IMPLEMENTASI LOGIKA *FUZZY* SEBAGAI PERINTAH GERAKAN TARI PADA ROBOT *HUMANOID* KRSI MENGGUNAKAN SENSOR KAMERA CMUCAM4

SKRIPSI

Diajukan Untuk Memenuhi Sebagian Persyaratan

Memperoleh Gelar Sarjana Teknik

Disusun Oleh:

GLADI BUANA

NIM. 0910630063 - 63

KEMENTERIAN PENDIDIKAN NASIONAL

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2013

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

LEMBAR PERSETUJUAN

IMPLEMENTASI LOGIKA *FUZZY* SEBAGAI PERINTAH GERAKAN TARI PADA ROBOT *HUMANOID* KRSI MENGGUNAKAN SENSOR KAMERA CMUCAM4

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan Untuk Memenuhi Sebagian Persyaratan
Memperoleh Gelar Sarjana Teknik

Disusun Oleh:

GLADI BUANA

NIM. 0910630063 - 63

Telah diperiksa dan disetujui oleh :

Ir. Purwanto, MT.	**M. Aziz Muslim, ST., MT., Ph.D**
NIP. 19540424 198601 1 001	**NIP. 19741203 200012 1 001**

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

LEMBAR PENGESAHAN

IMPLEMENTASI LOGIKA *FUZZY* SEBAGAI PERINTAH GERAKAN TARI PADA ROBOT *HUMANOID* KRSI MENGGUNAKAN SENSOR KAMERA CMUCAM4

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Disusun Oleh:

GLADI BUANA
NIM. 0910630063 - 63

Skripsi ini telah diuji dan dinyatakan lulus pada

tanggal 22 Juli 2013

MAJELIS PENGUJI

Goegoes Dwi Nusantoro, ST., MT.	**Ir. Retnowati, MT.**
NIP. 19711013 200604 1 001	**NIP. 19540720 198203 1 002**

Dr. Ir. Erni Yudaningtyas, MT.
NIP. 19650913 199002 2 001

Mengetahui
Ketua Jurusan Teknik Elektro

Dr. Ir. Sholeh Hadi Pramono, M.S.
NIP. 19580728 198701 1 001

PENGANTAR

Alhamdulillah, puji syukur penulis panjatkan kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi yang berjudul "Implementasi Logika *Fuzzy* sebagai Perintah Gerakan Tari pada Robot *Humanoid* KRSI Menggunakan Sensor Kamera CMUCAM4". Skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya.

Penulis menyadari bahwa penyusunan skripsi ini tidak lepas dari bantuan, bimbingan serta dorongan dari berbagai pihak. Pada kesempatan ini penulis menyampaikan rasa terima kasih kepada:

- Bapak Eko Sijanto, Ibu Sri Budi Utami, kakak Harsya Asul Rahman, serta adik Dewi Sri Mustikasari selaku keluarga besar penulis yang senantiasa mendoakan, memberikan nasihat, kasih sayang, perhatian, dan kesabaran selama ini,
- Bapak Dr. Ir. Sholeh Hadi Pramono, MS. sebagai Ketua Jurusan Teknik Elektro Universitas Brawijaya,
- Bapak M. Aziz Muslim, ST., MT, Ph.D sebagai Sekretaris Jurusan Teknik Elektro Universitas Brawijaya, sekaligus sebagai Dosen Pembimbing II atas segala bimbingan, pengarahan, gagasan, ide, saran serta motivasi yang telah diberikan,
- Bapak Ir. Purwanto, MT. sebagai Ketua Kelompok Dosen Keahlian Sistem Kontrol Jurusan Teknik Elektro Universitas Brawijaya, sekaligus sebagai Dosen Pembimbing I atas segala bimbingan, pengarahan, saran, kritik dan masukan yang telah diberikan,
- Staf Rekording, staf Pengajaran, dan staf Ruang Baca Jurusan Teknik Elektro yang telah membantu segala urusan penulis selama ini,
- Tim KRSI 2013 (IBRAHIM) Jurusan Teknik Elektro Universitas Brawijaya (Aka, Abu, Tansu, Muammar, Emon, Tegar, dan Bustanul) yang dalam setahun terakhir ini hampir setiap malam "ngerobot di GBE lantai 2" bekerjasama dengan penulis sehingga dapat mempersembahkan juara II divisi KRSI pada

tingkat regional IV tahun 2013 ini atas segala dukungan, pinjaman robot IBRAHIM-nya, alat-alatnya dan seluruh bantuannya,

- Rekan - rekan seperjuangan dalam pengerjaan skripsi, Aka, Sam, Adeck, Darmo, Along dkk atas segala dukungan dan bantuan yang telah diberikan,
- Irfan Habibur Rahman dan Riky Tri Yunardi atas skripsinya yang telah menjadi inspirasi untuk skripsi penulis,
- Teman - teman Laboratorium Sistem Kontrol Sam, Adeck, Darmo, Irjan, Sendok, Ika, Hamu, Salmi, dan aslab – aslab siskon lainnya, dan juga Mbak Eka (selaku Laboran Laboratorium Sistem Kontrol) atas doa, dukungan dan seluruh bantuannya,
- Anggota tim robotika TE-UB dari angkatan 2009, Aka, Andy, Eky, Firda, Iksan, Jatra, Juang, Mbah, Risma, Sam, Somad, Wito atas doa, dukungan, bantuan dan kebersamaan selama lebih kurang tiga tahun belakangan.
- Seluruh anggota tim robotika KRSI, KRI, KRCI (KRPAI), dan Nontek periode 2010-2013 atas segala bantuan dan semua kenangan indahnya.
- Semua anggota Ampere 2009 yang memberikan doa, semangat serta dukungan kepada penulis.
- Teman – teman el-kampank untuk doa, semangat, dan kebersamaan dengan penulis selama ini.

Seluruh teman - teman serta semua pihak yang tidak mungkin untuk dicantumkan namanya satu-persatu, terima kasih banyak atas segala bentuk bantuan dan dukungannya.

Dalam penyusunan skripsi ini, penulis menyadari bahwa skripsi ini belumlah sempurna, karena keterbatasaan ilmu dan kendala-kendala lain yang terjadi selama pengerjaan skripsi ini. Oleh karena itu, penulis mengharap kritik dan saran untuk penyempurnaan tulisan di masa yang akan datang. Penulis berharap, semoga tulisan ini dapat bermanfaat bagi kita semua.

Malang, Juli 2013

Penulis

DAFTAR ISI

PENGANTAR iv
DAFTAR ISI vi
DAFTAR GAMBAR ix
DAFTAR TABEL xii
ABSTRAK xiii
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 3
1.4 Tujuan 3
1.5 Sistematika Pembahasan 3
BAB II TINJAUAN PUSTAKA 5
2.1 Robot *Humanoid* 5
2.2 Kontes Robot Seni Indonesia (KRSI) 6
2.3 Motor Servo 8
2.4 Sensor Kamera (CMUcam4) 13
2.5 Mikrokontroler Atmel AVR ATMega128 15
2.6 Mikrokontroler Atmel AVR ATMega8 18
2.7 Logika *Fuzzy* 20
2.7.1 Struktur Dasar Kontrol Logika *Fuzzy* 21
2.7.2 Fungsi Keanggotaan 22
2.7.3 Kendali Logika *Fuzzy* 22
2.7.3.1 Fuzzifikasi 23
2.7.3.2 Kaidah Atur *Fuzzy* (*Fuzzy* Rule) 24
2.7.3.3 Metode Inferensi 24
2.7.3.4 Defuzzifikasi 25
2.8 Model Warna RGB 26
2.8.1 Warna Primer 27
2.8.2 Warna Sekunder dan Warna Tersier 27

BAB III METODELOGI PENELITIAN 29
3.1 Penentuan Spesifikasi Alat 29
3.2 Studi Literatur 30
3.3 Perancangan dan Pembuatan Alat 30
3.3.1 Perancangan dan Pembuatan Perangkat Keras (*Hardware*) 30
3.3.2 Perancangan dan Pembuatan Perangkat Lunak (*Software*) 30
3.4 Pengujian Alat 31
3.4.1 Pengujian Tiap Blok 31
3.4.2 Pengujian Keseluruhan Sistem 31
3.5 Pengambilan Kesimpulan 31
BAB IV PERANCANGAN DAN PEMBUATAN ALAT 32
4.1 Diagram Blok Sistem 32
4.2 Perancangan Perangkat Keras 35
4.2.1 Perancangan Mekanik Robot 35
4.2.2 Sensor Kamera 36
4.2.3 Rangkaian Minimum Sistem Mikrokontroler *Master* 37
4.2.4 Rangkaian Minimum Sistem Mikrokontroler *Slave* 39
4.3 Perancangan Kontrol Logika Fuzzy 40
4.3.1 Variabel Masukan dan Variabel Keluaran 40
4.3.1.1 Fungsi Keanggotaan Masukan pada Warna Dasar Merah 41
4.3.1.2 Fungsi Keanggotaan Masukan pada Warna Dasar Hijau 42
4.3.1.3 Fungsi Keanggotaan Masukan pada Warna Dasar Biru 43
4.3.1.4 Fungsi Keanggotaan Keluaran Logika *Fuzzy* 43
4.3.2 Kaidah Atur (*Rule Base*) Kontrol Logika *Fuzzy* 44
4.3.3 Metode Inferensi *Max-Min* 45
4.3.4 Defuzzifikasi 46
4.4 Perancangan Gerakan 48
4.5 Perancangan Perangkat Lunak 52
4.5.1 Pengaturan Sensor Kamera 52

4.5.2 Perancangan Algoritma Mikrokontroler *Slave* 53
4.5.2.1 Perancangan Subrutin Mengambil Data Unit *Decoder*.....54
4.5.2.2 Perancangan Subrutin Memilah Data 55
4.5.2.3 Perancangan Subrutin Mengubah Karakter ASCII ke Nilai *Integer*56
4.5.3 Perancangan Algoritma Mikrokontroler *Master*.....57
BAB V PENGUJIAN DAN ANALISIS 58
5.1 Pengujian Sensor Kamera 58
5.2 Pengujian Logika *Fuzzy* 61
5.3 Pengujian Keseluruhan Sistem.....67
BAB VI KESIMPULAN DAN SARAN.....72
6.1 Kesimpulan 72
6.2 Saran.....72
DAFTAR PUSTAKA 73
LAMPIRAN.....74

DAFTAR GAMBAR

Gambar 2.1. Robot *Bioloid* 6
Gambar 2.2. Lintasan Perlombaan KRSI 2013 7
Gambar 2.3. Motor Servo Parallax 9
Gambar 2.4. Konfigurasi Pin Pengkabelan Motor Servo 10
Gambar 2.5. Pengaturan Sudut Motor Servo 11
Gambar 2.6. Sensor Kamera CMUCAM4 13
Gambar 2.7. Konfigurasi Pin ATMega128 15
Gambar 2.8. Konfigurasi Pin ATMega8 19
Gambar 2.9. Pengendali *Fuzzy* 21
Gambar 2.10. Fungsi Keanggotaan Bentuk Triangular (T) 22
Gambar 2.11.Inferensi *Fuzzy* dengan Metode Max-min 25
Gambar 2.12. Metode Max Membership Principle 26
Gambar 2.13. Kombinasi Warna Benda 27
Gambar 2.14. Warna-warna Sekunder Hasil Kombinasi RGB 28
Gambar 2.15. Warna-warna Sekunder dari Kombinasi Warna Primer RGB 28
Gambar 2.16. Warna-warna Tersier dari Kombinasi Warna Primer 28
Gambar 4.1. Diagram Blok Sistem Pengaturan 32
Gambar 4.2. Diagram Blok Sistem Robot *Humanoid* 33
Gambar 4.3. Desain Mekanik Robot Tampak Depan 35
Gambar 4.4. Desain Mekanik Robot Tampak Samping Kanan 36
Gambar 4.5. Penampang Rangkaian CMUCAM4 37
Gambar 4.6. Rangkaian Minimum Sistem Mikrokontroler *Master* 38
Gambar 4.7. Rangkaian Minimum Sistem Mikrokontroler *Slave* 39
Gambar 4.8. Diagram Blok Perancangan Kontrol Logika *Fuzzy* 40

Gambar 4.9. Fungsi Keanggotaan Masukan pada Warna Dasar Merah 41

Gambar 4.10. Derajat Keanggotaan Warna Merah(μ_R) pada Data Masukan Merah 135 41

Gambar 4.11. Fungsi Keanggotaan Masukan pada Warna Dasar Hijau 42

Gambar 4.12. Derajat Keanggotaan Warna Hijau(μ_H) pada Data Masukan Hijau 140 42

Gambar 4.13. Fungsi Keanggotaan Masukan pada Warna Dasar Biru 43

Gambar 4.14. Derajat Keanggotaan Warna Biru (μ_B) pada Data Masukan Biru 147 43

Gambar 4.15. Fungsi Keanggotaan Keluaran Logika *Fuzzy* 44

Gambar 4.16. Ilustrasi Metode Inferensi Max-min (1).....45

Gambar 4.17. Ilustrasi Metode Inferensi Max-min (2).....46

Gambar 4.18. Ilustrasi Proses Defuzzifikasi *Max Membership Principle*.....47

Gambar 4.19. Lapangan KRSI 2013.....48

Gambar 4.20. Gerakan *Sembah Pembuka* 49

Gambar 4.21. Gerakan *Ngasak*.....50

Gambar 4.22. Gerakan *Membelah Awan* 51

Gambar 4.23. Gerakan Menghondari Zona Larang 51

Gambar 4.24. Gerakan *Memberi Cincin*.....52

Gambar 4.25. Diagram Alir Program Utama Kontrol Logika *Fuzzy*.....54

Gambar 4.26. Diagram Alir Subrutin Mengambil Data Unit *Decoder* 55

Gambar 4.27. Diagram Alir Subrutin Memilah data 56

Gambar 4.28. Diagram Alir Subrutin Mengubah Karakter ASCII ke Nilai *Integer*57

Gambar 4.29. Diagram Alir Perancangan Perangkat Lunak 57

Gambar 4.30. Diagram Alir Perancangan Sub-fungsi Gerakan Robot.....59

Gambar 5.1. Rangkaian Pengujian Sensor Kamera.....60

Gambar 5.2. Hasil Pembacaan Data ASCII RGB CMUCAM4 61

Gambar 5.3. Proses Logika *Fuzzy* Masukan Warna Biru 62
Gambar 5.4. Proses Logika Fuzzy Masukan Warna Biru Muda 62
Gambar 5.5. Proses Logika Fuzzy Masukan Warna Hijau.....63
Gambar 5.6. Proses Logika Fuzzy Masukan Warna Merah 64
Gambar 5.7. Proses Logika Fuzzy Masukan Warna Kuning.....64
Gambar 5.8. Proses Logika Fuzzy Masukan Warna Putih 65
Gambar 5.9. Proses Inferensi Logika *Fuzzy* Data Input Warna Merah 66
Gambar 5.10. Pengujian Zona Mulai (Mode Biru) Gerakan *Sembah Pembuka*....67
Gambar 5.11. Pengujian Zona Mulai (Mode Merah) Gerakan *Sembah Pembuka* 68
Gambar 5.12. Pengujian Zona I Gerakan *Ngasak* 69
Gambar 5.13. Pengujian Zona II Gerakan *Membelah Awan* 70
Gambar 5.14. Pengujian Menghindari Zona Larang 70
Gambar 5.15. Pengujian Zona III Gerakan *Memberi Cincin*.....71

DAFTAR TABEL

Tabel 4.1. Kaidah Atur Kombinasi warna RGB (*Red Green Blue*).....44

Tabel 5.1. Hasil Pengujian Logika *Fuzzy* 66

Tabel 5.2. Pengujian Zona Mulai (Mode Biru) 67

Tabel 5.3. Pengujian Zona Mulai (Mode Merah) 68

Tabel 5.4. Pengujian Zona I.....69

Tabel 5.5. Pengujian Zona II 69

Tabel 5.6. Pengujian Menghindari Zona Larang 70

Tabel 5.7. Pengujian Zona III 71

repository.ub.ac.id

ABSTRAK

Gladi Buana, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Juli 2013, *Implementasi Logika Fuzzy sebagai Perintah Gerakan Tari pada Robot Humanoid KRSI Menggunakan Sensor Kamera CMUCAM4*, Dosen Pembimbing: Ir. Purwanto, MT. dan M. Aziz Muslim, ST., MT., Ph.D.

Pada Kontes Robot Seni Indonesia (KRSI) robot diharuskan melakukan gerakan tari yang berbeda pada tiap zona lapangan. Zona lapangan ini ditandai dengan warna yang berbeda. Untuk keperluan tersebut, pada penelitian ini dikembangkan metode pengenalan warna untuk mendeteksi zona lapangan KRSI. Metode yang digunakan untuk pengaturan ini ialah logika *fuzzy*. Metode ini melalui beberapa tahap, yaitu fuzzifikasi, kaidah atur (*Rule-base)* dan inferensi, dan defuzzifikasi.

Penelitian ini menggunakan sensor kamera CMUCAM4 untuk pendeteksian warna pada lintasan. Data dari sensor diproses dengan logika *fuzzy* menggnakan mikrokontroler ATMega8 sebagai mikrokontroler *slave.* Data terkontrol dikirim ke mikrokontroler *master* yaitu ATMega128 untuk kemudian diproses dan dikirimkan ke motor DC servo sebagai aktuator.

Hasil pengujian menunjukkan bahwa metode logika *fuzzy* cukup efektif digunakan untuk mendeteksi warna lapangan. Dari beberapa hasil uji yang dilakukan robot *humanoid* dapat melaksanakan perintah sesuai dengan yang diinginkan.

Kata Kunci: Robot *Humanoid*, KRSI, Perintah Gerakan Tari, Logika *Fuzzy*, Sensor Kamera CMUCAM4.

UNIVERSITAS BRAWIJAYA

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan pesat dunia teknologi dewasa ini tak lepas dari peran serta kemajuan robotika yang didukung oleh perkembangan ilmu komputer dan sistem kontrol. Penelitian dari ilmuwan-ilmuwan saat ini untuk menunjang proses perindustrian maupun sebagai hiburan semata, banyak juga yang tertuju pada suatu mekanisme pergerakan robot menyerupai manusia (*humanoid* robot).

Kontes Robot Seni Indonesia (KRSI) merupakan suatu kontes resmi dari Direktorat Jenderal Pendidikan Tinggi (Dirjen Dikti) Indonesia dimana menampilkan berbagai macam kreativitas robot (dengan mengutamakan unsur-unsur seni di dalamnya) dari setiap mahasiswa seluruh perguruan tinggi yang ada di Indonesia. Tujuan dari kontes robot ini adalah untuk menumbuh-kembangkan kreativitas dan minat para mahasiswa dalam kemajuan teknologi, khususnya teknologi robotika yang selain diperuntukkan untuk sistemnya juga diharapkan dapat membantu kegiatan manusia serta mengaplikasikan ilmu pengetahuan dan teknologi ke dalam dunia nyata dan meningkatkan kepekaan mahasiswa dalam pengembangan bidang teknologi robotika, dan membudayakan iklim kompetitif di lingkungan perguruan tinggi. Divisi KRSI merupakan divisi dalam Kontes Robot mengambil tema robot penari untuk menampilkan seni budaya yang diinginkan sesuai dengan tema setiap tahun yang berbeda-beda.

Robot penari ini harus mampu berjalan mengikuti lintasan perlombaan yang telah disediakan dengan berbagai macam warna. Agar membuat robot ini berjalan dibutuhkan suatu sistem kaki yang terdiri dari beberapa motor servo. Sedangkan untuk membuat robot suatu sistem tangan juga dibutuhkan beberapa motor servo.

Lintasan dalam perlombaan KRSI terdiri dari bermacam-macam warna pada tiap-tiap zonanya, untuk mengetahui lintasan (dengan warna tertentu) mana yang harus dilewati dan juga sebagai pendeteksi tarian apa yang harus dilakukan pada zona yang sedang ditempati oleh robot diperlukan sistem pengambilan keputusan

dengan memanfaatkan pengolahan data yang diperoleh dari sensor kamera CMUcam4.

Cara pendeteksian dan pengambilan keputusan permasalahan tersebut sebenarnya sebelumnya juga pernah diteliti oleh Irfan Habibur Rahman, ST. lulusan S1 Teknik Elektro Universitas Brawijaya pada tahun 2012, yang mengangkat judul "Impelementasi Sensor Warna TCS 230 pada Robot Humanoid KRSI Sebagai Mode Tarian di Arena Perlombaan". Namun peneliti sendiri menyadari bahwa ada kekurangan dan kelemahan pada penelitian dengan penggunaan sensor warna TCS 230, sehingga peneliti memberikan saran agar menggunakan sensor kamera.

Kontrol Logika *Fuzzy* merupakan kontroler *Multiple input-multiple output*, dimana kontroler tersebut dapat memproses lebih dari satu input dan dapat menghasilkan output lebih dari satu pula. Sedangkan komponen warna dasar yang dapat dialokasikan sebagai input berjumlah tiga buah.

Berdasarkan kondisi-kondisi tersebut, dengan menggunakan metode kontrol logika *fuzzy* diharapkan dapat melakukan pengambilan keputusan dari permasalahan tersebut diatas. Sehingga dari mikrokontroler yang digunakan untuk mengatur putaran sudut motor servo akan didapatkan gerakan dari *humanoid* robot sesuai dengan yang diinginkan.

1.2 Rumusan Masalah

Berdasarkan uraian dalam latar belakang, dapat disusun rumusan masalah sebagai berikut :

1) Bagaimana merancang dan membuat robot *humanoid* yang dapat mengikuti lintasan perlombaan dan melakukan gerakan sesuai dengan yang direncanakan dengan menggunakan sensor kamera.
2) Bagaimana merancang dan membuat suatu sistem pengaturan menggunakan kontroler kontrol logika f*uzzy* yang dapat menghasilkan respon sesuai dengan yang direncanakan.

1.3 Batasan Masalah

Dalam Perancangan untuk sikripsi ini permasalahan dibatasi oleh hal-hal sebagai berikut :

1) Robot yang digunakan adalah IBRAHIM (Robot *Humanoid* KRSI Elektro Brawijaya 2013).
2) Lintasan yang digunakan sesuai aturan dalam Kontes Robot Seni Indonesia (KRSI) tahun 2013.
3) Pembahasan ditekankan pada penggunaan kontroler kontrol logika f*uzzy* pada sistem.
4) Pencahayaan dilakukan pada kondisi yang sama (intensitas cahaya ± 210 lux).
5) Sensor kamera yang digunakan adalah CMUcam4.
6) Aktuator yang digunakan adalah motor DC servo.
7) Pemrograman mikrokontroler (slave) *ATmega8* dengan menggunakan software *CodeVisionAVR C Compiler* untuk pengolahan data dari CMUcam4.
8) Pemrograman mikrokontroler (master) *ATmega128* dengan menggunakan software *CodeVisionAVR C Compiler* untuk pengiriman sinyal ke motor servo.

1.4 Tujuan

Penelitian ini bertujuan untuk merancang dan membuat robot *humanoid* yang dapat mengikuti lintasan perlombaan KRSI 2013 dan melakukan gerakan sesuai dengan yang direncanakan dengan menggunakan sensor kamera sebagai pendeteksi warna pada lintasan.

1.5 Sistematika Pembahasan

Skripsi ini terdiri dari enam bab dengan sistematika pembahasan sebagai berikut:

BAB I Pendahuluan

Membahas latar belakang, rumusan masalah, batasan masalah, tujuan dan sistematika pembahasan.

BAB II Tinjauan Pustaka

Membahas teori-teori yang mendukung dalam perancangan dan pembuatan alat.

BAB III Metodologi Penulisan

Membahas metode penelitian dan perencanaan alat.

BAB IV Perancangan dan Pembuatan Alat

Membahas perancangan kontroler kontrol logika *fuzzy* dan menerapkannya ke dalam *software* di komputer, sehingga sistem dapat bekerja dengan baik.

BAB V Pengujian dan Analisis

Membahas hasil pengujian sistem yang sudah dibuat, serta analisis hasil yang diperoleh.

Bab VI Kesimpulan dan Saran

Membahas kesimpulan perancangan ini dan saran-saran yang diperlukan untuk pengembangan selanjutnya.

BAB II
TINJAUAN PUSTAKA

Dalam merencanakan dan merealisasikan sebuah robot *humanoid* ini, maka dibutuhkan pemahaman tentang berbagai hal yang mendukung sistem ini. Pemahaman ini akan bermanfaat untuk merancang perangkat keras dan perangkat lunak sistem. Pengetahuan yang mendukung perencanaan dan realisasi alat meliputi robot *humanoid*, Kontes Robot Seni Indonesia (KRSI) 2013, motor servo, sensor kamera (CMUcam4), Mikrokontroler Atmel AVR ATMega128, Mikrokontroler Atmel AVR ATMega128, Kontrol Logika *Fuzzy*, dan Model Warna RGB.

2.1 Robot *Humanoid*

Robot telah mengalami perkembangan yang sangat pesat. Ketertarikan dunia pada bidang robotika begitu besar , dengan ditandai banyaknya penggunaan robot di berbagai bidang, seperti bidang industri, pertanian, kedokteran, dan juga militer. Tujuan utama pemanfaatan teknologi robotika pada berbagai bidang tersebut tidak lain adalah untuk mempermudah kerja manusia. Seperti definisi dari kata robot itu sendiri yang berarti “pekerja”, keberadaan robot untuk membantu pekerjaan manusia memang tidak bisa dipungkiri. Salah satu alasan penggunaan teknologi robot adalah karena robot memiliki kemampuan khusus untuk menyelesaikan suatu pekerjaan tertentu, dengan tingkat akurasi yang tinggi dan mampu mengerjakan sesuatu dalam waktu yang relatif lebih cepat jika dibandingkan dengan manusia.

Seiring dengan perkembangan zaman, robot kini memiliki program dan fungsi yang lebih kompleks. Robot dapat dikelompokkan menjadi beberapa jenis sesuai dengan spesifikasinya, misalkan fungsi dan alat gerak. Sesuai fungsinya, ada beberapa jenis robot, diantaranya robot penjelajah, robot industri, robot pemadam api, robot medis, robot nano, robot perang, dan robot kompetisi. Jenis robot sesuai dengan alat geraknya adalah robot beroda dan robot berkaki.

Robot *humanoid* termasuk dalam robot berkaki dua, robot berkaki dua adalah robot yang memiliki dua kaki dan bergerak menyerupai manusia atau hewan berkaki dua. Robot jenis ini bergerak dengan mengayunkan salah satu kaki, kemudian diikuti ayunan kaki lainnya. Kaki robot memiliki jumlah *Degree of Freedom (DOF)* atau

derajat kebebasan dengan jumlah yang relatif banyak. Untuk merepresentasikan secara keseluruhan kaki manusia dan hewan, diperlukan DOF dalam jumlah yang banyak yang sesuai dengan jumlah sendi dan otot pada kaki manusia dan hewan. Semakin banyak jumlah DOF yang dimiliki kaki robot, maka semakin banyak kerja dan kombinasi gerak kaki yang mampu dilakukan. Dengan begitu sangat memungkinkan bagi robot berkaki dua untuk melakukan pergerakan yang ekstrim seperti berlari, melompat, menaiki tangga, menendang bola, jongkok, duduk, dan berbagai macam gerakan yang tidak mungkin dilakukan oleh robot dengan kaki lebih dari dua. Hal tersebut yang menyebabkan tingginya apresiasi terhadap robot berkaki dua terutama dalam bidang ilmu pengetahuan. Gambar 2.1 menunjukkan salah satu robot *humanoid* yang diberi nama *Bioloid*.

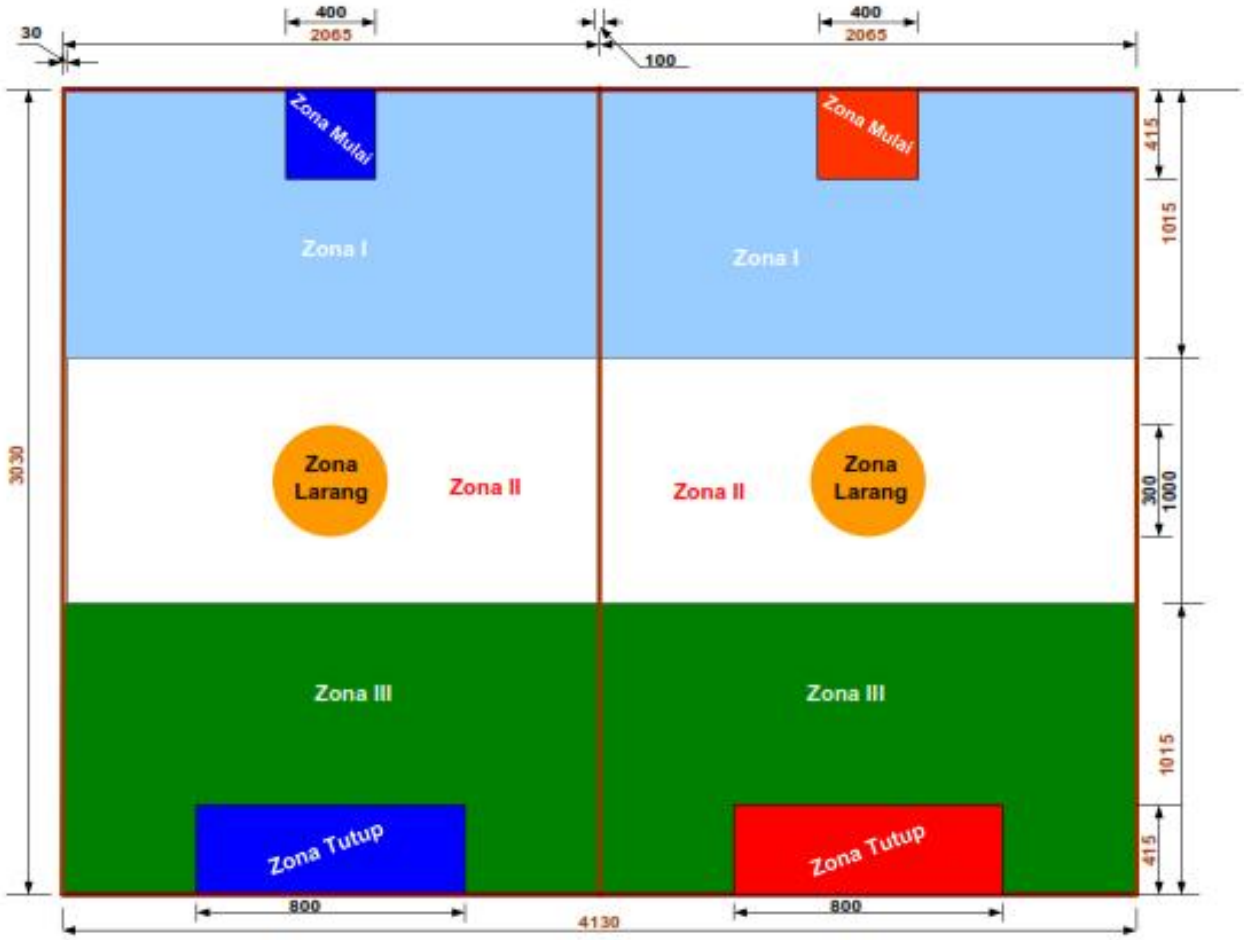
Gambar 2.1. Robot *Bioloid*
Sumber: Bioloid, 2013

2.2 Kontes Robot Seni Indonesia (KRSI)

Kontes Robot Seni Indonesia (KRSI) merupakan suatu ajang kompetisi perancangan dan pembuatan robot yang disertai dengan unsur-unsur seni dan budaya bangsa yang telah terkenal di bumi pertiwi. KRSI pertama kali diadakan pada tahun 2009 yang mengangkat tema Robot Penari Jaipong, pada tahun 2010 dengan mengangkat tema Robot Penari Pendet, tahun 2011 bertemakan Robot Penari Klono Topeng, dan tema Robot Penari Piring diangkat pada tahun 2012. Setiap tim peserta yang terdiri dari 3 (tiga) mahasiswa dengan seorang dosen pembimbing, diwajibkan untuk membuat satu robot untuk menampilkan seni budaya yang diinginkan sesuai tema kontes. Untuk KRSI 2013, sesuai dengan momentum yang tepat dalam gema

nasional membangkitkan kecintaan dan pelestarian budaya-budaya Nasional maka tema yang diangkat adalah “Robot Hanoman Duta”.

Robot penari yang digunakan harus menyerupai struktur tubuh manusia dengan tinggi 55±5cm diukur dari titik teratas robot hingga titik terbawah. Berat robot total maksimum adalah 30kg. Lintasan perlombaan berbentuk persegi-panjang berukuran (3000 x 2000)cm yang terdiri dari warna hijau gelap (zona III), putih (zona II), biru muda (zona I), kuning (zona larang), merah ataupun biru tua (zona pembuka dan tutup). Gambar 2.2 menunjukkan lapangan yang digunakan dalam pertandingan.



Gambar 2.2 Lintasan Perlombaan KRSI 2013

Sumber: DIKTI, 2013

Tiap arena satu robot memiliki tiga (5) zona dan zona larang, bila diurutkan dari zona mulai hingga tutup adalah Zona Mulai, Zona I, Zona II, Zona Larang, Zona III dan Zona Tutup. Dimana Zona Larang merupakan zona robot dilarang melintas. Tiap zona ini berfungsi sebagai pemandu gerakan. Gerakan tari harus dilakukan berdasarkan irama gamelan pengiring dengan cara mendengarkan langsung dari sistem audio gedung lomba menggunakan sensor (tanpa kabel).

Di Zona Mulai, setelah musik pengiring dimulai dan robot harus bisa melakukan sembah pembuka. Di Zona I robot harus melakukan gerak *Ngasak, kiprah* dan *kelat bahu*. Di Zona II robot melakukan gerak *Capingan, ulap-ulap,* dan *mbelah awan*. Gerak ini dilakukan berulang-ulang hingga akhir zona tengah. Saat robot melakukan gerak *Capingan, ulap-ulap,* dan *mbelah awan* bagian robot tidak diperbolehkan menyentuh Zona Larang. Di dalam Zona III ini robot harus melakukan gerak *putar gelung, memberi cincin* dan *gebesan*. Di Zona Tutup robot harus mampu melakukan gerak tari *sembah penutup tari Hanoman*. Ketika irama pengiring gerak tari berakhir menjelang menit ke tiga akan terdengar tepukan penonton. Ketika mendengar tepukan penonton ini robot harus menghentikan tariannya dan memberi salam hormat sebagai tanda telah selesainya tarian.

Musik gamelan pengiring tari robot adalah gamelan khas tari Hanoman Duta yang disediakan oleh panitia. Waktu yang disediakan untuk setiap unjuk kebolehan tari dalam lomba ini adalah tiga (3) menit lima puluh luma (55) detik sesuai dengan panjang atau durasi irama gamelan pengiring. Dalam setiap game tim diberikan kesempatan *retry* tidak dibatasi. Dalam setiap sesi pertandingan, dua robot dari tim peserta akan diletakkan diatas panggung (lapangan perlombaan) sesuai dengan warna tim awal, yaitu merah atau biru.

2.3 Motor Servo

Berbeda dengan motor DC dan motor Stepper, motor servo adalah sebuah motor dengan sistem *closed feedback* di mana posisi dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. Motor ini terdiri atas sebuah motor, serangkaian *internal gear*, potensiometer dan rangkaian kontrol. Potensiometer berfungsi untuk menentukan batas sudut putaran servo. Sedangkan sudut sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor.

Motor servo mampu bekerja dua arah *(CW* dan *CCW)* dimana arah dan sudut pergerakan rotornya dapat dikendalikan hanya dengan memberikan pengaturan *duty cycle* sinyal *PWM* pada bagian pin kontrolnya. Gambar 2.3 menunjukkan gambar motor servo.

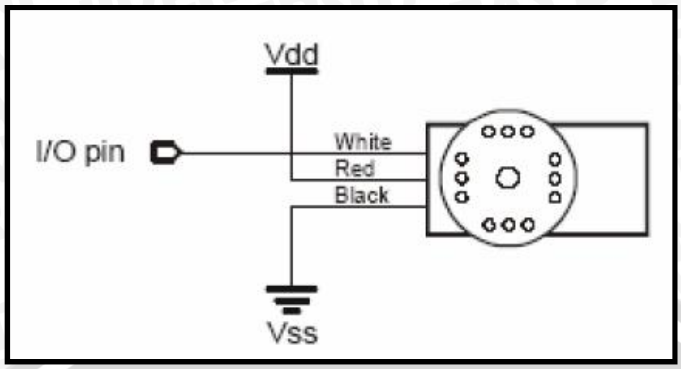
Gambar 2.3. Motor Servo Parallax

Sumber : Parallax, 2010

Motor servo merupakan motor yang berputar lambat, dimana biasanya ditunjukkan oleh rate putarannya yang lambat, namun demikian memiliki torsi yang kuat karena *internal gear*-nya. Karakteristik motor servo adalah sebagai berikut :

- Memiliki 3 jalur kabel : *power*, *ground, dan control* seperti ditunjukkan dalam Gambar 2.4
- Pin *Control* untuk mengendalikan posisi.
- Konstruksi didalamnya meliputi *internal gear*, *potensiometer*, dan *feedback control.*

Gambar 2.4 menunjukkan konfigurasi pin pengkabelan motor servo.



Gambar 2.4 Konfigurasi Pin Pengkabelan Motor Servo

Sumber : Parallax, 2010

Secara umum terdapat dua jenis motor servo, yaitu :

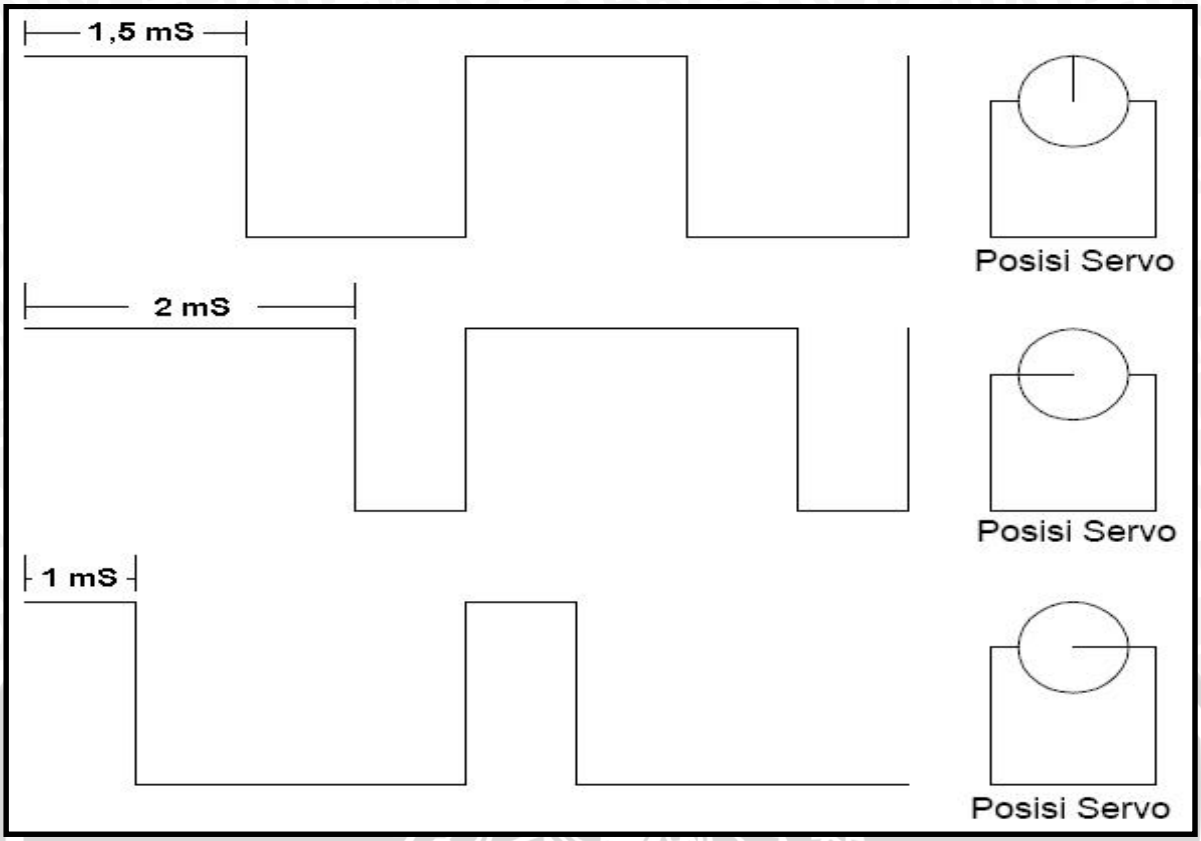
- Motor Servo *Standar 180°*

  Motor servo jenis ini merupakan motor yang hanya mampu bergerak dua arah (*CW* dan *CCW)* dan mempunyai *defleksi* masing-masing sudut mencapai 90° sehingga total *defleksi* sudut dari kanan – tengah – kiri adalah 180°.

- Motor Servo *Continuous*

  Motor servo jenis ini mampu bergerak dua arah *(CW* dan *CCW)* dan tanpa batasan *defleksi* sudut putar (dapat berputar secara kontinyu) sehingga motor ini berputar 360°.

Pengaturan sudut motor servo diperlukan untuk mengetahui gerakan dari motor servo dan pulsa yang harus diberikan untuk bergerak ke kanan atau bergerak ke kiri. Gambar 2.5 menunjukkan teknik PWM untuk mengatur sudut motor servo.



Gambar 2.5 Pengaturan Sudut Motor Servo

Sumber : Parallax, 2010

Dalam Gambar 2.5 diasumsikan bahwa saat diberikan sinyal periodik dengan lebar 1 ms maka motor servo akan bergerak dengan sudut 0^o, jika diberi sinyal 1,5 ms maka motor servo akan bergerak dengan sudut 90^o, dan jika diberi sinyal 2 ms maka motor servo akan bergerak dengan sudut 180^o. Dari kondisi tersebut bisa ditarik kesimpulan bahwa untuk mengatur sudut putaran motor servo diperlukan sinyal dengan lebar pulsa seperti pada persamaan berikut:

$$S = D\left(\frac{1000}{180}\right) + 1000\,\mu s \quad \text{atau}$$

$$S = (5{,}56)D + 1000\,\mu s$$

$$S = Lebar\ Pulsa\ dalam\ \mu s$$

$$D = Sudut\ putar\ servo\ dalam\ derajat$$

Motor servo yang digunakan adalah:

1.Merk **Hitec** dengan tipe **HS-7954SH**. Tipe tersebut memiliki spesifikasi sebagai berikut :

- Torsi: 29 kg-cm pada 7,4 V DC, 24 kg-cm pada 6,0 V DC, dan 20 kg-cm pada 4,8 V DC.
- Kecepatan: 0,12 detik/60° pada 7,4 V DC, 0,15 detik/60° pada 6 V DC dan 0,18 detik/60° pada 4,8 V DC.
- Dimensi: 40 x 20 x 37 mm.
- Berat: 68 gr.

2. Merk Hitec dengan tipe HS-5685MH. Tipe tersebut memiliki spesifikasi sebagai berikut :

- Torsi: 12,9 kg-cm pada 7,4 V DC dan 11,3 kg-cm pada 6,0 V DC.
- Kecepatan: 0,17 detik/60° pada 7,4 V DC dan 0,20 detik/60° pada 6 V DC.
- Dimensi: 40,6 x 19,8 x 37,8 mm.
- Berat: 60 gr.

3. Merk GWS dengan tipe micro 2BBMG. Tipe tersebut memiliki spesifikasi sebagai berikut :

- Torsi: 5,4 kg-cm pada 4,8 V DC.
- Kecepatan: 0,17 detik/60° pada 4,8 V DC.
- Dimensi: 28 x 14 x 29,8 mm.
- Berat: 28 gr.

2.4 Sensor Kamera (CMUcam4)

CMUcam4 adalah sebuah sensor komputer *vision* tertanam yang *programable* sepenuhnya. Prosesor utamanya adalah Parallax P8X32A (*Propeller Chip*) yang terhubung ke 9665 CMOS modul sensor kamera OmniVision. Kamera CMUcam4 ini memiliki spesifikasi sebagai berikut:

a. Resolusi (VGA) maksimum 640 x 480
b. *Image prosessing rate* 30 frame per detik
c. Komunikasi serial dengan *baudrate* 19200 *bits per second*
d. Menentukan titik tengah dari objek.
e. Mendapatkan nilai tengah warna dan variasinya.
f. Secara otomatis dapat mendeteksi warna objek
g. *Window* kamera yang dapat diubah-ubah.
h. Mentransfer gambar yang didapat melalui serial *port*.

Gambar 2.6 menunjukkan bentuk *hardware* dari kamera CMUcam4.

Gambar 2.6 Sensor Kamera CMUcam4
Sumber: CMUcam4 *Testing Guide* v1.02, 2012

CMUcam4 adalah *platform* perangkat keras yang dikembangkan oleh perusahaan *open source development.* Kamera ini ditergetkan kepada pengguna yang sudah terbiasa di bidang pengolahan gambar dasar dan kompatibel dengan

pemrograman mikrokontroler. Dapat digunakan untuk beberapa aplikasi *interfacing* dengan menggunakan komputer atau mikrokontroler melalui sistem komunikasi serial TTL atau RS232. Fungsi-fungsi dari kamera sebelumnya (CMUcam4) sudah terdapat pada kamera ini. Sehingga kamera ini cocok digunakan sebagai bahan riset. Berbagai penggunaannya antara lain:

a. Robotika.
b. Jaringan sensor.
c. Pelacakan objek.
d. Sistem pengawas.
e. Pendidikan.
f. Mainan interaktif.

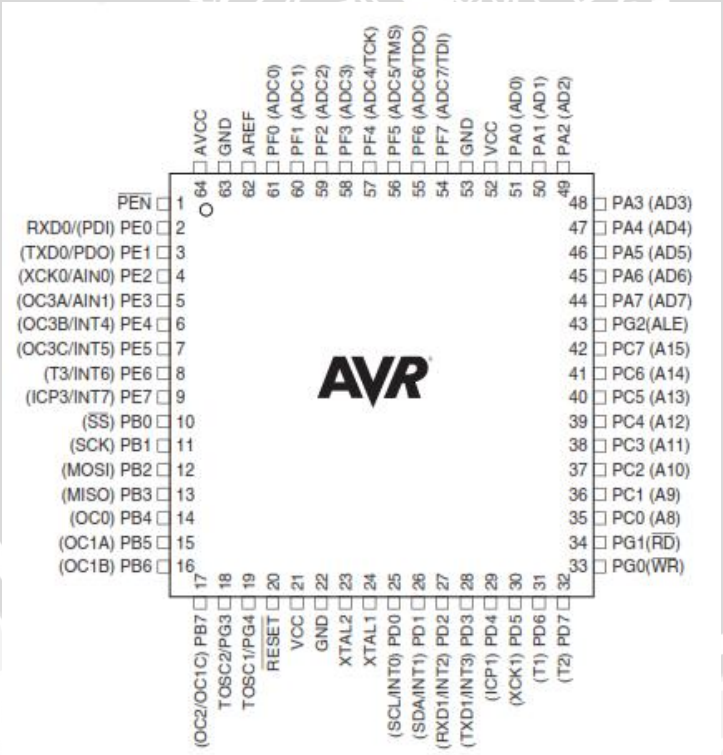
Kamera CMUcam4 bisa berkomunikasi melalui serial port dengan level TTL ataupun RS232 untuk mengirim perintah ke CMUcam4 dan mengambil data dari CMUcam4. Adapun parameter *default Baudrate-nya* sebesar 19200 (merupakan nilai *baudrate* maksimum).

Semua perintah yang dikirim menggunakan karakter ASCII (123 dikirim misalnya dengan mengirimkan 3 byte "123", tambahan 123 dapat dikirim sebagai nilai heksadesimal dengan mengirimkan "0x7B"). Setelah transmisi sukses dari perintah, "ACK" string dikembalikan oleh sistem. Jika ada masalah dalam sintaks transmisi, atau jika terjadi kesalahan pemindahan terdeteksi, sebuah "NCK" *string* dikembalikan. Setelah baik sebagai "ACK" atau "NCK" *string*, sebuah *prompt* '\ r' yaitu *return* (kembali). Ketika *prompt* ('\ r' diikuti oleh ':') dikembalikan, itu berarti bahwa kamera sedang menunggu perintah yang lain dalam keadaan siaga. Putih spasi (atau tab) memang penting dan digunakan untuk parameter argumen terpisah. Dikutip *string* diperlakukan sebagai parameter argumen tunggal - ini sangat berguna untuk operasi sistem file. The '\ r' (ASCII 13 - *carriage return*) digunakan untuk mengakhiri setiap baris dan mengaktifkan setiap perintah. Serial *timeout* untuk semua operasi *non-file* sistem 500 ms dan harus ada batas waktu serial untuk semua operasi sistem file.

2.5 Mikrokontroler Atmel AVR ATMega128

Memiliki fitur seperti:

- Memiliki 128Kbytes dari *In-System Selfprogramable Flash program memory*.
- EEPROM 4Kbyte.
- Internal SRAM 4Kbyte.
- *Write/erase cycles:* 10,000 *Flash*/100,000 EEPROM.
- Penyimpanan data: 20 tahun pada suhu 85°C/100 tahun pada suhu 25°C.
- Hingga 64Kbytes *Optional External Memory Space.*
- Komunikasi serial USART yang dapat diprogram.
- 2 buah 8-bit Timer/Counter dengan *Separate Prescalers* dan *Compare Modes*.
- 2 buah 16-bit Timer/Counter dengan *Separate Prescalers*, *Compare Mode*, dan *Capture Mode.*
- 53 jalur I/O *programable.*
- Tegangan operasi: 2.7 – 5.5 V untuk ATMega128L dan 4.5 – 5.5 Vuntuk ATMega128



Gambar 2.7 Konfigurasi Pin ATMega128

Sumber: ATMEL, 2011

Konfigurasi pin mikrokontroler ATMega128 meliputi:

- VCC : Suplai tegangan.
- GND : Ground.
- Port A (PA7..PA0) : *bi-directional* port I/O 8-bit dengan *internal pull-up resistor* (dipilih untuk setiap bit). Port A keluaran *buffer* memiliki karakteristik simetris *drive* dengan kedua *high sink* dan kemampuan sumber. Sebagai masukan, Port A pin yang rendah akan sumber arus eksternal ditarik jika *pull-up* resistor diaktifkan. Port pin A adalah *tri-stated* ketika kondisi reset menjadi aktif, bahkan jika *clock* tidak berjalan.
- Port B (PB7..PB0) : *bi-directional* port I/O 8-bit dengan *internal pull-up resistor* (dipilih untuk setiap bit). Port B keluaran *buffer* memiliki karakteristik simetris *drive* dengan kedua *high sink* dan kemampuan sumber. Sebagai masukan, Port B pin yang rendah akan sumber arus eksternal ditarik jika *pull-up* resistor diaktifkan. Port pin B adalah *tri-stated* ketika kondisi reset menjadi aktif, bahkan jika *clock* tidak berjalan.
- Port C (PC7..PC0) : *bi-directional* port I/O 8-bit dengan *internal pull-up resistor* (dipilih untuk setiap bit). Port C keluaran *buffer* memiliki karakteristik simetris *drive* dengan kedua *high sink* dan kemampuan sumber. Sebagai masukan, Port C pin yang rendah akan sumber arus eksternal ditarik jika *pull-up* resistor diaktifkan. Port pin C adalah *tri-stated* ketika kondisi reset menjadi aktif, bahkan jika *clock* tidak berjalan.
- Port D (PD7..PD0) : *bi-directional* port I/O 8-bit dengan *internal pull-up resistor* (dipilih untuk setiap bit). Port D keluaran *buffer* memiliki karakteristik simetris *drive* dengan kedua *high sink* dan kemampuan sumber. Sebagai masukan, Port D pin yang rendah akan sumber arus eksternal ditarik jika *pull-up* resistor diaktifkan. Port pin D adalah *tri-stated* ketika kondisi reset menjadi aktif, bahkan jika *clock* tidak berjalan.
- Port E (PE7..PE0) : *bi-directional* port I/O 8-bit dengan *internal pull-up resistor* (dipilih untuk setiap bit). Port E keluaran *buffer* memiliki karakteristik simetris *drive* dengan kedua *high sink* dan kemampuan sumber. Sebagai masukan, Port E pin yang rendah akan sumber arus eksternal ditarik

jika *pull-up* resistor diaktifkan. Port pin E adalah *tri-stated* ketika kondisi reset menjadi aktif, bahkan jika *clock* tidak berjalan.

- Port F (PF7..PF0) : Port F menyediakan konversi input analog ke A/D konverter. Jika A/D konverter tidak aktif, Port F juga menyediakan *bi-directional* port I/O 8-bit dengan *internal pull-up resistor* (dipilih untuk setiap bit). Port F keluaran *buffer* memiliki karakteristik simetris *drive* dengan kedua *high sink* dan kemampuan sumber. Sebagai masukan, Port F pin yang rendah akan sumber arus eksternal ditarik jika *pull-up* resistor diaktifkan. Port pin E adalah *tri-stated* ketika kondisi reset menjadi aktif, bahkan jika *clock* tidak berjalan. Jika *interface* JTAG dalam keadaan *enable*, resistor *pull-up* pada pin PF7 (TDI), PF5 (TMS), and PF4 (TVK) akan menjadi aktif sama halnya jika terjadi keadaan reset. Pin TDO dalam keadaan *tri-stated* jika tidak TAP *states* (yang mengeluarkan data) dimasukkan. Port F juga menyediakan fungsi dari *interface* JTAG.
- Port G (PG4..PG0) : *bi-directional* port I/O 5-bit dengan *internal pull-up resistor* (dipilih untuk setiap bit). Port G keluaran *buffer* memiliki karakteristik simetris *drive* dengan kedua *high sink* dan kemampuan sumber. Sebagai masukan, Port G pin yang rendah akan sumber arus eksternal ditarik jika *pull-up* resistor diaktifkan. Port pin G adalah *tri-stated* ketika kondisi reset menjadi aktif, bahkan jika *clock* tidak berjalan.
- RESET : Reset masukan. Pada saat pin ini *low level*, selama lebih dari panjang pulsa minimum akan menghasilkan reset, bahkan jika *clock* tidak berjalan. Pulsa lebih pendek tidak dijamin akan menghasilkan reset.
- XTAL1 : Masukan ke *Amplifier Oscillator* pembalik dan masukan ke internal clock operating circuit.
- XTAL2 : Keluaran dari *Amplifier Oscillator* pembalik.
- AVCC : Tegangan suplai untuk pin Port F dan A/D *Converter*. Pin ini harus dihubungkan secara eksternal dengan VCC, bahkan jika ADC tidak digunakan. Jika ADC digunakan, harus dihubungkan ke VCC melalui *low-pass* filter.
- AREF : Pin referensi analog untuk A/D *Converter*.

- PEN : Pemrograman *enable* pin untuk mode Pemrograman Serial (SPI), dan secara internal *pulled-high*. Dengan memegang pin ini *low* selama *Power-on Reset*, perangkat akan memasuki mode Pemrograman Serial SPI. PEN tidak memiliki fungsi selama operasi normal.

2.6 Mikrokontroler Atmel AVR ATmega 8

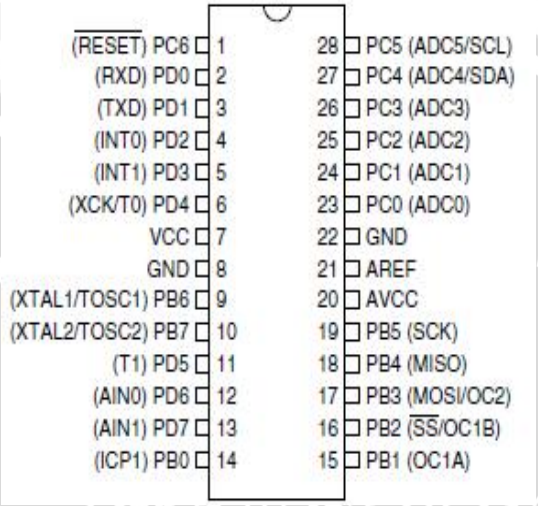
Secara umum, Mikrokontroler adalah suatu IC (*integrated circuit)* yang memiliki kemampuan untuk membuat keputusan berdasarkan sinyal dari luar (input), dan berdasarkan algoritma yang dibentuk menjadi suatu program tertentu. Pada dasarnya mikrokontroler memiliki mikroprosesor, *timer*, *counter*, perangkat I/O dan internal memori.

Sedangkan **ATMega8** menurut datasheet ATMega8 adalah mikrokontroler CMOS 8-bit berarsitektur AVR RISC yang memiliki 8K Bytes *ISP Flash*. Dengan mengeksekusi instruksi dalam clock yang sangat cepat, ATMega8 memiliki throughput mendekati 1 MIPS per MHZ, yang memungkinkan perancang sistem mengoptimalkan konsumsi daya terhadap kecepatan pemrosesan.

Sebagai suatu sistem kontrol mikrokontroler ATMega8 bila dibandingkan dengan mikroprosesor memiliki kemampuan dan segi ekonomis yang bisa diandalkan karena dalam mikrokontroler sudah terdapat RAM dan ROM sedangkan mikroprosesor didalamnya tidak terdapat keduanya. Secara umum konfigurasi yang dimiliki mikrokontroler ATMega8 adalah sebagai berikut :

- Sebuah CPU 8 bit dengan menggunakan teknologi dari Atmel.
- Jalur dua arah (*bidirectional*) yang digunakan sebagai saluran masukan atau keluaran yang dikontrol oleh *register* DDR.
- Dua buah *timer/counter* 8 bit dan sebuah *timer/counter* 16 bit.
- 23 jalur *programable* I/O (input/output).
- *Analog to Digital Converter (ADC)* 10-bit dan *Analog comparator* di dalam chip.
- Osilator internal dan rangkaian pewaktu.
- Sebuah komunikasi serial USART yang dapat diprogram.
- Sebuah *master/slave* serial SPI yang dapat diprogram.
- Sebuah *Two Wire Serial Interface*.

Masing-masing pin mikrokontroler ATMega8 mempunyai fungsi tersendiri. Dengan mengetahui fungsi masing-masing pin mikrokontroler ATMega8, perancangan aplikasi mikrokontroler ATMega8 akan lebih mudah dan maksimal. ATMega8 mempunyai 28 pin, susunan masing-masing pin ditunjukkan dalam Gambar 2.8.



Gambar 2.8 Konfigurasi Pin ATMega8

Sumber: ATMEL, 2010

Fungsi kaki-kaki PIN dalam ATMega8 sesuai dengan *data sheet* ATMega8 antara lain sebagai berikut:

- *Port* B (Pin B0..7), merupakan saluran masukan/keluaran dua arah dan juga mempunyai fungsi khusus. Fungsi khusus *Port* B diantaranya adalah : *Port* B0 {ICP (*Timer/counter input capture pin*)}, *Port* B1 {OC1A (*Timer/counter* 1 *output compare* A *match output*)}, *Port* B2 yang bisa digunakan sebagai SS (SPI *slave select input*) atau OC1B (*Timer/counter* 1 *output compare* B *match output*) , *Port* B3 yang bisa digunakan sebagai MOSI (SPI *bus master output/slave input*) atau OC2 (*timer/counter 2 compare match output*), *Port* B4 {MISO (SPI *bus master input/slave output*)}, *Port* B7 {SCK (SPI *bus serial clock*)}, *Port* B5 {SCK (SPI *bus serial clock*)}, *Port* B6 (XTAL1 & TOSC1 (*Timer Oscilator* pin1)), dan *Port* B7 (XTAL2 & TOSC2 (*Timer Oscilator* pin2)).

- *Port* C (Pin C0..5), merupakan saluran masukan/keluaran dua arah dan juga mempunyai fungsi khusus. Fungsi khusus *Port* C adalah sebagai ADC (*input* ADC *channel* 0..5), selain itu pada beberapa *Port* memiliki fungsi lain antara lain *Port* C4 (SDA (*Two-Wire serial bus data input/output line*)), dan *Port* C5 (SCL (*Two-Wire serial bus clock line*)).
- *Pin* C6 memiliki fungsi *RESET*, merupakan saluran dua masukan untuk mereset mikrokontroler dengan cara memberi masukan logika rendah.
- *Port* D (Pin D0..7), merupakan saluran masukkan/keluaran dua arah dan juga mempunyai fungsi khusus. Fungsi khusus dari *Port* D diantaranya adalah : *Port* D0 {RXD (USART *input pin*)}, *Port* D1 {TXD (USART *output pin*)}, *Port* D2 {INT0 (*Eksternal interupt 0 input*)}, *Port* D3 {INT1 (*Eksternal interupt 1 input*)}, *Port* D4 (T0 (*timer/counter0 eksternal counter input*) & XCK (USART *eksternal clock input/output*), *Port* D5 (T1 (*timer/counter* eksternel *counter input*)), *Port* D6 (AIN0 (*Analog comparator positive input*))dan *Port* D7 (AIN1 (*Analog comparator negative input*)).
- Pin 7 VCC, merupakan masukan untuk catu daya positif DC sebesar 5 volt.
- Pin 8 GND, merupakan *ground* dari seluruh rangkaian.
- Pin B6 dan Pin B7 (XTAL2 dan XTAL1), merupakan saluran untuk mengatur pewaktuan sistem. Untuk pewaktuan dapat menggunakan pewaktuan internal maupun eksternal.
- Pin 21 AREF, merupakan pin referensi analog untuk masukan ADC.
- Pin 22 GND, merupakan *ground* dari ADC.
- Pin 20 AVCC, merupakan catu untuk perangkat ADC.

2.7 Logika *Fuzzy*

Fuzzy secara harfiah berarti samar, sedangkan kebalikannya dalam hal ini adalah C*risp* yang secara harfiah berarti tegas. Dalam kehidupan sehari-hari nilai samar lebih akrab daripada nilai tegas. Temperatur/suhu tertentu biasa dinyatakan sebagai panas, agak panas, atau sangat dingin daripada dinyatakan dalam nilai terukur tertentu.

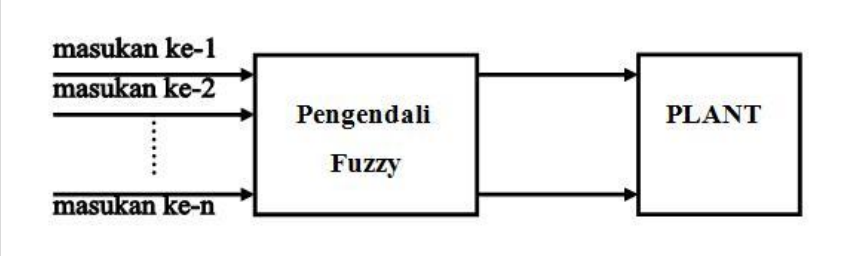
Tahun 1965 L.A. Zadeh memodifikasi teori himpunan yang disebut himpunan kabur (*Fuzzy Set*). Himpunan *fuzzy* didasarkan pada gagasan untuk

memperluas jangkauan fungsi karakteristik sehingga fungsi tersebut akan mencakup bilangan real pada interval [0,1]. Nilai keanggotaannya menunjukkan bahwa suatu nilai dalam semesta pembicaraan tidak hanya berada pada 0 atau 1, namun juga nilai yang terletak diantaranya. Dengan kata lain nilai kebenaran suatu hal tidak hanya bernilai benar atau salah. Nilai 0 menunjukkan salah, nilai 1 menunjukkan benar dan masih ada nilai-nilai yang terletak diantaranya.

Sejak tahun 1985 pengaturan berbasis logika *fuzzy* mengalami perkembangan pesat, terutama dalam hubungannya dengan penyelesaian masalah kendali yang bersifat tak linier, sulit dimodelkan, berubah karakteristiknya terhadap waktu (*time varying*) dan kompleks.

2.7.1 Struktur Dasar Kontrol Logika *Fuzzy*

Dalam sistem pengaturan dengan logika *fuzzy* dilibatkan suatu blok pengendali yang menerima satu atau lebih masukan dan mengumpankan satu atau lebih keluaran ke plant atau blok lain sebagaimana ditunjukan dalam Gambar 2.9



Gambar 2.9 Pengendali Fuzzy

Sumber : *Donald Coughanowr,1991*

Komponen utama penyusun *Fuzzy Logic Controller* adalah unit fuzzifikasi, *fuzzy inference*, basis pengetahuan dan unit defuzzifikasi.

Basis pengetahuan terdiri dari dua jenis (Yan : 94) :

- Basis data : Mendefinisikan parameter *fuzzy* sebagai bagian dari himpunan *fuzzy* dengan menentukan batas-batas fungsi keanggotaan pada semesta pembicaraan untuk tiap-tiap variabel.
- Basis aturan : Memetakan nilai masukan *fuzzy* menjadi nilai keluaran *fuzzy*.

2.7.2 Fungsi Keanggotaan

Fungsi keanggotaan menotasikan nilai kebenaran anggota-anggota himpunan *fuzzy*. Interval nilai yang digunakan untuk menentukan fungsi keanggotaan, yaitu nol

dan satu. Tiap fungsi keanggotaan memetakan elemen himpunan *crisp* ke semesta himpunan *fuzzy*.

Suatu himpunan *fuzzy* A dalam semesta pembicaraan U dinyatakan dengan fungsi keanggotaan, μ_A yang harganya berada dalam interval [0,1] (Kuswadi, 2000:27). Secara matematika hal ini dinyatakan dengan :

$$« _A : U \rightarrow [0,1] \qquad (2.1)$$

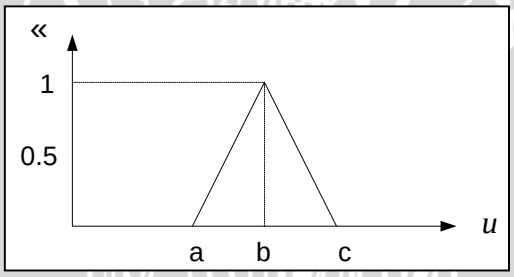
Berikut ini jenis fungsi keanggotaan yang akan digunakan yaitu fungsi keanggotaan T (*triangular*).

1. Fungsi keanggotaan bentuk Triangular

   Definisi fungsi triangular sebagai berikut:

$$T(u;a,b,c) = \begin{cases} 0 & u < a \\ \dfrac{u-a}{b-a} & a \le u \le b \\ \dfrac{c-u}{c-b} & b \le u \le c \\ 0 & u > c \end{cases} \qquad (2.2)$$

   Fungsi keanggotaan bentuk Tringular ditunjukkan dalam Gambar 2.10



Gambar 2.10 Fungsi Keanggotaan Bentuk Triangular (T)

Sumber : Jun Yan, 1994 : 19

Fungsi keanggotaan bentuk triangular ini digunakan bila diinginkan himpunan *fuzzy* mempunyai nilai proporsional terhadap nol maupun satu.

2.7.3 Kendali Logika *Fuzzy*

Kendali logika *fuzzy* adalah sistem berbasis aturan (*rule based system*) yang didalamnya terdapat himpunan aturan *fuzzy* yang mempresentasikan mekanisme pengambilan keputusan. Aturan yang dibuat digunakan untuk memetakan variabel input ke variabel output dengan pernyataan *If - Then.*

Kontroler ini akan menggunakan data tertentu *(crisp)* dari sejumlah sensor kemudian mengubahnya menjadi bentuk linguistik atau fungsi keanggotaan melalui proses fuzzifikasi. Lalu dengan aturan *fuzzy*, *inference engine* yang akan menentukan hasil keluaran *fuzzy*. Setelah itu hasil ini akan diubah kembali menjadi bentuk numerik melalui proses defuzzifikasi.

2.7.3.1 Fuzzifikasi

Proses fuzzifikasi merupakan proses untuk mengubah variabel non *fuzzy* (variabel numerik) menjadi variabel *fuzzy* (variabel linguistik). Nilai masukan-masukan yang masih dalam bentuk variabel numerik yang telah dikuantisasi sebelum diolah oleh pengendali logika *fuzzy* harus diubah terlebih dahulu ke dalam variabel *fuzzy*. Melalui fungsi keanggotaan yang telah disusun, maka dari nilai-nilai masukan tersebut menjadi informasi *fuzzy* yang berguna nantinya untuk proses pengolahan secara *fuzzy* pula. Proses ini disebut fuzzifikasi (Yan,1994:49). Proses fuzzifikasi diekspresikan sebagai berikut:

x = *fuzzifier* (x_0)

dengan:

x_0 = nilai *crisp* variabel masukan

x = himpunan *fuzzy* variabel yang terdefinisi

fuzzifier = operator fuzzifikasi yang memetakan himpunan *crisp* ke himpunan *fuzzy*

Pedoman memilih fungsi keanggotaan untuk proses fuzzifikasi, menurut Jun Yan, menggunakan :

1. Himpunan *fuzzy* dengan distribusi simetris.
2. Gunakan himpunan *fuzzy* dengan jumlah ganjil, berkaitan erat dengan jumlah kaidah (*rules*).
3. Mengatur himpunan *fuzzy* agar saling menumpuk.
4. Menggunakan fungsi keanggotaan bentuk segitiga atau trapesium.

2.7.3.2 Kaidah Atur *Fuzzy* (*Fuzzy* Rule)

Fuzzy rule adalah bagian yang menggambarkan dinamika suatu sistem terhadap masukan yang dikarakteristikan oleh sekumpulan variabel-variabel

linguistik dan berbasis pengetahuan seorang operator ahli. Pernyataan tersebut umumnya dinyatakan oleh suatu pernyataan bersyarat.

Dalam pengendali berbasis *fuzzy*, aturan pengaturan *fuzzy* berbentuk aturan " IF – THEN". Untuk sebuah sistem MISO *(Multi Input Single Output)* basis aturan pengaturan *fuzzy* berbentuk seperti berikut ini.

Rule 1 IF X_1 is A_{11} AND … AND x_m is A_{1m} THEN Y is B_1

Rule 2 IF X_1 is A_{21} AND … AND x_m is A_{2m} THEN Y is B_2

. . .

. . .

. . .

Rule n IF X_n is A_{n1} AND … AND x_m is A_{nm} THEN Y is B_n

Dengan X_j merupakan variabel masukan sistem , A_{ij} merupakan *fuzzy set* untuk X_j , Y merupakan variabel keluaran sistem, B_i merupakan *fuzzy set* untuk Y , AND adalah operator *fuzzy*.

2.7.3.3 Metode Inferensi

Metode inferensi merupakan proses untuk mendapatkan keluaran dari suatu kondisi masukan dengan mengikuti aturan-aturan yang telah ditetapkan. Keputusan yang didapatkan pada proses ini masih dalam bentuk *fuzzy* yaitu derajat keanggotaan keluaran.

Ada beberapa cara untuk mengidentifikasi aturan mana yang akan dipakai dengan menggunakan nilai keanggotaan masukan. Menurut Jun Yan, diantara bermacam-macam metode inferensi *fuzzy* ada dua metode yang paling sering digunakan pada kendali logika *fuzzy*, salah satunya yaitu :

Metode Inferensi MAX – MIN

Pada metode Max – Min aturan operasi minimum Mamdani digunakan untuk implikasi *fuzzy*. Persamaan aturan minimum adalah

$$\mu_{C'} = \bigcup_{1}^{n} \sim_i \wedge \mu ci \qquad (2.3)$$

dengan $\sim_i = \mu_{Ai}(x_0) \wedge \mu_{Bi}(y_0)$

Sebagai contoh , terdapat dua basis kaidah atur *fuzzy*, yaitu :

R_1 : Jika x adalah A_1 dan y adalah B_1 maka z adalah C_1

R_2 : Jika x adalah A_2 dan y adalah B_2 maka z adalah C_2

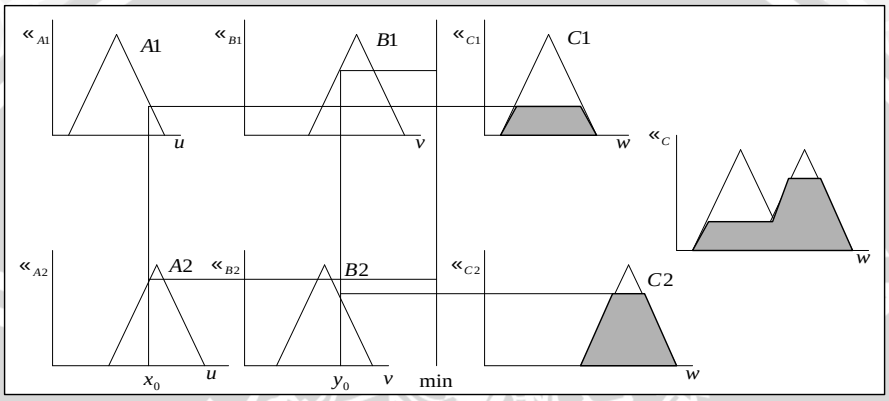
Pada metode penalaran MAX-MIN fungsi keanggotaan konsekuen dinyatakan dengan

$$\mu_{C'}(W) = \mu_{C'1} \vee \mu_{C'2} = [\sim_1 \wedge \mu_{C1}(w)] \vee [\sim_2 \wedge \mu_{C2}(w)] \quad (2.4)$$

dimana $\sim_1 = \mu_{A1}(x_0) \wedge \mu_{B1}(y_0)$ (2.5)

$\sim_2 = \mu_{A2}(x_0) \wedge \mu_{B2}(y_0)$ (2.6)

Lebih jelas metode ini dideskripsikan dalam Gambar 2.11



Gambar 2.11 Inferensi Fuzzy dengan Metode MAX-MIN

Sumber : Jun Yan, 1994 : 55

2.7.3.4 Defuzzifikasi

Defuzzifikasi adalah proses untuk mendapatkan nilai numerik dari data *fuzzy* yang dihasilkan dari proses inferensi (Yan, 1994 : 55). Proses defuzzifikasi dinyatakan sebagai berikut :

$$y_0 = defuzzifier(y) \quad (2.7)$$

dengan:

y : aksi kontrol *fuzzy*.

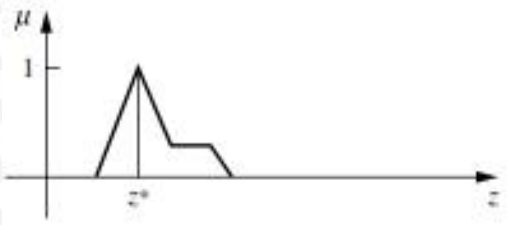
y_0 : aksi kontrol *crisp*.

defuzzifier : operator *defuzzifikasi*

Pada penelitian ini digunakan metode *max membership principle* (Ross, Timothy J. 1997). Metode ini dikenal juga sebagai metode tinggi, skema ini terbatas pada *output* fungsi memuncak. Metode ini didefinisikan sebagai berikut:

$\mu_C(z^*) \geq \mu_C(z)$ untuk semua nilai z ϵ Z (2.8)

dengan z* ialah nilai terdefuzzifikasi dan bisa dilihat pada Gambar 2.12.



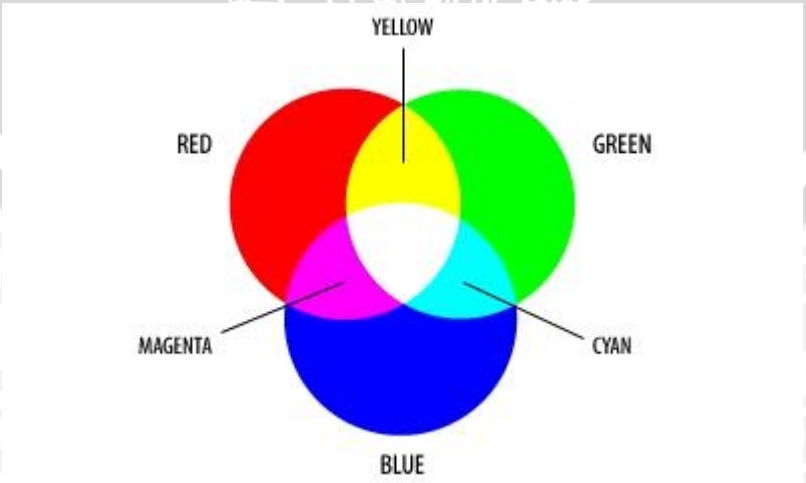
Gambar 2.12 Metode Max Membership Principle

Sumber: Timothy J. Ross, 1997

2.8 Model Warna RGB

Ketika mata melihat suatu benda, cahaya dari benda itu langsung merambat ke mata dan sinar yang dipantulkannya dikenali sebagai warna benda tersebut. Contohnya, benda berwarna merah akan tampak berwarna merah saat benda tersebut dikenai oleh sinar putih karena kecuali sinar merah, semua spektrum warna diserap oleh benda. Sehingga hanya sinar merah yang dipantulkan oleh benda dan benda tersebut tampak berwarna merah.

Warna komplementer dapat didefinisikan sebagai kombinasi dua warna yang menghasilkan warna putih apabila dijumlahkan. Beberapa warna komplementer yaitu warna kuning dengan biru, warna hijau dengan warna magenta, dan warna merah dengan warna cyan. Warna benda tak tembus cahaya akan memantulakn cahaya sedangkan warna benda tembus cahaya akan meneruskan cahaya. Warna sinar dapat dikelompokkan menjadi dua yaitu warna primer dan warna sekunder. Beberapa macam kombinasi warna benda ditunjukkan dalam Gambar 2.13.



Gambar 2.13 Kombinasi Warna Benda

Sumber: www.ilmugrafis.com,2013

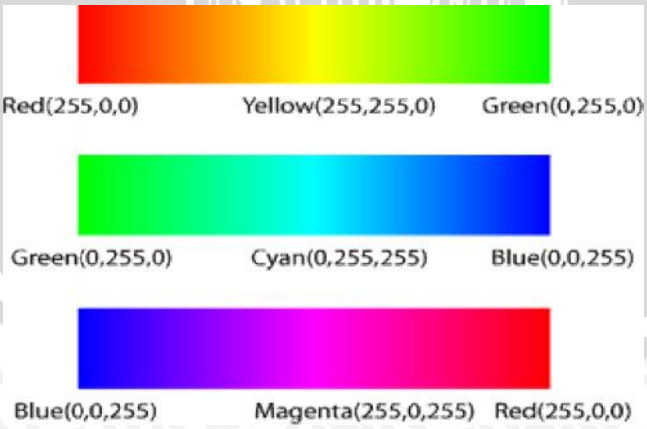
2.8.1 Warna Primer

Warna-warna yang ada pada citra digital pada dasarnya berasal dari 3 satuan warna dasar merah (*red*), hijau (*green*), dan biru (*blue*) atau yang sering disebut juga dengan istilah RGB. Warna apapun yang ada di dunia ini dan warna apapun yang ada pada citra digital merupakan pencampuran atau kombinasi dari warna RGB ini. Tiap warna memiliki skala dari 0-255 (8bit).

2.8.2 Warna Sekunder dan Warna Tersier

Dalam suatu ruang yang sama sekali tidak ada cahaya, maka ruangan tersebut adalah gelap total. Tidak ada signal gelombang cahaya yang diserap oleh mata diatau RGB (0,0,0). Apabila dimenambahkan cahaya merah pada ruangan tersebut, maka ruangan akan berubah warna menjadi merah misalnya RGB (255,0,0), semua benda dalam ruangan tersebut hanya dapat terlihat berwarna merah. Demikian apabila cahaya diganti dengan hijau atau biru.

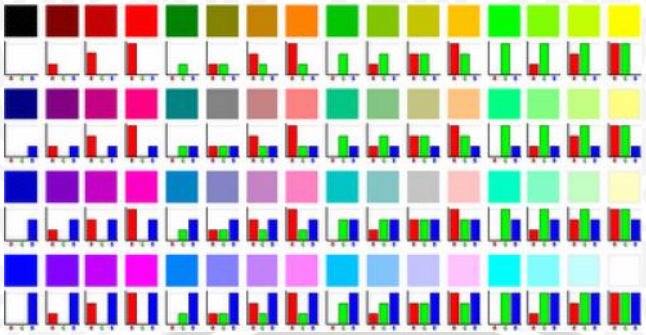
Apabila dimelanjutkannya dengan memberikan 2 macam cahaya primer dalam ruangan tersebut seperti (merah dan hijau), atau (merah dan biru) atau (hijau dan biru), maka ruangan akan berubah warna masing-masing menjadi kuning, atau magenta atau cyan. Warna-warna yang dibentuk oleh kombinasi dua macam cahaya tersebut disebut warna sekunder. Seperti yang terlihat pada Gambar 2.14 berikut.



Gambar 2.14 Warna-Warna Sekunder Hasil Kombinasi RGB

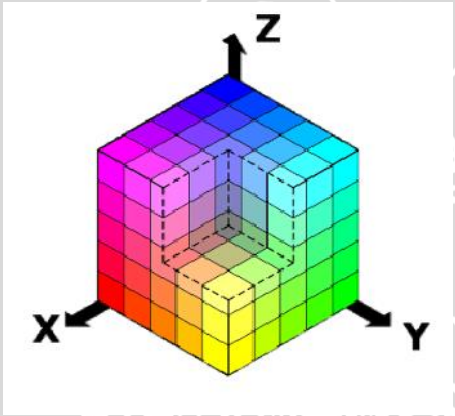
Sumber: www.pengantar-warna.blogspot.com, 2013

Bisa dilihat pula kombinasi warna RGB seperti pada Gambar 2.15.



Gambar 2.15 Warna-warna Sekunder dari Kombinasi Warna Primer RGB
Sumber: www.pengantar-warna.blogspot.com, 2013

Warna Tersier adalah warna yang hanya dapat terlihat apabila ada tiga cahaya primer, jadi apabila dinon-aktifkan salah satu cahaya, maka benda tersebut berubah warna. Contoh warna tersier seperti abu-abu, putih. Contoh kombinasi warna-warna primer tersebut dapat ditunjukkan dalam Gambar 2.16 berikut.



Gambar 2.16 Warna-warna Tersier dari Kombinasi Warna Primer
Sumber: www.pengantar-warna.blogspot.com, 2013

BAB III

METODOLOGI PENELITIAN

Untuk menyelesaikan rumusan masalah dan merealisasikan tujuan penelitian yang terdapat di bab pendahuluan maka diperlukan metode untuk menyelesaikan masalah tersebut.

3.1 Penentuan Spesifikasi Alat

Spesifikasi alat secara umum ditetapkan terlebih dahulu sebagai acuan dalam perancangan selanjutnya. Spesifikasi alat yang direncanakan adalah sebagai berikut :

1) Tinggi maksimal robot sesuai dengan Peraturan Kontes Robot Seni Indonesia (KRSI) 2013 adalah 55 ± 5 cm.
2) Robot berbahan dasar mika akrilik dan alumunium.
3) Robot memiliki struktur kepala, tangan dan kaki seperti struktur tubuh manusia.
4) Masing-masing kaki robot memiliki 6 buah motor DC servo (6 DOF).
5) Motor servo yang digunakan adalah Hitec tipe HS-7954SH dan HS-5685MH, dan GWS tipe micro 2BBMG.
6) Sensor kamera (CMUcam4) diletakan di depan perut.
7) Mikrokontroler yang digunakan adalah Atmel ATmega128 dan ATmega8.
8) Kemampuan kaki robot adalah berdiri tegak dengan dua kaki, berdiri satu kaki, gerak melangkah maju, dan gerak kepiting.
9) Kemampuan tangan robot menari sesuai rule yang telah ditentukan Peraturan Kontes Robot Seni Indonesia (KRSI) 2013.
10) Menggunakan catu daya baterai DC 11,1 V jenis *lithium polymer*.

3.2 Studi Literatur

Pengetahuan yang mendukung perencanaan dan realisasi alat meliputi:

1. Robot *Humanoid*
2. Kontes Robot Seni Indonesia (KRSI) 2013
3. Motor Servo
4. Sensor Kamera (CMUcam4)
5. Mikrokontroler Atmel AVR ATmega128
6. Mikrokontroler Atmel AVR ATmega8
7. Kontrol Logika *Fuzzy*
8. Model Warna RGB

3.3 Perancangan dan Pembuatan Alat

3.3.1 Perancangan dan Pembuatan Perangkat Keras (*Hardware*)

a. Pembuatan blok diagram lengkap sistem keseluruhan
b. Desain *Printed Circuit Board* (PCB) menggunakan *software* Eagle Layout Editor
c. Merakit perangkat keras masing-masing blok

3.3.2 Perancangan dan Pembuatan Perangkat Lunak (*Software*)

Setelah kita mengetahui seperti apa perangkat keras yang dirancang, maka kita membutuhkan perangkat lunak untuk mengatur kerja dari alat ini. Desain dan parameter yang telah dirancang kemudian diterapkan kedalam mikrokontroler ATMega128 dan ATMega8 dengan menggunakan bahasa C dan *compiler* CodeVision AVR.

3.4 Pengujian Alat

Untuk memastikan bahwa sistem ini berjalan sesuai yang direncanakan maka perlu dilakukan pengujian alat meliputi perangkat keras *(hardware)* yang dilakukan baik tiap blok rangkaian maupun keseluruhan sistem.

3.4.1 Pengujian Tiap Blok

Pengujian tiap blok dilakukan dengan tujuan untuk menyesuaikan nilai masukan dan nilai keluaran tiap-tiap blok sesuai dengan perancangan yang dilakukan sebelumnya.

3.4.2 Pengujian Keseluruhan Sistem

Pengujian sistem secara keseluruhan dilakukan dengan tujuan untuk mengetahui unjuk kerja alat setelah perangkat keras dan perangkat lunak diintegrasikan bersama.

3.5 Pengambilan Kesimpulan

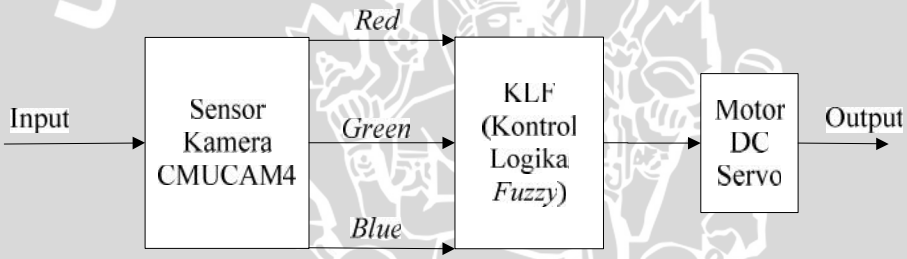
Pengambilan kesimpulan dilakukan setelah didapatkan hasil dari pengujian. Jika hasil yang diperoleh telah sesuai dengan spesifikasi yang direncanakan maka alat tersebut telah memenuhi harapan dan memerlukan pengembangan untuk penyempurnaannya.

BAB IV
PERANCANGAN DAN PEMBUATAN ALAT

Pada bab ini akan dijelaskan mengenai perancangan dan pembuatan *robot humanoid* mulai dari diagram blok sistem, desain mekanik robot, perancangan gerakan, perancangan perangkat keras, dan perancangan perangkat lunak. Perancangan dan pembuatan dilakukan secara bertahap dan sistematis, sehingga nantinya akan memudahkan dalam analisis sistem.

4.1 Diagram Blok Sistem

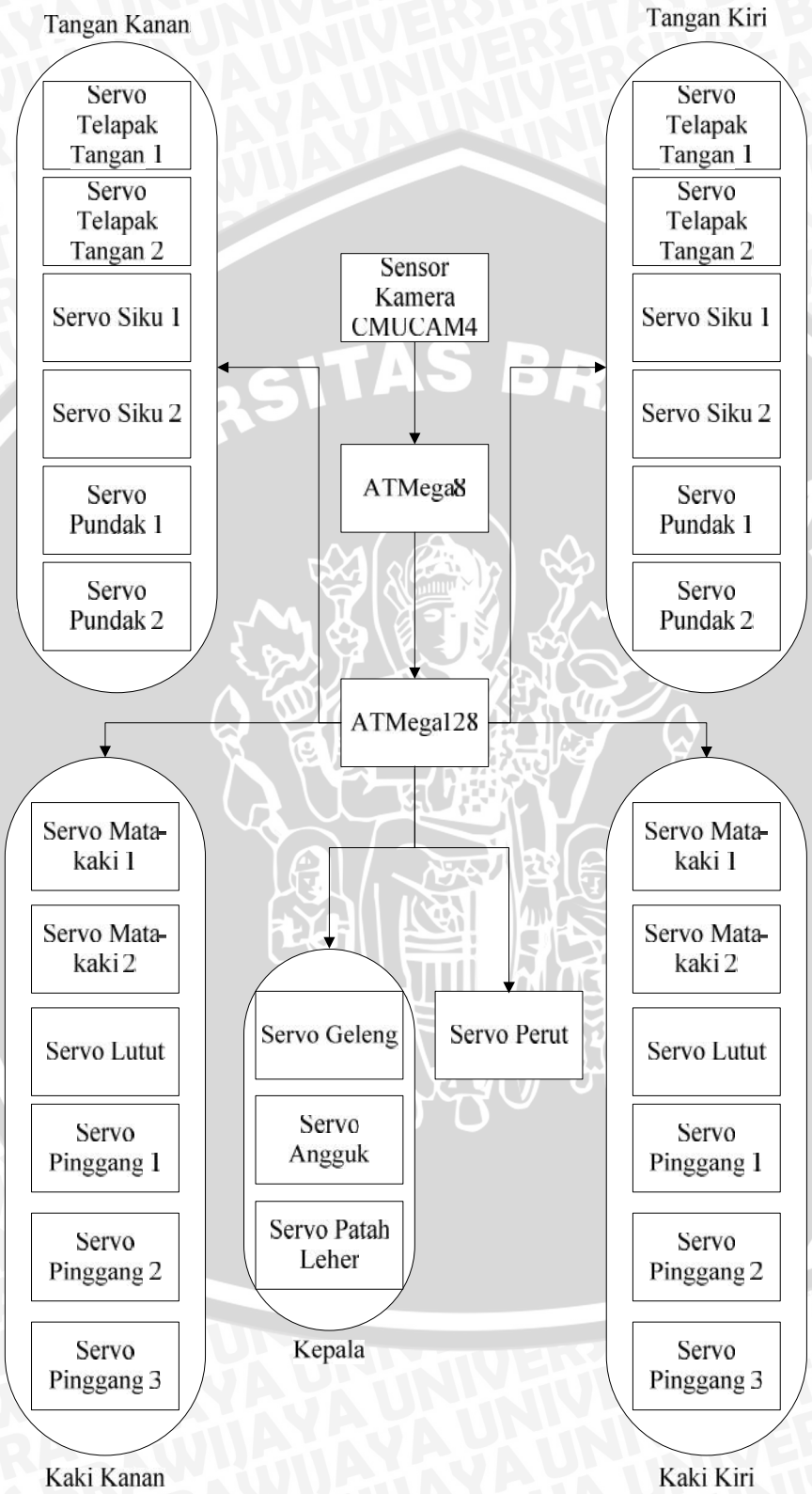
Pada perancangan alat diperlukan perancangan blok diagram sistem yang dapat menjelaskan sistem secara garis besar dan diharapkan alat dapat bekerja sesuai dengan rencana. Blok diagram tersebut dapat dilihat dari Gambar 4.1 berikut.



Gambar 4.1 Diagram Blok Sistem Pengaturan

Digram blok tersebut menjelaskan diagram blok sistem pengaturan pengolahan data dari sensor hingga ke aktuatornya (motor DC servo). *Input* berupa data yang dibaca sensor kamera CMUCAM4 dari pendeteksian warna pada lapangan KRSI 2013. Data keluaran sensor terpilah-pilah menjadi tiga komponen dasar warna yaitu merah, hijau, dan biru (RGB). Data RGB tersebut menjadi *input* dari setiap fungsi keanggotaan pada kontrol logika *fuzzy* (KLF). *Output* KLF berupa perintah yang harus dilakukan robot *humanoid* yang mana perintah tersebut terintregrasi ke dalam pengontrolan motor-motor DC servo sebagai perwakilan dari tiap-tiap sendi pada robot *humanoid.* Sedangkan *output*-nya sendiri ialah gerakan tari robot hasil dari pembacaan sensor kamera pada lapangan.

Sistem pergerakan robot *humanoid* secara keseluruhan ditunjukkan dalam Gambar 4.2



Gambar 4.2 Diagram Blok Sistem Robot *Humanoid*

Robot *humanoid* memiliki dua buah tangan (masing-masing tangan terdiri dari 6 buah motor DC servo), dua buah kaki (masing-masing kaki terdiri dari 6 buah motor DC servo), sebuah kepala yang terdapat 3 buah motor DC servo, dan satu buah motor DC servo pada perut. Dengan total keseluruhan dua puluh delapan buah motor DC servo yang tiap-tiap motornya memiliki fungsi khusus dalam menunjang pergerakan robot. Untuk penjelasan posisi dan fungsinya akan dijelaskan dalam sub bab perancangan mekanik robot.

Sensor kamera yang digunakan adalah modul kamera CMUCAM4. Sensor ini diletakkan di depan perut dangan peletakkan menghadap ke bawah agak miring ke atas, yang bertujuan agar warna yang dideteksi terletak di depan robot *humaoid* berada, sehingga dapat menghindari zona larang. Data keluaran dari modul CMUCAM4 sebagai sensor kamera akan menjadi *input* dari Mikrokontroler *slave* (ATMega8). Di dalam ATMega8 terjadi proses pengaturan KLF, kemudian *output* ATMega8 berupa kombinasi logika 3 bit dari 3 port *output* ATMega8 dikirim ke Mikrokontroler ATMega128 yang berperan sebagai master. Mikrokontroler ATMega128 menerjemahkan *output* ATMega8 kedalam fungsi kombinasi sudut yang diinginkan, sehingga *output* pada ATMega128 berupa program sudut-sudut yang terintegrasi ke motor DC servo terkait, sehingga kedua puluh delapan motor DC servo dapat melakukan sesuai dengan gerakan yang diinginkan.

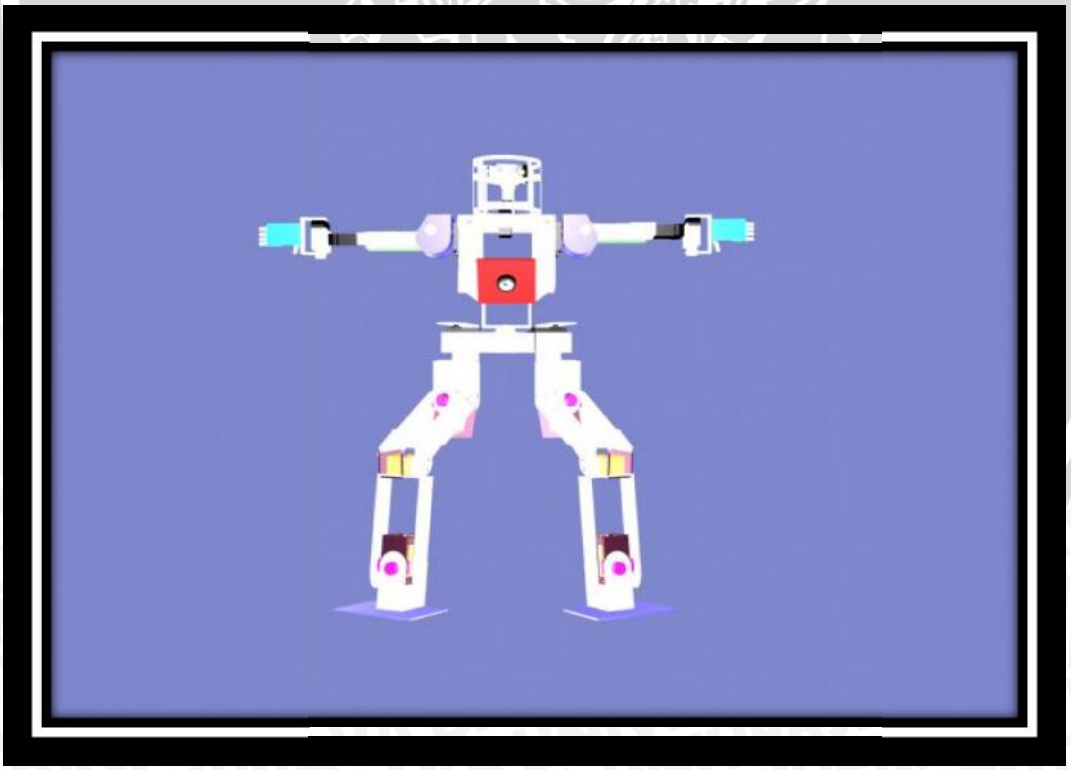
Pada perlombaan KRSI 2013, tiap-tiap zona pada lapangan memiliki warna yang berbeda-beda. Awal mula, robot berdiri tegak pada Zona Mulai, sensor warna akan mendeteksi warna lapangan (yaitu merah atau biru) dan Mikrokontroler akan memerintahkan robot melakukan gerakan *sembah pembuka*. Pada Zona I, sensor kamera mendeteksi warna biru muda dan robot akan melakukan gerakan *ngasak*. Pada Zona II, terdeteksi warna putih dan robot melakukan gerakan *capingan*. Pada pertengahan Zona II terdapat Zona Larang berwana kuning, maka jika terdeteksi zona tersebut, robot akan bergerak menghindari zona larang. Zona III berwarna hijau dan robot akan melakukan gerakan *memberi cincin.* dan akhirnya pada Zona Tutup yang berwarna merah atau biru, robot akan melakukan *sembah penutup* sebagai penutup gerakan tariannya.

4.2 Perancangan Perangkat Keras

4.2.1 Perancangan Mekanik Robot

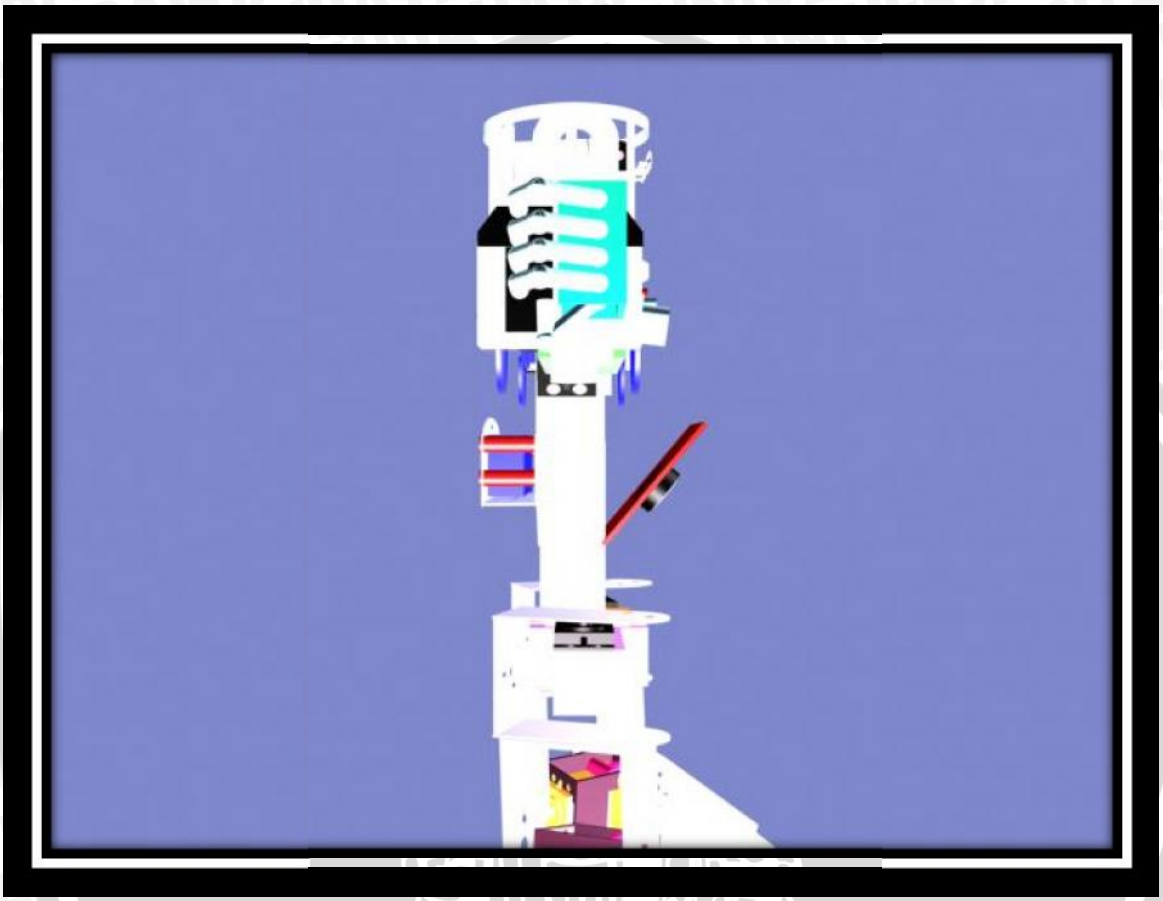
Perancangan mekanik robot mengikuti arutan Kontes Robot Seni Indonesia (KRSI) 2013. Tinggi robot secara keseluruhan 55±5cm diukur dari ujung bawah hingga ujung bawah secara vertikal. Rentang tangan maupun kaki robot tidak boleh melebihi 60cm diukur dari ujung tangan kanan hingga ujung tangan kiri ketika membuka selebar-lebarnya. Luas telapak kaki maksimum 150cm^2. Massa robot maksimal ialah 30kg.

Robot *Humanoid* selayaknya manusia terdiri dari dua buah kaki, dua buah tangan, kepala, dan badan. Robot *Humanoid* IBRAHIM memiliki keseluruhan 27 sendi (derajat kebebasan atau DOF). Masing-masing tangan terdiri dari 6 DOF, untuk kaki masing-masing terdapat 6 DOF, kepala 2 DOF, dan badan (perut) 1 DOF. Berikut desain mekanik yang digambar dengan menggunakan software *3ds Max 7*. Gambar 4.3 menunjukkan desain mekanik robot tampak depan.



Gambar 4.3 Desain Mekanik Robot Tampak Depan

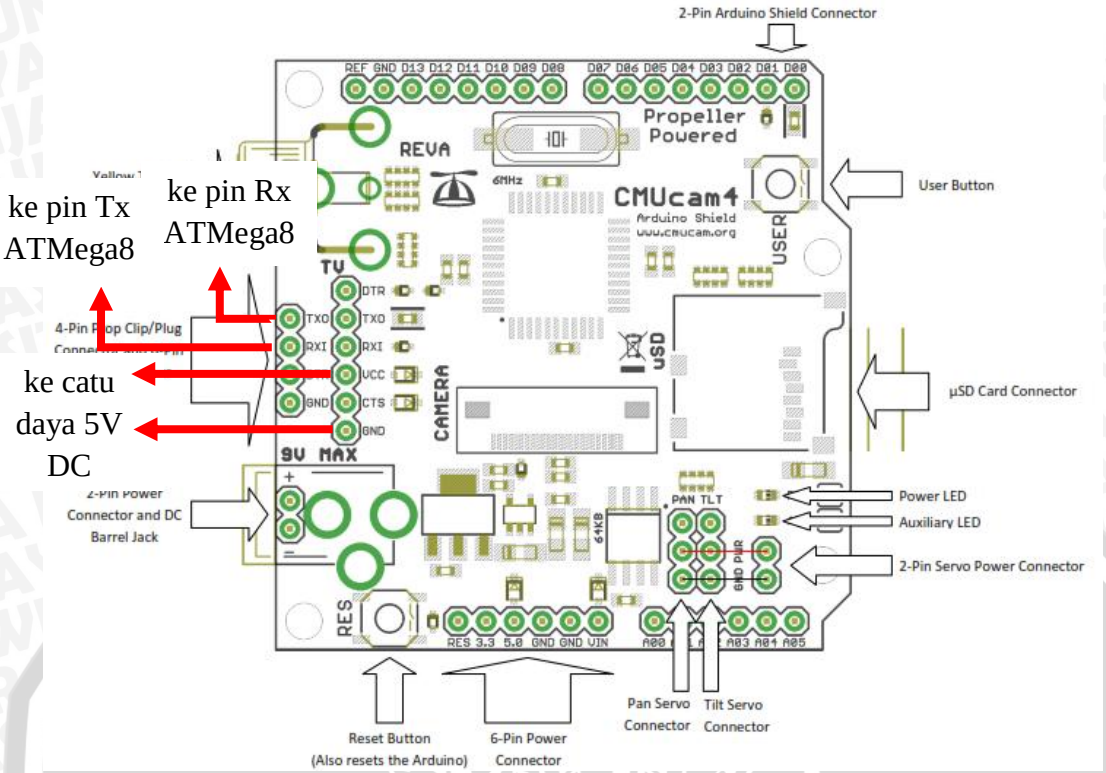
Seperti yang telah dijelaskan sebelumnya, sensor kamera diletakkan di depan perut dangan peletakkan menghadap ke depan miring ke bawah (pada Gambar 4.3 sensor kamera berbentuk persegi dengan warna merah). Gambar 4.4 menunjukkan posisi kamera dilihat dari samping



Gambar 4.4 Desain Mekanik Robot Tampak Samping Kanan

4.2.2 Sensor Kamera

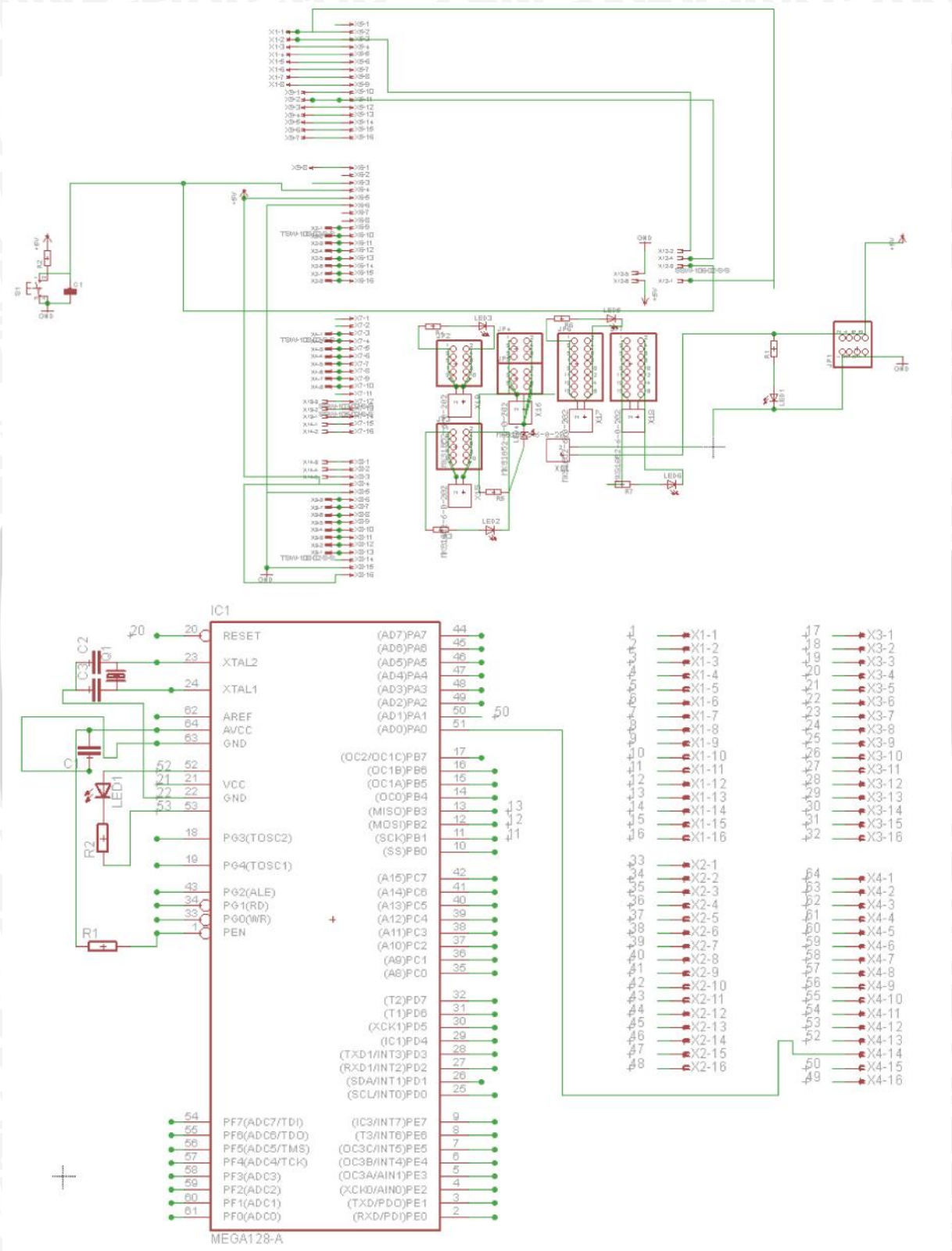
Pada perancangan robot ini, sensor kamera berfungsi sebagai pendeteksi warna pada lapangan KRSI. Data dari keluaran sensor ini akan menjadi masukan dari proses kontrol logika *fuzzy*. Penampang rangkaian sensor kamera CMUCAM4 dapat ditunjukkan dalam Gambar 4.5.



Gambar 4.5 Penampang Rangakaian CMUCAM4

4.2.3 Rangkaian Minimum Sistem Mikrokontroler *Master*

Pada robot ini digunakan mikrokontroler ATMega128 sebagai penerima keputusan proses logika *fuzzy* dari Mikrokontroler ATMega8 dan pengolah data utama dalam melakukan perintah terhadap aktuator (motor DC servo). Konfigurasi kaki I/O dalam mikrokontroler ATMega128 (mikrokontroler *master*) ditunjukkan dalam Gambar 4.6.



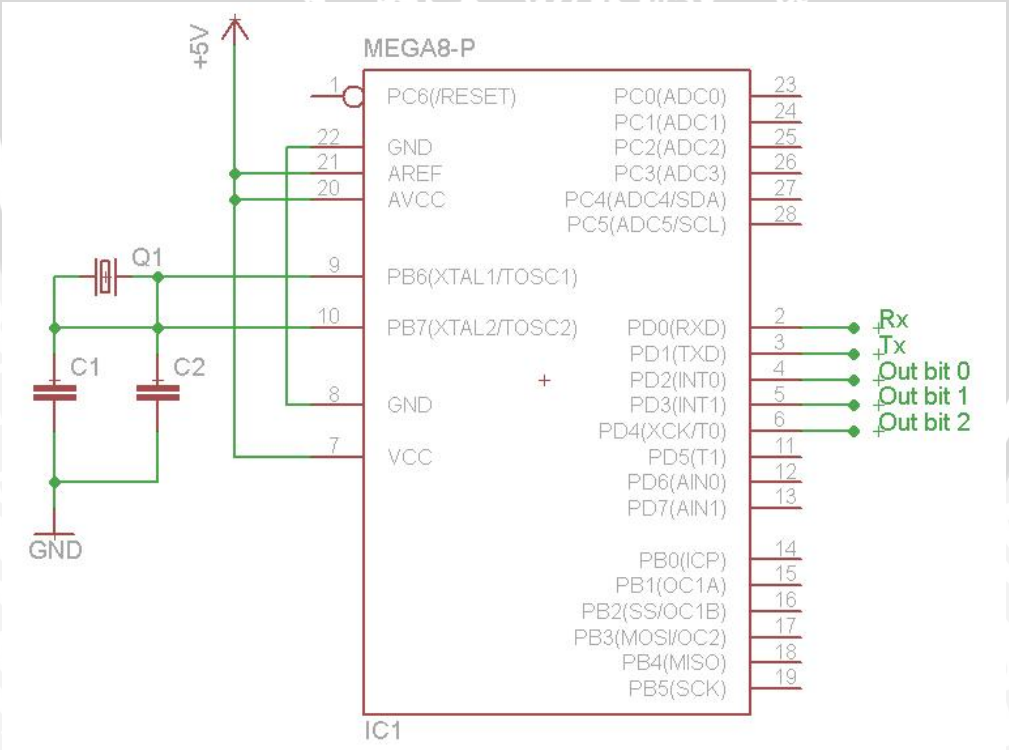
Gambar 4.6 Rangkaian Minimum Sistem Mikrokontroler Master

Rangkaian minimum sistem mikrokontoller master yang digunakan terdiri dari dua tingkat (dua *board*) sehingga *schematic* yang ditampilkan terpisah. Mikrokontroler ATMega128 mempunyai 3 port, dengan 21 jalur I/O yang dapat di program menjadi masukan atau keluaran. Pin yang digunakan antara lain:

- Pin D0-D7 : Sinyal kontrol servo tangan kiri dan kepala.
- Pin F1-F6 : Sinyal kontrol servo tangan kanan.
- Pin C0-C6 : Sinyal kontrol servo kaki kanan dan perut.
- Pin A1-A6 : Sinyal kontrol servo kaki kiri.
- Pin B0 : Sebagai *input* bit 0 dari mikrokontroler *slave.*
- Pin B1 : Sebagai *input* bit 1 dari mikrokontroler *slave.*
- Pin B2 : Sebagai *input* bit 2 dari mikrokontroler *slave.*

4.2.4 Rangkaian Minimum Sistem Mikrokontroler *Slave*

Pada robot ini juga digunakan mikrokontroler ATMega8 sebagai pengolah data sensor kamera dalam proses pengaturan kontrol logika *fuzzy* (KLF). Konfigurasi kaki I/O dari mikrokontroler ATMega8 (mikrokontroler *slave*) ditunjukkan dalam Gambar 4.7



Gambar 4.7. Rangkaian Minimum Sistem Mikrokontroler *Slave*

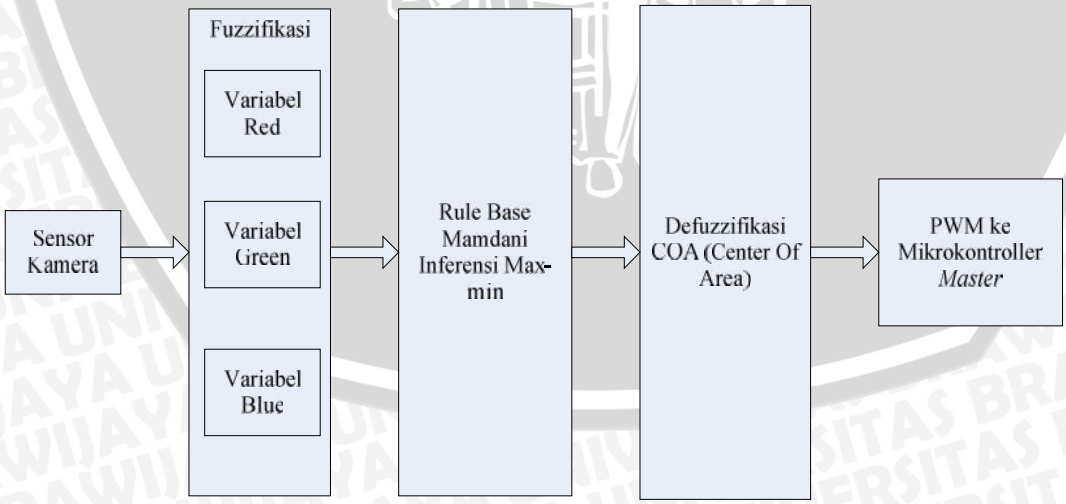
Mikrokontroler ATMega8 mempunyai 3 port, dengan 21 jalur I/O yang dapat di program menjadi masukan atau keluaran. Pin yang digunakan antara lain:

- Pin D0-D1 : Rx dan Tx untuk transmisi data dan komunikasi dengan sensor kamera CMUCAM4.
- Pin D2 : Sebagai *output* logika *fuzzy* bit 0 yang akan dikirim ke mikrokontroler *master*.
- Pin D3 : Sebagai *output* logika *fuzzy* bit 1 yang akan dikirim ke mikrokontroler *master*.
- Pin D4 : Sebagai *output* logika *fuzzy* bit 3 yang akan dikirim ke mikrokontroler *master*.
- Pin B.6 : Dihubungkan dengan kaki osilator 16 MHz.
- Pin B.7 : Dihubungkan dengan kaki osilator 16 MHz.

4.3 Perancangan Kontrol Logika Fuzzy

4.3.1 Variabel Masukan dan Variabel Keluaran

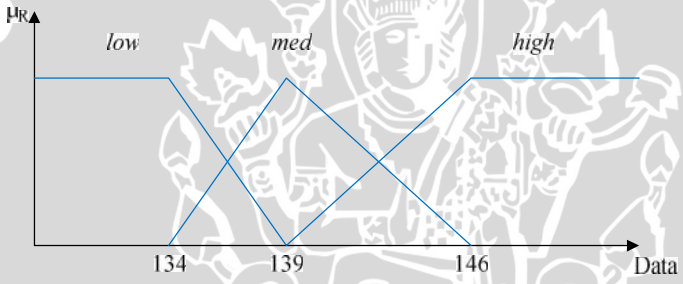
Variabel masukan dari sensor kamera yang digunakan ada tiga, yaitu komponen warna dasar merah, hijau, dan biru, sedangkan variabel keluarannya berupa PWM (*Pulse Width Modulation*) yang akan dijadikan *input* pada Mikrokontroler *master*. Berikut Gambar 4.8 menunjukkan blok diagram dari perancangan kontrol logika *fuzzy*.



Gambar 4.8 Diagram Blok Perancangan Kontrol Logika *Fuzzy*

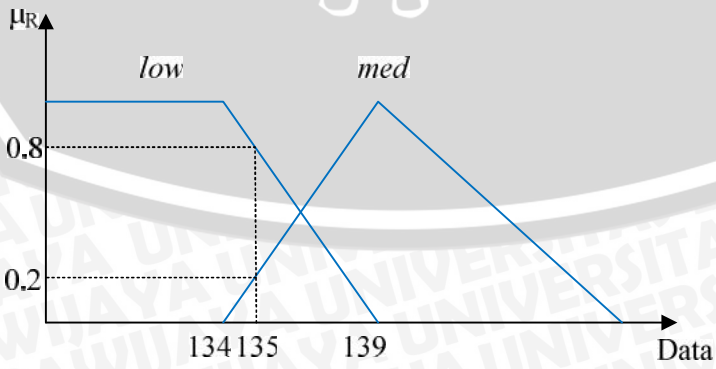
4.3.1.1 Fungsi Keanggotaan Masukan pada Warna Dasar Merah

Pada fungsi keanggotaan *input* variabel warna dasar merah terdiri dari tiga kurva *membership function* (fungsi keanggotaan), yaitu *Low*, *Med*, dan *High*. Kurva *low* menandakan bahwa warna terdeteksi mengandung unsur komponen warna dasar merah yang rendah nilainya. Begitu pula pada *med* untuk nilai sedang, dan *high* untuk nilai yang dianggap tinggi. Penentuan parameter nilai batas dari tiap-tiap kurva sendiri menggunakan salah satu metode yaitu *intuition*. Metode ini nampak sederhana, itu hanya berasal dari kapasitas manusia untuk mengembangkan fungsi keanggotaan melalui kecerdasan bawaan mereka sendiri dan pemahaman (Timothy J. Ross, 1995: 92). Meski menggunakan intuisi namun penulis memberikan nilai juga berdasarkan referensi dari kombinasi nilai komponen warna dasar (RGB) dengan menggunakan software desain grafis pada komputer. Fungsi keanggotaan masukan pada warna dasar merah dapat ditunjukkan pada Gambar 4.9



Gambar 4.9 Fungsi Keanggotaan Masukan pada Warna Dasar Merah

Misalnya, data warna terdeteksi untuk komponen warna dasar merah adalah 135, maka derajat keanggotaan warna merah (μ_R) didapat dengan cara sebagai berikut.



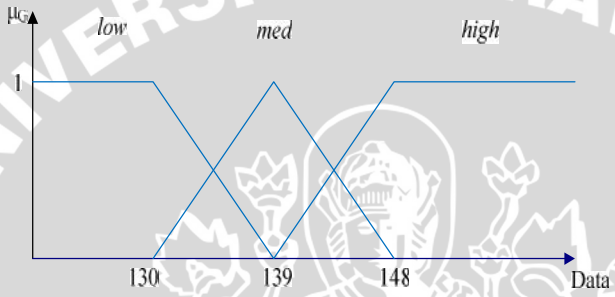
Gambar 4.10 Derajat Keanggotaan Warna Merah(μ_R) pada Data Masukan Merah 135

Dengan perhitungan μ_R low $= \frac{139-135}{139-134} = 0.8$; μ_R *med* $= \frac{135-134}{139-134} = 0.2$; μ_R *high* = 0

Sehingga nilai derajat keanggotaan μ_R ialah *low*=0.8, *med*=0.2, dan *high*=0.

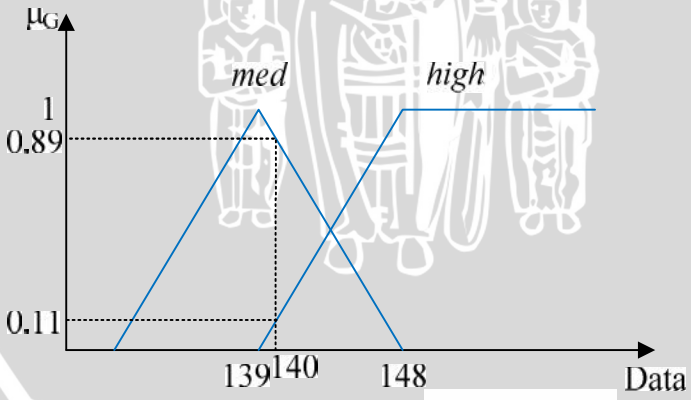
4.3.1.2 Fungsi Keanggotaan Masukan pada Warna Dasar Hijau

Seperti pada warna dasar merah, warna dasar hijau juga menggunakan tiga kurva fungsi keanggotaan, yaitu *low* untuk nilai rendah, *med* untuk nilai sedang, dan *high* untuk nilai tinggi. Metode *intuition* juga diterapkan pada penentuan nilai parameter batas tiap-tiap kurvanya. Gambar 4.11 berikut menunjukkan fungsi keanggotaan masukan pada warna dasar hijau.



Gambar 4.11 Fungsi Keanggotaan Masukan pada Warna Dasar Hijau

Misalnya, data warna terdeteksi untuk komponen warna dasar hijau adalah 140, maka derajat keanggotaan warna hijau (μ_H) didapat dengan cara sebagai berikut.



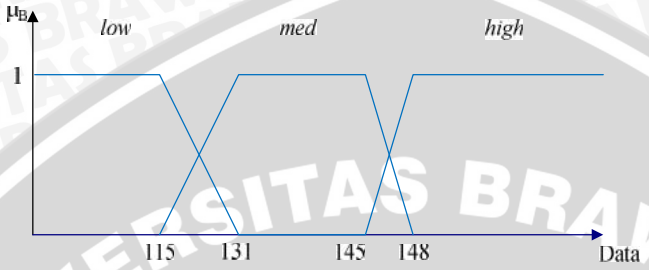
Gambar 4.12 Derajat Keanggotaan Warna Hijau(μ_G) pada Data Masukan Hijau 140

Dengan perhitungan μ_G *low* = 0; μ_G *med* $= \frac{148-140}{148-139} = 0.89$; μ_G *high* $= \frac{140-139}{148-139} = 0.11$.

Sehingga nilai derajat keanggotaan μ_G ialah *low*=0, *med*=0.89, dan *high*=0.11.

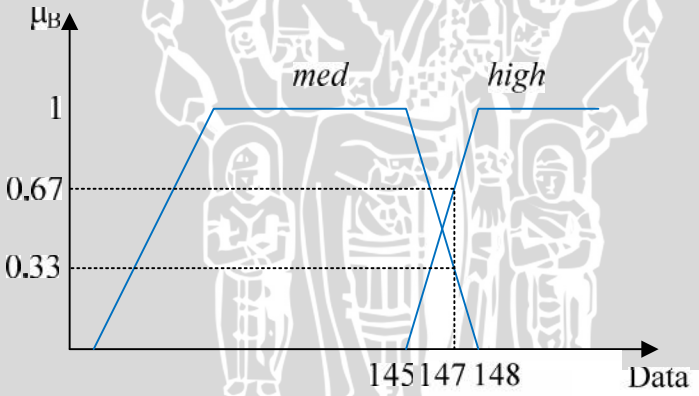
4.3.1.3 Fungsi Keanggotaan Masukan pada Warna Dasar Biru

Warna dasar biru juga terdiri dari tiga kurva fungsi keanggotaan, yaitu *low* pada nilai rendah, *med* untuk nilai sedang, dan *high* untuk nilai tinggi. Penentuan nilai parameter batas tiap-tiap kurva juga menerapkan metode *intuition.* Gambar 4.13 menunjukkan fungsi keanggotaan masukan pada warna dasar biru.



Gambar 4.13 Fungsi Keanggotaan Masukan pada Warna Dasar Biru

Misalnya, data warna terdeteksi untuk komponen warna dasar biru adalah 147, maka derajat keanggotaan warna biru (μ_B) didapat dengan cara sebagai berikut.



Gambar 4.14 Derajat Keanggotaan Warna Biru(μ_B) pada Data Masukan Biru 147

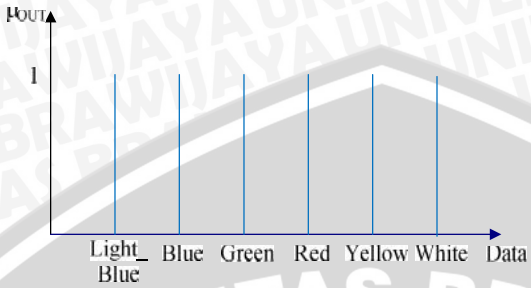
Dengan perhitungan μ_B *low* = 0; $\mu_B\, med = \frac{148-147}{148-145} = 0.33$; $\mu_B\, high = \frac{147-145}{148-145} = 0.67$.

Sehingga nilai derajat keanggotaan μ_R ialah *low*=0, *med*=0.33, dan *high*=0.67.

4.3.1.4 Fungsi Keanggotaan Keluaran Logika *Fuzzy*

Fungsi keanggotaan keluaran logika *fuzzy* berbentuk *singleton*. Pemilihan fungsi keanggotaan berbentuk *singleton* dengan tujuan untuk menghemat memori dan mempercepat eksekusi program.

Fungsi keanggotaan keluaran logika *fuzzy* terdiri dari 6 kurva, yaitu merah, biru, biru muda, putih, kuning dan hijau. Fungsi keanggotaan keluaran logika *fuzzy* ditunjukkan dalam Gambar 4.15.



Gambar 4.15 Fungsi Keanggotaan Keluaran Logika *Fuzzy*

4.3.2 Kaidah Atur (*Rule Base*) Kontrol Logika *Fuzzy*

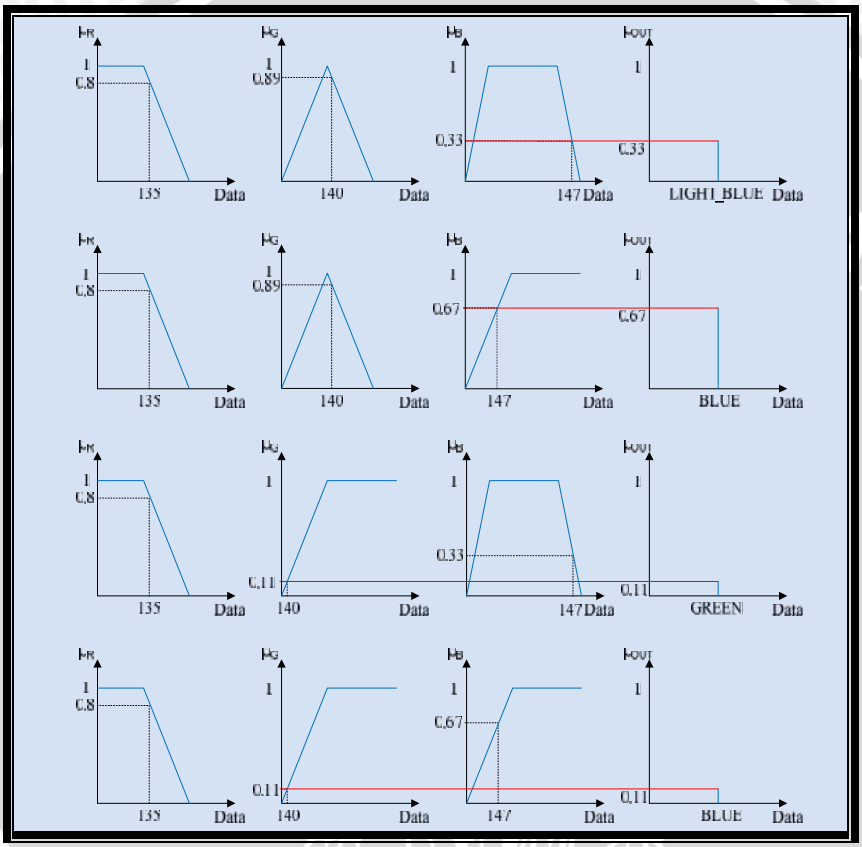
Penentuan kaidah atur kontrol logika fuzzy didasarkan pada pengalaman dan disusun dalam bentuk penalaran Jika-Maka (*If-Then*). Setelah masukan variabel *crips* diubah menjadi masukan *fuzzy*, selanjutnya diproses sesuai dengan kaidah aturnya. Metode penalaran *fuzzy* yang dipergunakan kontrol logika fuzzy adalah metode MAX-MIN. Kaidah atur kombinasi warna RGB dapat ditunjukkan dalam Tabel 4.1.

Tabel 4.1 Kaidah atur kombinasi warna RGB (*Red Green Blue*)

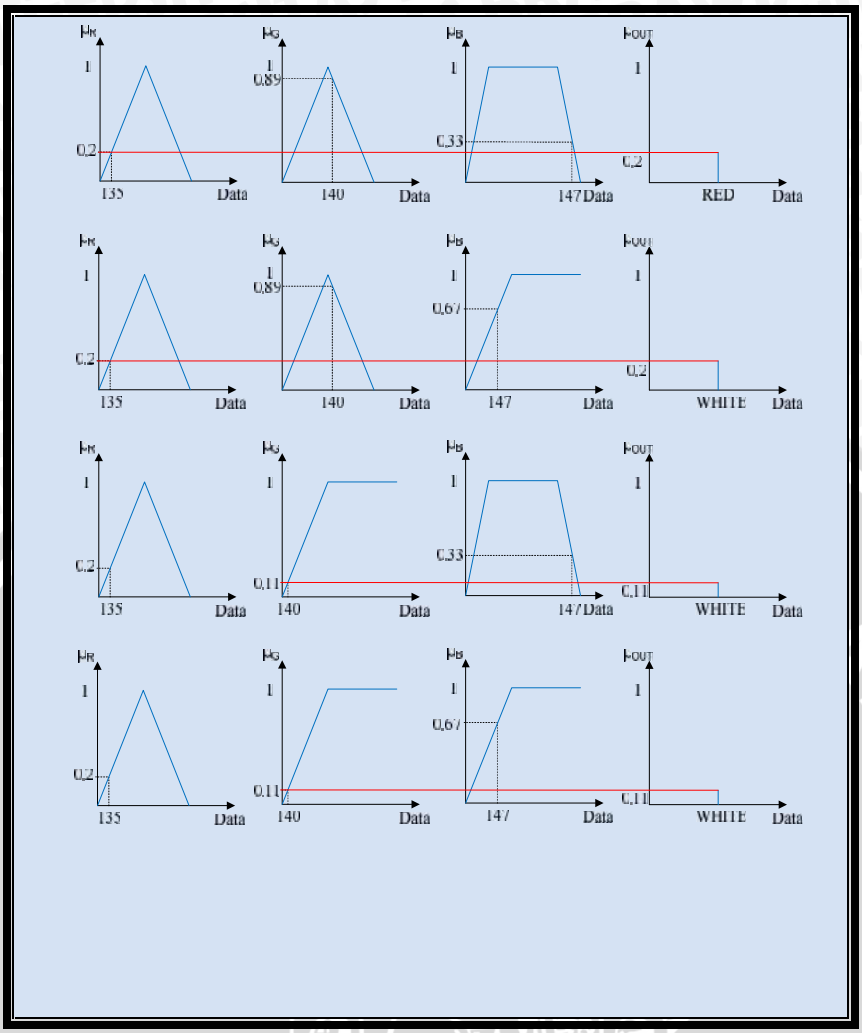
	R	*low*	*med*	*high*
G	**B**			
low	*low*	Light_Blue	Red	Red
med		Light_Blue	Yellow	Yellow
high		Green	Yellow	Yellow
low	*med*	Light_Blue	Red	Red
med		Light_Blue	White	White
high		Green	White	White
low	*high*	Blue	Red	Red
med		Blue	Red	White
high		Blue	White	White

4.3.3 Metode Inferensi *Max-min*

Visualisasi proses pengolahan masukan ketika sensor kamera membaca data warna dasar merah (R) 135, warna dasar hijau (G) terbaca data 140, dan warna dasar biru (B) terbaca data 147 seperti yang telah disebutkan pada *point* sebelumnya (4.3.1.1 – 4.3.1.3) dengan metode MAX-MIN menggunakan kaidah atur kombinasi warna RGB yang telah ditetapkan dapat ditunjukkan dalam Gambar 4.16 dan Gambar 4.17.



Gambar 4.16 Ilustrasi Metode Inferensi Max-min (1)



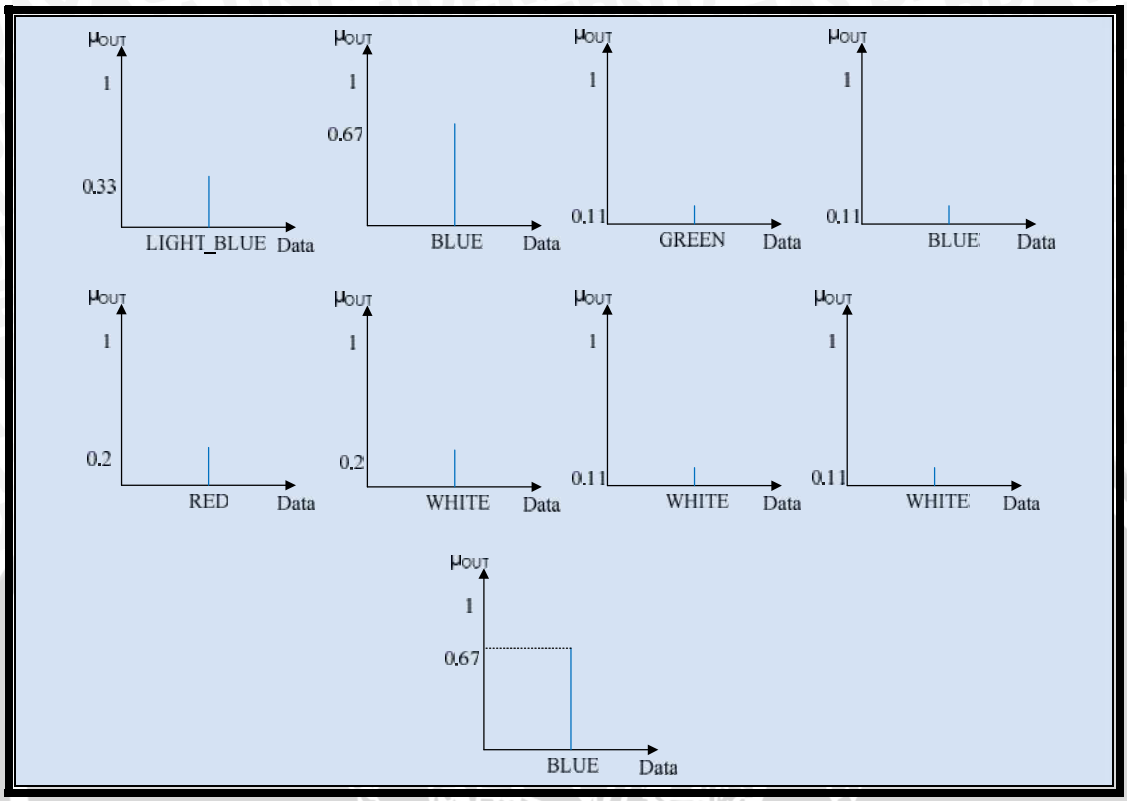
Gambar 4.17 Ilustrasi Metode Inferensi Max-min (2)

4.3.4 Defuzzifikasi

Defuzzifikasi adalah proses mengubah keluaran *fuzzy* menjadi keluaran *crips*. Hasil defuzzifikasi inilah yang digunakan untuk memberi sinyal perintah gerakan yang harus dilakukan robot *humanoid* kepada mikrokontroler *master*. Metode defuzzifikasi yang digunakan adalah *Max membership principle*. Metode ini digunakan melihat kondisi nilai *output* logika *fuzzy* yang tidak memungkinkan untuk dibuat nilai kuantisasi secara bertingkat.

Seperti permasalahan pada *point* 4.3.3, ketika sensor kamera membaca data warna dasar merah (R) 135, warna dasar hijau (G) terbaca data 140, dan warna dasar biru (B) terbaca data 147 maka nilai data yang muncul 0.67 (seperti pada Gambar

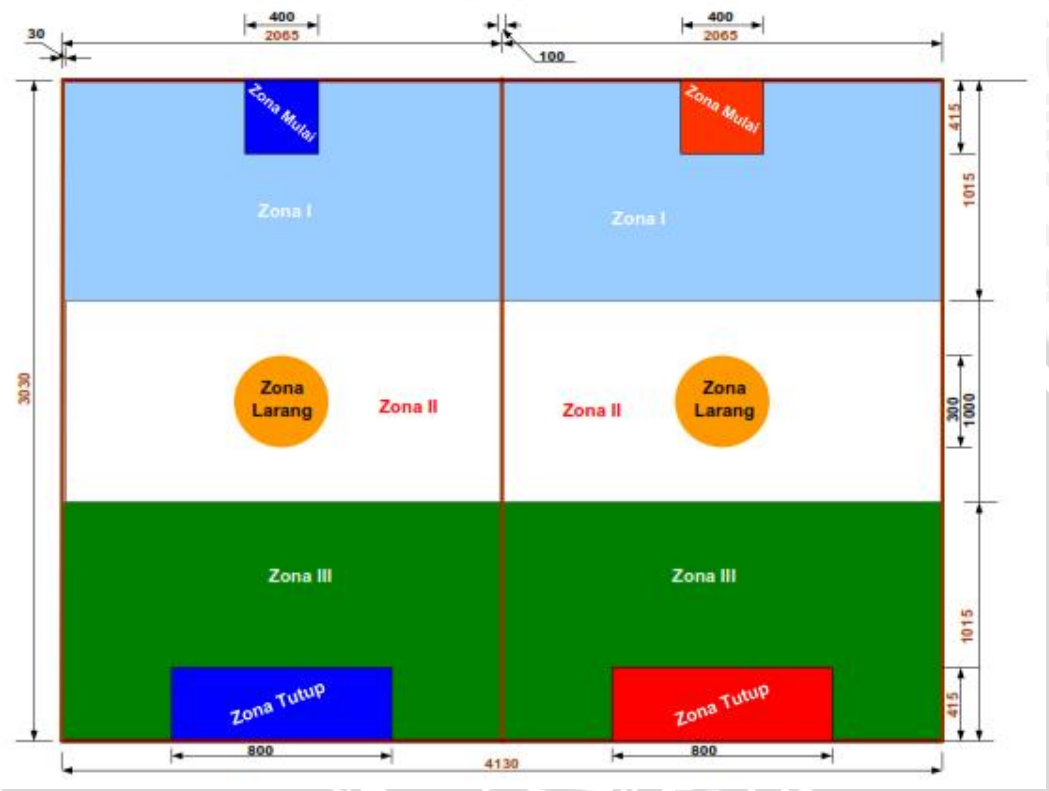
4.18), yang berarti dari pembacaan sensor tersebut pada lapangan dihasilkan keputusan warna biru (BLUE).



Gambar 4.18 Ilustrasi Proses Defuzzifikasi *Max Membership Principle*

4.4 Perancangan Gerakan

Lapangan KRSI tiap-tiap zona terdapati perbedaan warna seperti yang terlihat dalam Gambar 4.19.



Gambar 4.19 Lapangan KRSI 2013

Pada tiap-tiap zona, robot harus melakukan gerakan tari yang berbeda-beda pula. Gerakan tari ini berkaitan dengan nilai keluaran logika *fuzzy*. Jika logika *fuzzy* menerjemahkan hasil pendeteksian sensor kamera sebagai warna merah, maka mikrokontroler *slave* ATMega8 akan mengirim sinyal 3 bit dari pin D *output* berupa (PORTD.2=0;PORTD.3=0;PORTD.4=1) dan mikrokontroler *master* akan memerintahkan sinyal pada motor-motor DC servo sebagai aktuator untuk melakukan gerakan *sembah pembuka*. Seperti yang terlihat pada Gambar 4.20 berikut.

Gambar 4.20 Gerakan *Sembah Pembuka*

Jika logika *fuzzy* menerjemahkan hasil pendeteksian sensor kamera sebagai warna biru, maka mikrokontroler *slave* ATMega8 akan mengirim sinyal 3 bit dari pin D *output* berupa (PORTD.2=0;PORTD.3=1;PORTD.4=0) dan mikrokontroler *master* akan memerintahkan robot untuk melakukan gerakan *sembah pembuka* juga. Seperti yang terlihat pada Gambar 4.20.

Apabila warna biru muda yang telah terjemahkan oleh logika *fuzzy*, maka mikrokontroler *slave* ATMega8 akan mengirim sinyal 3 bit dari pin D *output* berupa (PORTD.2=1;PORTD.3=0;PORTD.4=0) dan mikrokontroler *master* akan memerintahkan sinyal pada motor-motor DC servo sebagai aktuator untuk melakukan gerakan *ngasak.* Seperti yang terlihat pada Gambar 4.21.

Gambar 4.21 Gerakan *Ngasak*

Saat logika *fuzzy* menerjemahkan hasil pendeteksian sensor kamera sebagai warna putih, maka mikrokontroler *slave* ATMega8 akan mengirim sinyal 3 bit dari pin D *output* berupa (PORTD.2=0;PORTD.3=1;PORTD.4=1) dan mikrokontroler *master* akan memerintahkan robot untuk melakukan gerakan *membelah awan* juga. Seperti yang terlihat pada Gambar 4.22.

Gambar 4.22 Gerakan *Membelah Awan*

Ketika logika *fuzzy* menerjemahkan hasil pendeteksian sensor kamera sebagai warna kuning, maka mikrokontroler *slave* ATMega8 akan mengirim sinyal 3 bit dari pin D *output* berupa (PORTD.2=1;PORTD.3=0;PORTD.4=1) dan mikrokontroler *master* akan memerintahkan robot untuk melakukan gerakan *menghindari zona larang* juga. Seperti yang terlihat pada Gambar 4.23.

Gambar 4.23 Gerakan *Menghindari Zona Larang*

Saat logika *fuzzy* menerjemahkan hasil pendeteksian sensor kamera sebagai warna kuning, maka mikrokontroler *slave* ATMega8 akan mengirim sinyal 3 bit dari pin D *output* berupa (PORTD.2=1;PORTD.3=1;PORTD.4=0) dan mikrokontroler

master akan memerintahkan robot untuk melakukan gerakan *menghindari zona larang* juga. Seperti yang terlihat pada Gambar 4.24.

Gambar 4.24 Gerakan *Memberi Cincin*

4.5 Perancangan Perangkat Lunak

4.5.1 Pengaturan Sensor Kamera

Pada penelitian ini CMUcam3 komunikasinya menggunakan sistem komunikasi USART. Format pengiriman data untuk melakukan penjejakan menggunakan perintah GM (*Get Mean*) dengan format sebagai berikut:

GM [Rmean Gmean Bmean Rmedian Gmedian Bmedian Rmode Gmode Bmode RstDev GstDev BStDev] \r

Paket pengiriman data yang akan diterima adalah sebagai berikut :

S 98 97 98 148 149 148 197 198 197 8 8 8

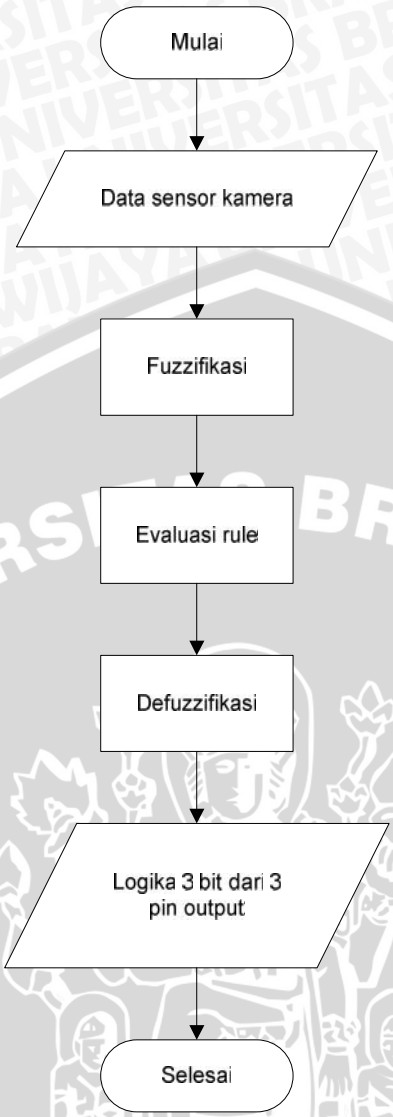
Merupakan tipe paket data ASCII yang diterima untuk memonitor warna.

Pengiriman perintah tersebut dilakukan dari mikrokontroler *slave* menuju sensor kamera CMUCAM4.

4.5.2 Perancangan Algoritma Mikrokontroler *Slave*

Mikrokontroler *slave* yang digunakan ialah mikrokontroler ATMega8. Mikrokontroler ini berperan sebagai tempat proses data dari sensor kamera CMUCAM4 dan proses KLF (kontrol logika *fuzzy*). Proses penerimaan data dari sensor kamera menggunakan komunikasi USART, sehingga Mikrokontroler *slave* dapat melakukan proses KLF dari data sensor kamera.

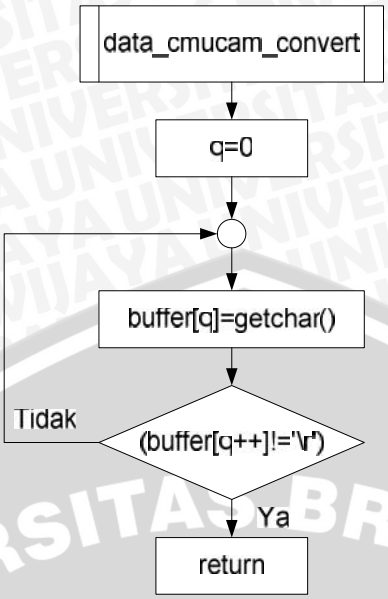
Selanjutnya proses awal dari kontroler logika *fuzzy* yaitu fuzzifikasi sesuai perancangan yang telah dibuat. Kemudian seluruh derajat keanggotaan yang telah didapatkan di evaluasi dengan *rule* yang telah ditentukan. Penulisan *rule* disusun menggunakan tipe data larik (*array*) atau bisa dikatakan seperti matriks agar proses eksekusi lebih cepat dan menghemat memori. Proses inferensinya dilakukan dengan menggunakan metode *Max-min*. Proses defuzzifikasi dilakukan dengan menghitung hasil evaluasi *rule* yang telah dilakukan dengan menggunakan metode *Max membership principle*. Hasil logika *fuzzy* yang berupa logika 3 bit dari 3 pin *output* mikrokontroler *slave* selanjutnya dikirimkan pada Mikrokontroler *master* (Mikrokontroler ATMega128). Diagram alir program utama kontrol logika *fuzzy* ditunjukan dalam Gambar 4.25.



Gambar 4.25 Diagram Alir Program Utama Kontrol Logika *Fuzzy*

4.5.2.1 Perancangan Subrutin Mengambil Data Unit *Decoder*

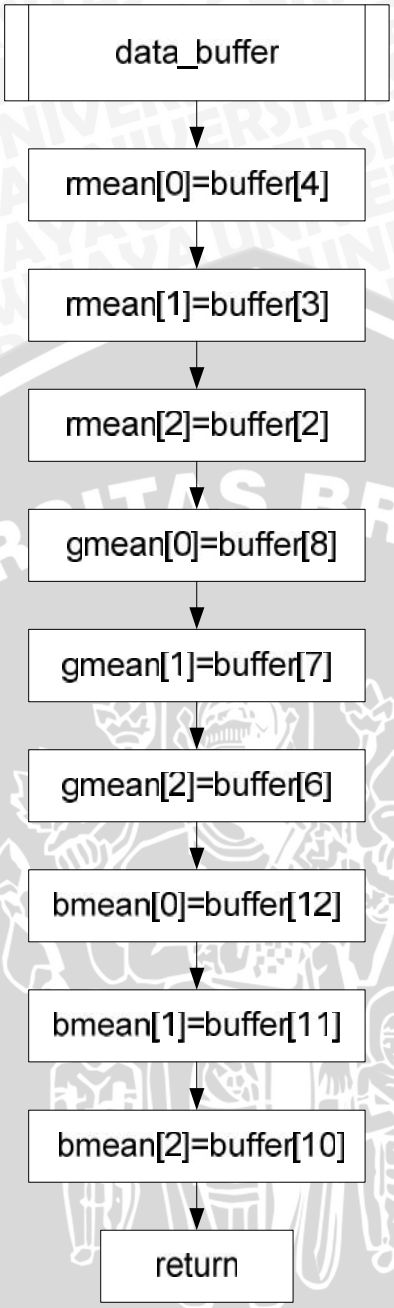
Subrutin mengambil data unit *decoder* ini berfungsi untuk menerima data dari kamera CMUCAM4 dengan mengunakan sistem komunikasi serial yang sebelumnya diperintah oleh *main controller*. Data serial yang diterima berupa karakter-karakter yang dikirim satu persatu dengan *baudrate* 19200 *bits per second*. Kemudian data-datanya akan dimasukkan dalam *array* yang diberi nama *buffer*. Dalam perancangan perangkat lunak ini menggunakan 13 buah *array* yang tiap *array*-nya berisi satu buah karakter. Diagram alir subrutin mengambil data unit *decoder* ditunjukkan dalam Gambar 4.26.



Gambar 4.26 Diagram Alir Subrutin Mengambil Data Unit *Decoder*

4.5.2.2 Perancangan Subrutin Memilah Data

Subrutin memilah data ini berfungsi untuk memudahkan dalam pengolahan data yang diterima dari unit *decoder* pada unit *main controller* yang sebelumnya sudah di dimasukkan dalam *array* yang diberi nama *buffer*. Dalam subrutin ini data-data yang dipisah berjumlah tiga buah data informasi keadaan objek antara lain data nilai warna merah pada objek (rmean), data nilai warna hijau pada objek (gmean), dan nilai nilai warna biru pada objek (bmean) di kamera. Diagram alir subrutin memilah data ditunjukkan dalam Gambar 4.27.

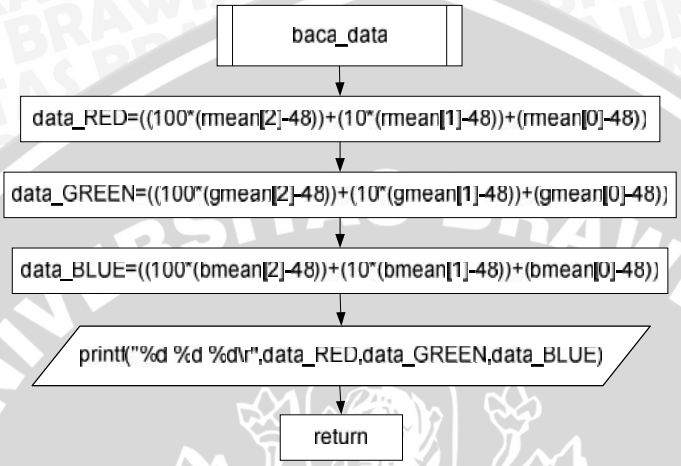


Gambar 4.27 Diagram Alir Subrutin Memilah Data

4.5.2.3 Perancangan Subrutin Mengubah Karakter ASCII ke Nilai *Integer*

Subrutin mengubah karakter ASCII ke nilai integer ini berfungsi untuk mengubah tipe data yang awalnya adalah karakter ASCII menjadi nilai *integer*. Pengubahan ini perlu dilakukan karena untuk memproses data-data informasi keadaan objek pada mikrokontroler harus berupa nilai *integer*. Dalam subrutin ini

program akan mengambil nilai ratusan dari data yang akan diubah, kemudian mengubahnya menjadi nilai *integer*. Selanjutnya program akan mengambil nilai puluhan dan mengubahnya menjadi nilai *integer*, dan yang terakhir program akan mengambil data satuan dan mengubahya menjadi nilai *integer*. Diagram alir subrutin mengubah karakter ASCII ke nilai *integer* ini ditunjukkan dalam Gambar 4.28.



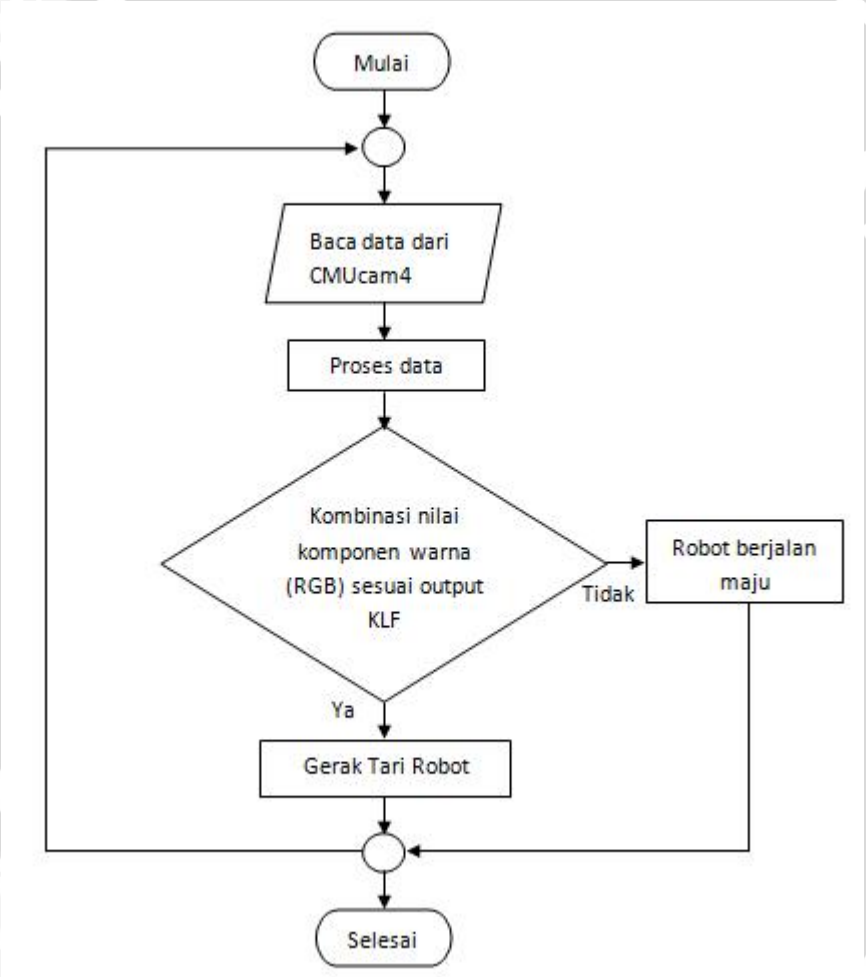
Gambar 4.28 Diagram Alir Subrutin Mengubah Karakter ASCII Ke Nilai *Integer*

4.5.3 Perancangan Algoritma Mokrokontroller *Master*

Mikrokontroler *master* berperan sebagari penerima hasil keputusan kontol logika *fuzzy* dari Mikrokontroler *slave* dan mengirimkan sinyal perintah kepada motor-motor DC servo sesuai dengan keputusan kontrol logika *fuzzy*. Sinyal perintah ke motor DC tersebut berupa sinyal lebar pulsa yang akan menentukan besar sudut motor DC servo terkait. Data terbaca dari Mikrokontroler *slave* berupa logika 3 bit dari 3 pin *output* dimana warna hasil keputusan kontrol logika *fuzzy*, logika 001 (Port D4-D2) untuk warna “biru muda”, “biru” 010, “merah” 011, “hijau” 100, “kuning” 101, dan “putih” 110.

Sehubungan bahwa dengan tiap-tiap zona berbeda warna dan berbeda gerakan tariannya pula (seperti yang telah dijelaskan pada poin 4.3 perancangan mekanik) maka digunakan penalaran *if-then* untuk eksekusi programnya. Dimulai *start* awal, untuk logika terbaca warna merah berarti robot berada pada Zona Mulai mode lapangan merah, maka eksekusi program gerakan tari *salam pembuka.* Perlu diketahui sebelumnya bahwa gerakan tari *salam pembuka* hingga yang terakhir

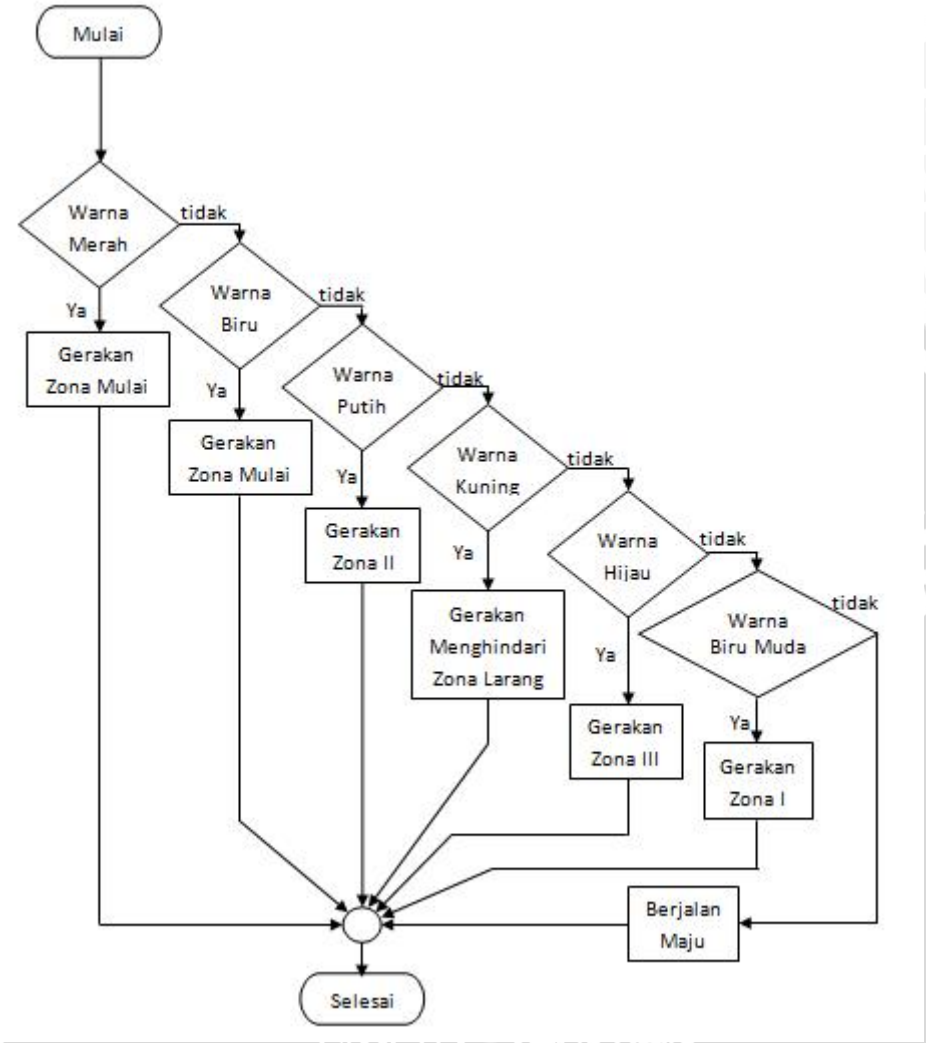
gerakan tari *salam penutup*, program kombinasi sinyal sudut motor DC servonya telah tersimpan dalam memori Mikrokontroler *master*. Selesai melakukan gerakan tari *salam pembuka*, robot akan berjalan maju hingga menemui zona lain (dalam artian terdeteksi warna lain selain warna merah atau warna sebelumnya). Kemudian robot akan melakukan gerakan sesuai hasil keputusan warna dari kontrol logika *fuzzy*-nya. Dan hal tersebut akan terus berlangsung hingga Zona Tutup dan robot selesai menjalankan tugasnya. Diagram alir perancangat perangakat lunak secara keseluruhan dapat dilihat pada Gambar 4.29.



Gambar 4.29 Diagram Alir Perancangan Perangkat Lunak

"Proses data" yaitu proses logika *fuzzy* terjadi pada mikrokontroler *slave*, sedangkan *output fuzzy* tersebut akan dikirim ke mikrokontroler *master* untuk kemudian terjadi proses "Gerak Tari Robot" yang akan berintegrasi dengan motor-

motor DC servo sebagai aktuatornya. Gambar 4.30 berikut menunjukkan diagram alir perancangan sub-fungsi gerakan robot.



Gambar 4.30 Diagram Alir Perancangan sub-fungsi Gerakan Robot

BAB V
PENGUJIAN DAN ANALISIS

Pengujian alat meliputi pengujian sensor kamera, pengujian logika *fuzzy*, dan pengujian keseluruhan sistem.

5.1 Pengujian Sensor Kamera

a. Tujuan

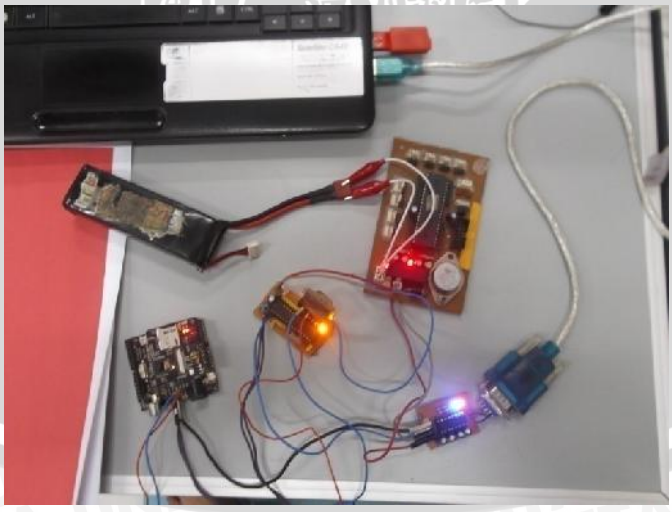
untuk mengetahui sistem komunikasi serta cara mengirim perintah dan data yang dikeluarkan dari kamera CMUCAM4.

b. Peralatan yang digunakan

- Modul kamera CMUCAM4
- Mikrokontroler ATMega8 beserta rangkaian minimum sistemnya.
- RS232
- Laptop dengan perangkat lunak *Parallax Serial Terminal* untuk mengetahui register apa saja yang bisa digunakan untuk memerintah kamera dan data apa saja yang dikeluarkan dari sensor.

c. Langkah pengujian

Peralatan dihubungkan seperti yang terlihat pada Gambar 5.1.



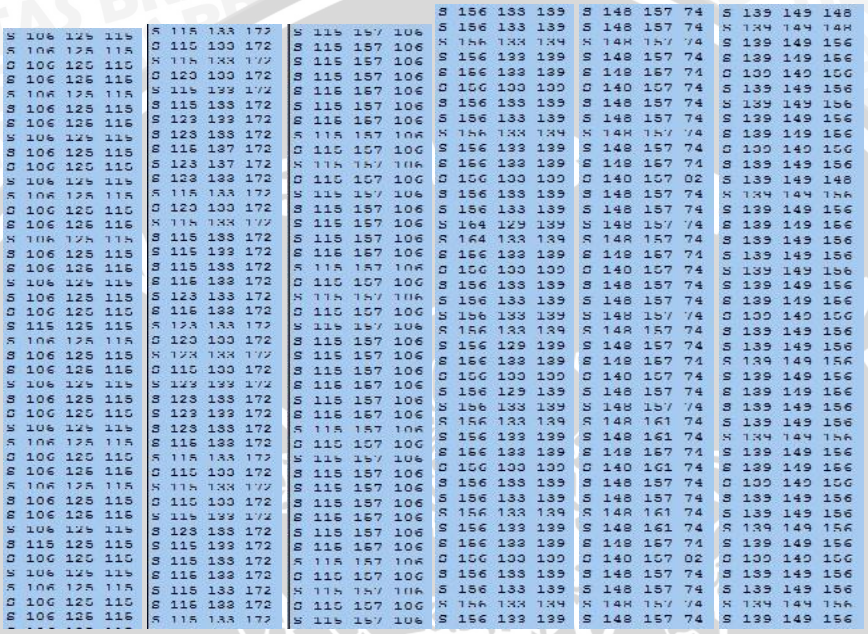
Gambar 5.1 Rangkaian Pengujian Sensor Kamera

Diberi catu daya sesuai dengan kebutuhan yaitu 5V. Modul kamera CMUCAM4 diarahkan pada objek warna yang akan dideteksi. Kemudian

mengirim perintah dari *software Parallax Serial Terminal* ke modul kamera dan merekam hasil pembacaan data tersebut.

d. Hasil pengujian

Hasil pengiriman perintah dan penerimaan data CMUCAM4 pada *Parallax Serial Terminal* ditunjukkan dalam Gambar 5.2 berikut.

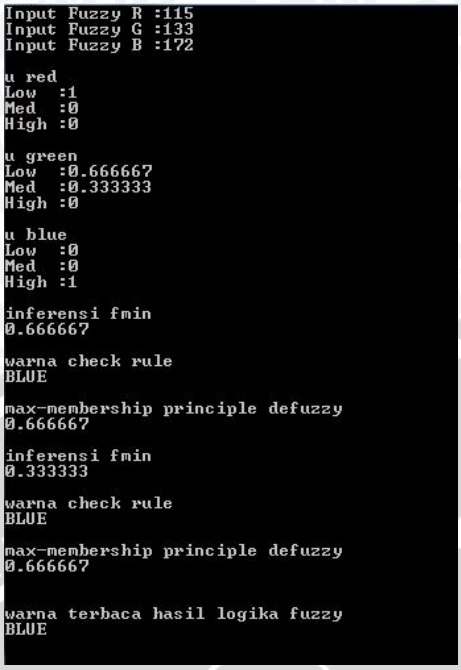


Gambar 5.2 Hasil Pembacaan Data ASCII RGB CMUCAM4 (Light_Blue,Blue,Green,Red,Yellow,White)

Data terebut bertipe karakter ASCII. Pembacaannya yaitu dari kiri ke kanan ialah data komponen warna dasar merah (*Red*), data komponen warna dasar hijau (*Green*), dan data komponen warna dasar biru (*Blue*).

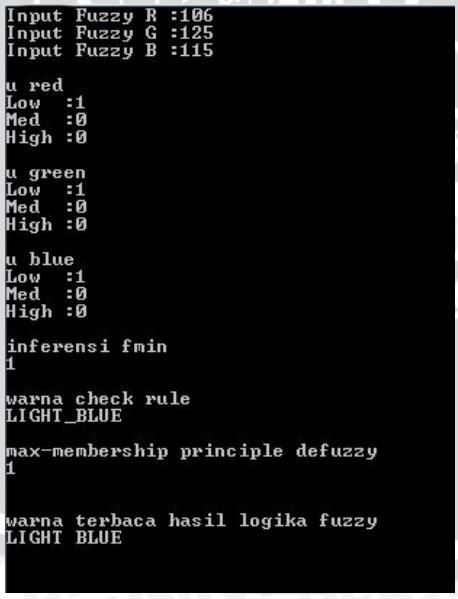
5.2 Pengujian Logika *Fuzzy*

Pengujian ini dilakukan untuk mengetahui performa sistem logika *fuzzy* yang akan diimplementasikan pada robot *humanoid* dalam mengolah data warna lintasan. Menggunakan perangkat lunak Dev C++ untuk mengetahui keluaran yang dihasilkan dari program logika *fuzzy* yang telah dibuat. Gambar 5.3 hingga Gambar 5.8 menunjukkan tampilan proses logika *fuzzy* dari data masukan warna objek terdeteksi.



Gambar 5.3 Proses Logika *Fuzzy* Masukan Warna Biru

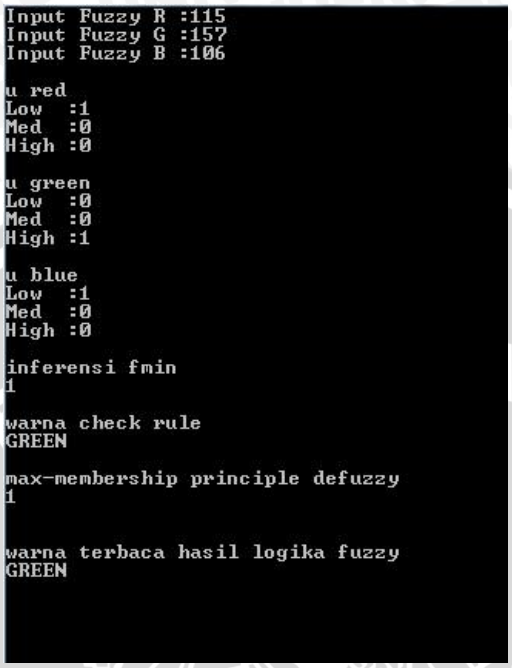
Proses sistem logika *fuzzy* dari data hasil pembacaan warna biru didapat hasil seperti pada Gambar 5.3. Dari hasil tersebut warna terbaca logika *fuzzy* adalah warna biru, sesuai dengan warna objek pembacaan sensor kamera CMUCAM4.



Gambar 5.4 Proses Logika *Fuzzy* Masukan Warna Biru Muda

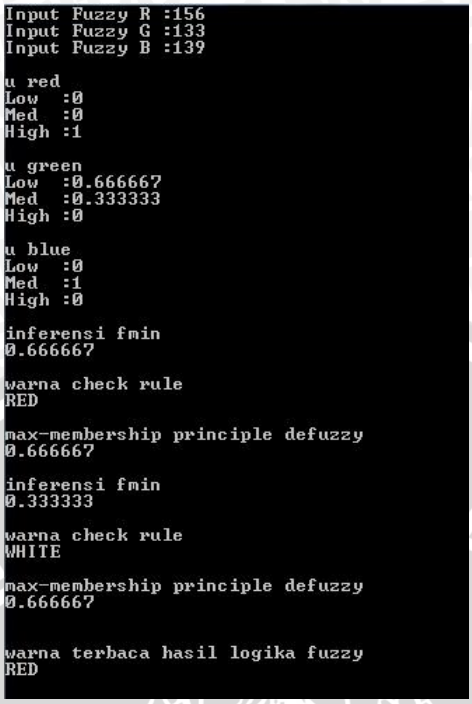
Proses sistem logika *fuzzy* dari data hasil pembacaan warna biru muda didapat hasil seperti pada Gambar 5.4. Dari hasil tersebut warna terbaca logika *fuzzy* adalah

warna biru muda, sesuai dengan warna objek pembacaan sensor kamera CMUCAM4.



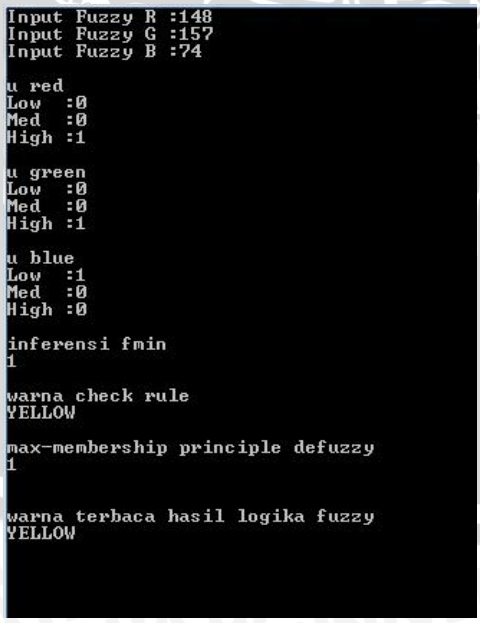
Gambar 5.5 Proses Logika *Fuzzy* Masukan Warna Hijau

Proses sistem logika *fuzzy* dari data hasil pembacaan warna hijau didapat hasil seperti pada Gambar 5.5. Dari hasil tersebut warna terbaca logika *fuzzy* adalah warna hijau, sesuai dengan warna objek pembacaan sensor kamera CMUCAM4.



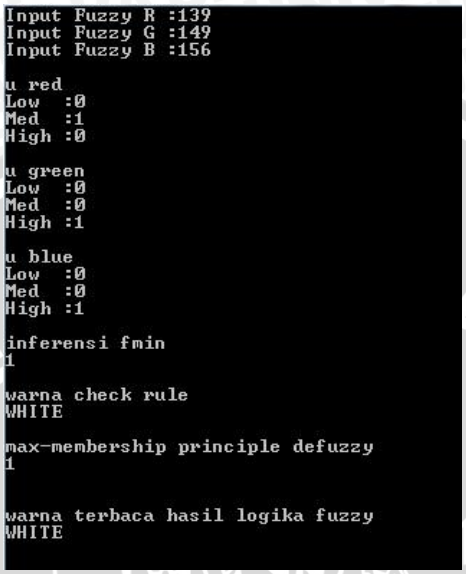
Gambar 5.6 Proses Logika *Fuzzy* Masukan Warna Merah

Proses sistem logika *fuzzy* dari data hasil pembacaan warna merah didapat hasil seperti pada Gambar 5.6. Dari hasil tersebut warna terbaca logika *fuzzy* adalah warna merah, sesuai dengan warna objek pembacaan sensor kamera CMUCAM4.



Gambar 5.7 Proses Logika *Fuzzy* Masukan Warna Kuning

Proses sistem logika *fuzzy* dari data hasil pembacaan warna kuning didapat hasil seperti pada Gambar 5.7. Dari hasil tersebut warna terbaca logika *fuzzy* adalah warna kuning, sesuai dengan warna objek pembacaan sensor kamera CMUCAM4.



Gambar 5.8 Proses Logika *Fuzzy* Masukan Warna Putih

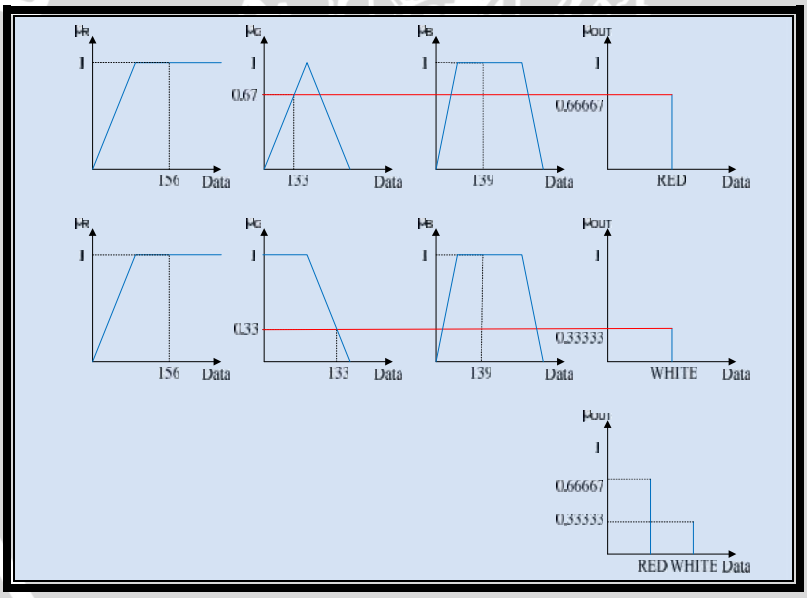
Proses sistem logika *fuzzy* dari data hasil pembacaan warna putih didapat hasil seperti pada Gambar 5.8. Dari hasil tersebut warna terbaca logika *fuzzy* adalah warna putih, sesuai dengan warna objek pembacaan sensor kamera CMUCAM4.

Pengujian ini dilakukan pada masukan warna-warna pada lintasan yang dapat ditunjukkan dalam Tabel 2.

Tabel 5.1 Hasil Pengujian Logika *Fuzzy*

No	Warna	*Input Fuzzy*						Inferensi	Perhitungan *Output*	Warna Terbaca
		R	μ(R)	G	μ(G)	B	μ(B)			
1	Biru Muda	106	low=1	125	low=1	115	low=1	μ(1)=1	1	LIGHT_BLUE
2	Biru	115	low=1	133	low=0.67 med=0.33	172	high=1	μ(1)=0.67 μ(2)=0.33	0.67	BLUE
3	Hijau	115	low=1	157	high=1	106	low=1	μ(1)=1	1	GREEN
4	Merah	156	high=1	133	low=0.67 med=0.33	139	med=1	μ(1)=0.67 μ(2)=0.33	0.67	RED
5	Kuning	148	high=1	157	high=1	74	low=1	μ(1)=1	1	YELLOW
6	Putih	139	med=1	149	high=1	156	high=1	μ(1)=1	1	WHITE

Dapat dilihat dari Tabel 5.1 bahwa hasil warna terbaca proses logika *fuzzy* telah sesuai dengan data *input* hasil pembacaan sensor kamera CMUCAM4. Berikut pemaparan proses logika *fuzzy* yang terjadi dari data *input* warna merah.



Gambar 5.9 Proses Inferensi Logika *Fuzzy* Data *Input* Warna Merah

Hasil inferensi dari rule menunjukkan ada dua nilai yang muncul, yaitu *RED* dan *WHITE*. Berdasarkan metode *max membership principle*, maka dengan nilai $\mu_{OUT\ (RED)}$ sebesar 0.66667 yang lebih besar dari $\mu_{OUT\ (WHITE)}$ maka keputusan warna terbaca yang diambil ialah warna merah.

5.3 Pengujian Keseluruhan Sistem

Pengujian sistem secara menyeluruh dilakukan dengan prosedur sebagai berikut:

1. Menyusun semua rangkaian yang dibutuhkan.
2. Menulis program *slave* pada mikrokontroler ATMega8
3. Menulis program *master* pada mikrokontroler ATMega128.
4. Memberikan catu daya pada sistem.
5. Menguji pembacaan warna sensor kamera terhadap objek terdeteksi

Hasil pengujian sistem secara keseluruhan pada Gambar 5.10 hingga Gambar 5.15. Hasil pengujian sistem secara keseluruhan juga disajikan dalam bentuk tabel, dapat dilihat pada Tabel 5.2 hingga 5.7.

Tabel 5.2 Pengujian Zona Mulai (Mode Biru)

Pengujian ke-	Warna yang Dideteksi	Gerakan yang dilakukan
1	Biru	sembah pembuka
2	Biru	sembah pembuka
3	Biru	sembah pembuka

Gambar 5.10 Pengujian Zona Mulai (Mode Biru) Gerakan *Sembah Pembuka*

Tabel 5.3 Pengujian Zona Mulai (Mode Merah)

Pengujian ke-	Warna yang Dideteksi	Gerakan yang dilakukan
1	Merah	sembah pembuka
2	Merah	sembah pembuka
3	Merah	sembah pembuka

Gambar 5.11 Pengujian Zona Mulai (Mode Merah) Gerakan *Sembah Pembuka*

Tabel 5.4 Pengujian Zona I

Pengujian ke-	Warna yang Dideteksi	Gerakan yang dilakukan
1	Biru_Muda	*Ngasak*
2	Biru_Muda	*Ngasak*
3	Biru_Muda	*Ngasak*

Gambar 5.12 Pengujian Zona I Gerakan *Ngasak*

Tabel 5.5 Pengujian Zona II

Pengujian ke-	Warna yang Dideteksi	Gerakan yang dilakukan
1	Putih	*Membelah Awan*
2	Putih	*Membelah Awan*
3	Putih	*Membelah Awan*

Gambar 5.13 Pengujian Zona II Gerakan *Membelah Awan*

Tabel 5.6 Pengujian Menghindari Zona Larang

Pengujian ke-	Warna yang Dideteksi	Gerakan yang dilakukan
1	Kuning	*Menghindari Zona Larang*
2	Kuning	*Menghindari Zona Larang*
3	Kuning	*Menghindari Zona Larang*

Gambar 5.14 Pengujian Menghindari Zona Larang

Tabel 5.7 Pengujian Zona III

Pengujian ke-	Warna yang Dideteksi	Gerakan yang dilakukan
1	Hijau	*Memberi Cincin*
2	Hijau	*Memberi Cincin*
3	Hijau	*Memberi Cincin*

Gambar 5.15 Pengujian Zona III Gerakan *Memberi Cincin*

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan perancangan dan hasil pengujian, dapat diambil kesimpulan:

1. Data dari sensor kamera dapat diperoleh dengan menggunakan komunikasi USART.
2. Dari beberapa hasil pengujian logika fuzzy untuk pembacaan 6 warna yang berbeda didapatkan hasil yang sama dengan hitungan teorinya, dengan tingkat keberhasilan pada pengujian tersebut 100%.
3. Dari 3 hasil pengujian sistem secara menyeluruh pada masing-masing zona, diperoleh hasil yang sesuai dengan yang diharapkan, dengan tingkat keberhasilan pada pengujian tersebut 100%.

6.2 Saran

Pada penelitian ini pencahayaan dilakukan pada kondisi yang sama (intensitas cahaya ±210 lux). Untuk penelitian lanjutan perlu dikembangkan metode untuk mengatasi kesalahan deteksi warna akibat perbedaan pencahayaan.

DAFTAR PUSTAKA

Atmel. 2011. *8-bit AVR with 128KBytes In-System Programmable Flash ATmega128, ATmega128L*. San Jose: Atmel.

Atmel. 2007. *8-bit AVR with 8KBytes In-System Programmable Flash ATmega8, ATmega8L*. San Jose: Atmel.

Coughanowr, D.R. 1991. *Process System–Analysis and Control*. USA: Prentice Hall.

Datasheet CMUCAM4. 2012. *CMUCAM4 Board Layout and Port*. Kwabena W. Agyeman, Anthony Rowe, and Carnegie Mellon University.

Datasheet CMUCAM4. 2012. *CMUCAM4 Command List v1.00*. Kwabena W. Agyeman, Anthony Rowe, and Carnegie Mellon University.

Datasheet CMUCAM4. 2012. *CMUCAM4 Testing Guide v1.02*. Kwabena W. Agyeman, Anthony Rowe, and Carnegie Mellon University.

DIKTI. 2013. Panduan Kontes Robot Seni Indonesia 2013. Jakarta: DIKTI.

Habibur, Irfan R. 2012. Implementasi Sensor Warna TCS 230 pada Robot Humanoid KRSI Sebagai Mode Tarian di Arena Pelombaan. Malang: Brawijaya University.

Kuswadi, Son. 2007. Kendali Cerdas. Yogyakarta: ANDI.

Ogata, Katsuhiko. 1997. *Modern Control Engineering (Third Edition).* USA: Prentice-Hali.

Parallax. 2010. *Parallax Standard Servo.* California: Parallax.

Rahman, Irfan Habibur. 2012. Impelementasi Sensor Warna TCS 230 pada Robot Humanoid KRSI Sebagai Mode Tarian di Arena Perlombaan. Skripsi Program Sarjana. Universitas Brawijaya.

Reznik, Leonid. 1997. *Fuzzy Controllers*. Melbourne: Newnes.

Ross, Timothy J. 1997. *Fuzzy Logic With Engineering Applications.* Singapore: McGraw-Hill.

Sigit, Riyanto. 2007. Robotika, Sensor, dan Aktuator. Yogyakarta: Graha Ilmu.

Standardisasi Industri Grafika (ATGMI). 2008. Model Warna RGB. http://pengantar-warna.blogspot.com/2008/10/model-warna-rgb.html. Diakses 1 Juli 2013

Yan,J.,Ryan,M., dan Power,J. 1993. *Using Fuzzy Logic*. NewYork: Prentice Hall.

Yunardi , Riky Tri. 2011. Implementasi Kamera CMUcam3 pada Robot Tank yang Dapat Mengikuti Objek Berwarna. Skripsi Program Sarjana. Universitas Brawijaya.

LAMPIRAN

LAMPIRAN I

FOTO ALAT

Gambar 1 Robot *Humanoid* IBRAHIM

LAMPIRAN II

LISTING PROGRAM

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.04.4a Advanced
Automatic Program Generator
© Copyright 1998-2009 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project : Skripsi
Version : Fuzzy_cmucam4
Date    : 22/06/2013
Author  : NeVaDa
Company : glad
Comments:


Chip type           : ATmega8
Program type        : Application
AVR Core Clock frequency: 16,000000 MHz
Memory model        : Small
External RAM size   : 0
Data Stack size     : 256
*****************************************************/

#include <mega8.h>
#include <delay.h>
// Standard Input/Output functions
#include <stdio.h>
#include <iostream>
#include <conio.h>
#include <math.h>
#define LOW 0
#define MED 1
```

```
#define HIGH 2
#define LIGHT_BLUE 0
#define BLUE 1
#define GREEN 2
#define RED 3
#define YELLOW 4
#define WHITE 5


char buffer[],rmean[],gmean[],bmean[];
int data_RED, data_GREEN, data_BLUE, data_RE, data_GR, data_BL;
int data_r[],data_g[],data_b[];

void data_cmucam(void)
{
int q;
do
  {
  buffer[q]=getchar();
  }
while
  (buffer[q++]=='\r');
}

void data_rmean(void)
{
if ((buffer[1]==' ')&&(buffer[3]==' ')&&(buffer[5]==' ')&&(buffer[7]==' '))
  {
  rmean[0]=buffer[2];
  rmean[1]='0';
  rmean[2]='0';
  gmean[0]=buffer[4];
```

```
  gmean[1]='0';
  gmean[2]='0';
  bmean[0]=buffer[6];
  bmean[1]='0';
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[3]==' ')&&(buffer[6]==' ')&&(buffer[8]==' '))
  {
  rmean[0]=buffer[2];
  rmean[1]='0';
  rmean[2]='0';
  gmean[0]=buffer[5];
  gmean[1]=buffer[4];
  gmean[2]='0';
  bmean[0]=buffer[7];
  bmean[1]='0';
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[3]==' ')&&(buffer[7]==' ')&&(buffer[9]==' '))
  {
  rmean[0]=buffer[2];
  rmean[1]='0';
  rmean[2]='0';
  gmean[0]=buffer[6];
  gmean[1]=buffer[5];
  gmean[2]=buffer[4];
  bmean[0]=buffer[8];
  bmean[1]='0';
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[3]==' ')&&(buffer[5]==' ')&&(buffer[8]==' '))
  {
```

```
  rmean[0]=buffer[2];
  rmean[1]='0';
  rmean[2]='0';
  gmean[0]=buffer[4];
  gmean[1]='0';
  gmean[2]='0';
  bmean[0]=buffer[7];
  bmean[1]=buffer[6];
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[3]==' ')&&(buffer[6]==' ')&&(buffer[9]==' '))
  {
  rmean[0]=buffer[2];
  rmean[1]='0';
  rmean[2]='0';
  gmean[0]=buffer[5];
  gmean[1]=buffer[4];
  gmean[2]='0';
  bmean[0]=buffer[8];
  bmean[1]=buffer[7];
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[3]==' ')&&(buffer[7]==' ')&&(buffer[10]==' '))
  {
  rmean[0]=buffer[2];
  rmean[1]='0';
  rmean[2]='0';
  gmean[0]=buffer[6];
  gmean[1]=buffer[5];
  gmean[2]=buffer[4];
  bmean[0]=buffer[9];
  bmean[1]=buffer[8];
```

```
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[3]==' ')&&(buffer[5]==' ')&&(buffer[9]==' '))
  {
  rmean[0]=buffer[2];
  rmean[1]='0';
  rmean[2]='0';
  gmean[0]=buffer[4];
  gmean[1]='0';
  gmean[2]='0';
  bmean[0]=buffer[8];
  bmean[1]=buffer[7];
  bmean[2]=buffer[6];
  }
else if ((buffer[1]==' ')&&(buffer[3]==' ')&&(buffer[6]==' ')&&(buffer[10]==' '))
  {
  rmean[0]=buffer[2];
  rmean[1]='0';
  rmean[2]='0';
  gmean[0]=buffer[5];
  gmean[1]=buffer[4];
  gmean[2]='0';
  bmean[0]=buffer[9];
  bmean[1]=buffer[8];
  bmean[2]=buffer[7];
  }
else if ((buffer[1]==' ')&&(buffer[3]==' ')&&(buffer[7]==' ')&&(buffer[11]==' '))
  {
  rmean[0]=buffer[2];
  rmean[1]='0';
  rmean[2]='0';
  gmean[0]=buffer[6];
```

```
  gmean[1]=buffer[5];
  gmean[2]=buffer[4];
  bmean[0]=buffer[10];
  bmean[1]=buffer[9];
  bmean[2]=buffer[8];
  }
else if ((buffer[1]==' ')&&(buffer[4]==' ')&&(buffer[6]==' ')&&(buffer[8]==' '))
  {
  rmean[0]=buffer[3];
  rmean[1]=buffer[2];
  rmean[2]='0';
  gmean[0]=buffer[5];
  gmean[1]='0';
  gmean[2]='0';
  bmean[0]=buffer[7];
  bmean[1]='0';
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[4]==' ')&&(buffer[7]==' ')&&(buffer[9]==' '))
  {
  rmean[0]=buffer[3];
  rmean[1]=buffer[2];
  rmean[2]='0';
  gmean[0]=buffer[6];
  gmean[1]=buffer[5];
  gmean[2]='0';
  bmean[0]=buffer[8];
  bmean[1]='0';
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[4]==' ')&&(buffer[8]==' ')&&(buffer[10]==' '))
  {
```

```
  rmean[0]=buffer[3];
  rmean[1]=buffer[2];
  rmean[2]='0';
  gmean[0]=buffer[7];
  gmean[1]=buffer[6];
  gmean[2]=buffer[5];
  bmean[0]=buffer[9];
  bmean[1]='0';
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[4]==' ')&&(buffer[6]==' ')&&(buffer[9]==' '))
  {
  rmean[0]=buffer[3];
  rmean[1]=buffer[2];
  rmean[2]='0';
  gmean[0]=buffer[5];
  gmean[1]='0';
  gmean[2]='0';
  bmean[0]=buffer[8];
  bmean[1]=buffer[7];
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[4]==' ')&&(buffer[7]==' ')&&(buffer[10]==' '))
  {
  rmean[0]=buffer[3];
  rmean[1]=buffer[2];
  rmean[2]='0';
  gmean[0]=buffer[6];
  gmean[1]=buffer[5];
  gmean[2]='0';
  bmean[0]=buffer[9];
  bmean[1]=buffer[8];
```

```
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[4]==' ')&&(buffer[8]==' ')&&(buffer[11]==' '))
  {
  rmean[0]=buffer[3];
  rmean[1]=buffer[2];
  rmean[2]='0';
  gmean[0]=buffer[7];
  gmean[1]=buffer[6];
  gmean[2]=buffer[5];
  bmean[0]=buffer[10];
  bmean[1]=buffer[9];
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[4]==' ')&&(buffer[6]==' ')&&(buffer[10]==' '))
  {
  rmean[0]=buffer[3];
  rmean[1]=buffer[2];
  rmean[2]='0';
  gmean[0]=buffer[5];
  gmean[1]='0';
  gmean[2]='0';
  bmean[0]=buffer[9];
  bmean[1]=buffer[8];
  bmean[2]=buffer[7];
  }
else if ((buffer[1]==' ')&&(buffer[4]==' ')&&(buffer[7]==' ')&&(buffer[11]==' '))
  {
  rmean[0]=buffer[3];
  rmean[1]=buffer[2];
  rmean[2]='0';
  gmean[0]=buffer[6];
```

```
  gmean[1]=buffer[5];
  gmean[2]='0';
  bmean[0]=buffer[10];
  bmean[1]=buffer[9];
  bmean[2]=buffer[8];
  }
else if ((buffer[1]==' ')&&(buffer[4]==' ')&&(buffer[8]==' ')&&(buffer[12]==' '))
  {
  rmean[0]=buffer[3];
  rmean[1]=buffer[2];
  rmean[2]='0';
  gmean[0]=buffer[7];
  gmean[1]=buffer[6];
  gmean[2]=buffer[5];
  bmean[0]=buffer[11];
  bmean[1]=buffer[10];
  bmean[2]=buffer[9];
  }
else if ((buffer[1]==' ')&&(buffer[5]==' ')&&(buffer[7]==' ')&&(buffer[9]==' '))
  {
  rmean[0]=buffer[4];
  rmean[1]=buffer[3];
  rmean[2]=buffer[2];
  gmean[0]=buffer[6];
  gmean[1]='0';
  gmean[2]='0';
  bmean[0]=buffer[8];
  bmean[1]='0';
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[5]==' ')&&(buffer[8]==' ')&&(buffer[10]==' '))
  {
```

```
  rmean[0]=buffer[4];
  rmean[1]=buffer[3];
  rmean[2]=buffer[2];
  gmean[0]=buffer[7];
  gmean[1]=buffer[6];
  gmean[2]='0';
  bmean[0]=buffer[9];
  bmean[1]='0';
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[5]==' ')&&(buffer[9]==' ')&&(buffer[11]==' '))
  {
  rmean[0]=buffer[4];
  rmean[1]=buffer[3];
  rmean[2]=buffer[2];
  gmean[0]=buffer[8];
  gmean[1]=buffer[7];
  gmean[2]=buffer[6];
  bmean[0]=buffer[10];
  bmean[1]='0';
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[5]==' ')&&(buffer[7]==' ')&&(buffer[10]==' '))
  {
  rmean[0]=buffer[4];
  rmean[1]=buffer[3];
  rmean[2]=buffer[2];
  gmean[0]=buffer[6];
  gmean[1]='0';
  gmean[2]='0';
  bmean[0]=buffer[9];
  bmean[1]=buffer[8];
```

```
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[5]==' ')&&(buffer[8]==' ')&&(buffer[11]==' '))
  {
  rmean[0]=buffer[4];
  rmean[1]=buffer[3];
  rmean[2]=buffer[2];
  gmean[0]=buffer[7];
  gmean[1]=buffer[6];
  gmean[2]='0';
  bmean[0]=buffer[10];
  bmean[1]=buffer[9];
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[5]==' ')&&(buffer[9]==' ')&&(buffer[12]==' '))
  {
  rmean[0]=buffer[4];
  rmean[1]=buffer[3];
  rmean[2]=buffer[2];
  gmean[0]=buffer[8];
  gmean[1]=buffer[7];
  gmean[2]=buffer[6];
  bmean[0]=buffer[11];
  bmean[1]=buffer[10];
  bmean[2]='0';
  }
else if ((buffer[1]==' ')&&(buffer[5]==' ')&&(buffer[7]==' ')&&(buffer[11]==' '))
  {
  rmean[0]=buffer[4];
  rmean[1]=buffer[3];
  rmean[2]=buffer[2];
  gmean[0]=buffer[6];
```

```
    gmean[1]='0';
    gmean[2]='0';
    bmean[0]=buffer[10];
    bmean[1]=buffer[9];
    bmean[2]=buffer[8];
    }
  else if ((buffer[1]==' ')&&(buffer[5]==' ')&&(buffer[8]==' ')&&(buffer[12]==' '))
    {
    rmean[0]=buffer[4];
    rmean[1]=buffer[3];
    rmean[2]=buffer[2];
    gmean[0]=buffer[7];
    gmean[1]=buffer[6];
    gmean[2]='0';
    bmean[0]=buffer[11];
    bmean[1]=buffer[10];
    bmean[2]=buffer[9];
    }
  else if ((buffer[1]==' ')&&(buffer[5]==' ')&&(buffer[9]==' ')&&(buffer[13]==' '))
    {
    rmean[0]=buffer[4];
    rmean[1]=buffer[3];
    rmean[2]=buffer[2];
    gmean[0]=buffer[8];
    gmean[1]=buffer[7];
    gmean[2]=buffer[6];
    bmean[0]=buffer[12];
    bmean[1]=buffer[11];
    bmean[2]=buffer[10];
    }

  }
```

```
void baca_data(void)
{
 data_RED = ((100*(rmean[2])) + (10*(rmean[1])) + (rmean[0]));
 data_GREEN = ((100*(gmean[2])) + (10*(gmean[1])) + (gmean[0]));
 data_BLUE = ((100*(bmean[2])) + (10*(bmean[1])) + (bmean[0]));

}
// Declare your global variables here
//using namespace std;
float alpha=0,alpha2[][][],alpha_nilaiout=0,alpha_nilaiout2=0;
float triangle(int value,float x0,float x1, float x2);
float grade(int value, float x0, float x1);
float reverse_grade(int value, float x0, float x1);

float red[];
float green[];
float blue[];
float warna_out[];
float hasil;

void fuzzy();
void check_rule();
void defuzzy();
unsigned char warna_rule[3][3][3]={
                  {
                    {LIGHT_BLUE,LIGHT_BLUE,GREEN};
                    {LIGHT_BLUE,LIGHT_BLUE,GREEN};
                    {GREEN,BLUE,BLUE};
                  }
                  {
                    {RED,YELLOW,YELLOW};
```

```
                {RED,WHITE,WHITE};
                {RED,WHITE,WHITE};
            }
            {
                {RED,YELLOW,YELLOW};
                {RED,WHITE,WHITE};
                {RED,WHITE,WHITE};
            }
};

float triangle(int value, float x0, float  x1,float x2)
{
float result;
float x;
x = value;
if((x <= x0) || (x >= x2))
      {result = 0;}
else if(x == x1)
      {result = 1;}
else
      {result =(((-x)+x2)/(x2-x1));}
return result;
}

float grade(int value,float x0, float x1)
{
float result;
float x;
x = value;
if(x <= x0)
      {result = 0;}
```

```
else if(x >= x1)
        {result = 1;}
else
        {result = ((x-x0)/(x1-x0));}
return result;
}

float reverse_grade(int value,float x0,float  x1)
{
float result;
float x;
x = value;
if(x <= x0)
        {result = 1;}
else if(x >= x1)
        {result = 0;}
else
        {result = (((-x)+x1)/(x1-x0));}
return result;
}

float trapesium(int value,float x0,float x1,float x2,float x3)
{
float result;
float x;
x = value;

if((x <= x0)||(x>=x3)) {result = 0;}
else if((x>=x1) && (x<=x2)){result = 1;}
else {result = ((-x+x3)/(x3-x2));}
return result;
}
```

```
void fuzzy()
{
red[LOW]    =   reverse_grade(data_RED;134;139);          //134,142
red[MED]    =   triangle(data_RED,134;139;146);           //134,142,150
red[HIGH]   =   grade(data_RED;139;146);                  //142,150

green[LOW]  =   reverse_grade(data_GREEN;130;139);
green[MED]  =   triangle(data_GREEN,130;139;148);
green[HIGH] =   grade(data_GREEN;139;148);

blue[LOW]   =   reverse_grade(data_BLUE;123;131);
blue[MED]   =   trapesium(data_BLUE;123;131145;148);
blue[HIGH]  =   grade(data_BLUE;145;148);

}

void check_rule()
{
float u_min;
char x,y,z;
for (x = 0; x < 10; x++)
{
warna_out[x] = 0;
}

for (x = 0; x < 3; x++)    //red
{
   for (y = 0; y < 3; y++)     //blue
   {
      for (z = 0; z < 3; z++)     //green
      {
```

```
            if (red[x] > 0 && green [z] > 0 && blue[y] > 0)
            {
            u_min = (float)fmin(red[x],green[z]);
            u_min = (float)fmin(blue[y],u_min);

            alpha=alpha+u_min;

            if (warna_rule[x][y][z]=LIGHT_BLUE)
            {alpha_nilaiout2=35*u_min;}
            else if (warna_rule[x][y][z]=BLUE)
            {alpha_nilaiout2=70*u_min;}
            else if (warna_rule[x][y][z]=GREEN)
            {alpha_nilaiout2=105*u_min;}
            else if (warna_rule[x][y][z]=RED)
            {alpha_nilaiout2=140*u_min;}
            else if (warna_rule[x][y][z]=YELLOW)
            {alpha_nilaiout2=175*u_min;}
            else if (warna_rule[x][y][z]=WHITE)
            {alpha_nilaiout2=210*u_min;}

            alpha_nilaiout=alpha_nilaiout2;
            }
        }
        }
}
}

float defuzzy()
{
float result_warna;
result_warna = (float) (alpha_nilaiout / alpha);
}
```

```
void main(void)
{
// Declare your local variables here
int i;
// Input/Output Ports initialization
// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=Out
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=0 State0=T
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out
Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
TCCR0=0x00;
TCNT0=0x00;
```

```
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 250,000 kHz
// Mode: Fast PWM top=00FFh
// OC1A output: Non-Inv.
// OC1B output: Non-Inv.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0xD5;
TCCR1B=0x2C;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
```

```
// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
MCUCR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 19200
UCSRA=0x12;
UCSRB=0x6B;
UCSRC=0xA5;
UBRRH=0x77;
UBRRL=0x22;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

delay_ms(10000);
printf("RS\\r");
printf("RS\\r");
printf("SW\\r");
printf("GM\\r");
```

```
while(1)

for(j=0;j<20;j++)
{
  {
    {
    for(i=0;i<20;i++)
    {
     data_cmucam();
     data_rmean();
     baca_data();
    data_r[i]=data_RE;
    data_g[i]=data_GR;
    data_b[i]=data_BL;
    }
    putchar('1');
    for(i=0;i<20;i++)
    {
    data_r[i]=data_r[i] + data_r[i+1];
    data_r[i+1]=data_r[i];
    data_g[i]=data_g[i] + data_g[i+1];
    data_g[i+1]=data_g[i];
    data_b[i]=data_b[i] + data_b[i+1];
    data_b[i+1]=data_b[i];
    }
    putchar('2');
    data_RED=data_r[19]/20;
    data_GREEN=data_g[19]/20;
    data_BLUE=data_b[19]/20;
    data_RED = 115;
```

```
        data_GREEN = 133;
        data_BLUE = 172;
        fuzzy();
        check_rule();
        hasil = defuzzy();
        hasil=100;

        if (hasil<=70) {PORTD.2=1;PORTD.3=0;PORTD.4=0;};          //35-->0.68V
LB
        else if ((hasil>=69)&&(hasil<=105))
{PORTD.2=0;PORTD.3=1;PORTD.4=0;};          //70-->1.37V      B
        else if ((hasil>=104)&&(hasil<=140))
{PORTD.2=1;PORTD.3=1;PORTD.4=0;};          //105-->2.06V      G
        else if ((hasil>=139)&&(hasil<=175))
{PORTD.2=0;PORTD.3=0;PORTD.4=1;};          //140-->2.75V      R
        else if ((hasil>=174)&&(hasil<=210))
{PORTD.2=1;PORTD.3=0;PORTD.4=1;};          //175-->3.43V          Y
        else if (hasil>=209) {PORTD.2=0;PORTD.3=1;PORTD.4=1;};          //210--
>4.12V                    W
        else {PORTD.2=1;PORTD.3=1;PORTD.4=1;};      //255-->5V

        printf("%d %d %d\r",data_RED,data_GREEN,data_BLUE);
        }
}

}
         }
```

LAMPIRAN III

DATA SHEET