan *fuzzy* interval tipe-2 sama dengan tipe-1, persamaan (3.15) kemudian menjadi,

$$\mu_{\tilde{A}}(x) = \int_u 1/u \, , u \in J \subseteq [0,1] \qquad (3.19)$$

3.3.2. Operasi Pada Himpunan Fuzzy Interval Tipe-2

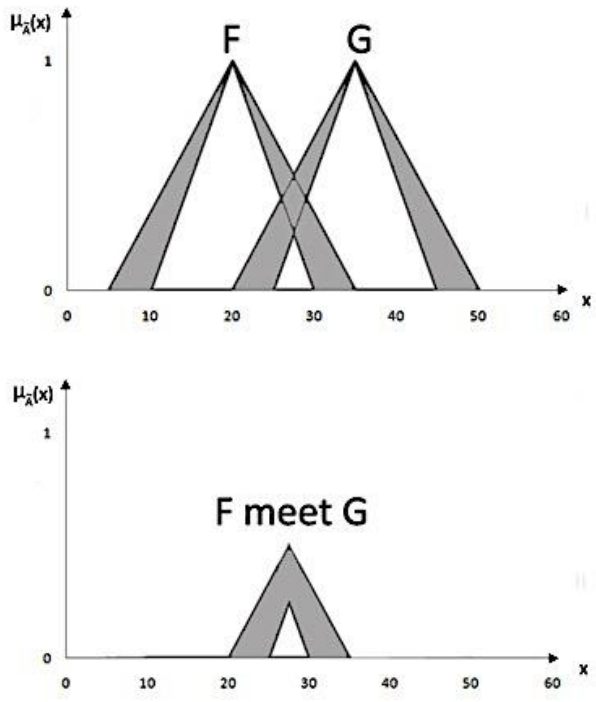
Operasi-operasi matematis yang dilakukan dalam proses fuzzifikasi, inferensi dan defuzzifikasi pada SLF interval tipe-2 ada dua macam yaitu meet (⊓) atau irisan, dan join (⊔) atau gabungan. Kedua operasi matematis tersebut dapat dijelaskan sebagai berikut:

3.3.2.1. Meet/Intersection

Operasi matematis ini berkaitan dengan operasi matenatis irisan pada himpunan fuzzy tipe-1 dan dilambangkan dengan notasi ⊓. Dimisalkan F = $\int_{v\in F} 1/v$ dan G = $\int_{w\in G} 1/w$, dengan v ∈ [l_f,r_f] dan w ∈ [l_f,r_f]. Hasil dari operasi meet dari F dan G adalah Q = F ⊓ G, atau dalam definisi t-norm dinyatakan sebagai,

$$Q=F \sqcap G = \int_{q\in[l_f * l_g, r_f * r_g]} \frac{1}{q} \qquad (3.20)$$

dimana q = v * w.



Gambar 3.8 Operasi Meet Pada Himpunan *Fuzzy* Interval Tipe-2
(Karnik &Mendel, 2009)

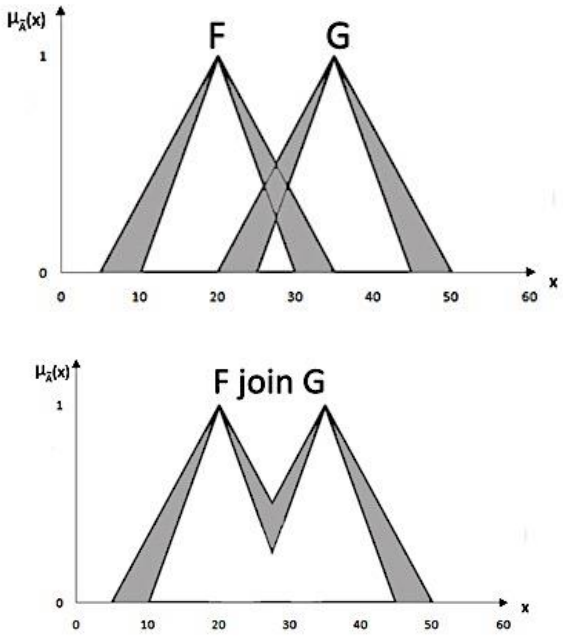
3.3.2.2. Join/Union

Operasi matematis ini berkaitan dengan operasi matematis gabungan pada himpunan fuzzy tipe-1, dan dilambangkan dengan notasi ⊔. Dimisalkan F = $\int_{v\in F} 1/v$ dan

$G = \int_{w \in G} 1/w$, dengan $v \in [l_f, r_f]$ dan $w \in [l_f, r_f]$. Hasil dari operasi join dari F dan G adalah $Q = F \sqcup G$, atau dalam definisi t-norm dinyatakan sebagai:

$$Q = F \sqcup G = \int_{q \in [l_f \vee l_g, r_f \vee \vee r_g]} \frac{1}{q} \tag{3.21}$$

dimana $q = v \vee w$.



Gambar 3.9 Operasi Join Pada Himpunan *Fuzzy* Interval Tipe-2
(Karnik & Mendel, 2009)

3.3.3. Fuzzifikasi

Proses Fuzzifikasi pada himpunan fuzzy interval tipe-2 adalah proses memetakan besaran tegas menjadi nilai keanggotaan pada himpunan fuzz*y* interval tipe-2. Fungsi keanggotaan dari himpunan fuzzy interval tipe-2 ditentukan oleh batas atas dan batas bawah dari FOU, yaitu LMF dan UMF. UMF dan LMF adalah himpunan fuzzy tipe-1 yang berada tepat pada nilai-nilai tertinggi dan terendah dari FOU. UMF dilambangkan dengan garis diatas $\overline{\mu}$, sedangkan LMF dilambangkan dengan garis dibawah $\underline{\mu}$. Sehingga, derajat keanggotaan dari x pada himpunan fuzz*y* interval tipe-2 A^I dilambangkan sebagai,

$$\mu_{A^I}(x) = \int_{a \in \left[\underline{\mu}_{A^I}(x), \overline{\mu}_{A^I}(x)\right]} \frac{1}{a} \qquad (3.22)$$

3.3.4. Fuzzy Inference System pada Fuzzy Interval Tipe-2

Tidak banyak perbedaan antara FIS pada fuzzy tipe-1 dan fuzzy tipe-2, keduanya adalah proses pengambilan keputusan dari *antecedent* yang nilainya didapat dari proses sebelumnya. Perbedaannya adalah, terdapat dua nilai yang dihasilkan dari fuzzifikasi, yaitu derajat keanggotaan pada UMF dan LMF. Terdapat dua macam komposisi pada SLF interval tipe-2, yaitu:

a. Komposisi Extended Sup – Star

Proses ini mirip dengan komposisi max-min pada SLF tipe-1. Langkah pertama pada proses ini adalah memperoleh *firing set*, yaitu himpunan nilai-nilai yang digunakan untuk memotong himpunan kesimpulan. Hasil potongan pada himpunan kesimpulan kemudian digabungkan dengan menentukan nilai-nilai maksimum dari setiap *firing set*, menghasilkan *fired set* pada *consequent*. Untuk Himpunan fuzzy interval tipe-2, *fired set* $\underline{f}^I = [\underline{f}^I, \overline{f}^I]$ dari sistem dengan *p* kaidah, menggunakan komposisi extended sup-star didefinisikan sebagai:

$$\underline{f}^I = \sup_{x \in X} \int_{X_1} \dots \int_{X_p} \left[\underline{\mu}_{\underline{\tilde{X}}_1}(x_1) * \underline{\mu}_{\underline{\tilde{F}}_k^I}(x_1)\right] * \dots * \left[\underline{\mu}_{\underline{\tilde{X}}_1}(x_p) * \underline{\mu}_{\underline{\tilde{F}}_k^I}(x_p)\right] / x \qquad (3.23)$$

dan

$$\overline{f}^I = \sup_{x \in X} \int_{X_1} \dots \int_{X_p} \left[\overline{\mu}_{\tilde{X}_1}(x_1) * \overline{\mu}_{\tilde{F}_k^I}(x_1)\right] * \dots * \left[\overline{\mu}_{\tilde{X}_1}(x_p) * \overline{\mu}_{\tilde{F}_k^I}(x_p)\right] - x \qquad (3.24)$$

b. Komposisi Produk

Pada proses ini, proses penghitungan *firing set* dan komposisinya dilakukan dengan melakukan operasi produk (pengali) dari tiap nilai-nilai derajat keanggotaan, atau didefinisikan sebagai:

$$\underline{f}^{l} = \underline{\mu}_{\tilde{F}_1^l}(x_1) * \ldots * \underline{\mu}_{\tilde{F}_p^l}(x_p) \tag{3.25}$$

dan

$$\overline{f}^{l} = \overline{\mu}_{\tilde{F}_1^l}(x_1) * \ldots * \overline{\mu}_{\tilde{F}_p^l}(x_p) \tag{3.26}$$

3.3.5. Proses Defuzzifikasi

Blok pemprosesan keluaran terdiri atas dua proses, yaitu reduksi tipe dan defuzzifikasi. Reduksi tipe bertujuan untuk mengubah himpunan fuzzy tipe-2 yang dihasilkan pada proses sebelumnya menjadi himpunan fuzz*y* tipe-1. Himpunan yang telah direduksi tipe kemudian didefuzzifikasi menjadi nilai tegas yang merupakan keluaran dari SLF interval tipe-2.

3.3.5.1. Reduksi Tipe

Reduksi tipe adalah suatu proses yang khusus dimiliki oleh SLF tipe-2. Ini adalah proses mereduksi himpunan fuzzy tipe-2 menjadi himpunan-himpunan fuzzy tipe-1. Terdapat tiga macam reduksi tipe yang dapat digunakan dalam sebuah SLF tipe-2, yaitu:

Reduksi Tipe *Centroid*

Pereduksi tipe *centroid* mengkomposisikan semua himpunan kesimpulan dengan menghitung nilai gabungannya. Pereduksi tipe *centroid* didefinisikan sebagai:

$$Y_c(x) = \int_{\theta_1} \ldots \int_{\theta_N} \left[\mu_{D_1}(\theta_1) * \ldots * \mu_{D_N}(\theta_N)\right] \Big/ \frac{\sum_{i=1}^{N} y_i\theta_i}{\sum_{i=1}^{N} \theta_i} \tag{3.27}$$

Dimana $D_i = \mu_{\tilde{B}}(y_i)$ dan $\theta_i \in D_i$.

Pada SLF interval tipe-2 persamaan (3.27) kemudian menjadi,

$$Y_c(x) = \int_{\theta_1} \ldots \int_{\theta_N} 1 \Big/ \frac{\sum_{i=1}^{N} y_i\theta_i}{\sum_{i=1}^{N} \theta_i} \tag{3.28}$$

Reduksi Tipe *Height*

Pereduksi tipe *height* menggantikan himpunan keluaran dari tiap kaidah dengan singleton yang berada pada nilai maksimum dari himpunan keluaran dan kemudian menghitung *centroid* dari himpunan fuzzy tipe-1 yang terdiri atas singleton-singleton ini. Pereduksi tipe *height* didefinisikan sebagai:

$$Y_h(x) = \int_{\theta_1} \cdots \int_{\theta_M} \left[\mu_{D_1}(\theta_1) * \ldots * \mu_{D_M}(\theta_M)\right] \Big/ \frac{\sum_{l=1}^{N} \bar{y}^l \theta_l}{\sum_{l=1}^{N} \theta_l} \tag{3.29}$$

Untuk fuzzy interval tipe-2, persamaan diturunkan menjadi,

$$Y_h(x) = \int_{\theta_1} \cdots \int_{\theta_M} 1 \Big/ \frac{\sum_{l=1}^{N} \bar{y}^l \theta_l}{\sum_{l=1}^{N} \theta_l} \tag{3.30}$$

Reduksi Tipe *Center of Sets*

Pereduksi tipe ini menggantikan setiap himpunan keluaran dengan singleton yang berada pada centroidnya dan kemudian menghitung *centroid* dari himpunan tipe-1 yang terdiri atas singleton-singleton ini. Pereduksi tipe *center of sets* (COS) didefinisikan sebagai:

$$Y_{cos}(x) = \int_{y_l \in Y_i} \cdots \int_{y_N \in Y_i} \int_{f_1 \in F^i(x)} \cdots \int_{f_N \in F^i(x)} \left[1 \Big/ \frac{\sum_{i=1}^{N} f_i y_i}{\sum_{i=1}^{N} f_i} \right] \tag{3.31}$$

Dimana $f^l(x) = [\underline{f}^I, \overline{f}^I]$ dan $y_i = [y_l^i, y_r^i]$ nilai singleton yang menggantikan himpunan keluaran. Untuk melakukan reduksi tipe ini, nilai-nilai *centroid* dari himpunan fuzzy dari keluaran harus dihitung terlebih dahulu menggunakan:

$$c_{\tilde{A}} = \int_{\theta_1} \cdots \int_{\theta_N} \left[\mu_{D_1}(\theta_1) * \ldots * \mu_{D_N}(\theta_N)\right] \Big/ \frac{\sum_{i=1}^{N} x_i \theta_i}{\sum_{i=1}^{N} \theta_i} \tag{3.32}$$

dimana $D_i = \mu_{\tilde{A}}(x_i)$, dan $\theta_i \in D_i$

3.3.5.2. Algoritma Reduksi Tipe Karnik-Mendel

Telah disebutkan ketiga jenis pereduksi tipe yang dapat digunakan pada SLF interval tipe-2. Namun, untuk menemukan seluruh komposisi dari himpunan fuzzy tipe-1 dari reduksi tipe pada domain yang kontinu adalah langkah yang tidak praktis (Mendel 2007). Karnik dan Mendel (1999) menyatakan bahwa untuk mendapatkan nilai pendekatan dari himpunan tereduksi hanya bergantung pada nilai paling kiri *(y_l)* dan paling kanan *(y_r)* dari himpunan tereduksi. Di dalam pernyataannya, Karnik dan Mendel juga memperkenalkan algoritma iteratif yang dapat digunakan untuk menemukan nilai-nilai tersebut.

Pada tahun 2000, Karnik dan Mendel memperkenalkan suatu algoritma iteratif yang dapat digunakan untuk mereduksi tipe dari himpunan fuzzy interval tipe-2. Proses reduksi tipe pada sistem logika fuzzy tipe-2 interval tersebut diawali dengan mengasumsikan bahwa $y^i{}_r$ dan $y^i{}_l$ (i = 1 ... M, dengan M adalah jumlah kaidah) diurutkan secara menaik, algoritma karnik-mendel untuk menghitung y_r dapat dideskripsikan sebagai berikut (Dongrui, 2013):

1. Inisialisasi nilai f_n

   Hitung semua nilai f_n dengan (3.33)

   $$f^n = \frac{\underline{f}^n + \overline{f}^n}{2}, n = 1,2,3,..,N \tag{3.33}$$

   Dan hitung y_r dengan persamaan (3.34),

   $$y' = \frac{\sum_{n=1}^{N} \overline{y}^n f^n}{\sum_{n=1}^{N} f^n} \tag{3.34}$$

   Nilai dari y' berbeda-beda, tergantung pada reduksi tipe yang digunakan.

2. Menentukkan $k(1 \leq k \leq N-1)$ agar:

   $$y_k{}^r \leq y \leq y_{k+1}{}^r \tag{3.35}$$

3. Set:

   $$f^n = \begin{cases} \underline{f}^n, n \leq k \\ \overline{f}^n, n > k \end{cases} \tag{3.36}$$

   Dan menghitung y_r dengan persamaan (3.37),

$$y_r = \frac{\sum_{n=1}^{N} \overline{y}^{\,n} f^n}{\sum_{n=1}^{N} f^n} \tag{3.37}$$

4. Periksa apakah nilai y' sama dengan y_r
   a. Apabila keduanya bernilai sama, maka $y' = y_r$
   b. Apabila tidak sama, ubah nilai $y' = y_r$ dan ulangi langkah 2 sampai y_r ditemukan.

Sedangkan langkah-langkah untuk menentukan y_l adalah sebagai berikut:

1. Inisialisasi nilai f_n

   Hitung semua nilai f_n dengan persamaan (3.38),

$$f^n = \frac{\underline{f}^{\,n} + \overline{f}^{\,n}}{2}, n = 1,2,3,..,N \tag{3.38}$$

   Dan hitung y' dengan persamaan (3.39),

$$y' = \frac{\sum_{n=1}^{N} \underline{y}^{\,n} f^n}{\sum_{n=1}^{N} f^n} \tag{3.39}$$

2. Menentukkan $k(1 \leq k \leq N-1)$ agar :

$$y_{k^l} \leq y \leq y_{k+1^l} \tag{3.40}$$

3. Set:

$$f^n = \begin{cases} \overline{f}^{\,n}, n \leq k \\ \underline{f}^{\,n}, n > k \end{cases} \tag{3.41}$$

   Dan menghitung y_l dengan persamaan (3.42),

$$y_l = \frac{\sum_{n=1}^{N} \underline{y}^{\,n} f^n}{\sum_{n=1}^{N} f^n} \tag{3.42}$$

4. Periksa apakah nilai y' sama dengan y_l

a. Apabila keduanya bernilai sama, maka $y' = y_l$
b. Apabila tidak sama, ubah nilai $y' = y_l$ dan ulangi langkah 2 sampai y_l ditemukan.

3.3.5.3. Defuzzifikasi

Nilai tegas dari suatu himpunan tereduksi dapat dihitung dengan merata-rata nilai paling kanan dan paling kiri dari himpunan tereduksi, atau dapat dinyatakan sebagai:

$$y = \frac{y_l + y_r}{2} \tag{3.43}$$

Dimana y_l merupakan hasil perhitungan untuk linguistik kiri dan y_r adalah hasil perhitungan untuk linguistik kanan.

BAB 4

Pemodelan Sistem Robotik

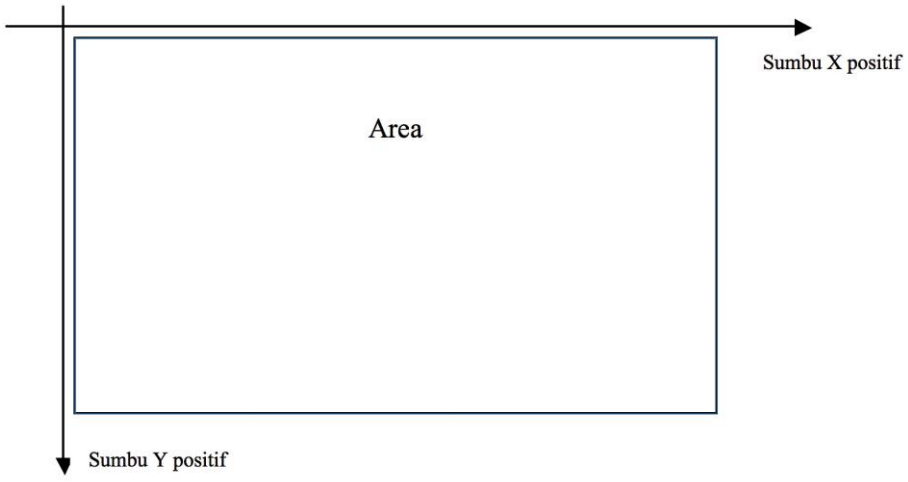
Dalam simulasi yang dilakukan, pengujian ini akan membahas mengenai desain sistem yang dibangun, seperti *setup* sistem, desain lingkungan kerja, desain model robot, desain sensor, desain parameter input, output, dan export data.

4.1. Model Objek Kendali

Pada sub-bab berikut akan membahas analisis mengenai data-data model yang diperlukan untuk membangun sistem secara keseluruhan. Bahasan akan dijelaskan dalam bentuk pemodelan visual sesuai dalam bentuk simulasi. Pemodelan visual diperlukan untuk menggambarkan sesuatu secara nyata dalam bahasa pemrograman. Semua pemodelan sistem akan menjadi dasar dalam pembangunan sistem koordinasi ini.

4.2.1. Model Area

Area yang digunakan pada simulasi ini dimodelkan sebagai bidang kartesian dengan 2 dimensi (absis = x, ordinat = y) dan di dalam area terdapat hambatan berbentuk kotak dan robot berbentuk lingkaran yang masing-masing memiliki koordinat penanda letak pada area. Model area dapat dilihat pada Gambar 4.1.

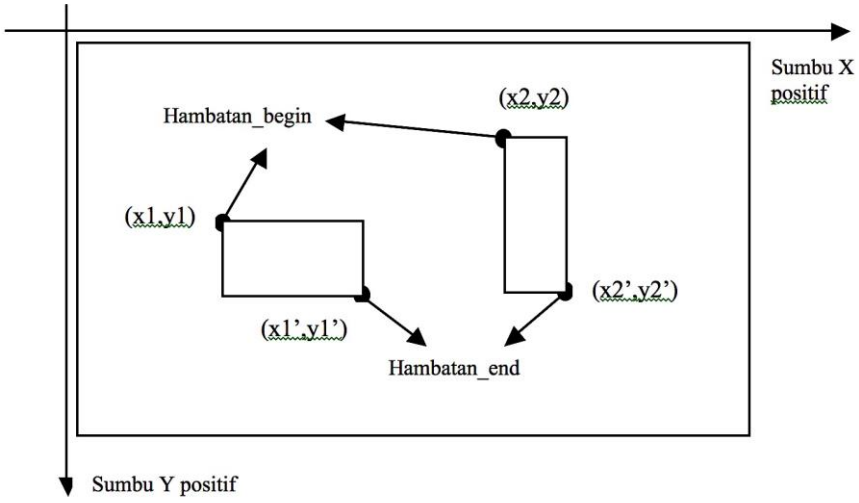


Gambar 4.1. Model Area

Pemodelan area digunakan untuk membandingkan besar area pada simulator dengan besar lingkungan pada dunia nyata. Skala pada model area ini adalah 1 pixel: 1.4 cm dan berlaku pada semua objek-objek yang berada pada simulator robot ini.

4.2.2. Model Hambatan

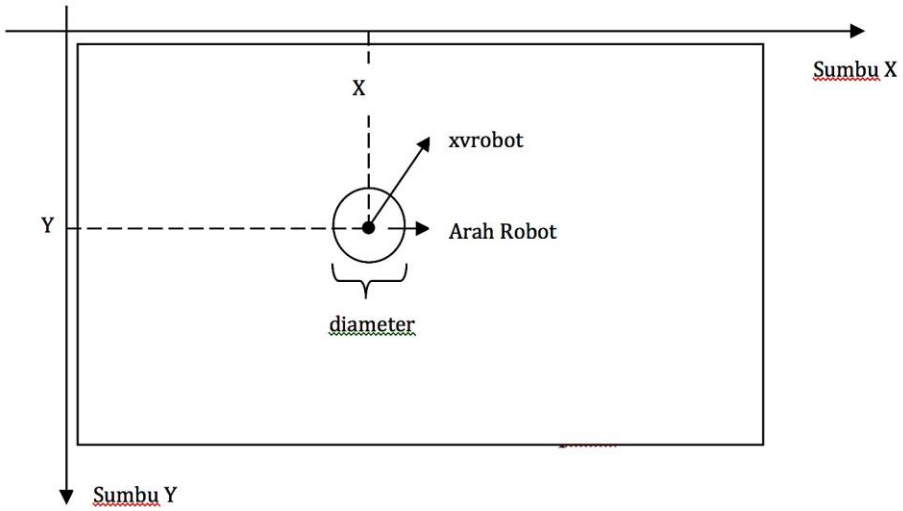
Hambatan pada simulasi ini adalah bidang persegi atau persegi panjang yang memiliki koordinat penanda letak di area. Besar bidang dari hambatan ditentukan oleh koordinat awal dan koordinat akhir di setiap ujung bidang. Jumlah dan letak hambatan pada simulasi ini statis namun besar persegi panjang berbeda. Model hambatan dapat dilihat pada Gambar 4.2. Hambatan_awal dan Hambatan_akhir masing-masing memiliki koordinat *X* dan *Y* yang menjadi penanda letak pada area.



Gambar 4.2. Model Hambatan

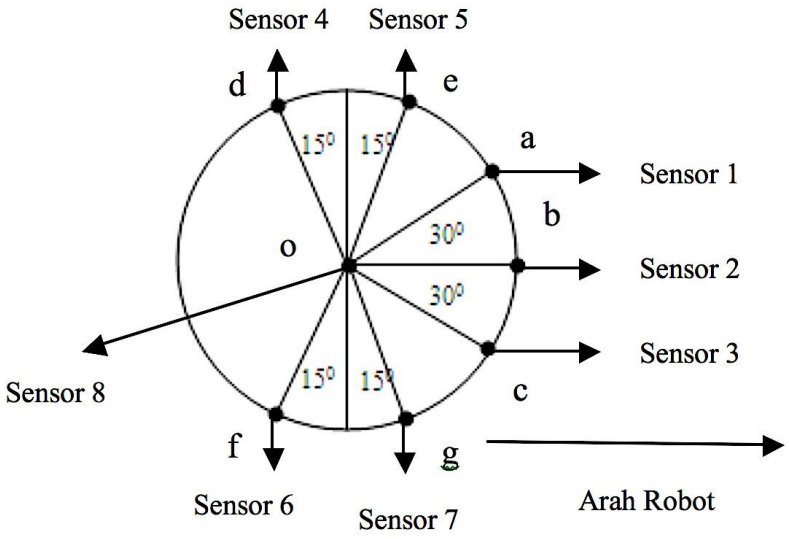
4.2.3. Model Robot

Pada simulasi ini robot digambarkan dengan bidang lingkaran yang membutuhkan nilai kecepatan dan sudut untuk bergerak. Diameter pada bidang lingkaran ditentukan berdasarkan lebar robot. koordinat robot akan menentukan letak robot pada area, ditandai dengan titik pusat pada bidang area tersebut. Gambar 4.3 memperlihatkan letak robot pada area.



Gambar 4.3. Model Robot Pada Area

Model robot memiliki 8 sensor yang digunakan oleh 3 perilaku yaitu menghindar halangan, mengikuti dinding, dan pencari target. Sensor-sensor tersebut masing-masing diletakan pada bagian depan, samping, dan atas. Gambar 4.4, memperlihatkan letak masing-masing sensor.

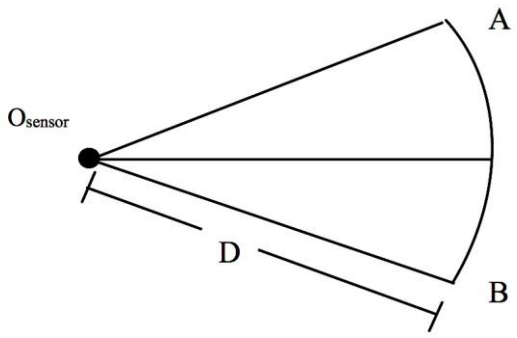


Gambar 4.4. Model Robot Berdasarkan Posisi Sensor

Pada Gambar 4.4, setiap sensor memiliki jarak rotasi terhadap titik b sebagai arah robot. Sensor 2 berada tepat di titik b sehingga tidak memiliki jarak rotasi. Sensor 1 dan 3 berada pada -30^0 dan $+30^0$ dari titik b, ditandai dengan sudut *aob* dan sudut *aob*. Sensor 5 dan 7 berada pada -75^0 dan $+75^0$ dari titik b, ditandai dengan sudut *eob* dan sudut *gob*. Sensor 4 dan 6 berada pada -105^0 dan $+105^0$ dari titik b, ditandai dengan sudut *dob* dan sudut *fob*. Sensor 8 berada pada titik o titik tengah robot.

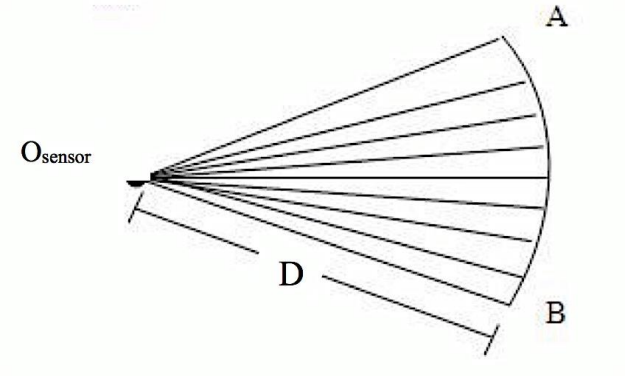
4.2.4. Sensor Pendeteksi Jarak

Sensor pendeteksi jarak yang digunakan para robot digambarkan sebagai titik pada area dan letak dari setiap sensor tersebut mengikuti pergerakan robot. Koordinat setiap sensor ditentukan dengan perhitungan besar sudut pemisah dari arah robot. Sensor memiliki jarak baca maksimum (D) dan besar bentang bacaan ($AO_{sensor}B$) yang akan menentukan jarak antara halangan dengan sensor (J). Gambar 4.5 menjelaskan pemodelan sensor pendeteksi jarak.



Gambar 4.5. Model Sensor Pendeteksi Jarak

Seperti yang diperlihatkan pada Gambar 4.5, sensor memiliki jarak baca maksimum (D) dan besar bentang bacaan ($AO_{sensor}B$) dan bentang sensor dapat digambarkan menjadi lebih dari satu garis dari titik OA. Hal tersebut di rancang sesuai dengan karakteristik sistem sensor yang diangkat sebagai model sistem pendeteksi.

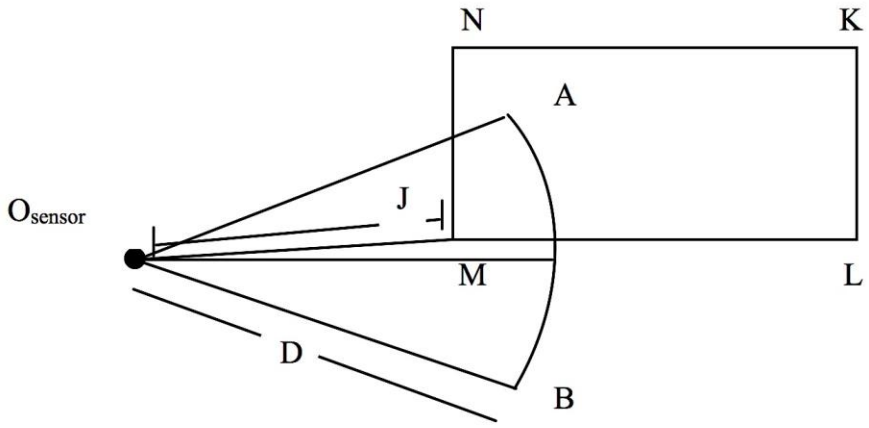


Gambar 4.6 Himpunan Jarak Maksimum

Berdasarkan Gambar 4.6, perhitungan jarak sensor terdekat dapat dihitung apabila hambatan (JKLM) berada dalam jarak bentang (AoB) dan lebih kecil dari jarak baca maksimum (D), maka jarak halangan dengan sensor yang didefinisikan seperti pada persamaan (4.1) berikut,

$$f(\text{Jarak}): \text{J Jika KLMN} < \text{d and JKLM} \in \text{AOsensorB} \qquad (4.1)$$

Gambar 4.6 memperlihatkan pemodelan dari persamaan (4.1),



Gambar 4.7. Pemodelan Perhitungan Jarak Sensor

Hasil bacaan dari setiap sensor didapat dari mengimplementasikan algoritma *line clipping* Liang-Barsky untuk mendapatkan jarak terdekat antar titik sensor dengan halangan.

Algoritma *line clipping* Liang-Barsky adalah algoritma yang menggunakan persamaan garis singgung untuk menghitung jarak antar satu titik dengan titik yg bersinggungan dengan garis lain. Pada Gambar 4.8, garis RS memiliki titik koordinat (x_1, x_2) dan titik koorfinat (y_1, y_2) sebagai atributnya dan hambatan dengan nama KLMN. Dalam bentuk persamaan matematis, pemodelan perhitungan jarak sensor pada Gambar 4.8 dijabarkan dengan persamaan garis singgung. Titik T pada garis RS yang bersinggungan dengan garis KN dapat dijabarkan dengan persamaan (4.2) berikut,

$$\left.\begin{aligned} x &= x_1 + (x_2 - x_1) * m \\ y &= y_1 + (y_2 - y_1) * m \end{aligned}\right\} \quad 0 < m < 1 \qquad (4.2)$$

$jika, m = 0\ maka\ x = x_1, y = y_1$

$jika, m = 1\ maka\ x = x_2, y = y_2$

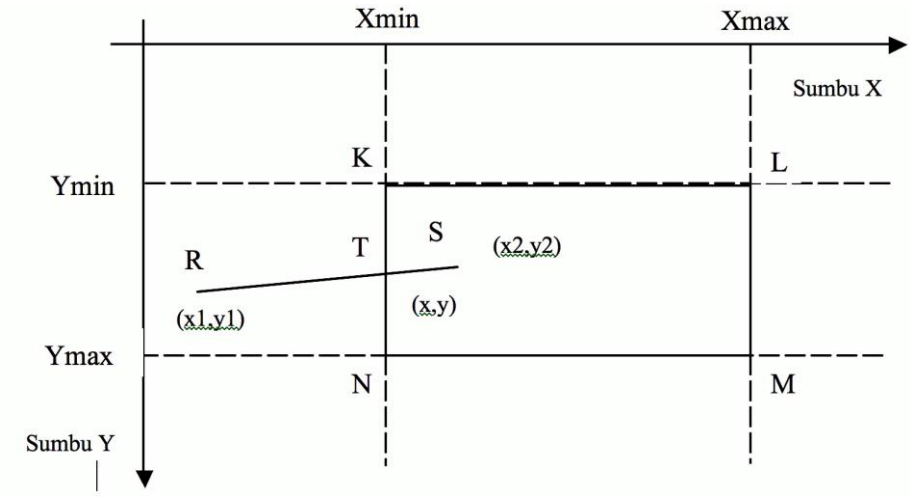
$jika\ x_2 - x_1 = dx,\ y_2 - y_1 = dy$

maka:

$$\begin{aligned} x &= x_1 + dx * m \\ y &= y_1 + dy * m \end{aligned} \qquad (4.3)$$

dengan kondisi:

$$\begin{aligned} &X_{min} \leq x \leq X_{max} \quad \text{dan} \\ &Y_{min} \leq y \leq Y_{max} \end{aligned} \qquad (4.4)$$



Gambar 4.8 Pemodelan Algoritma Liang-Barsky

Jika persamaan (4.3) disubtitusikan pada persamaan (4.4), maka diperoleh persamaan (4.5) berikut,

$$X_{min} \leq x_1 + d_x * m \leq X_{max}$$
$$Y_{min} \leq y_1 + d_y * m \leq Y_{min} \quad (4.5)$$

Persamaan (4.5) dapat ditulis kembali menjadi,

$$-d_x * m \leq x_1 - X_{min} \quad \sim \text{dinding kiri (1)}$$
$$d_x * m \leq X_{max} - x_1 \quad \sim \text{dinding kanan (2)}$$
$$-d_y * m \leq y_1 - Y_{min} \quad \sim \text{dinding atas (3)} \quad (4.6)$$
$$d_y * m \leq Y_{min} - y_1 \quad \sim \text{dinding bawah (4)}$$

Dari persamaan (4.6) didapat persamaan umum untuk menghitung gradien garis

$$p_i * m \leq q_i, dengan\ i = \{1,2,3,4\}$$
$$m = \frac{q_i}{p_i} \quad (4.7)$$

Dengan memasukan nilai m0 = 0 dan m1 = 1, nilai m pada persamaan (4.7) untuk masing-masing kondisi adalah sebagai berikut,

$$jika, P\ < 0\ dan\ m\ > m_i\ maka\ m_i = m \quad (4.8)$$

$$jika\ P > 0\ dan\ m\ < m_i\ maka\ m_i = m \quad (4.9)$$

dimana i = {0,1}.

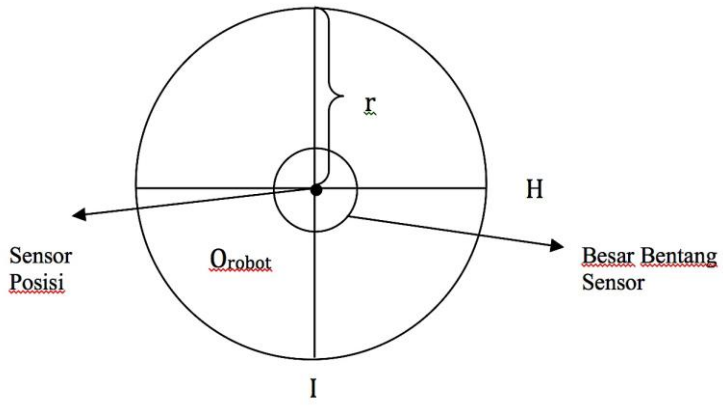
Selanjutnya apabila persamaan (4.7) dengan kondisi yang dijelaskan pada persamaan (4.8) dan persamaan (4.9) didapat persamaan (4.10) yang merupakan umum untuk mencari nilai m,

$$m_1 = MAX\ (\{\frac{q_i}{P_i} |\ P_i < 0, i = 1,2,3,4\}\ U\ \{0\})$$
$$m_2 = MIN\ (\{\frac{q_i}{P_i} |\ P_i > 0, i = 1,2,3,4\}\ U\ \{1\}) \quad (4.10)$$

Maka jarak titik T yang menjadi titik potong antar garis RS dan garis KN dengan titik R yang menjadi titik awal garis dapat dihitung dengan persamaan phytagoras.

4.2.5. Sensor Pencari Target

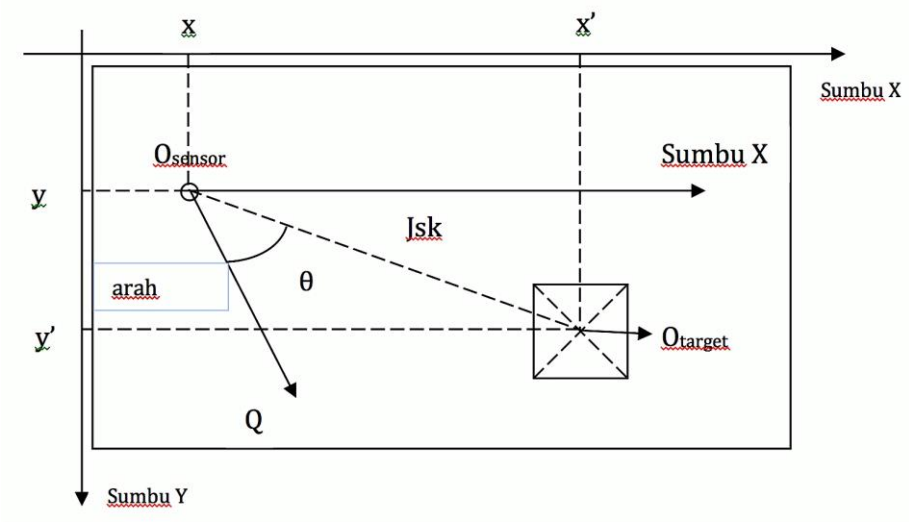
Sensor pencari target digambarkan tidak jauh berbeda dengan sensor pedeteksi jarak yaitu titik pada area dan letaknya selalu mengikuti pergerakan dari robot. Koordinat sensor ini sama dengan koordinat robot dikarenakan letaknya yang berada pada pusat robot.



Gambar 4.9 Model Sensor Posisi

Sensor pencari target berfungsi untuk menghitung jarak robot dengan target yang dituju. Pada Gambar 4.9, Besar jarak baca dari sensor (r) disesuaikan dengan besar model

area dan bentang sensor posisi berbentuk lingkaran dengan titik pusat Orobot. Besar sudut salah satu dari 4 juring lingkaran tersebut adalah 90^0 ditandai dengan sudut HOrobotI. Sensor posisi bertugas untuk menghitung jarak sensor dengan target dan jarak rotasi sebagai penanda arah menuju target. Perhitungan jarak (Jsk) antar titik sensor (Osensor) dengan titik target (Otarget) dan sudut QOsensorOtarget (θ) sebagai jarak rotasi antara arah robot dengan target dijelaskan pada Gambar 4.10.



Gambar 4.10. Model Perhitungan jarak

Jarak antar titik O_{sensor} yang memiliki atribut (x,y) pada model area dan titik O_{target} yang memiliki atribut (x',y') pada model area dapat dihitung dengan persamaan phytagoras persamaan (4.11),

$$J_{sk} = \sqrt{(x' - x)^2 + (y' - y)^2} \tag{4.11}$$

Besar arah robot adalah jarak rotasi antara titik Q dengan sumbu X positif, ditandai dengan sudut $QO_{sensor}X$. Target pada model area juga memiliki jarak rotasi terhadap sumbu X ditandai dengan sudut $O_{target}O_{sensor}X$. Sehingga jarak rotasi antara arah robot dengan arah target dapat dihitung dengan persamaan (4.12),

$$QO_{sensor}O_{target} = QO_{sensor}X - O_{target}O_{sensor}X \tag{4.12}$$

Jika sudut $QO_{sensor}O_{target}$ bernilai plus, maka target berada pada sebelah kiri dari arah robot, dan jika sudut $QO_{sensor}O_{target}$ bernilai minus, maka target berada pada sebelah kanan dari arah robot.

4.3. Model Pengendali Fuzzy

Hirarki perilaku fuzzy adalah sistem koordinasi perilaku akan menjadi sistem yang mengkoordinasikan pergerakan robot secara keseluruhan. Sub bab ini akan menjelaskan perilaku koordinasi sebagai perilaku pengatur dan perilaku-perilaku yang diaturnya sebagai perilaku utama. Hirarki perilaku fuzzy adalah sistem yang bertujuan untuk mengarahkan robot berdasarkan sejumlah perilaku yang masing-masing memberikan navigasi untuk pergerakan robot berdasarkan area disekitarnya.

Berdasarkan model robot yang telah dijelaskan sebelumnya, kedelapan sensor yang memberikan nilai bacaan kepada tiga perilaku utama. Perilaku pertama yaitu penghindar halangan akan mengelola masukan dari 3 sensor di depan robot. Perilaku kedua yaitu mengikuti dinding akan mengelola masukan dari 2 sensor masing-masing di samping kanan dan kiri robot. Dan perilaku terakhir yaitu pencari target akan mengelola masukan dari sensor yang berada pada pusat robot. Keluaran dari masing-masing perilaku ini yang kemudian dijadikan dasar oleh hirarki perilaku fuzzy untuk menentukan langkah yang tepat. Langkah tersebut akan dipakai oleh robot dalam bergerak berupa nilai kecepatan baru dan perubahan sudut.

4.3.1. Perilaku Menghindar Halangan

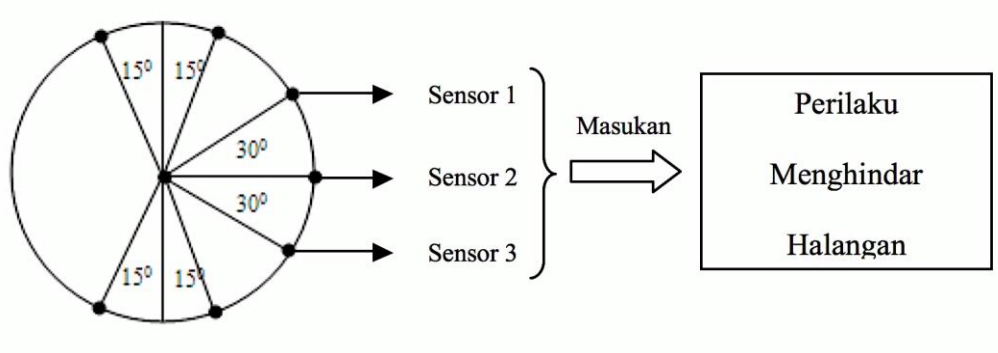
Fungsi utama dari perilaku ini adalah menjaga robot agar tidak bertabrakan dengan hambatan di depannya. Perilaku ini mendapat masukan dari 3 sensor yang berada di depan robot. Gambar 4.11 akan menjelaskan pemodelan dari perilaku ini. Nilai keluaran dari perilaku ini adalah kecepatan dan jarak yang dihitung dari nilai bacaan jarak dari sensor 1, 2, dan 3 yang dihitung dengan nilai keanggotaan atau fungsi keanggotaan.

4.3.2. Perilaku Mengikuti Dinding

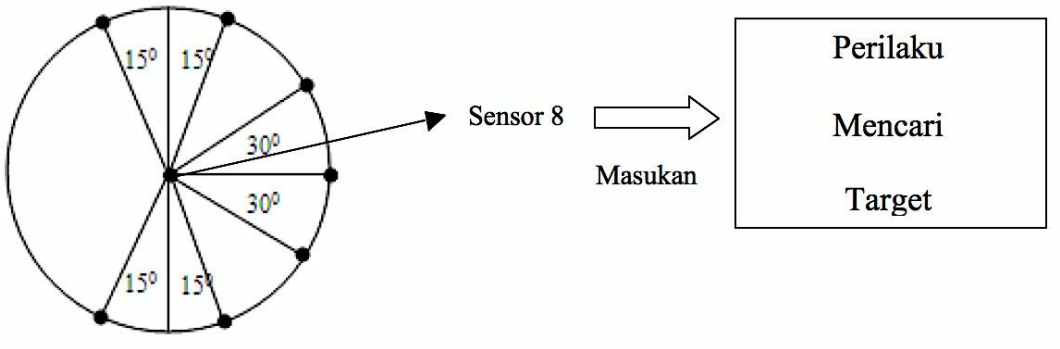
Fungsi utama dari perilaku ini adalah menjaga robot agar tetap berjalan di samping dinding yang berada di dalam lingkungannya. Perilaku ini mendapat masukan dari 2 sensor yang berada masing-masing di samping robot. Gambar 4.13 akan menjelaskan pemodelan dari perilaku ini. Nilai keluaran dari perilaku ini adalah kecepatan dan jarak yang dihitung dari nilai bacaan jarak dari sensor 4, 5, 6, dan 7 yang dihitung dengan nilai keanggotaan atau *membership function*.

4.3.3. Perilaku Pencari Target

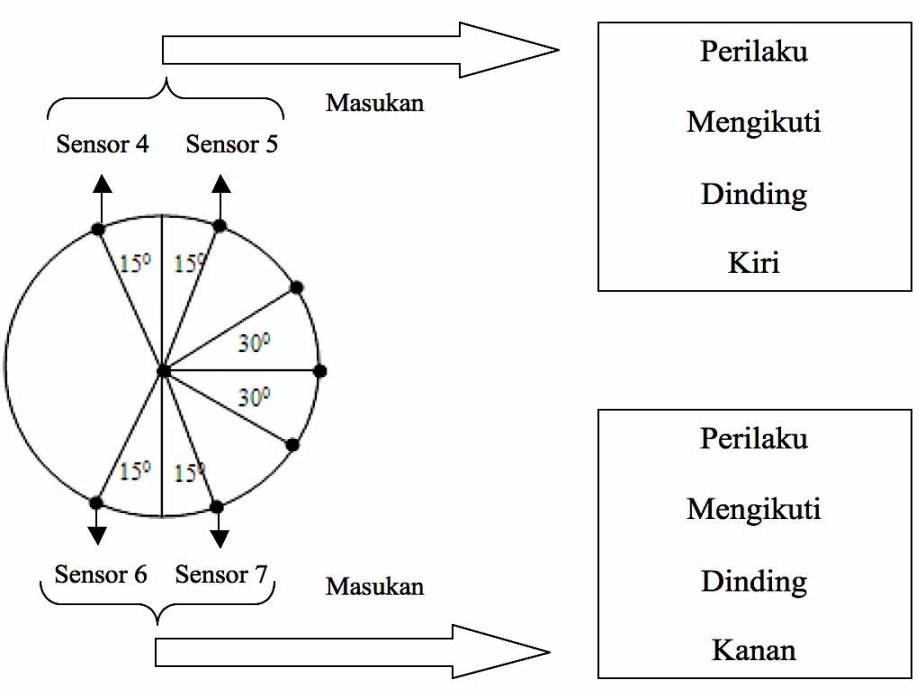
Fungsi utama dari perilaku ini adalah menjaga robot agar tetap berjalan mengarah pada target yang telah ditetapkan di dalam lingkungan. Perilaku ini mendapat masukan dari 1 sensor yang berada pusat robot. Gambar 4.12 akan menjelaskan pemodelan dari perilaku ini. Nilai keluaran dari perilaku ini adalah kecepatan dan jarak yang dihitung dari nilai bacaan jarak dari sensor 8 yang dihitung dengan nilai keanggotaan atau *membership function*.



Gambar 4.11. Model Prilaku Menghindar Halangan



Gambar 4.12. Model Prilaku Mencari Target

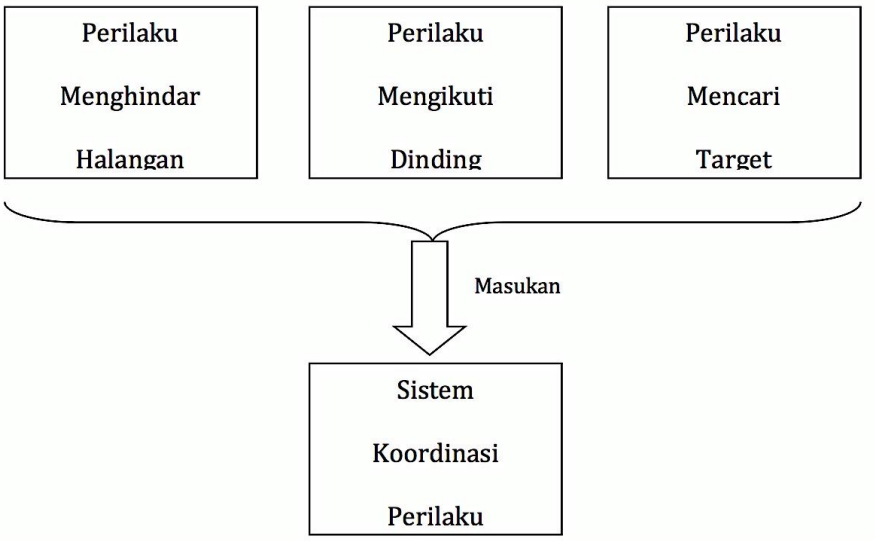


Gambar 4.13. Model Prilaku Mengikuti Dinding

4.3.4. Sistem Koordinasi Perilaku Dengan Logika Hirarki Fuzzy

Robot dinamis yang disimulasikan memiliki 3 buah perilaku (menghindar halangan, mengikuti dinding, mencari target), dan memberikan nilai kecepatan dan perubahan sudut tergantung dari kaedah masing-masing perilaku. Nilai-nilai tersebut akan dijadikan masukan untuk sistem koordinasi dengan logika *fuzzy hierarchy*. Dan sistem ini dirancang dengan satu keluaran, sehingga robot akan memiliki dua sistem koordinasi yang

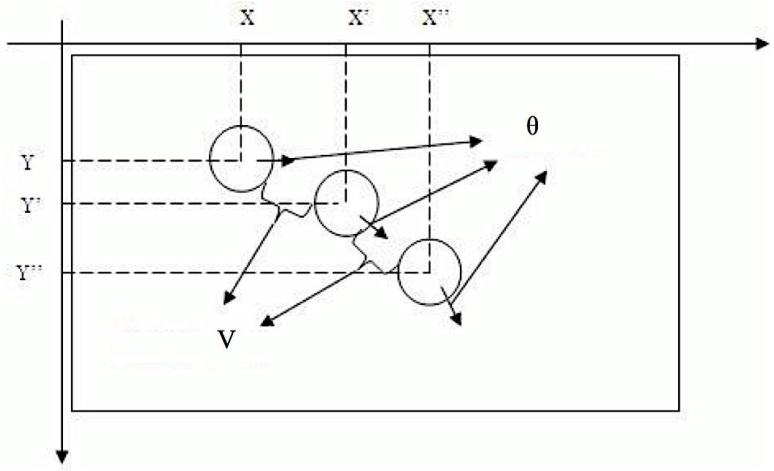
masing-masing memberikan keluaran berupa kecepatan dan perubahan sudut. Gambar 4.14 akan menjelaskan skema sistem koordinasi perilaku tersebut.



Gambar 4.14. Skema Sistem Koordinasi Perilaku

4.3.5. Keluaran Hirarki Perilaku Fuzzy

Hasil akhir yang diinginkan dari perilaku utama ini adalah kecepatan dan arah yang paling tepat dalam memandu pergerakan robot berikutnya. Kecepatan baru ditandai dengan variabel V dan arah baru ditandai dengan variabel θ. Gambar 4.15 akan menjelaskan pemodelan keluaran perilaku ini.



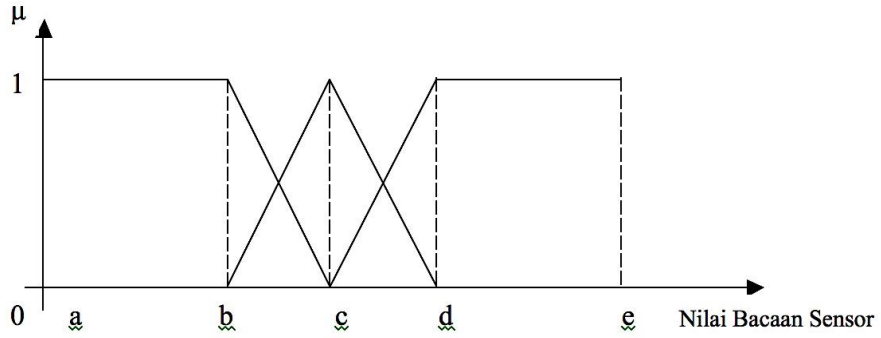
Gambar 4.15. Model Hasil Keluaran Hirarki Perilaku Fuzzy

4.3.6. Fungsi Keanggotaan

Nilai keanggotaan x pada fungsi keanggotaan didefinisakan sebagai μ(x) yang bernilai $0 \leq \mu x \leq 1$,

$$f(x) : \mu x \rightarrow \{ 0 \leq \mu x \leq 1\} \tag{4.13}$$

Terdapat dua jenis fungsi keanggotaan yaitu fungsi keanggotaan sebab (antecedent) dan fungsi keanggotaan akibat (consequent).



Gambar 4.16 Fungsi Keanggotaan Hirarki

Gambar 4.16, menjelaskan fungsi yang membagi nilai dari bacaan sensor menjadi 3 bagian yaitu dekat, sedang, dan jauh. Kategori-kategori tersebut akan menentukan aturan yang digunakan pada sistem pengendali fuzzy. Pada Gambar 4.16, jika *x* adalah nilai dari bacaan sensor dan *x* berada di antara titik a dan b maka nilai keanggotaan dari sistem fuzzy didapat dari persamaan (4.14),

$$\begin{aligned} \mu_{abc}(x) &= (x - a)/(b - a) \\ \mu_{bcd}(x) &= (b - x)/(b - a) \end{aligned} \tag{4.14}$$

Nilai dari keanggotaan x bernilai ganda dikarenakan x berada pada 2 fungsi keanggotaan yaitu MF (*abc*) dan MF (*bcd*).

BAB 5

KINERJA SISTEM KENDALI FUZZY

Pada bab ini akan di uraikan bagimana kinerja sistem kendali yang telah dirancang. Dua jenis sistem kendali logika fuzzy yang akan diuji yaitu sistem logika fuzzy tipe-1 dan tipe-2. Hal ini dilakukan untuk melihat bagaimana proses masing-masing pengendali jika diimplementasikan pada sistem robotik.

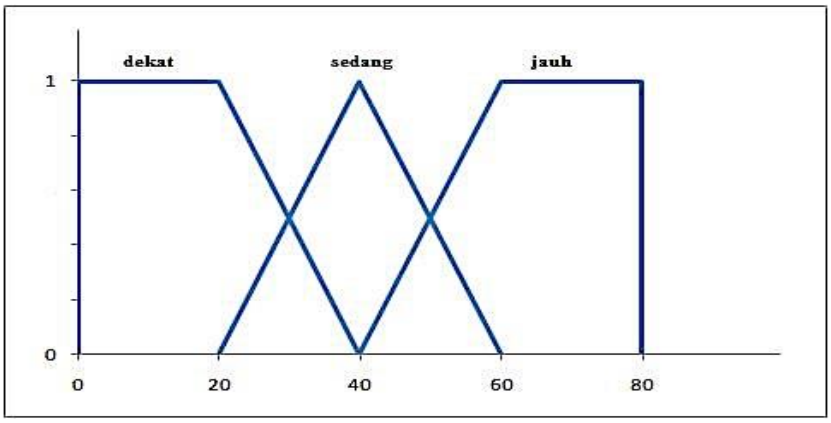
5.1. Sistem Logika Fuzzy Tipe-1

Pada pengujian ini akan dibahas cara kerja sistem berdasarkan perilaku dan lingkungan yang dimiliki oleh sistem. Pengujian ini dibagi menjadi 2 bagian yaitu pengujian dengan 2/3 perilaku pada lingkungan mudah, sulit, dan tidak terstruktur dan perhitungan jarak sensor.

5.1.1. Model Pengendali

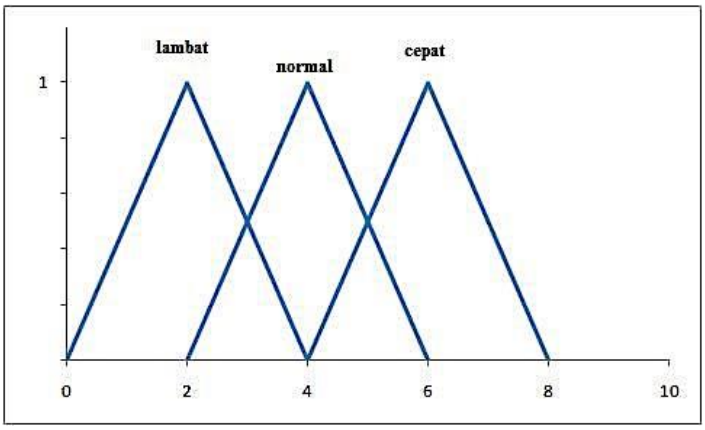
5.1.1.1. Fungsi Keanggotaan Masukan dan Keluaran

Nilai masukan yang dibaca oleh sensor adalah jarak antara dinding dengan titik sensor. Agar nilai jarak dapat diproses menggunakan sistem pengendali logika *fuzzy*, maka jarak harus memiliki fungsi keanggotaan sendiri. Fungsi keanggotaan dari jarak menggunakan bentuk fungsi keanggotaan trapesium dan segitiga, dimana setiap linguistiknya yaitu dekat, sedang dan jauh sedangkan fungsi keanggotaan jarak seperti pada Gambar 5.1.

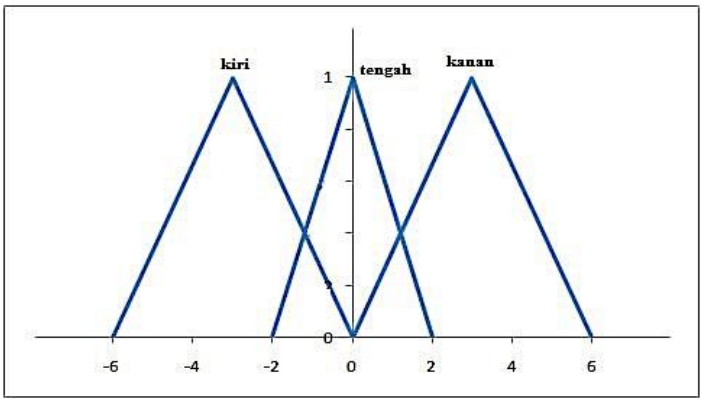


Gambar 5.1. Fungsi Keanggotaan Jarak

Pada keluaran *fuzzy*, hasil yang di dapat adalah berupa nilai perubahan sudut dan nilai kecepatan baru. Kedua nilai tersebut memiliki fungsi keanggotaan yang sama banyaknya, yaitu tiga linguistik dan hanya memakai bentuk fungsi keanggotaan segitiga seperti terlihat pada Gambar III - 7. Linguistik untuk fungsi keanggotaan perubahan sudut yaitu kiri, tengah dan kanan. Linguistik untuk fungsi keanggotaan kecepatan yaitu lambat, normal dan cepat.



Gambar 5.2. Fungsi Keanggotaan Dari Kecepatan



Gambar 5.3. Fungsi Keanggotaan Dari Perubahan Sudut

Pada sistem pengendali pengikut dinding terdapat dua basis aturan pengikut dinding bagian kanan dan pengikut dinding bagian kiri seperti pada Tabel 5.1. Setiap basis pengetahuan akan di gunakan untuk menentukan nilai hasil fuzifikasi yang disimpan dalam inferensi. Pada proses defuzifikasi, metode yang dipakai adalah metode *centroid* untuk menghitung pusat gravitasi dari himpunan keluaran fuzz*y* pada interval a ke b.

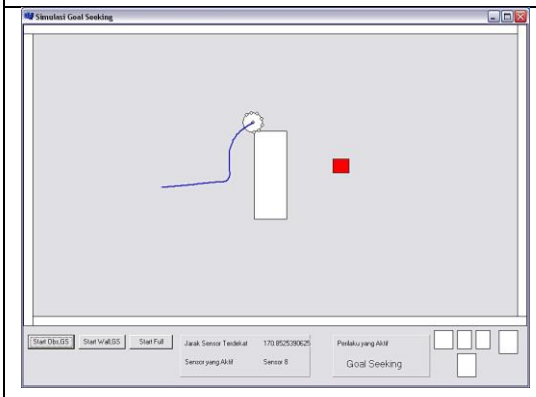
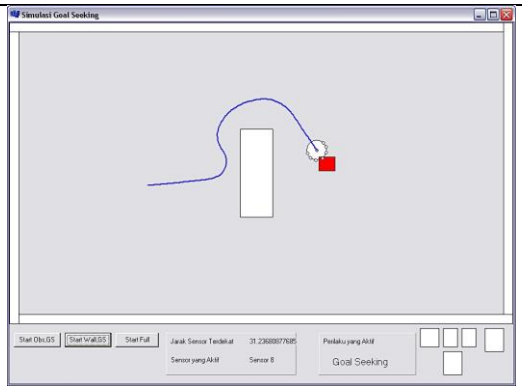
Tabel 5.1. Contoh Basis Aturan Prilaku Pengikut Dinding

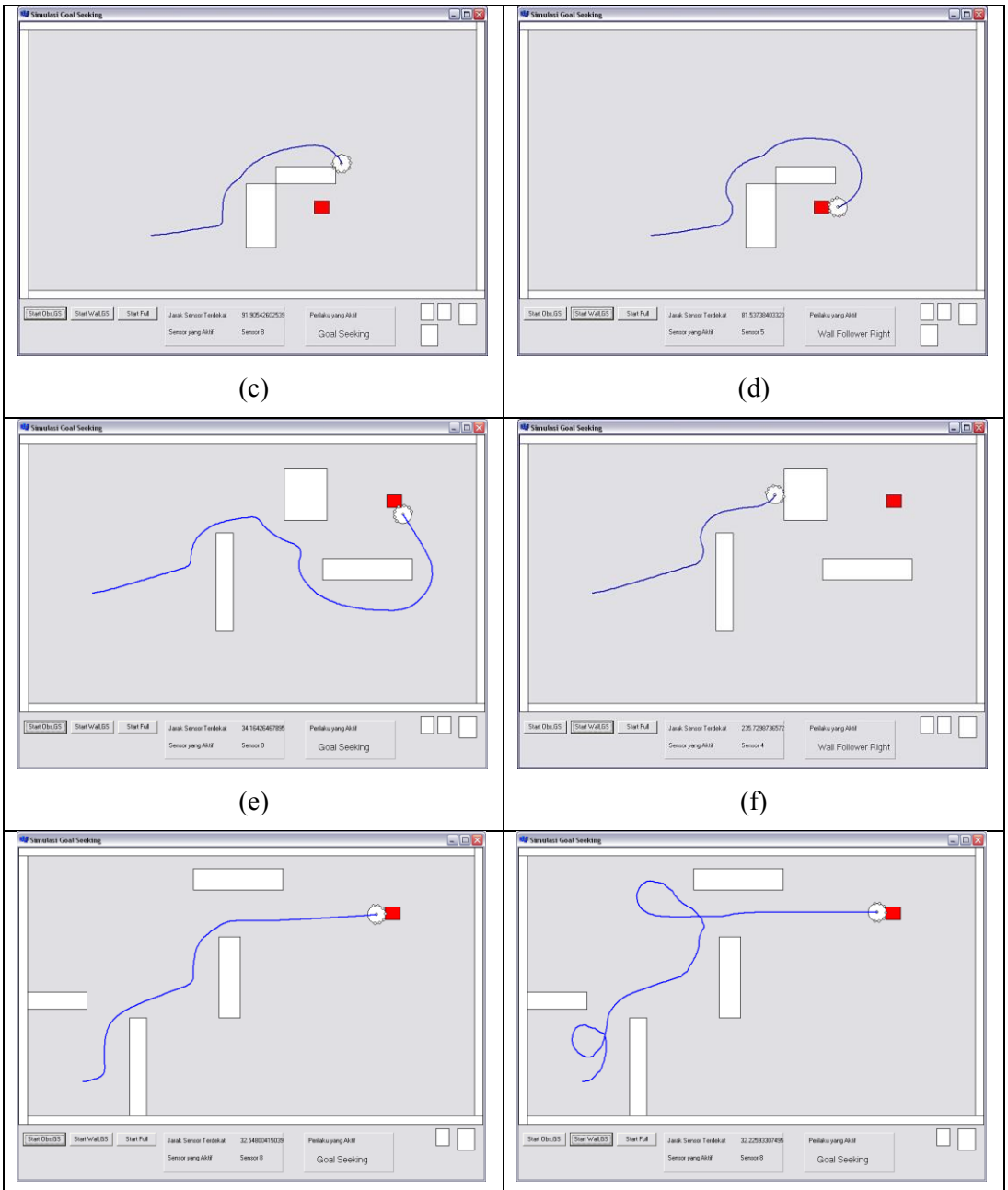
DINDING KANAN			
Sensor 0	**Sensor 1**	**Kecepatan Gerak**	**Arah Perubahan Sudut**
dekat	dekat	lambat	Kiri
dekat	sedang	lambat	Kiri
dekat	jauh	lambat	Kiri
sedang	dekat	normal	Lurus
sedang	sedang	normal	Lurus
sedang	jauh	normal	Lurus
jauh	dekat	cepat	Kanan
jauh	sedang	cepat	Kanan
jauh	jauh	cepat	Kanan
DINDING KIRI			
Sensor 2	**Sensor 3**	**Kecepatan Gerak**	**Arah Perubahan Sudut**
dekat	dekat	lambat	Kanan
dekat	sedang	lambat	Kanan
dekat	jauh	lambat	Kanan
sedang	dekat	normal	Lurus
sedang	sedang	normal	Lurus
sedang	jauh	normal	Lurus
jauh	dekat	cepat	Kiri
jauh	sedang	cepat	Kiri
jauh	jauh	cepat	Kiri

5.1.2. Kinerja Pengendali untuk 2 Prilaku

Kinerja yang ingin dilihat saat penggunaan fuzzy hirarki adalah bagaimana semua prilaku dari sistem robotik dapat bekerja sama dan berkoordinasi dalam mengaktifkan prosedur pengendalian. Dalam hal ini prilaku yang akan dilihat hanya 3 yaitu menghindar halangan, mengikuti dinding dan mencari target. Sebagai hasil dari algoritma yang telah digunakan dalam pengendalian robot dapat dilihat pada Gambar 5.4 berikut ini. Dalam gambar tersebut yang digunakan adalah hanya 2 prilaku yang aktif secara bergantian.

Pada Gambar 5.4. (a) dan (c), perilaku yang digunakan adalah perilaku penghindar halangan dan pencari target. Terdapat kelemahan pada koordinasi 2 perilaku ini ketika mendekati sudut halangan. Permasalahannya adalah ketika 3 sensor didepan robot tidak mendeteksi halangan maka robot akan mendekati target sedangkan di samping kanan atau kiri robot masih berada halangan yang tidak terdeteksi. Pada gambar 5.4. (b) dan (d), perilaku yang digunakan adalah perilaku pengikut dinding dan pencari target. Terdapat kelemahan pada koordinasi 2 perilaku ini ketika mendekati halangan. Permasalahannya adalah ketika robot mendekati halangan dari depan maka robot akan merubah sudutnya secara *crips* dan semakin dekat dengan halangan.

Penghindar Halangan dan Pencari Target	Pengikut Dinding dan Pencari Target
	
(a)	(b)



(c) (d)

(e) (f)

(g) (h)

(a) dan (b) 2 perilaku dengan satu halangan, (c) dan (d) 2 perilaku dengan dua halangan, (e) dan (f) 2 perilaku dengan 3 halangan, (g) dan (h) 2 perilaku dengan 4 halangan

Gambar 5.4. Perilaku Robot Dengan Halangan

Pada Gambar 5.4 (e) dan (g), perilaku yang digunakan adalah perilaku penghindar halangan dan pencari target. Nilai awal yang dimasukan dalam aturan fuzzy perilaku penghindar halangan adalah ketika sensor depan mendeteksi jarak maka robot akan

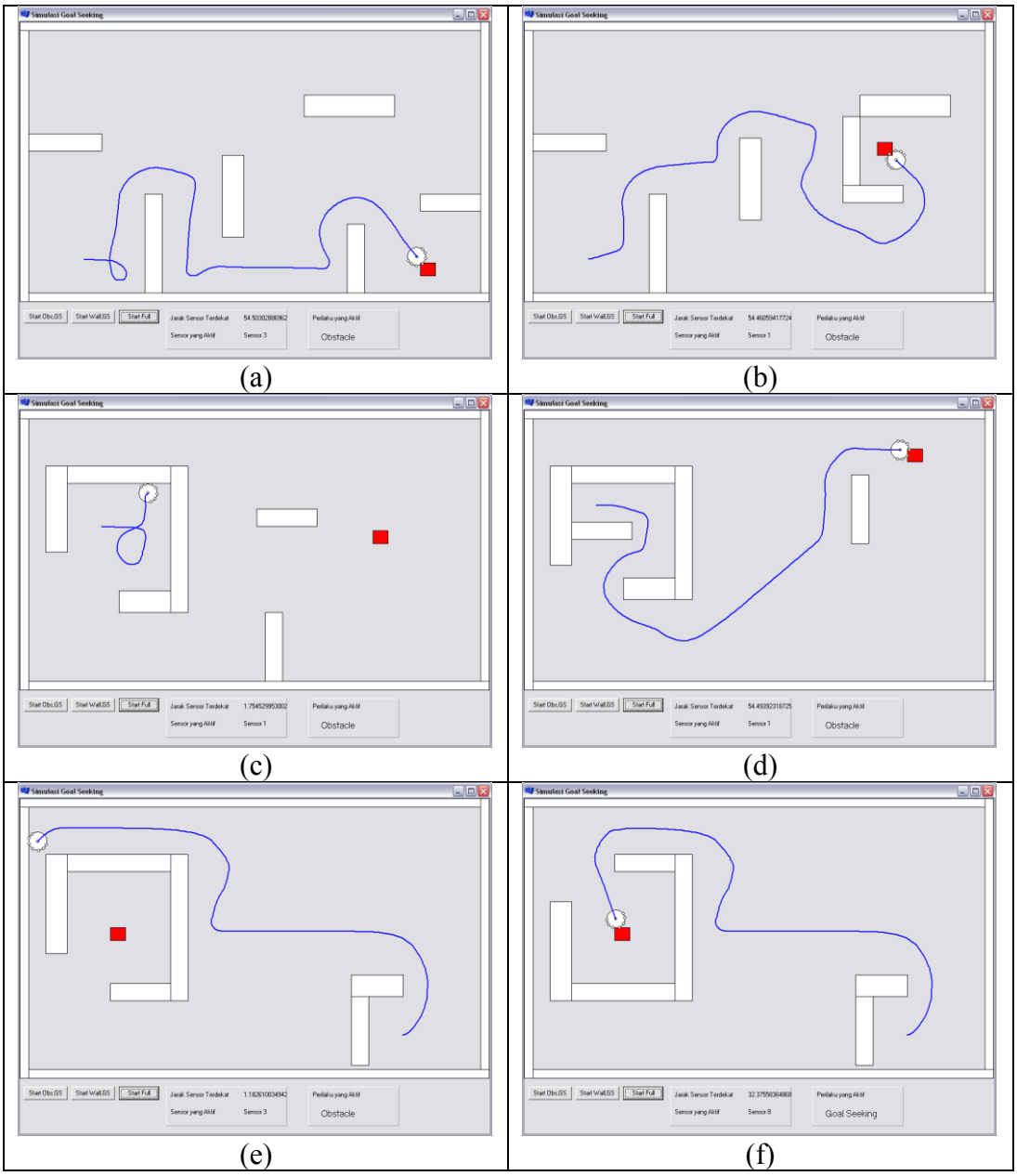
memutar sudutnya kekanan. Maka sesuai Gambar 5.4. (e), jarak yang ditempuh menuju target memutar. Pada Gambar 5.4. (f) dan (h), perilaku yang digunak an adalah perilaku pengikut dinding dan pencari target. Terdapat kelemahan pada koordinasi 2 perilaku ini ketika mendekati halangan. Permasalahannya adalah ketika robot mendekati halangan dari depan maka robot akan merubah sudutnya secara *crips* dan pada kondisi ini robot tidak memiliki pengetahuan untuk memutar sudutnya sesuai dengan lingkungan.

5.1.3. Kinerja Pengendali untuk 3 Prilaku

Perilaku yang digunakan pada Gambar 5.5 adalah penghindar halangan, pengikut dinding, dan pencari target. Pada Gambar 5.5. (a), lingkungan yang digunakan adalah lingkungan tidak terstuktur 1. Robot pada awalnya memutar hampir 360 derajat dikarenakan robot mendeteksi target terdapat di sebelah kanan arah robot namun terdapat halangan disebelah kanan hingga ia berputar dan mencari jalan lain.

Pada Gambar 5.5. (b) dan (d), lingkungan yang digunakan adalah lingkungan tidak terstuktur 2 dan 4. Robot tidak menemukan kesulitan pada pencarian target dengan lingkungan ini. Ditunjukan dengan tidak ada lintasan yang tidak efisien menuju target. Pada Gambar 5.5. (c), lingkungan yang digunakan adalah lingkungan tidak terstuktur 3. Robot menemukan kesulitan pada pencarian target dengan lingkungan ini dikarenakan pada saat robot tidak mendeteksi halangan pada sebelah kiri robot dan mendeteksi target berada pada sebelah kanan robot.

Pada Gambar 5.5. (e) dan (f), lingkungan yang digunakan adalah lingkungan tidak terstuktur 5 dan 6. Lingkungan ini memiliki sedikit perbedaan pada pintu menuju target. Robot menemukan kesulitan pada pencarian target dengan lingkungan 5 dikarenakan pada saat robot memiliki nilai awal yang dimasukan dalam aturan fuzzy perilaku penghindar halangan adalah ketika sensor depan mendeteksi jarak maka robot akan memutar sudutnya kekanan dan menemuni jalan buntu dan tidak dapat berputar. Robot tidak menemukan kesulitan ketika pintu menuju target dirubah menjadi diatas seperti pada lingkungan 6.



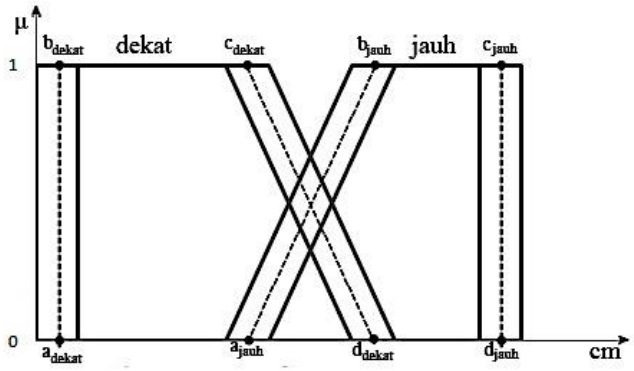
Gambar 5.5. Koordinasi Perilaku Sesuai Dengan Lingkungan Kompleks Dan Tidak Terstruktur

5.2. Sistem Logika Fuzzy Tipe-2 Interval

5.2.1. Model Pengendali

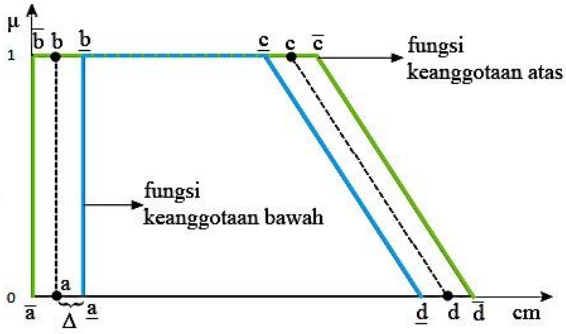
5.2.1.1. Fuzzifikasi

Dalam perancangan sistem kontrol logika fuzzy tipe-2, masing-masing *antecedent* memiliki dua fungsi keanggotaan, yaitu Jauh dan Dekat. Banyak dan bentuk himpunan fungsi keanggotaan *antecedent* statis, yaitu dua buah dengan bentuk himpunan trapesium, seperti pada Gambar 5.6, sedangkan parameter himpunannya dapat diubah oleh pengguna.



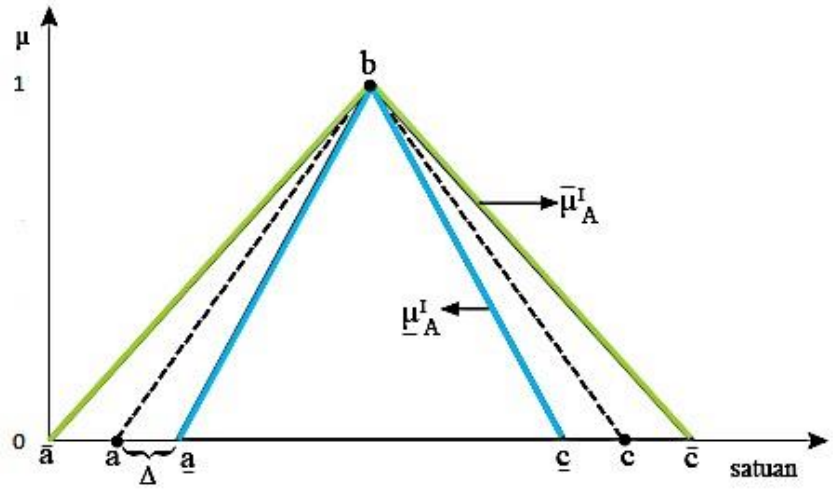
Gambar 5.6. A*ntecedent* dari SLF penghindar halangan

Parameter dari *antecedent* yang dapat dimasukkan oleh pengguna berupa nilai-nilai dari a,b,c,dan d, serta lebar pita Δ seperti pada Gambar 5.6. Lebar pita yang diberikan oleh pengguna kemudian akan digunakan untuk menentukan FOU dari himpunan interval *fuzzy* tipe-2 dari *antecedent.*



Gambar 5.7. Parameter-parameter pada Fungsi Keanggotaan trapesium

Himpunan keluaran dari SLF adalah himpunan *fuzzy* interval tipe-2, masing-masing berupa kecepatan dan perubahan sudut dari *mobile robot*, masing-masing memiliki tiga fungsi keanggotaan segitiga seperti pada Gambar 5.7. Banyak dan jenis fungsi keanggotaan pada masing-masing *consequent* adalah statis, yaitu tiga buah dengan fungsi keanggotaan segitiga. Terdapat tiga parameter untuk fungsi keanggotaan segitiga sehingga untuk menentukan FOU dari masing-masing himpunan fungsi keanggotaan, pengguna harus memasukkan nilai-nilai a,b,c dan lebar pita Δ seperti pada Gambar 5.8.



Gambar 5.8. Parameter-Parameter Pada Fungsi Keanggotaan Segitiga

5.2.1.2. Kaidah dan Inferensi

Banyak kombinasi kaidah yang ada pada aplikasi adalah sebanyak 8 kaidah, banyak dari kaidah didapatkan dari kombinasi 3 masukan yang masing-masing memiliki dua fungsi keanggotaan. Kombinasi dari kaidah-kaidah tersebut dapat dilihat pada Tabel 5.2 berikut ini,

Tabel 5.2 Kombinasi masukan untuk tiap-tiap kaidah

Sensor0	Sensor1	Sensor2
Dekat	Dekat	Dekat
Dekat	Dekat	Jauh
Dekat	Jauh	Dekat
Dekat	Jauh	Jauh
Jauh	Dekat	Dekat
Jauh	Dekat	Jauh
Jauh	Jauh	Dekat
Jauh	Jauh	Jauh

Tiap-tiap baris dari kaidah kemudian akan digunakan untuk menentukan nilai hasil fuzzifikasi yang disimpan dalam inferensi. Metode inferensi yang digunakan pada aplikasi adalah inferensi komposisi extended sup-star, seperti pada persamaan 3.22 (Bab 3) dan persamaan 3.23 (Bab 3). Untuk untuk menentukan nilai hasil inferensi, dicari nilai minimum dari $\overline{f}$ dan $\underline{f}$ untuk masing-masing baris kaidah. Nilai-nilai ini kemudian disimpan sebagai $\overline{f}_{min}$ dan $\underline{f}_{min}$. Untuk mendapatkan nilai $\overline{f}_{min}$ untuk tiap-tiap baris kaidah dilakukan langkah-langkah seperti berikut ini :

Simpan nilai $\overline{f}$ *untuk masing-masing* x_1, x_2, x_3 *masukan dari sensor sebagai* $\overline{f_{1dekat}}$, $\overline{f_{2dekat}}$, $\overline{f_{3dekat}}$ *dan* $\overline{f_{1jauh}}$, $\overline{f_{2jauh}}$, $\overline{f_{3jauh}}$

Tentukan nilai $\overline{f}^{in1}$ *sesuai dengan kombinasi kaidah pada Tabel 5.2*

Jika sensor 0 = dekat, $\overline{f}^{in1} = \overline{f_{1dekat}}$

Jika sensor 0 = jauh, $\overline{f}^{in1} = \overline{f_{1jauh}}$

Tentukan nilai $\overline{f}^{in2}$ *sesuai dengan kombinasi kaidah pada Tabel 5.2*

Jika sensor 1 = dekat, $\overline{f}^{in2} = \overline{f_{2dekat}}$

Jika sensor 1 = jauh, $\overline{f}^{in2} = \overline{f_{2jauh}}$

Tentukan nilai $\overline{f}^{in3}$ *sesuai dengan kombinasi kaidah pada Tabel 5.2*

Jika sensor 2 = dekat, $\overline{f}^{in3} = \overline{f_{3dekat}}$

Jika sensor 2 = jauh, $\overline{f}^{in3} = \overline{f_{3jauh}}$

Tentukan nilai $\overline{f}_{min} = min(\overline{f}^{in1}, \overline{f}^{in2}, \overline{f}^{in3})$

Nilai $\overline{f}_{min}$dari tiap-tiap kombinasi masukan dihitung dan disimpan secara paralel dengan nilai $\underline{f}_{min}$. Algoritma untuk mendapatkan nilai $\overline{f}_{min}$ untuk sebuah baris kombinasi kaidah adalah:

Simpan nilai $\underline{f}$ *untuk masing-masing* x_1,x_2,x_3 *masukan dari sensor sebagai* $\underline{f}_{1dekat}$, $\underline{f}_{2dekat}$, $\underline{f}_{3dekat}$ *dan* $\underline{f}_{1jauh}$, $\underline{f}_{2jauh}$, $\underline{f}_{3jauh}$

Tentukan nilai $\underline{f}^{in1}$ *sesuai dengan kombinasi kaidah pada Tabel 5.2*

Jika sensor 0 = dekat, $\underline{f}^{in1} = \underline{f}_{1dekat}$,

Jika sensor 0 = jauh, $\underline{f}^{in1} = \underline{f}_{1jauh}$,

Tentukan nilai $\underline{f}^{in2}$ *sesuai dengan kombinasi kaidah pada Tabel 5.2*

Jika sensor 1 = dekat, $\underline{f}^{in2} = \underline{f}_{2dekat}$,

Jika sensor 1 = jauh, $\underline{f}^{in2} = \underline{f}_{2jauh}$

Tentukan nilai $\underline{f}^{in3}$ *sesuai dengan kombinasi kaidah pada Tabel 5.2*

Jika sensor 2 = dekat, $\underline{f}^{in3} = \underline{f}_{3dekat}$,

Jika sensor 2 = jauh , $\underline{f}^{in3} = \underline{f}_{3jauh}$

Tentukan nilai $\overline{f}_{min} = min(\underline{f}^{in1}, \underline{f}^{in2}, \underline{f}^{in3})$

Nilai dari $\overline{f}_{min}$ dan $\underline{f}_{min}$ dari masing-masing kombinasi kaidah dihitung dan disimpan sebagai himpunan $\overline{f}_{min1}, \overline{f}_{min2}, \dots, \overline{f}_{minN}$ dan $\underline{f}_{min1}$, $\underline{f}_{min2}$,..., $\underline{f}_{minN}$ untuk N = banyak kaidah, yang dalam hal ini adalah delapan.

5.2.1.3. Reduksi Tipe Dan Defuzzifikasi

Pada SLF tipe-2, sebelum dilakukan defuzzifikasi, harus terlebih dahulu dilakukan reduksi pada himpunan-himpunan *fuzzy* tipe-2. Aplikasi ini menggunakan reduksi tipe *center of sets*, dimana nilai himpunan tereduksi adalah y_l dan y_r yang merupakan pendekatan nilai tengah dari himpunan *fuzzy* hasil inferensi. Pada reduksi tipe *center of sets*, seperti yang telah disebutkan pada nilai-nilai centroid dari masing-masing himpunan keluaran telah disimpan dalam memori sebagai $c_{rlambat}$ $c_{rsedang}$, c_{rcepat} dan $c_{llambat}$, $c_{lsedang}$,

c_{lcepat} untuk himpunan keluaran kecepatan, dan c_{rkiri}, $c_{rtengah}$, c_{rkanan} dan c_{lkiri} , $c_{ltengah}$, c_{lkanan} untuk himpunan keluaran arah.

Langkah-langkah untuk menghitung nilai y_l pada SLF dengan himpunan keluaran adalah sebagai berikut:

Tentukan nilai-nilai yli untuk i=1,2,...,banyak kaidah, sesuai dengan keluaran dari tiap kombinasi baris kaidah.

$y_{li} = c_{llambat}$, *jika keluaran = lambat*

$y_{li} = c_{lsedang}$, *jika keluaran = sedang*

$y_{li} = c_{lcepat}$, *jika keluaran = cepat*

Urutkan himpunan y_{li} *beserta urutan kombinasi kaidahnya secara menaik untuk i=1,2,...,banyak kaidah.*

Hitung nilai-nilai $f_{min\,i} = \overline{f}_{min\,i} + \underline{f}_{min\,i} \Big/ 2$ *untuk i=1,2,...,banyak kaidah*

Hitung nilai $y_l' = \frac{\sum_{i=1}^{N} y_{li} f_{min\,i}}{\sum_{i=1}^{N} f_{min\,i}}$, *dengan N= banyak kaidah*

Tentukan nilai $y_l'' = -1$

Tentukan nilai k_lama =0
Tentukan nilai k_baru = 0
Tentukan nilai break=false;
While (break=false) do
Tentukan nilai k=0
Cari nilai k_baru, dimana $y_{li} \leq f_{min\,k\_baru} \leq y_{l(i+1)}$ *dengan i=1,2,...,banyak kaidah*
Jika k_baru $\leq$ *k_lama, break=true;*
Hitung nilai $y_l'' = \frac{\sum_{i=1}^{k} y_{li} \underline{f}_{min\,i} + \sum_{i=k+1}^{N} y_{li} \overline{f}_{min\,i}}{\sum_{i=1}^{k} \underline{f}_{min\,i} + \sum_{i=k+1}^{N} \overline{f}_{min\,i}}$, *dimana N =banyak kaidah*
Jika yl' = yl" break = true
Jika $y_l' \neq y_l''$
$y_l' = y_l''$
k_lama=k_baru
endwhile
Tentukan nilai $y_l = yl''$

Nilai *yr* pada SLF dengan keluaran himpunan keanggotaan kecepatan kemudian juga dihitung dengan langkah-langkah sebagai berikut :

Tentukan nilai-nilai y_{ri} *untuk i=1,2,...,banyak kaidah, sesuai dengan keluaran dari tiap kombinasi baris kaidah.*

$y_{ri} = c_{rlambat}$, *jika keluaran = lambat*

$y_{ri} = c_{rsedang}$, *jika keluaran = sedang*

$y_{ri} = c_{rcepat}$, *jika keluaran = cepat*

Urutkan himpunan y_{ri} beserta urutan kombinasi kaidahnya secara menaik untuk i=1,2,...,banyak kaidah.

Hitung nilai-nilai $f_{min\,i} = \overline{f}_{min\,i} + \underline{f}_{min\,i} \Big/ 2$ *untuk i=1,2,...,banyak kaidah*

Hitung nilai $y_r' = \frac{\sum_{i=1}^{N} y_{ri} f_{min\,i}}{\sum_{i=1}^{N} f_{min\,i}}$, *dengan N= banyak kaidah*

Tentukan nilai $y_r'' = -1$

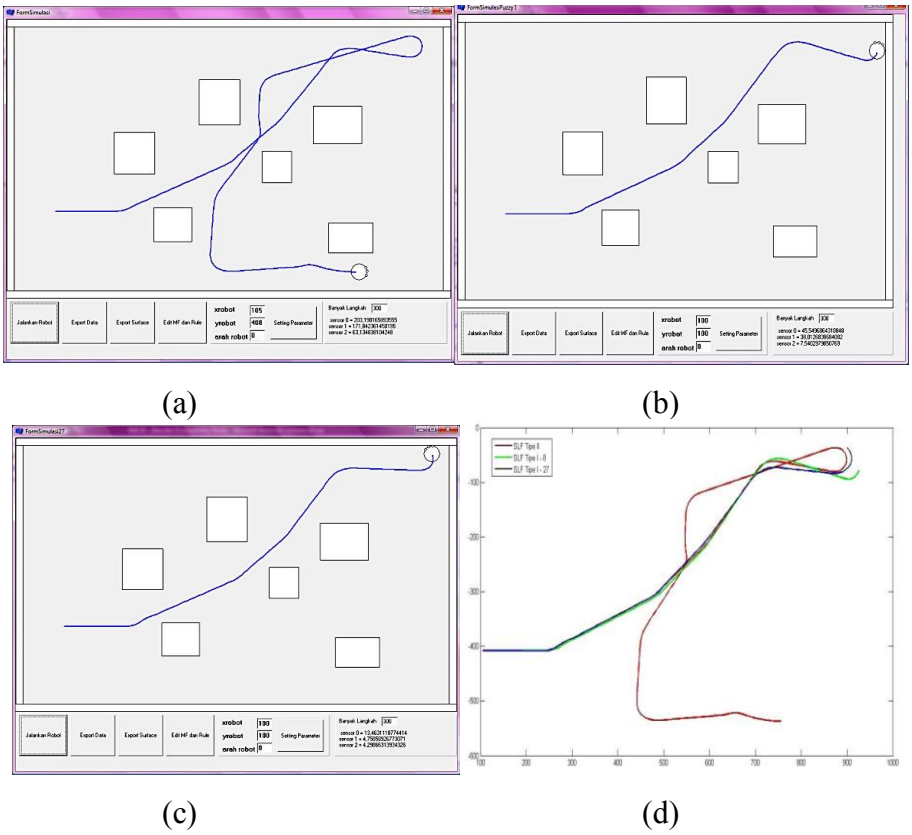
Tentukan nilai k_lama =0
Tentukan nilai k_baru = 0
Tentukan nilai break=false;
While (break=false) do
    Tentukan nilai k=0
    Cari nilai k_baru, dimana $y_{ri} \leq f_{min\,k\_baru} \leq y_{r(i+1)}$ *dengan i=1,2,...,banyak kaidah*
    Jika k_baru ≤ k_lama, break=true;
    Hitung nilai $y_r'' = \frac{\sum_{i=1}^{k} y_{ri} \overline{f}_{min\,i} + \sum_{i=k+1}^{N} y_{ri} \underline{f}_{min\,i}}{\sum_{i=1}^{k} \overline{f}_{min\,i} + \sum_{i=k+1}^{N} \underline{f}_{min\,i}}$ *dimana N =banyak kaidah*
    Jika yl' = yl" break = true
    Jika $y_l' \neq y_l''$
        $y_l' = y_l''$
        k_lama=k_baru
endwhile
Tentukan nilai $y_l = yl''$

Hasil defuzzifikasi dari SLF tipe-2 didapatkan dengan menjumlahkan *yl* dan *yr* yang telah didapatkan dari hasil reduksi tipe dan membaginya dengan dua.

5.2.2. Kinerja Pengendali

Parameter input dalam sistem yang dibangun adalah jarak antara robot dengan diding dibagian kanan atau kiri robot. Jarak tersebut akan dideteksi oleh sensor untuk kemudian diproses oleh SLF tipe-2 interval. Sedangkan parameter output dalam sistem yang dibangun adalah kecepatan dan perubahan sudut. Nilai kecepatan dalam sistem yang dibangun memiliki batasan dari 10 – 100 rpm dan nilai perubahan sudut dalam sistem yang dibangun memiliki batasan dari -7^o s.d 7^o. Dalam SLF tipe-1 dan tipe-2 nilai kecepatan dan perubahan sudut tersebut dinyatakan dalam fungsi keanggotaan berbentuk segitiga. Setelah sistem pengendali dirancang dengan tahapan dan spesifikasi yang telah ditentukan, maka robot di uji pada lingkungan simulasi. Dalam Gambar 5.9 lingkungan penguji cukup kompleks, dimana terdapat 6 buah halangan dan robot harus berjalan tanpa

menabrak dinding maupun hambatan. Pada pengujian ini dilakukan perbandingan antara SLF tipe-1 dan SLF tipe-2 dengan hasil sebagai berikut.



(a) (b)

(c) (d)

Gambar 5.9. Pergerakan pada lingkungan tidak terstruktur (a) SLF Tipe-2 (b) SLF Tipe-1 (8 kaidah) (c) SLF Tipe-1 (27 kaidah) (d) Plotting trajektori dari ketiga SLF

Gambar 5.9 (a), (b), dan (c) adalah trajektori dari SLF Tipe-2, Tipe-1 dengan 8 kaidah, dan Tipe-1 dengan 27 kaidah pada lingkungan tidak terstruktur, sedangkan Gambar 5.9 (d) adalah plotting trajektori dari ketiga SLF. Terlihat bahwa pada lingkungan tidak terstruktur hanya robot dengan SLF tipe-2 yang tidak mengalami crash saat, sedangkan kedua SLF Tipe-1 menabrak dinding untuk 2 kondisi.

Daftar Pustaka

Carelli, Ricardo dan Freira, E. Oliveira. 2003." Corridor navigation dan wall-following stable control for sonar-based mobile robots". *Robotics dan Autonomous Systems.* 45. 235 – 247.

Coupland, Simon. 2003. *Type-2 Fuzzy Control of a Mobile Robot*. Leicester: De Montfort University.

Dongrui, Wu 2005. *Design and Analysis of Type-2 Fuzzy Logic Systems*. Singapura: University of Singapore.

Dwiono, Wakhyu 2005. *Perancangan dan Penerapan Sistem Kendali Fuzzy Bertipe 2 Interval*. Bandung: Institut Teknologi Bandung.

Eeles, P, Houston, K & Kozaczynski, W. 2002. *Building J2EE™ Applications with the Rational Unified Process*. Boston: Pierson Education, Inc

Hagras, Hani. 2009. "Practical dan Applications Aspects of Type-2 FLCs". *IEEE International Conference on Fuzzy Systems*, Jeju Island, Korea.

Karnik, Nilesh N. et. al 1999. "Type-2 Fuzzy Logic Systems". *IEEE Transactions On Fuzzy Systems.* 7(6) . 643-657.

Karnik, Nilesh N. dan Mendel, J. 2001, "Centroid of a type-2 fuzzy set," *Information Sciences*. 132. 195-220.

Kruchten, Philippe 2001, *The Rational Unified Process: An Introduction*. Addison-Wesley.

Mendel, Jerry M. 2007, "Type 2 Fuzzy Sets dan Systems: an Overview", *IEEE Computational Intelligence Magazine*. 2 (1). 20 – 29.

Mendel, Jerry M. 2009. "Introduction to Type-2 Fuzzy Sets and Systems". *Lecture Material*. University of Southern California.

Nurmaini, S. and Primanita, A. 2012. Modeling of mobile robot system with control strategy based on type-2 fuzzy logic. *International Journal of Information and Communication Technology Research*, *2*(3), 235-242.

.

Nurmaini, S., Zaiton, S., and Norhayati, D. 2009. An embedded interval type-2 neuro-fuzzy controller for mobile robot navigation. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on* (pp. 4315-4321). IEEE.

Nurmaini, S., and Hashim, S. Z. M. 2008. An embedded fuzzy type-2 controller based sensor behavior for mobile robot. In *Intelligent Systems Design and Applications, 2008. ISDA'08. Eighth International Conference on* (Vol. 2, pp. 29-34). IEEE.

Nurmaini, S. and Tutuko, B., 2011, July. A new control architecture in mobile robot navigation based on IT2neuro-fuzzy controller. In *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on* (pp. 1-7). IEEE.

Negnevitsky, Michael 2005, *Artificial Intelligence: A Guide to Intelligent System*, Second edition, Addison-Wesley.

Ogata, Katsuhiko. 2002. *Modern Control Engineering*, Fourth edition, Prentice-Hall.

Okatan, A and Dimirovski, Georgi M. 2002. "Fuzzy Logic Navigation And Control Of A Non-Holonomic Vacuum Cleaner". *Proceedings of the 10th Mediterranean Conference on Control dan Automation*. Lisbon, Portugal, July 9-12.

Safiotti, Alessandro. 1997. "The uses of fuzzy logic in autonomous robot navigation". *Soft Computing*. 180-197.

Son, C. 1999. "Intelligent process model for robotic part assembly in a partially unstructured environment". *Proceedings of IEE Control Theory Application.* 146 (3). 282-288.

Tunstel, E., de Oliver, M., Berman, S. "Fuzzy Behavior Hierarchies for Multi-Robot Control", *Internaional Jounal of Intelligent Systems*, California Institute of Technology, Pasadena, 2002, vol. 17, pp. 499-470

Zadeh, Lotfi A. 1994. "Fuzzy logic: issues, Contention, and Perspective", Computer Science Division and the Electronics Research Laboratory, Department of EECS, University of California. p. 1.

Zadeh, Lotfi A. 1975. "The concept of a linguistic variable dan its application to approximate reasoning". *Information Science*. 8. 199–249.

Zhang, Ziying, et. al. 2008. "Research on Hierarchical Fuzzy Behavior Learning Of Autonomous Robot". *International Conference on Internet Computing in Science dan Engineering*.

Index

o

HOW TO THINK RICH SERIES (BOOK 1) REVISED EDITION

o

HOW TO THINK RICH SERIES (BOOK 1) REVISED EDITION