RANCANG BANGUN
MOBILE FIRE FIGHTING ROBOT
MENGGUNAKAN JARINGAN SYARAF TIRUAN

SKRIPSI

Oleh

MUHAMMAD SYAFIUDDIN
04 03 03 710 6

DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNIK UNIVERSITAS INDONESIA
GANJIL 2007/2008

RANCANG BANGUN
MOBILE FIRE FIGHTING ROBOT
MENGGUNAKAN JARINGAN SYARAF TIRUAN

SKRIPSI

Oleh

MUHAMMAD SYAFIUDDIN
04 03 03 710 6

SKRIPSI INI DIAJUKAN UNTUK MELENGKAPI SEBAGIAN
PERSYARATAN MENJADI SARJANA TEKNIK

DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNIK UNIVERSITAS INDONESIA
GANJIL 2007/2008

PERNYATAAN KEASLIAN SKRIPSI

Saya menyatakan dengan sesungguhnya bahwa skripsi dengan judul:

RANCANG BANGUN *MOBILE FIRE FIGHTING ROBOT*
MENGGUNAKAN JARINGAN SYARAF TIRUAN

yang dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada Program Studi Teknik Elektro, Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia, sejauh yang saya ketahui bukan merupakan tiruan atau duplikasi dari skripsi yang sudah dipublikasikan dan atau pernah dipakai untuk mendapatkan gelar kesarjanaan di lingkungan Universitas Indonesia maupun di Perguruan Tinggi atau Instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

Depok, 4 Januari 2008

Muhammad Syafiuddin
NPM. 0403037106

PENGESAHAN

Skripsi dengan judul:

RANCANG BANGUN *MOBILE FIRE FIGHTING ROBOT*
MENGGUNAKAN JARINGAN SYARAF TIRUAN

dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada Program Studi Teknik Elektro, Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia. Skripsi ini telah diajukan pada sidang ujian skripsi pada tanggal 4 Januari 2008 dan dinyatakan memenuhi syarat/sah sebagai skripsi pada Program Studi Teknik Elektro, Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia.

Depok, 4 Januari 2008
Dosen Pembimbing

Dr.Ir. Feri Yusivar, M.Eng
NIP. 132 090 912

UCAPAN TERIMA KASIH

Puji syukur kepada Allah SWT atas segala rahmat dan karunia-Nya sehingga skripsi ini dapat diselesaikan. Shalawat dan salam semoga senantiasa tercurahkan kepada Nabi Muhammad SAW. Penulis ingin mengucapkan terima kasih kepada Bapak:

Dr. Ir. Feri Yusivar, M.Eng

selaku dosen pembimbing yang telah banyak meluangkan waktunya untuk memberikan saran, bimbingan, dan pengarahan sehingga skripsi ini dapat diselesaikan dengan baik.

Muhammad Syafiuddin
NPM 04 03 03 710 6
Departemen Teknik Elektro

Dosen Pembimbing
Dr.Ir. Feri Yusivar, M.Eng

RANCANG BANGUN *MOBILE FIRE FIGHTING ROBOT* MENGGUNAKAN JARINGAN SYARAF TIRUAN

ABSTRAK

Skripsi ini membahas perancangan dan pembuatan perangkat keras dan perangkat lunak robot pemadam api yang mampu bergerak dengan baik pada lingkungan sekitar tanpa pengendalian manusia. Robot mempunyai tugas untuk mencari api di dalam ruangan dan mematikannya. Robot bergerak di dalam labirin sebagai lingkungannya.

Agar dapat bergerak dengan baik dalam lingkungannya, jaringan syaraf tiruan diterapkan sebagai pengendali pergerakan robot pemadam api. Jaringan syaraf tiruan yang digunakan adalah jaringan syaraf tiruan *back propagation*. Robot ini menggunakan mikrokontroller AVR dari Atmel Corporation yang berjenis ATMEGA32.

Analisa dilakukan dengan mengamati pergerakan robot didalam labirin. Robot dapat bergerak dengan baik didalam labirin tanpa mengalami tabrakan dengan dinding labirin. Hal ini menunjukkan bahwa jaringan syaraf tiruan dapat digunakan sebagai salah satu pengendali pergerakan robot.

Kata Kunci : Robot, Jaringan Syaraf Tiruan, Mikrokontroller

Muhammad Syafiuddin
NPM 04 03 03 710 6
Electrical Engineering Department

Counselor
Dr.Ir. Feri Yusivar, M.Eng

PRACTICAL APPLICATION OF MOBILE FIRE FIGHTING ROBOT USING ARTIFICIAL NEURAL NETWORK

ABSTRACT

This research was conducted to practical application of hardware and software for mobile fire fighting robot. The main task of the robot is finding out a flame and extinguishing it. The flame is placed somewhere in rooms. To get into the destination room, it has to avoid obstacles along the path of labirin.

Artificial neural network is used to control the movement of robot. This research using artificial neural network back propagation. AVR microcontroller from Atmel Corporation (ATMEGA 32) is used for movement process in a labirin.

Analyze is done on movement of robot in a labirin. Robot has ability running in a labirin and without crash the wall of labirin. This result of this research is an artificial neural network algorithm which can be used as artificial intelligence of the robot.

Keywords : Robotics, Artificial Neural Network, Microcontroller,

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 LATAR BELAKANG

Perkembangan dunia elektronika yang semakin pesat dengan perkembangan mikroprosesor dan mikrokontroler yang semakin cepat dan pintar dalam pemrosesan data sangat mendukung perkembangan dunia robotika. Hal ini mendorong robot supaya dapat menjadi representasi manusia dan mempunyai sifat serta dapat melaksanakan fungsi-fungsi sebagaimana manusia. Dengan kata lain, peran robot otonom menjadi lebih signifikan sebagai pembantu manusia dalam menjalankan aktifitas kehidupannya.

Robot saat ini yang banyak diteliti di seluruh dunia adalah robot koloni (*colony robot*). Robot ini berfungsi antara lain sebagai robot untuk menelusuri daerah-daerah berbahaya dimana manusia tidak bisa memasukinya (*dangerous-zone robot*) seperti robot pemadam api, robot penyapu ranjau, dan robot-robot pionir dalam menjelajahi daerah yang belum terjamah manusia seperti Robot penjelajah planet Mars. Dalam perencanaan dan perancangan robot tersebut yang masih menjadi masalah adalah sistem pergerakan robot, yaitu: bagaimana robot pemadam api dapat bergerak dengan baik di dalam suatu lingkungan tanpa menabrak halangan.

Pada skripsi ini *mobile fire fighting* robot menggunakan jaringan syaraf tiruan untuk mengatasi permasalahan pergerakan robot dalam suatu lingkungan. Lingkungan yang digunakan berupa labirin yang terdiri dari beberapa ruangan. Algoritma ini diharapkan dapat membuat robot berjalan lurus dengan baik dalam lorong atau jalur tertentu tanpa menabrak dinding labirin.

1.2 TUJUAN PENELITIAN

Skripsi ini bertujuan untuk mengaplikasikan jaringan syaraf tiruan sebagai salah satu algoritma dalam pergerakan robot pemadam api. Dengan algoritma ini, robot dapat bergerak dalam labirin tanpa menabrak dinding labirin.

1.3 BATASAN MASALAH

Skripsi ini membahas rancang bangun robot otonom pemadam api menggunakan jaringan syaraf tiruan *back propagation* sebagai pengatur gerakan robot di dalam *labirin*. Proses pelatihan jaringan dilakukan di Matlab 7.0 sedangkan proses pengujian (pergerakan robot) dilakukan dalam mikrokontroller AVR dari Atmel Corporation yang berjenis ATMEGA32.

1.4 METODOLOGI PENELITIAN

Metodologi penelitian ini adalah sebagai berikut:

1. Perancangan perangkat keras dan perangkat lunak
2. Pembuatan perangkat keras dan perangkat lunak
3. Analisis dan ujicoba keseluruhan fungsi robot

1.5 SISTEMATIKA PENULISAN

Skripsi ini ditulis dalam 5 bab sesuai panduan pembuatan skripsi. Bab pertama berisi mengenai pendahuluan yang terdiri dari latar belakang, tujuan penelitian, batasan masalah, metodologi penelitian, dan sistematika penulisan. Bab kedua berisi dasar teori yang terdiri dari teori-teori yang akan berhubungan dengan penelitian yang akan dilakukan, yaitu: dasar teori jaringan syaraf tiruan *backpropagation* yang digunakan dalam robot ini sebagai pengendali pergerakan (*movement*). Bab ketiga membahas mengenai perangkat keras dan perangkat lunak yang terdiri dari sensor-sensor yang digunakan oleh robot ini untuk mengetahui keadaan lingkungan sehingga dapat mengatur pergerakan dan rangkaian elektronika sebagai pengolah input dari sensor serta membahas aktuator dan pengendali pergerakan robot yang menggunakan mikrokontroler dan perancangannya dalam mengendalikan robot ini. Selain itu juga membahas

mengenai perancangan perangkat lunak untuk akuisisi data dari sensor-sensor, flowchart, algoritma dan langkah-langkah untuk pergerakan robot didalam labirin dengan jaringan syaraf tiruan *backpropagation*. Bab keempat berisi hasil ujicoba dan analisis pembentukan jaringan syaraf tiruan, serta ujicoba perilaku robot sesuai dengan fungsi robot, serta ujicoba keseluruhan fungsi robot secara penuh dan analisisnya dalam labirin. Bab kelima berisi kesimpulan dari keseluruhan skripsi.

BAB II
LANDASAN TEORI

2.1 JARINGAN SYARAF TIRUAN

Jaringan syaraf tiruan didefinisikan sebagai suatu sistem pemrosesan informasi yang mempunyai karakteristik menyerupai jaringan syaraf manusia. Jaringan syaraf tiruan tercipta sebagai suatu generalisasi model matematis dari pemahaman manusia yang didasarkan atas asumsi sebagai berikut [1]:

1. Pemrosesan informasi terjadi pada elemen sederhana yang disebut neuron.
2. Sinyal mengalir di antara sel syaraf / neuron melalui suatu sambungan penghubung.
3. Setiap sambungan penghubung memiliki bobot yang bersesuaian. Bobot ini akan digunakan untuk menggandakan / mengalikan isyarat yang dikirim melaluinya.
4. Setiap sel syaraf akan menerapkan fungsi aktivasi terhadap isyarat hasil penjumlahan berbobot yang masuk kepadanya untuk menentukan isyarat keluarannya.

Model syaraf dalam jaringan syaraf tiruan ditunjukkan dengan kemampuan dalam emulasi, analisis, prediksi, dan asosiasi. Kemampuan yang dimliki oleh jaringan syaraf tiruan dapat digunakan untuk belajar dan menghasilkan aturan atau operasi dari beberapa contoh atau *input* yang dimasukkan dan membuat prediksi tentang kemungkinan *output* yang akan muncul atau menyimpan karakteristik dari *input* yang disimpan kepadanya.

Jaringan syaraf tiruan merupakan suatu bentuk arsitektur terdistribusi paralel dengan sejumlah besar *node* dan hubungan antar *node* tersebut. Tiap titik hubungan dari satu *node* ke *node* yang lain mempunyai harga yang diasosiasikan dengan bobot. Setiap *node* memiliki suatu nilai yang diasosiasikan sebagai nilai aktivasi *node*.

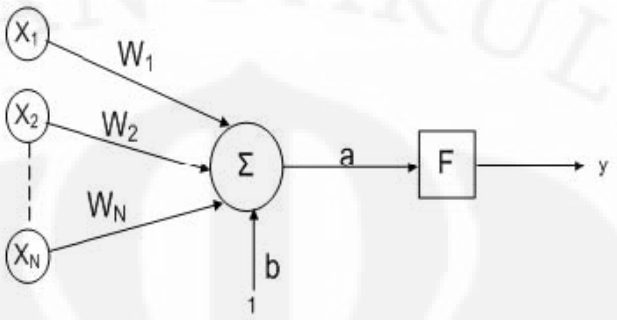
Karakteristik jaringan syaraf tiruan ditentukan oleh [2]:

1. Pola hubungan antar – neuron (disebut dengan arsitektur jaringan).
2. Metode penentuan bobot – bobot sambungan (disebut dengan pelatihan atau

proses belajar jaringan).

3. Fungsi aktivasi.

Neuron dimodelkan dari penyederhanaan sel syaraf manusia yang sebenarnya. Gambar 2.1 merupakan struktur *unit* jaringan syaraf tiruan sederhana.



Gambar 2.1 Struktur jaringan syaraf tiruan sederhana

Pada Gambar 2.1 tersebut sebuah neuron akan mengolah N *input*($x_1, x_2, .., x_N$) yang masing-masing memiliki bobot $w_1, w_2, ..., w_N$ dan bobot bias b, dengan rumus [3]:

$$a = (\sum_{i=1}^{N} x_i w_i) + b \quad \text{.....(2.1)}$$

Kemudian fungsi aktivasi F akan mengaktivasi *a* menjadi *output* jaringan. Jaringan syaraf tiruan dirancang dengan menggunakan suatu aturan yang bersifat menyeluruh di mana seluruh model jaringan memiliki konsep dasar yang sama. Arsitektur sebuah jaringan akan menentukan keberhasilan target yang akan dicapai karena tidak semua permasalahan dapat diselesaikan dengan arsitektur yang sama. Ada banyak model jaringan yang dapat digunakan sebagai jaringan syaraf tiruan. Salah satu jenis algoritma yang sering digunakan adalah jaringan syaraf tiruan *back propagation*.

Fungsi aktivasi merupakan bagian penting dalam tahapan perhitungan keluaran dari suatu jaringan syaraf tiruan. Beberapa fungsi aktivasi yang digunakan dalam jaringan syaraf tiruan adalah:

1. Fungsi undak biner

$$y = f(x) = \left\{ \frac{1 untuk X \geq 0}{0 untuk X \leq 0} \right\} \quad \text{.....(2.2)}$$

2. Fungsi *Sigmoid Biner*

$$y = f(x) = \frac{1}{1+\exp(-\sigma x)}$$

$$y = f'(x) = \sigma f(x)\left[1 - f(x)\right] \quad \textbf{(2.3)}$$

dimana : σ = konstanta

3. Fungsi *Sigmoid bipolar*

$$y = g(x) = 2f(x) - 1 = \frac{2}{1+\exp(-\sigma x)} - 1$$

$$= \frac{1-\exp(-\sigma x)}{1+\exp(-\sigma x)}$$

$$g'(x) = \frac{\sigma}{2}\left[1 + g(x)\right]\left[1 - g(x)\right] \quad \textbf{(2.4)}$$

dimana : σ = konstanta

4. *Tansig*

$$y = f(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$$

$$f'(x) = \left[1 + f(x)\right]\left[1 - f(x)\right] \quad \textbf{(2.5)}$$

5. *Purelin*

$$y = f(x) = x$$

$$f'(x) = 1 \quad \textbf{(2.6)}$$

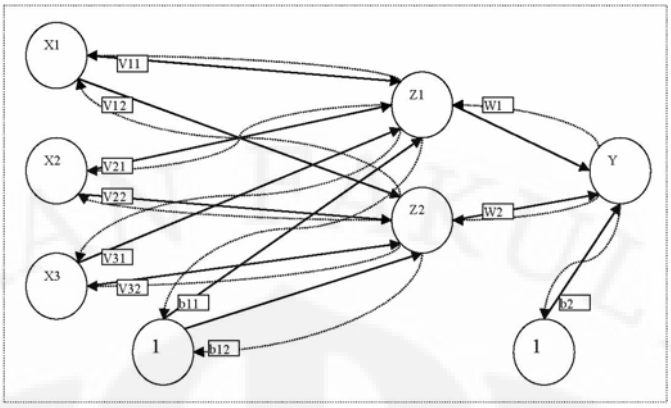
Sebagian besar jaringan syaraf tiruan melakukan penyesuaian bobot-bobotnya selama menjalani prosedur latihan. Jaringan syaraf tiruan *back propagation* merupakan salah satu jenis algoritma jaringan syaraf tiruan yang banyak digunakan, penjelasan mengenai jaringan syaraf tiruan *back propagation* lebih lengkapnya akan dijelaskan pada sub bab berikutnya.

2.2 JARINGAN SYARAF TIRUAN *BACK PROPAGATION*

Jaringan syaraf tiruan *back propagation* merupakan salah satu jenis jaringan syaraf *multilayer*. Jaringan ini dapat menyelesaikan masalah - masalah yang rumit dibandingkan jaringan syaraf tiruan *single layer*. Pada jaringan syaraf tiruan *back propagation* diberikan sepasang pola yang terdiri atas pola masukan dan pola keluaran yang diinginkan. Ketika suatu pola diberikan kepada jaringan, bobot – bobot diubah untuk memperkecil perbedaan pola keluaran dan pola yang diinginkan. Pelatihan dilakukan berulang – ulang sehingga semua pola yang dikeluarkan jaringan dapat memenuhi pola yang diinginkan.

Algoritma *backpropagation* menggunakan *error output* (perbedaan antara pola keluaran dengan pola yang diinginkan) untuk mengubah nilai bobot - bobotnya dalam arah mundur (*backward*). Untuk mendapatkan *error* ini, arah perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu. Pada saat perambatan maju, *neuron-neuron* diaktifkan dengan menggunakan fungsi aktivasi yang dapat dideferensiasikan seperti *sigmoid*, *Tansig*, atau *purelin*.

Arsitektur jaringan *backpropagation* dapat dilihat pada Gambar 2.2. Pada gambar tersebut, jaringan terdiri atas 3 *unit* (*neuron*) pada lapisan *input* yaitu X1, X2, dan X3; 1 lapisan tersembunyi dengan 2 *unit* (*neuron*), yaitu Z1 dan Z2; serta 1 *unit* (*neuron*) pada lapisan *output*, yaitu y. Bobot yang menghubungkan X1, X2, dan X3 dengan *neuron* pertama pada lapisan tersembunyi, adalah V11, V21, dan V31 (Vij: bobot yang menghubungkan *neuron input* ke-i ke *neuron* ke-j pada lapisan yang menghubungkan *neuron input* ke-i ke *neuron* ke-j pada lapisan tersembunyi), b11 dan b12 adalah bobot bias yang menuju ke *neuron* pertama dan kedua pada lapisan tersembunyi. Bobot yang menghubungkan Z1 dan Z2 dengan *neuron* pada lapisan *output*, adalah W1 dan W2. Bobot bias b2 menghubungkan lapisan tersembunyi dengan lapisan *output*. Fungsi aktivasi antara lapisan *input* dengan lapisan tersembunyi dan antara lapisan tersembunyi dengan lapisan *output* tidak diperlihatkan pada Gambar 2.2.



Gambar 2.2 Arsitektur jaringan syaraf tiruan *backpropagation*

Berikut adalah langkah-langkah algoritma pelatihan jaringan syaraf tiruan *backpropagation* [4]:

1. Menentukan pola masukan dan pola keluaran yang diinginkan.
2. Inisialisasi bobot (ambil bobot awal dengan nilai acak yang cukup kecil).
3. Menetapkan maksimum epoh, target *error*, dan *learning rate*(α). Banyaknya epoh menetukan banyaknya iterasi yang dilakukan. Target *error* menentukan keberhasilan proses pelatihan dimana pelatihan dikatakan berhasil jika *error output* yang terjadi sama dengan atau lebih kecil dari target *error* yang ditetapkan. *Learning rate* digunakan untuk mengatur perbaikan bobot pada setiap langkah pelatihan.
4. Inisialisasi: Epoh=0, MSE=1.
   Dimana: MSE(*Mean Square Error*) = *error* rata-rata kuadrat
5. Melakukan langkah – langkah berikut selama epoh < maksimum epoh dan MSE > target *error*:
   1) Epoh = Epoh + 1
   2) Untuk setiap pasangan elemen jaringan syaraf tiruan akan dilakukan proses pembelajaran dalam dua tahap (feedforward dan backpropagation), yaitu sebagai berikut:

   - ***Feedforward* :**

     a) Tiap-tiap *unit input* (X_i,i=1,2,3..n) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua *unit* pada lapisan yang ada di atasnya (lapisan tersembunyi),

b) Tiap-tiap *unit* pada suatu lapisan tersembunyi (Z_j, j=1,2,3,…,p) menjumlahkan sinyal – sinyal *input* terbobot, menggunakan persamaan sebagai berikut:

$$z\_in_j = b1_j + \sum_{i=1}^{n} x_i v_{ij} \quad \textbf{.....(2.7)}$$

dimana:

z_in_j = sinyal pada lapisan tersembunyi *node* ke-j

$b1_j$ = bias pada lapisan tersembunyi *node* ke-j

x_i = sinyal masukan

v_{ij} = bobot antara lapisan tersembunyi *node* ke-j dengan lapisan *input node* ke-i

Selanjutnya digunakan fungsi aktivasi untuk menghitung sinyal *output*nya, dengan menggunakan persamaan sebagai berikut:

$$z_j = f\left(z\_in_j\right) \quad \textbf{.....(2.8)}$$

dimana:

z_j = sinyal keluaran pada lapisan tersembunyi ke-j setelah aktivasi

dan dikirimkan sinyal tersebut ke semua *unit* di lapisan atasnya(*unit-unit output*).

c) Tiap – tiap *unit output* (Y_k, k=1,2,3,…,m) menjumlahkan sinyal – sinyal *input* terbobot, dengan menggunakan persamaan sebagai berikut:

$$y\_in_k = b2_k + \sum_{i=1}^{p} z_j w_{jk} \quad \textbf{.....(2.9)}$$

dimana:

y_in_k = sinyal pada lapisan keluaran *node* ke-k

$b2_k$ = bias pada lapisan keluaran *node* ke-k

w_{jk} = bobot antara lapisan tersembunyi *node* ke-j dengan lapisan *output node* ke-k

Selanjutnya digunakan fungsi aktivasi untuk menghitung sinyal *output*nya, dengan menggunakan persamaan sebagai berikut:

$$y_k = f(y\_in_k) \quad \textbf{(2.10)}$$

dimana:

y_k = sinyal pada lapisan keluaran *node* ke-k setelah aktivasi

- ***Backpropagation***

a) Tiap – tiap *unit output* (Y_k, k=1,2,3,...,m) menerima target pola yang berhubungan dengan pola *input* pembelajaran, selanjutnya menghitung *error* informasinya, dengan menggunakan persamaan berikut:

$$\delta_k = (t_k - y_k) f'(y\_in_k) \quad \textbf{(2.11)}$$

dimana:

t_k = target pada keluaran *node* ke-k

δ_k = *error* informasi lapisan keluaran k

Kemudian dihitung nilai koreksi bobot antara lapisan tersembunyi ke-j dan lapisan keluaran ke-k (koreksi bobot nantinya akan digunakan untuk memperbaiki nilai bobot w_{jk}) , dengan menggunakan persamaan berikut:

$$\Delta w_{jk} = \alpha \delta_k z_j \quad \textbf{(2.12)}$$

dimana:

α = konstanta belajar

Δw_{jk} = koreksi bobot antara lapisan tersembunyi ke-j dan lapisan keluaran ke-k

Disamping itu dihitung juga nilai koreksi bias pada lapisan keluaran ke-k (yang nantinya akan digunakan untuk memperbaiki nilai bias $b2_k$), dengan menggunakan persamaan berikut:

$$\Delta b2_k = \alpha \delta_k \quad \textbf{(2.13)}$$

dimana:

$\Delta b2_k$ = koreksi bias pada lapisan keluaran ke-k

b) Tiap – tiap *unit* tersembunyi (Z_j, j=1,2,3,...,p) menjumlahkan hasil kali delta inputnya (δ_k) dengan bobotnya masing-masing, dengan menggunakan persamaan berikut:

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \quad \textbf{(2.14)}$$

dimana:

δ_in_j = *error* informasi *input* lapisan tersembunyi j

Selanjutnya nilai δ_in_j dikalikan dengan turunan dari fungsi aktivasi pada lapisan tersembunyi untuk menghitung informasi *error*, dengan menggunakan persamaan berikut:

$$\delta_j = \delta\_in_j f'\left(z\_in_j\right) \quad \textbf{(2.15)}$$

dimana:

δ_j = *error* informasi lapisan tersembunyi j

Kemudian dihitung nilai koreksi bobot antara lapisan *input* ke-i dan lapisan tersembunyi ke-j (koreksi bobot nantinya akan digunakan untuk memperbaiki nilai bobot v_{ij}), dengan menggunakan persamaan berikut:

$$\Delta v_{ij} = \alpha \delta_j x_i \quad \textbf{(2.16)}$$

dimana:

Δv_{ij} = koreksi bobot antara lapisan tersembunyi ke-j dan lapisan masukan ke-i

Disamping itu dihitung juga nilai koreksi bias pada lapisan tersembunyi ke-j (yang nantinya akan digunakan untuk memperbaiki nilai $b1_j$), dengan menggunakan persamaan berikut:

$$\Delta b1_j = \alpha \delta_j \quad \text{.....(2.17)}$$

dimana:

$\Delta b1_j$ = koreksi bias pada lapisan tersembunyi ke-j

c) Selanjutnya tiap-tiap *unit output* (Y_k, k=1,2,3,…,m) memperbaiki bias dan bobotnya (j=0,1,2,…,p), dengan menggunakan persamaan berikut:

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk} \quad \text{.....(2.18)}$$

$$b2_k(baru) = b2_k(lama) + \Delta b2k \quad \text{.....(2.19)}$$

Begitu juga tiap-tiap *unit* tersembunyi (Z_j,j=1,2,3,…,p) memperbaiki bias dan bobotnya (i=0,1,2,…,n), dengan menggunakan persamaan berikut:

$$v_{ij}(baru) = vi_j(lama) + \Delta v_{ij} \quad \text{.....(2.20)}$$

$$b1_j(baru) = b1_j(lama) + \Delta b1_j \quad \text{.....(2.21)}$$

3) Kemudian dihitung nilai MSE (*Mean Square Error*, dengan menggunakan persamaan berikut:

$$MSE = \sum_{l=1}^{n} \sum_{k=1}^{m} \frac{(t_{lk} - y_{lk})^2}{n * m} \quad \text{.....(2.22)}$$

dimana:

t_{lk} = keluaran yang dikehendaki untuk sampel data ke-l dan unit keluaran ke-k

y_{lk} = keluaran JST untuk sampel ke-l dan unit keluaran ke-k

m = jumlah unit keluaran

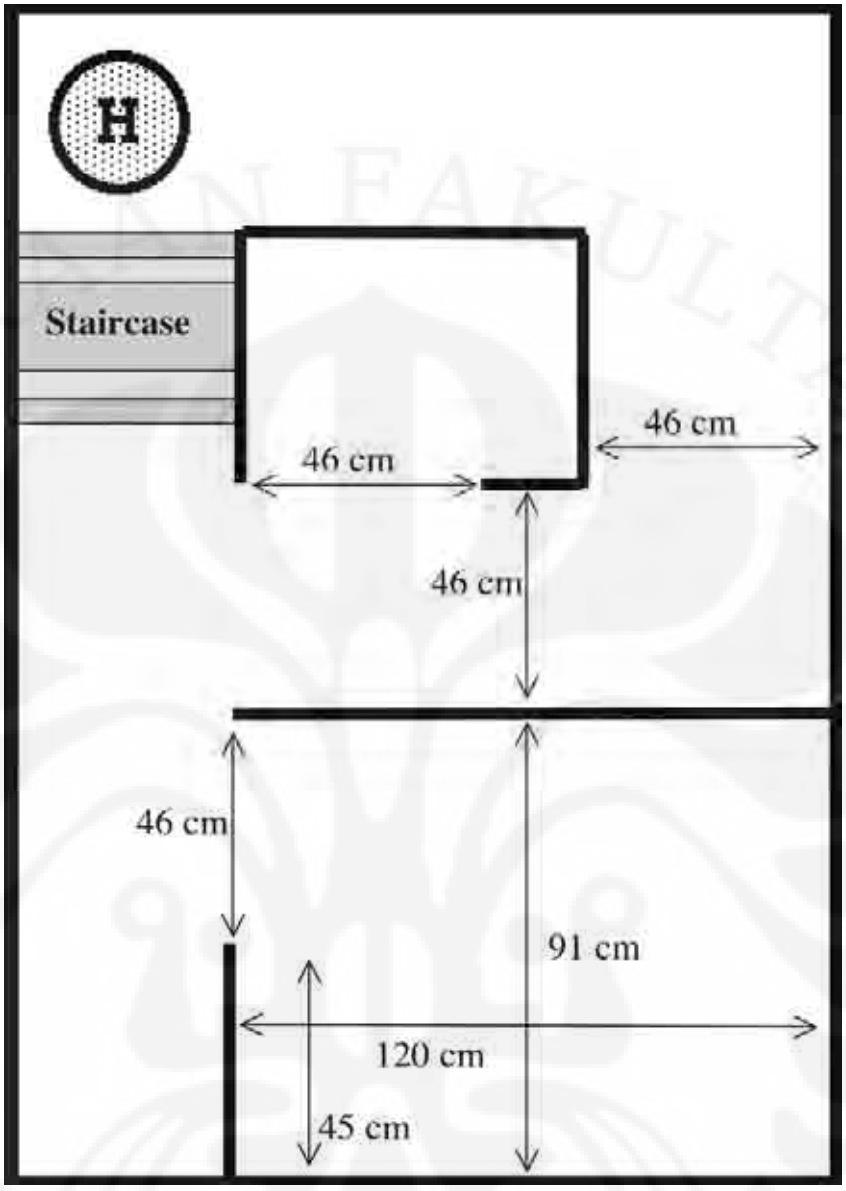
n = jumlah sampel data

BAB III
PERANGKAT KERAS dan PERANGKAT LUNAK SISTEM

3.1 MOBILE FIRE FIGHTING ROBOT

Robot yang dirancang dan dibuat pada skripsi ini, disesuaikan dengan tema pertandingan Kontes Robot Cerdas Indonesia dengan tema "Robot Pemadam Api" yang diadaptasi dari "Fire-Fighting Robot" dari Trinity College, Connecticut, Amerika Serikat. Robot pemadam api ini merupakan robot dengan orientasi tujuan (*Goal-Oriented Robot*). Robot pemadam api hampir sama dengan robot koloni (*Colony Robot*) yang sedang dikembangkan oleh laboratorium-laboratorium di dunia yang bertujuan untuk bekerja di daerah berbahaya untuk dimasuki manusia.

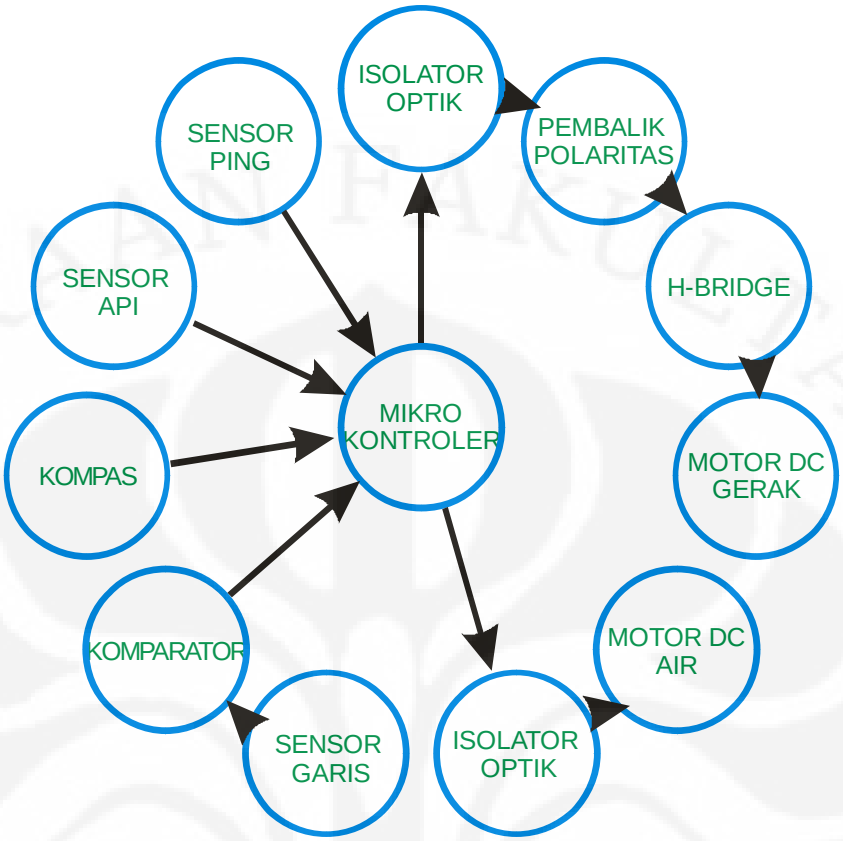
Prinsip pertandingan kontes robot pemadam api ini adalah robot harus dapat menjelajahi ruangan yang menyerupai rumah dengan ruangan-ruangannya (*house-like labirin*) tanpa menyentuh dinding, mencari api lalu mematikan api tersebut dan kemudian kembali lagi ke tempat awal robot tersebut mulai bergerak, seperti yang terlihat pada Gambar 3.1. Gambar tersebut merupakan modifikasi dari lapangan yang digunakan dalam Kontes Robot Cerdas Indonesia. Dimana pada kontes robot tersebut ada empat ruangan yang harus dijelajahi oleh robot sedangkan pada skripsi ini robot hanya bergerak menjelajahi dua buah ruangan. Pada setiap pintu masuk ruangan, terdapat pita berwarna putih dengan lebar 3 cm dan panjang sepanjang pintu ruangan tersebut. Titik H merupakan tempat posisi start robot dan stair case merupakan undakan anak tangga.



Gambar 3.1 Denah labirin yang digunakan robot pemadam api

3.2 BLOK DIAGRAM SISTEM ELEKTRIK ROBOT

Keseluruhan perangkat yang mengatur kerja robot dapat dijelaskan melalui blok diagram yang melambangkan bagian-bagian robot yang menentukan kinerja robot secara keseluruhan. Blok diagram keseluruhan bagian robot dapat dilihat pada Gambar 3.2.



Gambar 3.2 Blok diagram keseluruhan bagian robot

Suatu sistem yang otomatis atau otonomi selalu membutuhkan sensor-sensor untuk mengenali keadaan sekitarnya. Semakin banyak sensor yang digunakan maka robot semakin menyerupai manusia yang otonomi tetapi semakin banyak juga hasil dari sensor yang diproses oleh pengendali utama. Sensor-sensor yang dibutuhkan untuk *mobile fire fighting robot* adalah sebagai berikut:

1. Sensor Garis, sensor yang digunakan untuk mendeteksi apakah robot sudah memasuki ruangan.
2. Sensor Jarak, sensor yang digunakan adalah sensor PING, sensor ini menggunakan media pengukuran berupa bunyi ultrasonik. Sensor ini digunakan untuk mengukur posisi robot terhadap dinding labirin.
3. Sensor Api, sensor yang digunakan adalah UVTron. Sensor ini mengukur intensitas ultraviolet yang dipancarkan oleh api.
4. Sensor Kompas, sensor yang digunakan adalah Devantech Compass. Sensor ini digunakan untuk proses gerakan berbelok pada *mobile fire fighting robot* ini.

Selain sensor, robot juga membutuhkan rangkaian elektronika sebagai pengolah masukan dari sensor-sensor yang digunakan dan aktuator sebagai pelaksana keluaran dari hasil pengolahan tersebut. Rangkaian elektronika yang digunakan pada mobile fire fighting robot adalah sebagai berikut:

1. Rangkaian Komparator
2. Rangkaian Isolator Optik
3. Rangkaian Pembalik Polaritas
4. Rangkaian *H-Bridge*

Sedangkan aktuator yang digunakan pada *mobile fire fighting robot* ini adalah sebagai berikut:

1. Dua buah motor DC, merupakan motor pabrikan MAXON dengan tipe A MAX 32 15 watt. Motor ini sebagai motor penggerak roda.
2. Sebuah motor DC pemompa air, merupakan motor yang berfungsi sebagai penggerak wiper pada mobil.

Keseluruhan penjelasan mengenai seluruh komponen yang digunakan pada *mobile fire fighting robot* ini dapat dilihat secara lengkap pada bagian lampiran.

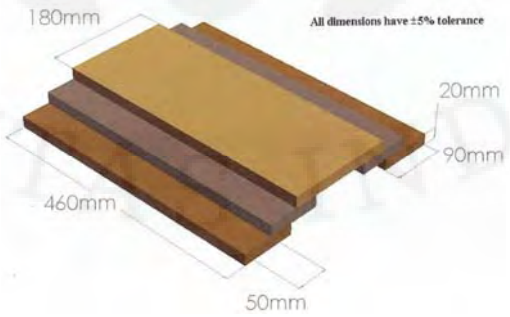
3.3 KONSTRUKSI ROBOT

Robot pada skripsi ini di buat menyerupai mobil dengan desain roda berupa roda tank, seperti yang terlihat pada Gambar 3.3. Robot berukuran 26 x 26 x 25 cm. Robot dibuat dari bahan acrylic bewarna bening seperti kaca dan dibuat 2 lantai dengan pembagian sebagai berikut:

1. Lantai pertama sebagai tempat *power supply* berupa baterai dengan tipe SLA, tempat aktuator berupa motor DC sebagai penggerak roda dan pemompa air, serta tempat tangki berisi air yang digunakan sebagai media untuk mematikan api.
2. Lantai Kedua sebagai tempat peletakan rangkaian elektronika dan sensor-sensor yang digunakan pada robot ini.

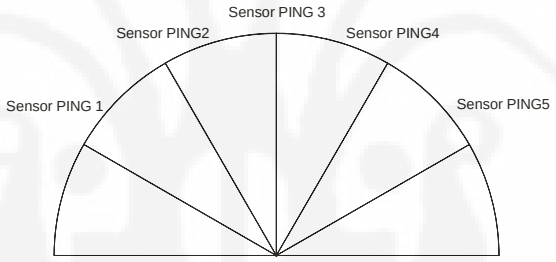
Gambar 3.3 Robot dari berbagai posisi

Roda pada robot ini dibuat mirip roda pada tank. Pemilihan model roda tank dimaksudkan agar robot dapat bergerak di berbagai lintasan, baik yang rata maupun yang bergelombang. Selain itu, dengan desain roda seperti roda tank, robot dapat melewati halangan berupa undakan anak tangga seperti Gambar 3.4.



Gambar 3.4 Undakan anak tangga

Peletakan sensor pada posisi yang tepat sangat membantu kerja dari robot tersebut. Pada robot ini sensor garis diletakkan pada bagian bawah lantai pertama dengan posisi 0.5 cm dari lantai labirin. Sensor Api diletakkan pada lantai kedua, agar dapat dengan leluasa mendeteksi adanya api. Sensor kompas diletakkan pada ketinggian 10 cm diatas lantai kedua, agar terbebas dari pengaruh medan magnet yang dihasilkan oleh benda-benda yang ada pada robot ini. Konfigurasi peletakan sensor jarak PING ditunjukkan oleh Gambar 3.5. Konfigurasi ini bertujuan agar robot dapat dengan cepat mendeteksi adanya persimpangan atau belokan pada lintasan yang akan dilaluinya. Sehingga robot dapat melakukan gerakan berbelok seperti kendaraan roda empat. Selain itu diharapkan robot dapat menghindari rintangan disekitarnya selama dalam perjalanan. Hal ini dikarenakan konfigurasi sensor seperti ini dapat meng-*cover* daerah sekeliling robot. Namun untuk gerakan berbelok seperti kendaraan roda empat pada umumnya belum diterapkan dalam skripsi ini.



Gambar 3.5 Konfigurasi letak sensor jarak PING

3.4 PERANGKAT LUNAK SISTEM

3.4.1 Identifikasi Masalah pada *Mobile Fire Fighting Robot*

Dari kondisi labirin yang akan dijadikan sebagai lingkungan pergerakan robot, terdapat beberapa kondisi atau kasus yang harus dihadapi oleh *mobile fire fighting robot*, yaitu:

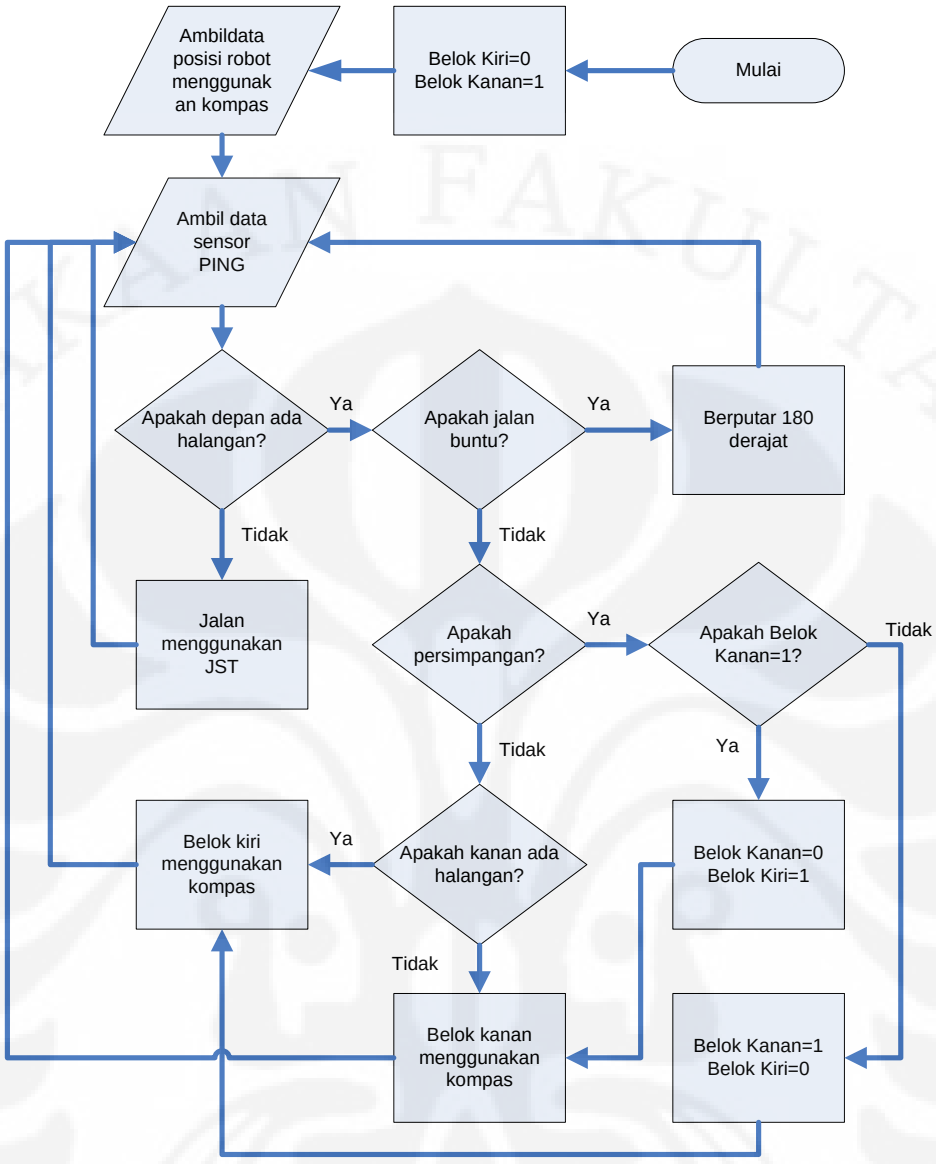
1. Robot harus dapat bergerak dengan baik tanpa menabrak dinding, harus dapat memilih kemana arah berbelok dari robot jika menemui persimpangan dan jalan buntu.
2. Robot harus dapat memasuki ruangan dan keluar dari ruangan.
3. Robot harus dapat mencari api didalam ruangan dan mematikannya.
4. Robot harus dapat kembali ke tempatnya semula (titik H).

3.4.2 Algoritma Penyelesaian Masalah Pada Mobile Fire Fighting Robot

Dari sub bab 3.4.1 dapat disimpulkan bahwa ada empat masalah yang harus ditangani oleh robot. Penyelesaian masalah tersebut dapat dilihat dari algoritma dibawah ini. Algoritma penyelesaian masalah yang pertama dapat dilihat pada Gambar 3.6.

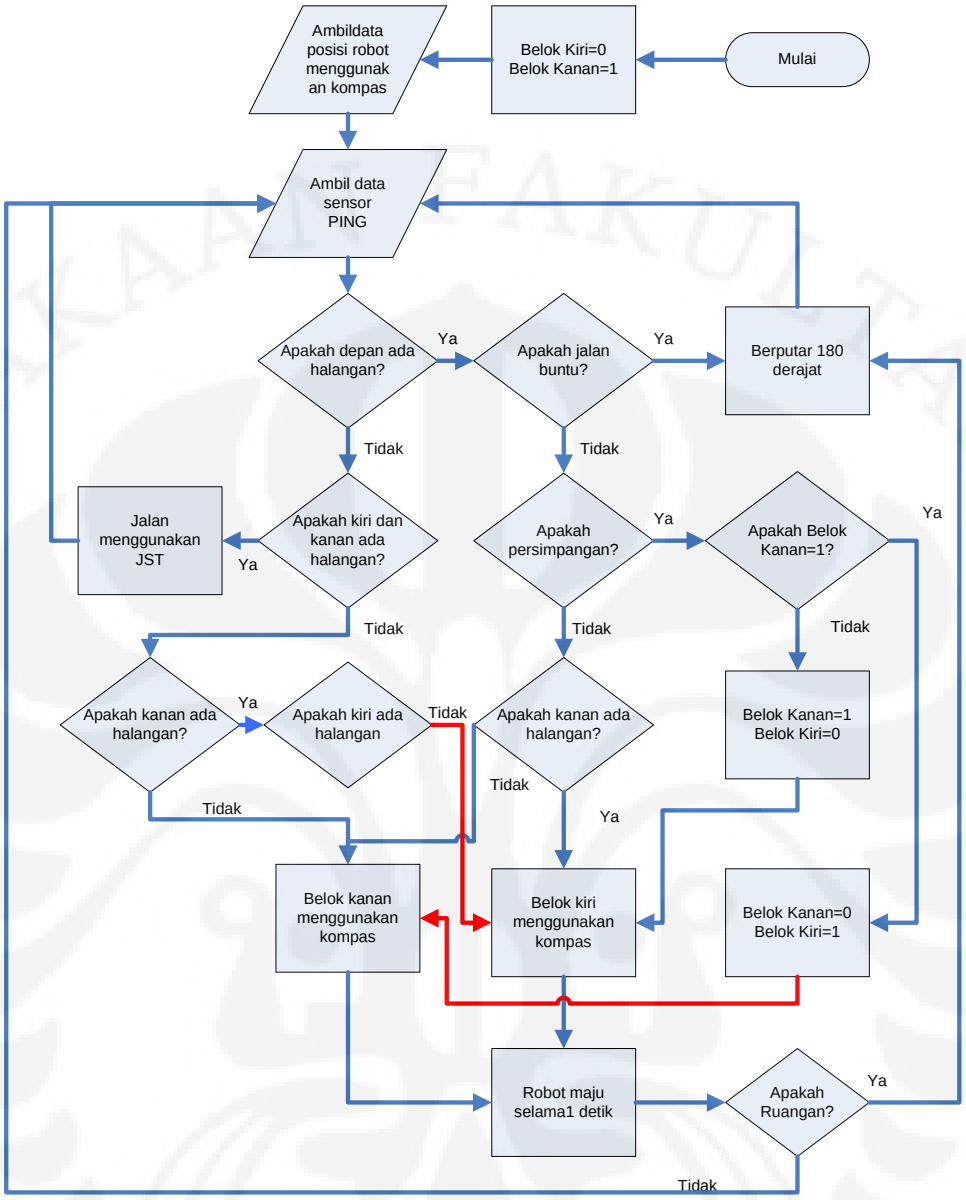
Setelah dimulai robot mendefinisikan suatu variabel belok kiri dan belok kanan dengan nilai awal masing-masing adalah 0 dan 1. Nilai ini akan menentukan arah belok dari robot tersebut jika menemukan persimpangan. Jika variabel belok kanan bernilai 1 maka robot akan berbelok kekanan dan akan mengubah nilai variabel belok kanan menjadi 0 dan belok kiri menjadi 1 sedangkan jika variabel belok kiri bernilai 1 maka robot akan berbelok kekiri dan akan mengubah nilai variabel belok kanan menjadi 1 dan belok kiri menjadi 0. Dengan algoritma ini di harapkan robot tidak bergerak dijalur yang sama berulang kali.

Selanjutnya, robot akan mengecek posisi robot terhadap medan magnet bumi. Posisi ini akan menjadi referensi saat melakukan gerak berbelok. Robot melakukan gerakan berbelok sebesar 90^o dari posisi referensi terakhir sebelum berbelok. Sesaat sebelum bergerak robot akan mengecek apakah didepan robot ada halangan dengan menggunakan sensor PING 3. Jika tidak ada, maka robot akan bergerak menggunakan algoritma jaringan syaraf tiruan menelusuri labirin. Jika ada halangan di depan robot, maka robot akan mengecek apakah halangan juga ada dikiri dan kanan robot. Jika halangan hanya ada dikiri, maka robot akan berbelok ke kanan. Jika halangan hanya ada di kanan, maka robot akan berbelok ke kiri. Namun jika ada halangan dikiri dan kanan berarti robot memasuki jalan buntu. Jika hal ini terjadi, robot akan berputar 180^o dari posisinya dan melakukan pengecekan sensor jarak lagi.



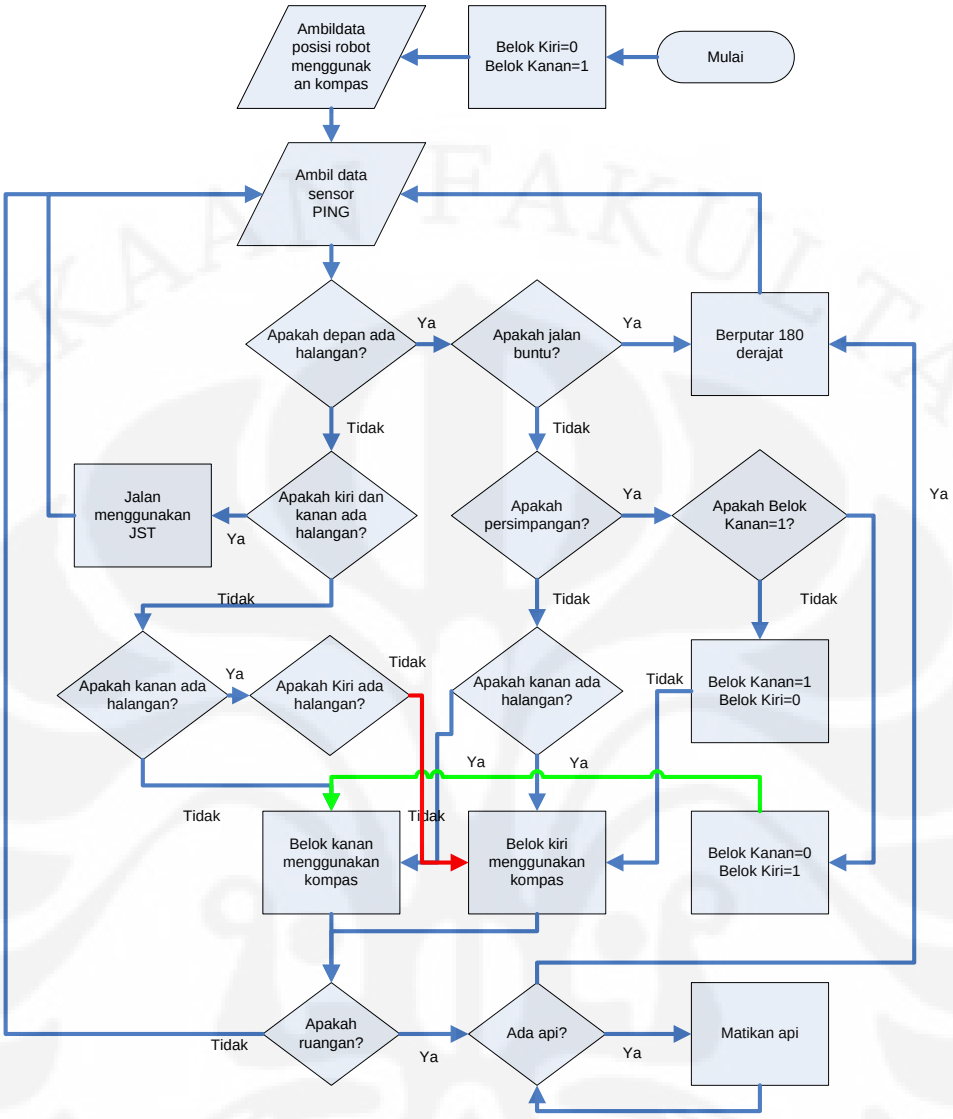
Gambar 3.6 Algoritma mengelilingi labirin

Algoritma penyelesaian masalah kedua dapat dilihat pada Gambar 3.7. Robot akan selalu berbelok setiap ada sisi yang kosong selama didepannya belum ada halangan. Setelah berbelok robot akan mengetahui apakah ia sedang memasuki ruangan atau tidak melalui sensor garis yang terpasang pada bagian bawah robot. Robot sedang memasuki ruangan jika sensor garis bernilai 0 (saat posisi sensor berada diatas garis putih. Jika hal ini terjadi robot akan bergerak memutar 180^{o} dan kembali mencari ruang yang lain.



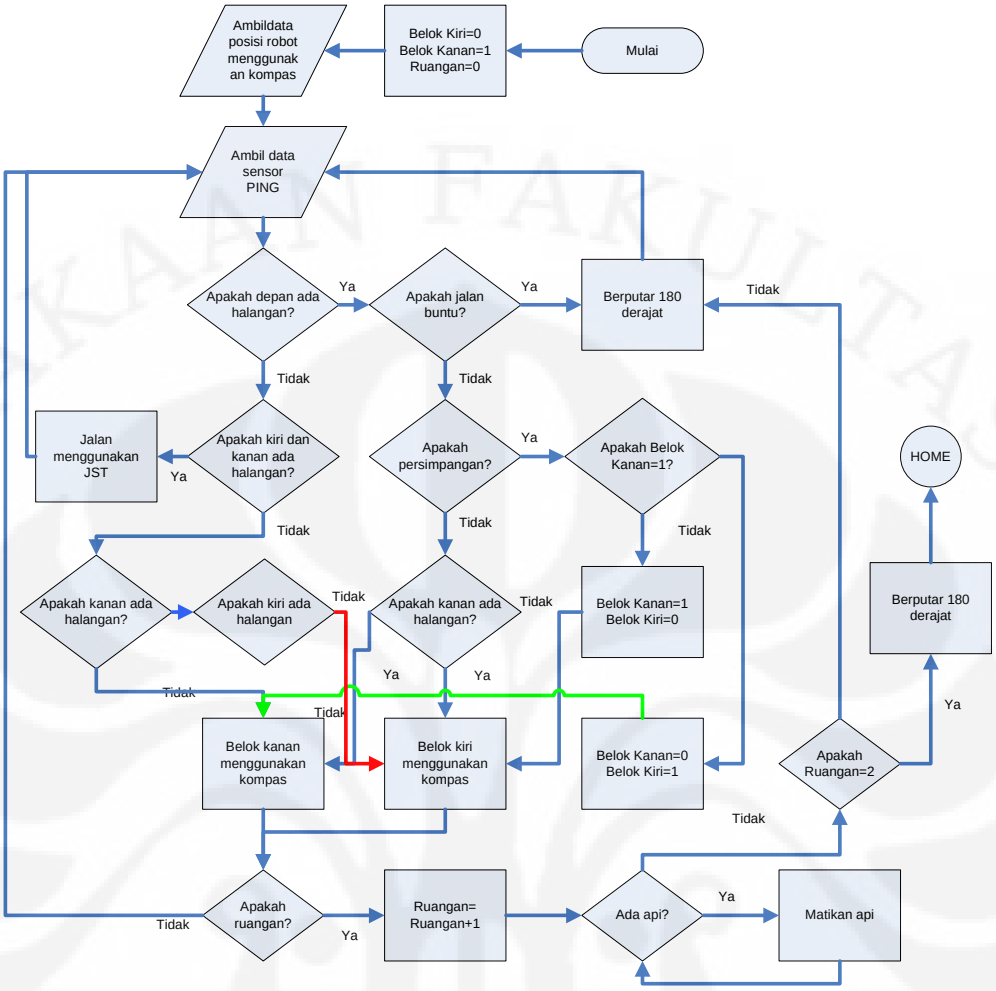
Gambar 3.7 Algoritma memasuki ruangan

Algoritma penyelesaian masalah ketiga dapat dilihat pada Gambar 3.8. Setelah robot mengetahui bahwa ia sedang memasuki ruangan, maka robot akan berhenti dan melakukan pemeriksaan terhadap ruangan tersebut dengan sensor api. Sensor api ini mempunyai jangkauan sampai 5m. Jika ada api maka robot akan mematikan api tersebut. Setelah api mati maka robot akan berputar 180^{o} untuk keluar dari ruangan. Begitu juga jika tidak ada api di dalam ruangan tersebut.



Gambar 3.8 Algoritma mencari dan mematikan api

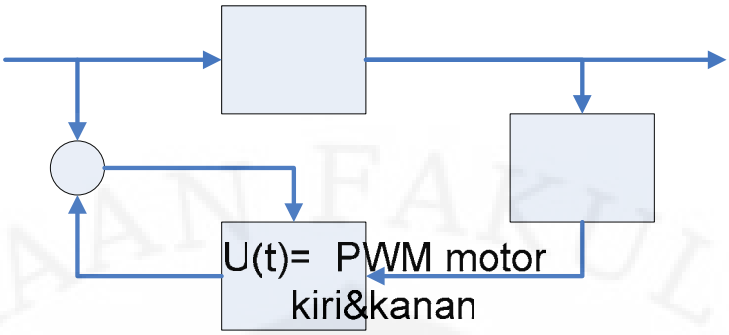
Algoritma penyelesaian masalah keempat dapat dilihat pada Gambar 3.9. Algoritma ini merupakan algoritma keseluruhan gerak robot di dalam ruangan. Robot dapat kembali ke posisi semula jika ruangan yang telah diperiksa berjumlah dua. Variabel ruangan diatur sebelum robot bergerak, dengan nilai awal adalah 0. Variabel ini akan bertambah setiap kali robot memasuki ruangan. Robot kembali ke posisi home dengan mengunakan algoritma mengelilingi labirin (algoritma pertama) dimana robot akan mengabaikan semua ruangan. Robot akan terus bergerak hingga mencapai posisi home. Posisi ini ditunjukkan dengan lingkaran bewarna putih. Jika sensor garis menemukan lingkaran tersebut maka robot akan berhenti. Hal ini menandakan robot sudah kembali ke posisi home (start).



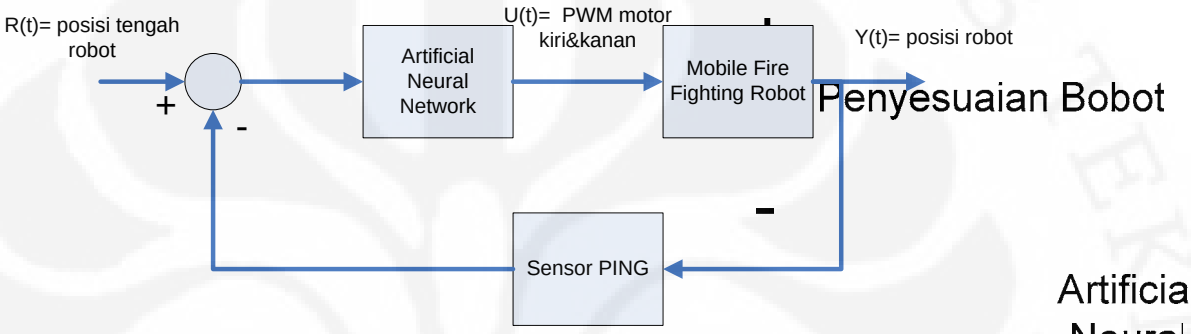
Gambar 3.9 Algoritma keseluruhan gerak robot hingga kembali ke posisi awal.

3.4.3 Flowchart Pelatihan dan Pengujian Jaringan Syaraf Tiruan

Percobaan terdiri dari 2 tahap, yaitu proses pelatihan jaringan syaraf tiruan, dan proses identifikasi atau pengujian jaringan syaraf tiruan yang telah dilatih. Blok diagram dari pelatihan dan pengujian atau identifikasi jaringan syaraf tiruan dapat dilihat pada Gambar 3.12. dan Gambar 3.13. Pada proses pelatihan diberikan PWM (tegangan) pada motor kiri dan kanan dengan besar tegangan yang tertentu. Selanjutnya dilihat pengaruh PWM tersebut pada posisi robot menggunakan sensor PING. Posisi ini digunakan sebagai input untuk proses pelatihan jaringan syaraf tiruan. Jaringan syaraf tiruan akan mengeluarkan PWM yang akan dibandingkan dengan PWM yang diberikan di awal. Perbedaan (*error*) yang terjadi digunakan untuk memperbaiki bobot pada jaringan.



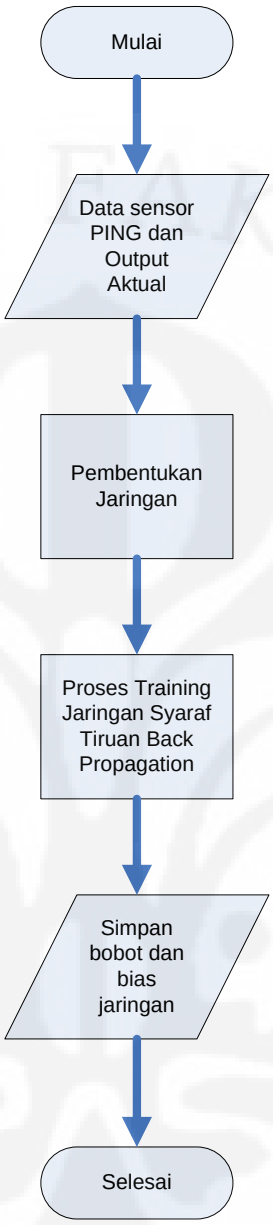
Gambar 3.10 Blok diagram pelatihan jaringan syaraf tiruan



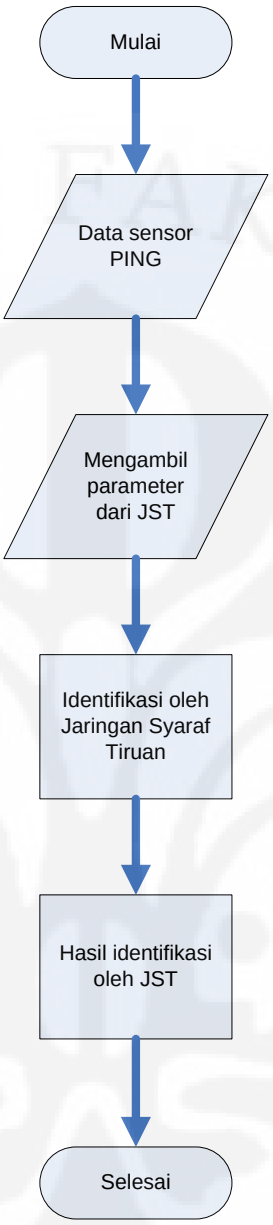
Gambar 3.11 Blok diagram aplikasi jaringan syaraf tiruan

Pada proses aplikasi, jaringan syaraf tiruan dengan menggunakan bobot yang didapat dari pelatihan menghasilkan output berupa PWM untuk motor kiri dan kanan pada robot. Pemberian PWM ini akan memberikan efek perubahan posisi pada robot. Posisi robot akan menjadi feedback yang akan dibandingkan dengan posisi referensi (posisi robot pada bagian tengah labirin) sehingga robot diharapkan dapat berjalan lurus.

Proses pelatihan untuk menentukan bobot dari jaringan syaraf tiruan dilakukan dengan menggunakan MATLAB 7.0, sedangkan pengujian dilakukan dengan menggunakan Mikrokontroller ATMEGA32. Flowchart dari pelatihan dan pengujian atau identifikasi jaringan syaraf tiruan dapat dilihat padaGambar 3.12. dan Gambar 3.13.



Gambar 3.12 Flowchart pelatihan jaringan syaraf tiruan



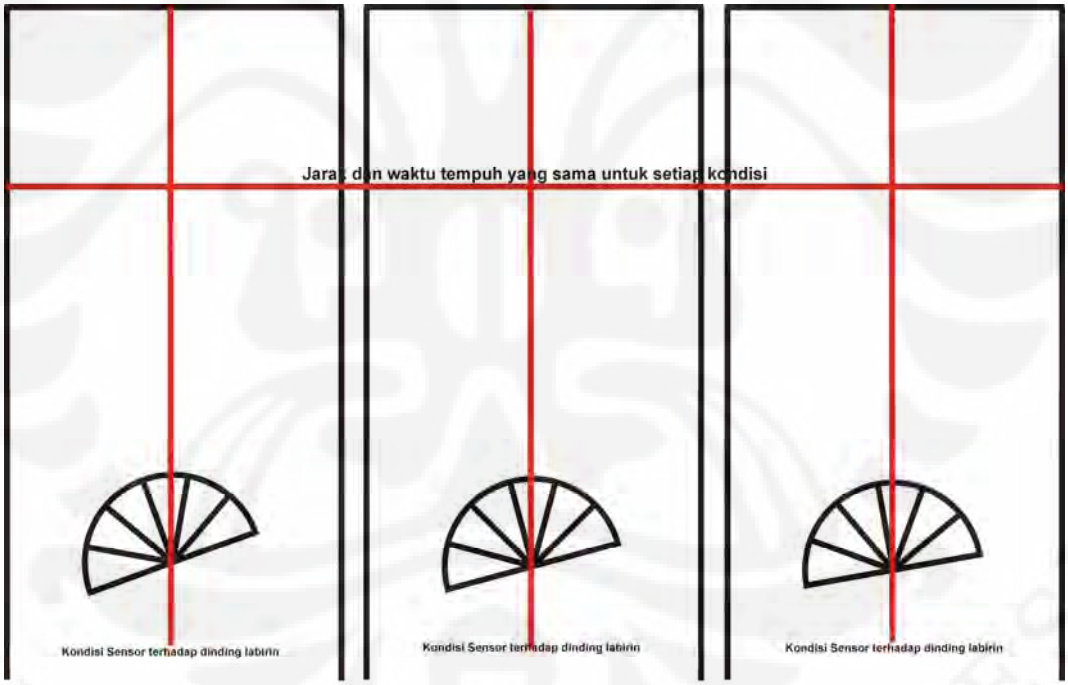
Gambar 3.13 Flowchart pengenalan jaringan syaraf tiruan

3.4.4 Langkah Pengerjaan

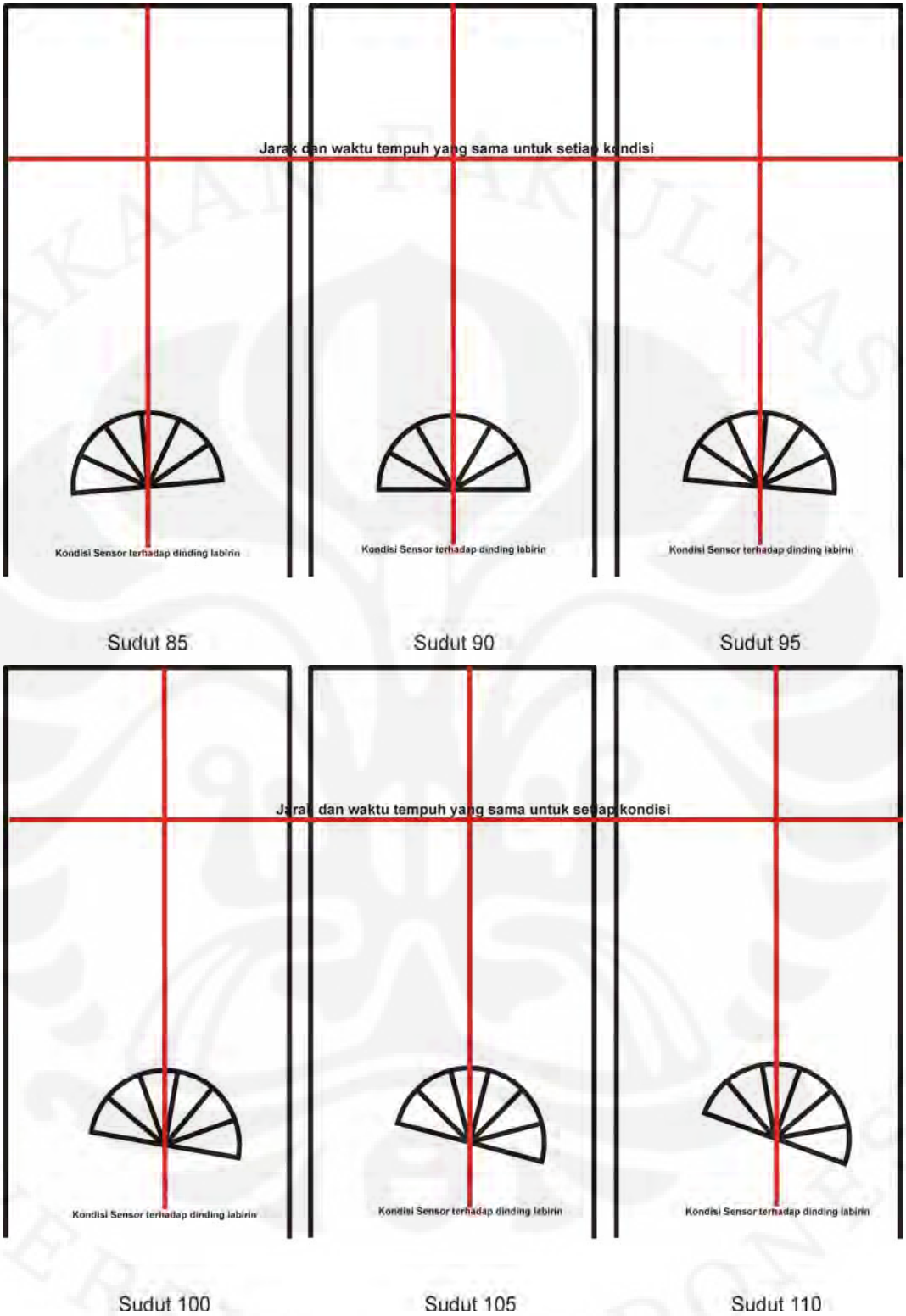
3.4.4.1 Pengambilan Data Sensor Jarak dan Keluaran Motor

Untuk melakukan proses pelatihan dibutuhkan suatu data berupa masukan dan keluaran yang sudah diketahui nilai kebenarannya. Namun pada robot ini data tidak dapat direkam karena keterbatasan dari sistem robot tersebut. Selain itu, tidak semua robot buatan tangan(*handmade*) mempunyai karakteristik yang sama. Perbedaan ini disebabkan ketidaktepatan dalam pemasangan roda dan motor. Akibatnya, tidak jarang ada perbedaan putaran antara roda kiri dan kanan

meskipun diberikan tegangan yang sama. Oleh karena itu diperlukan suatu pendekatan untuk mencari data masukan dan keluaran yang akan digunakan sebagai data pelatihan. Secara garis besar, proses pengambilan data adalah meletakkan robot pada berbagai kondisi di dalam labirin dimana robot harus dapat kembali ke posisi lurus dengan output tertentu. Hal yang dilakukan adalah mengambil data dari empat buah sensor jarak PING (sensor PING 1, sensor PING 2, sensor PING 4 dan sensor PING 5) kemudian memberikan kecepatan tertentu pada motor sebelah kiri dan sebelah kanan agar posisi robot kembali ke posisi normal. Dengan catatan waktu yang diperlukan untuk kembali dari berbagai posisi ke posisi semula adalah sama. Data yang diambil dari keempat sensor PING menunjukkan posisi robot terhadap titik tengah jalanan pada labirin, seperti pada Gambar 3.12.



Sudut 70 Sudut 75 Sudut 80



Gambar 3.14 Gambar posisi robot untuk pengambilan data

Hasil percobaan terlampir pada Tabel 3.1. Data ini kemudian digunakan sebagai input dan output aktual pada proses pelatihan jaringan syaraf tiruan *back propagation.*

Tabel 3.1Data masukan sensor jarak dan keluran motor

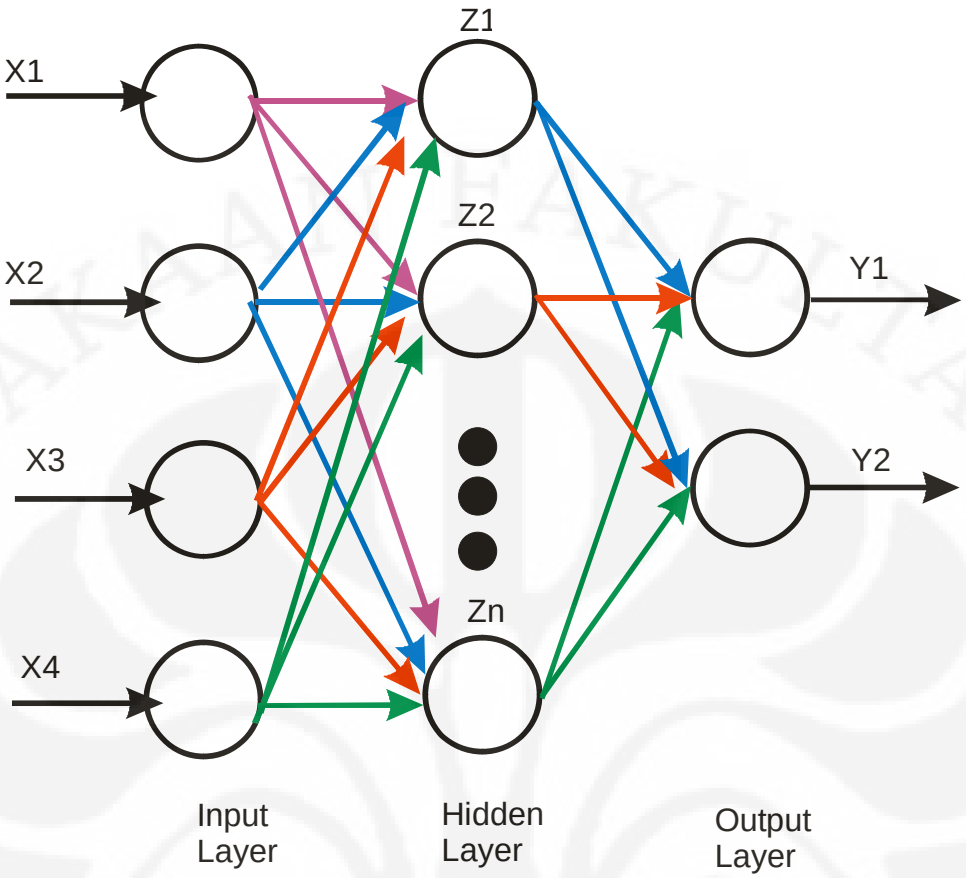
Sudut Posisi Robot	Jarak PING1	Jarak PING2	Jarak PING4	Jarak PING5	PWM Motor Kiri	PWM Motor Kanan
70^o	4	6	47	17	100	20
75^o	5	8	37	15	95	30
80^o	5	10	20	13	95	45
85^o	7	13	20	11	85	55
90^o	9	15	15	9	45	47
95^o	11	19	13	8	50	85
100^o	13	35	12	7	45	100
105^o	15	36	8	5	30	100
110^o	17	45	5	4	20	100

3.4.4.2 Jaringan Syaraf Tiruan Back Propagation untuk Pelatihan

Input dari jaringan syaraf tiruan diambil dari proses diatas. Jadi jaringan syaraf tiruan menggunakan 4 *node* pada *layer* masukan dan 2 *node* pada *layer output*. Pelatihan jaringan syaraf tiruan dilakukan dengan Matlab. Pada pelatihan, tidak ditekankan momentum maupun konstanta belajar, pelatihan dilakukan untuk mencari parameter jaringan (bobot dan bias) yang kemudian dapat digunakan untuk pengenalan.

Pada percobaan, proses pelatihan dilihat dan dibandingkan dengan mengubah – ubah jumlah *node* pada lapisan tersembunyi, dengan satu *hidden layer*, sebesar apa jumlah *node* pada *hidden layer* menentukan keberhasilan pergerakan robot.

Gambar 3.15 merupakan gambar jaringan syaraf tiruan dengan 4 *node* pada *input layer*, m *node* pada *hidden layer* dan 2 *node* pada *output layer* (bias tidak disertakan pada Gambar 3.15).



Gambar 3.15 Jaringan syaraf tiruan dengan 4 *node* pada *input layer*, m *node* pada *hidden layer* dan 2 *node* pada *output layer*

Proses pelatihan dilakukan dengan memberikan pasangan parameter *input* dan *output*, dari semua data yang akan dilatih sebagai *input* dan dipasangkan dengan *output* representasi keluaran seperti pada Tabel 3.1Data masukan sensor jarak dan keluran motor akan dijadikan parameter pelatihan jaringan. Proses pelatihan jaringan syaraf tiruan dilakukan dengan perintah:

1. Pembuatan jaringan:

net = newff([0 jumlahmax;],[JNH JNO],{FA1 FA2});

net.trainParam.goal = besar toleransi kesalahan;

dimana:

jumlahmax = nilai maksimal pada *node input*.

JNH = jumlah *node* pada lapisan tersembunyi

JNO = jumlah *node* pada lapisan keluaran

FA1 = fungsi aktivasi pada lapisan tersembunyi

FA2 = fungsi aktivasi pada lapisan keluaran

Pada percobaan jumlahmax ditetapkan 50, dengan asumsi bahwa *input* jaringan tidak akan melebihi 50 (karena jarak robot dengan dinding labirin maksimal 50 cm, meskipun jangkauan sensor 3 – 300 cm). JNO ditetapkan 2, karena keluaran jaringan adalah 2. FA1 ditetapkan 'tansig', yang berarti fungsi *sigmoid bipolar*, dan FA2 ditetapkan '*purelin*'. Besar toleransi kesalahan ditetapkan sebesar 1e-5.

2. Memasukkan parameter

Sebelum dilatih, semua pasangan *input output* dari semua data harus dimasukkan, untuk selanjutnya dilatih, semua *input* dimasukkan ke dalam suatu variabel misalnya semua_input, dan semua*output* juga dimasukkan dalam suatu variabel misalnya semua_output.

3. Melatih jaringan

net = train(net,semua_input,semua_output);

Setelah semua parameter dimasukkan, yakni net untuk jaringan, semua_input untuk *input*, dan semua_output untuk *output* yang akan dilatih, jarigan tersebut akan dilatih dengan fungsi train() yangsudah didapat pada matlab.

3.4.4.3 Jaringan Syaraf Tiruan Back Propagation untuk Pengujian

Pada proses pelatihan didapat bobot dan bias yang digunakan untuk proses pergerakan robot. Proses pergerakan dilakukan pada mikrokontroler AVR dari Atmel Corporation yang berjenis ATMEGA32, dengan menggunakan perintah :

a) Definisikan variabel – variabel untuk perhitungan

int neuron_in[5] = jumlah node pada lapisan *input*

float neuron_hidden[16] = jumlah node pada lapisan *hidden*

float neuron_hidden_out[16] = jumlah node pada lapisan *hidden* setelah fungsi aktivasi

float neuron_out[3] = jumlah node pada layer *output*

float neuron_out_out[3] = jumlah node pada layer *output* setelah fungsi aktivasi

int jumlahInput = n = banyaknya *node* lapisan *input*;

int jumlahLayer = p = banyaknya *node* pada lapisan *hidden*;

int jumlahOutput = m = banyaknya *node* pada lapisan *output*;

b) Definisikan variabel – variabel bobot dan bias dengan nilai yang diambil pada pelatihan:

float weight_in_hidden[5][16] = nilai bobot dan bias dari lapisan *input* ke lapisan *hidden* yang didapat dari pelatihan

float weight_hidden_out[16][3]= nilai bobot dan bias dari lapisan hidden ke lapisan *output* yang didapat dari pelatihan

c) Pengenalan atau pengujian(dengan menggunakan algoritma *backpropagation* bagian *feedforward* yang telah dipaparkan pada dasar teori):

a) Tiap-tiap *unit input* (X_i,i=1,2,3..n) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua *unit* pada lapisan yang ada di atasnya (lapisan tersembunyi).

b) Tiap – tiap *unit* pada suatu lapisan tersembunyi (Z_j, j=1,2,3,…,p) menjumlahkan sinyal – sinyal *input* terbobot (sesuai persamaan.2.7)

```
for(j=1;j<=p;j++){
    sigma=0;
    for(i=1;i<=n;i++){
            sigma+=neuron_in[i]*weight_in_hidden[i][j];
    }//end for1
    neuron_hidden[j]=weight_in_hidden[0][j]+sigma;
    neuron_hidden_out[j]=bip_sig(neuron_hidden[j]);
}//end for
```

dan kirimkan sinyal tersebut ke semua *unit* di lapisan atasnya(*unit-unit output*). Sinyal yang dikirim adalah neuron_hidden_out[].

c) Tiap – tiap *unit output* (Y_k, k=1,2,3,…,m) menjumlahkan sinyal – sinyal *input* terbobot (sesuai dengan persamaan 2.9).

```
for (k=1;k<=m;k++){
    sigma=0;
    for(j=1;j<=p;j++){
            sigma+=neuron_hidden_out[j]*weight_hidden_out[j][k];
```

```
    }//end for1
    neuron_out[k]=weight_hidden_out[0][k]+sigma;
    neuron_out_out[k]=purelin(neuron_out[k]);
  }//end for
```

Karena fungsi aktivasinya adalah *purelin* yang mempunyai persamaan f(x) = x, maka f(neuron_out[k]) = neuron_out[k], dengan kata lain hasil sinyal setelah aktivasi sama dengan sebelum aktivasi.

BAB IV
HASIL UJICOBA dan ANALISIS

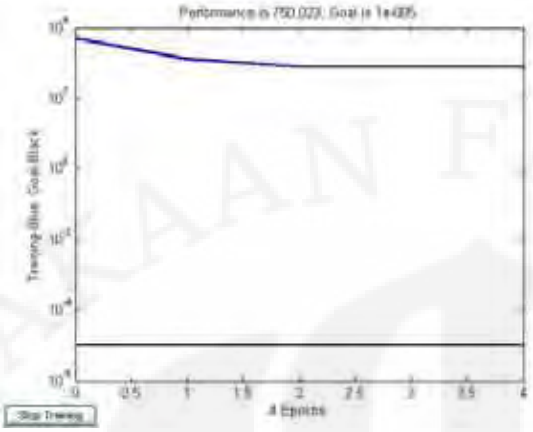
4.1 HASIL UJICOBA dan ANALISIS PEMBENTUKAN JARINGAN

4.2.1 Hasil Ujicoba Pembentukan Jaringan

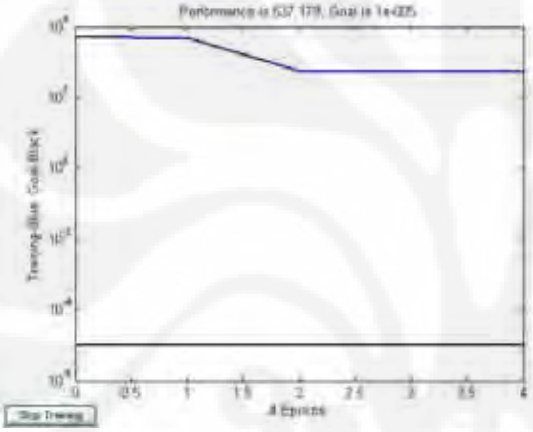
Pada pembentukan jaringan, bobot – bobot dan bias yang terbentuk berbeda – beda, bobot – bobot dan bias tersebut dilatih sedemikian sehingga jaringan dapat mengenali ke semua data *input* beserta pasangan *output*nya. Pada skripsi ini, error pelatihan diatur sebesar 1e-5 untuk masukan jaringan yang ideal. Proses pelatihan dapat saja gagal bila pada pelatihan jaringan sudah tidak dapat lagi memperbaiki bobotnya untuk mendapatkan keluaran yang sesuai. Tabel 4.1 memperlihatkan hasil percobaan pelatihan yang dilakukan di Matlab 7.0, dengan jumlah percobaan merupakan berapa kali jaringan coba dibuat apabila jaringan yang dibuat sebelumnya gagal. Gambar dibawah menunjukkan grafik dari proses pelatihan dengan jumlah node pada hidden layer yang berbeda-beda.

Tabel 4.1Uji coba pembentukkan jaringan

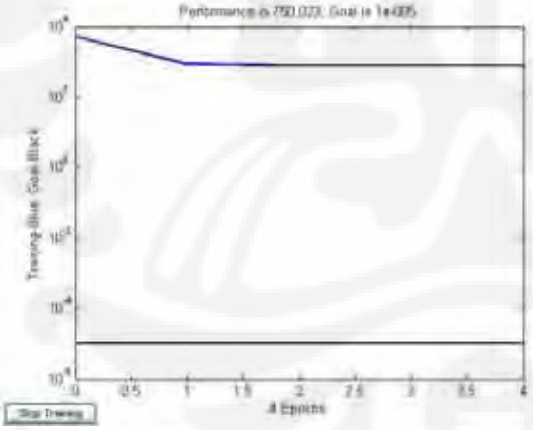
jumlah *node*	hasil	jumlah percobaan
1	gagal	10
2	gagal	10
3	gagal	10
4	gagal	10
5	gagal	10
6	gagal	10
7	gagal	10
8	gagal	10
9	gagal	10
10	gagal	10
15	berhasil	4
20	berhasil	1
25	berhasil	2
30	berhasil	2
35	berhasil	1
40	berhasil	1



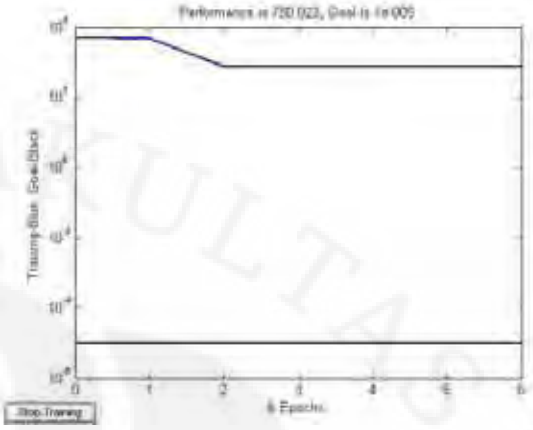
Gambar 4.1 Pelatihan 1 node hidden layer



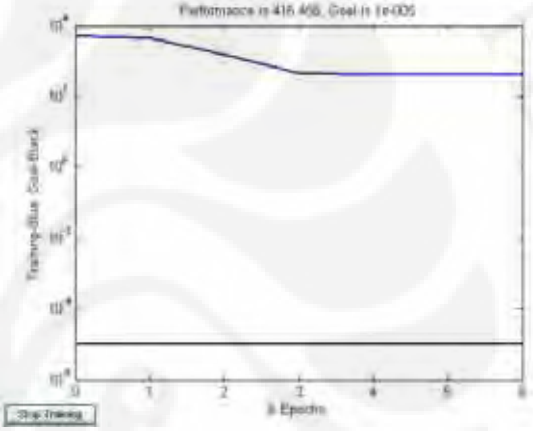
Gambar 4.2 Pelatihan 2 node hidden layer



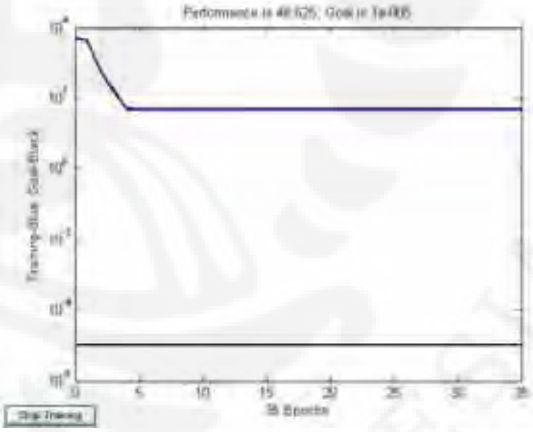
Gambar 4.3 Pelatihan 3 node hidden layer



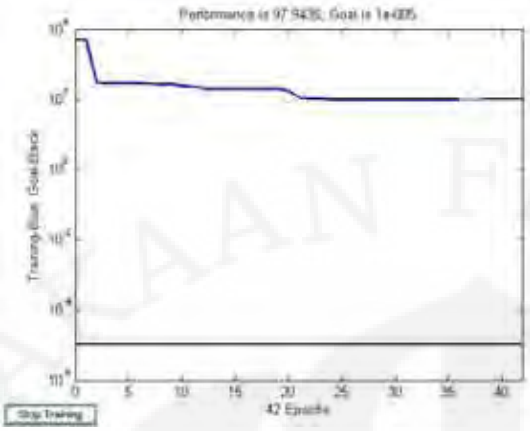
Gambar 4.4 Pelatihan 4 node hidden layer



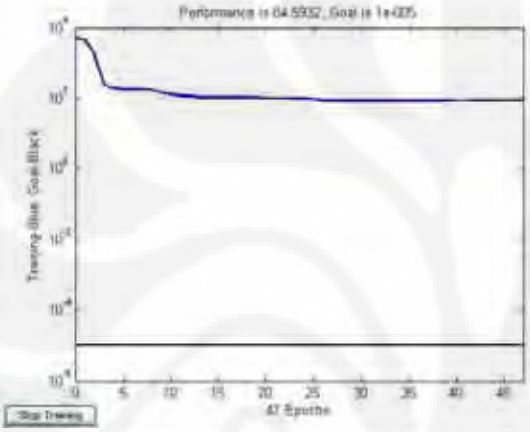
Gambar 4.5 Pelatihan 5 node hidden layer



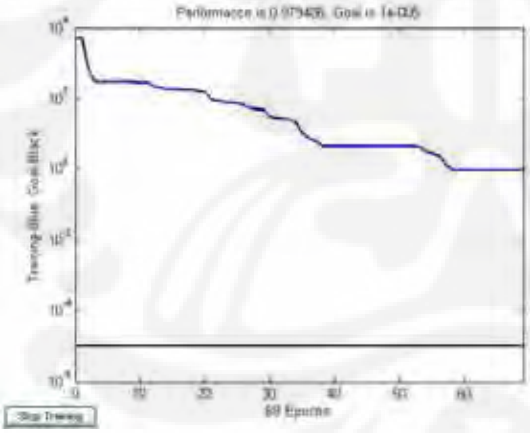
Gambar 4.6 Pelatihan 6 node hidden layer



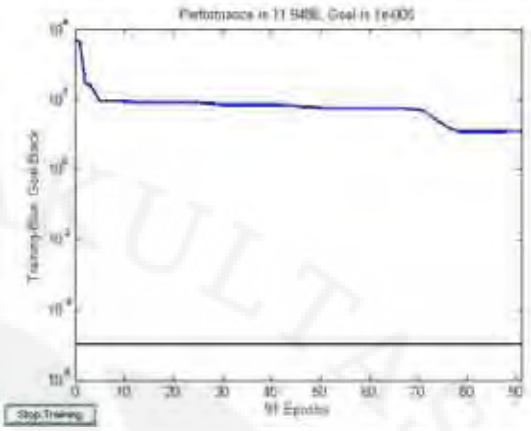
Gambar 4.7 Pelatihan 7 node hidden layer



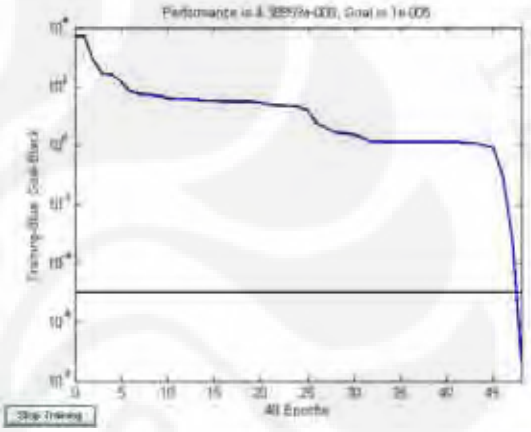
Gambar 4.8 Pelatihan 8 node hidden layer



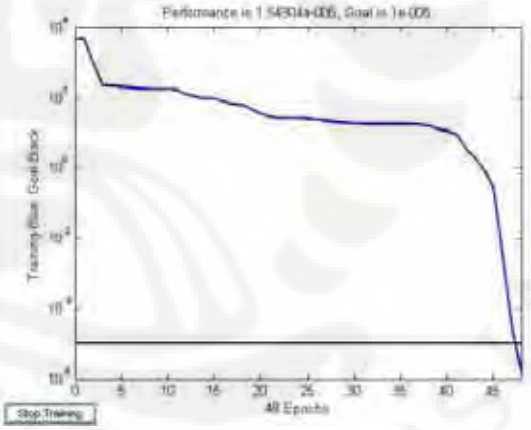
Gambar 4.9 Pelatihan 9 node hidden layer



Gambar 4.10 Pelatihan 10 node hidden layer

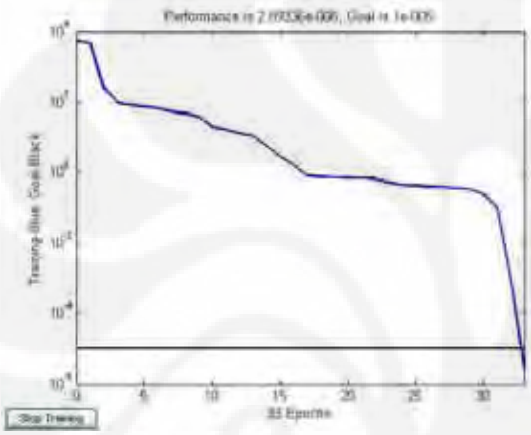


Gambar 4.11 Pelatihan 15 node hidden layer

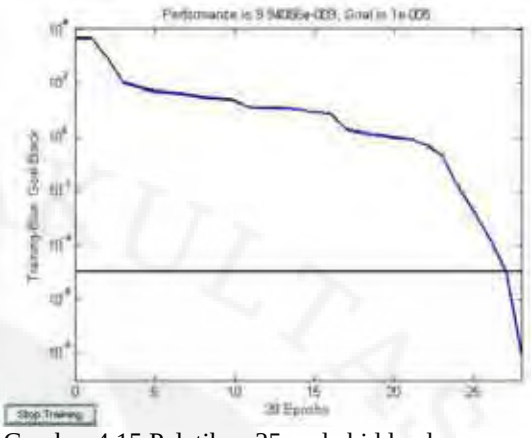


Gambar 4.12 Pelatihan 20 node hidden layer

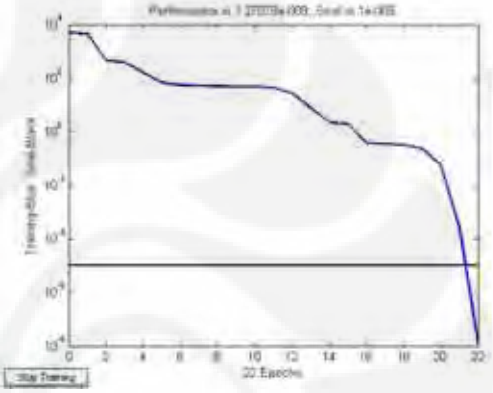
Gambar 4.13 Pelatihan 25 node hidden layer



Gambar 4.14 Pelatihan 30 node hidden layer



Gambar 4.15 Pelatihan 35 node hidden layer



Gambar 4.16 Pelatihan 40 node hidden layer

4.2.2 Analisis Pembentukan Jaringan

Proses pelatihan dikatakan berhasil jika hasil pelatihan mencapai target yang telah ditetapkan. Pada skripsi ini terget pelatihan ditetapkan berupa toleransi *error* sebesar 1e-5 . Pada Tabel 4.1 dapat dilihat bahwa untuk 1 sampai 10 *node* pada *hidden layer* proses pelatihan yang dilakukan mengalami kegagalan setelah dilakukan 10 kali pembentukan jaringan dengan inisialisasi nilai bobot dan bias secara acak. Hal ini disebabkan oleh jumlah *node* yang tidak mencukupi bagi jaringan untuk dapat menemukan bobot yang tepat untuk ke semua pasangan data pelatihan.

Pada Tabel 4.1 juga terlihat banyaknya percobaan untuk beberapa *node* tidak hanya satu kali, percobaan ini jika diulang akan ditemukan beberapa nilai jumlah percobaan yang berbeda di setiap *node*nya, hal ini disebabkan oleh penentuan bobot yang secara acak pada awal pembentukan jaringan. Pada percobaan ini dapat dilihat bahwa penentuan bobot acak sangat menentukan

keberhasilan jaringan, bobot acak yang tepat dapat membuat jaringan dapat dilatih hingga mendapatkan *error* hanya sebesar target pelatihan. Tetapi bobot acak awal yang tidak tepat dapat membuat jaringan tidak dapat menemukan bobot yang tepat selama pelatihan, yang berarti bobot tidak dapat lagi diperbaiki.

Pada grafik pelatihan dengan jumlah node hidden layer yang bervariasi, garis hitam menunjukkan target pelatihan dan garis biru menunjukkan kurva pergerakan pelatihan untuk mencapai target. Performance (garis biru) menunjukkan nilai dari pelatihan yang dilakukan sedangkan Goal (garis hitam) menunjukkan nilai dari target pelatihan. Pelatihan dikatakan berhasil jika nilai performance lebih kecil atau sama dengan nilai goal.

Pada grafik tersebut dapat dilihat berapa banyak iterasi (epoch) yang dilakukan agar target pelatihan (toleransi kesalahan) terpenuhi. Untuk jumlah node hidden layer yang gagal mencapai target pelatihan, terlihat bahwa semakin kecil jumlah node maka banyaknya iterasi yang dilakukan semakin sedikit untuk mencapai nilai konstan (mendatar). Sedangkan untuk jumlah node hidden layer yang berhasil mencapai target pelatihan, terlihat bahwa semakin banyak jumlah node maka iterasi yang dilakukan semakin sedikit untuk mencapai target pelatihan.

Pada beberapa jumlah node pada hidden layer yang gagal memperbaiki bobot, terlihat juga kurva grafik yang berbentuk garis yang mendatar. Hal ini menunjukkkan bahwa pelatihan mengalami kejenuhan dan tidak berhasil mendekati target pelatihan (toleransi kesalaahan) yang diinginkan. Dibandingkan dengan jumlah node pada hidden layer yang berhasil memperbaiki bobot untuk mencapai target pelatihan yang diinginkan, kurva grafik berbentuk garis yang mengalami penurunan menuju target yang diinginkan (toleransi kesalahan = 1e-5).

Jumlah *node* terendah pada *hidden layer* agar target pelatihan tercapai adalah 15 *node*, jadi untuk skala minimum penggunaan JST untuk pergerakan robot dengan konfigurasi 4 *node input* dan 2 *node output* dapat dilakukan hanya dengan 15 *node*, yang berarti dalam implementasinya dapat menghemat memori dan mempercepat proses perhitungan.

4.2 ANALISIS KESELURUHAN FUNGSI ROBOT

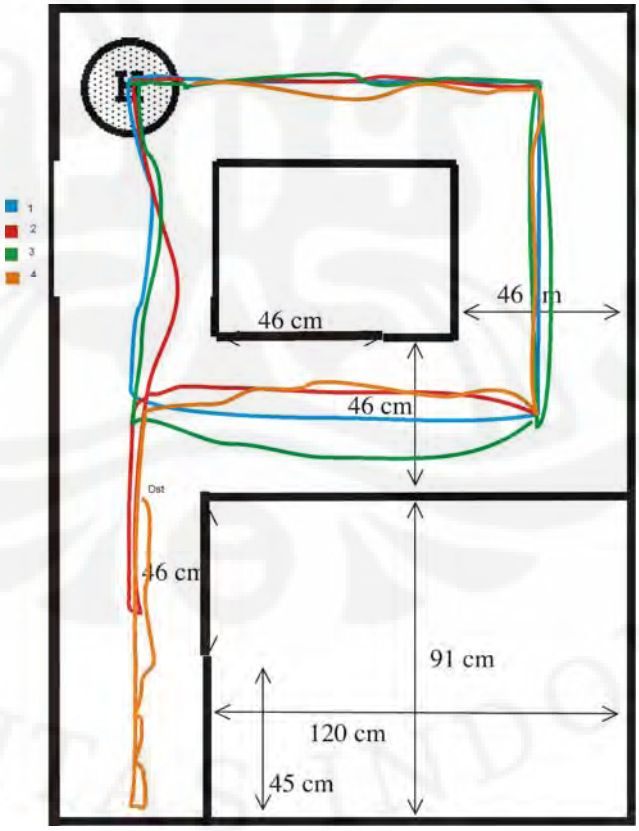
Nilai bobot dan bias yang didapat dari hasil pembentukan jaringan di Matlab, selanjutnya diterapkan pada pergerakan robot dengan menggunakan mikrokntroler AVR dari Atmel Corporation yang berjenis ATMEGA32. Layer tersembunyi (hidden) menggunakan node sebanyak 15 buah. Pemilihan ini didasarkan beberapa hal,yaitu:

1. Node minimal layer hidden yang dapat membentuk jaringan untuk kondisi robot pada skripsi ini sebanyak 15 buah.
2. Semakin kecil jumlah node pada semua layer akan meminimal jumlah memori mikorontroler yang digunakan.
3. Semakin kecil jumlah node pada semua layer akan meminimalkan waktu yang diperlukan untuk memproses pengendali jaringan syaraf tiruan. Sehingga dapat merespon perubahan kondisi dengan lebih cepat.

4.2.1 Ujicoba Pergerakan Robot Saat Kedua Ruangan Tertutup

Pengujian dilakukan untuk melihat pergerakan robot mengelilingi labirin, tanpa memasuki ruangan. Hasil pengujian berupa hasil pengamatan pergerakan robot yang dapat dilihat pada Gambar 4.17. Secara lengkap dapat dilihat dari CD yang terlampir. Posisi awal robot sejajar dengan garis pergerakan robot. Ujicoba ini menggunakan algoritma mengelilingi labirin, seperti yang sudah dijelaskan di bab sebelumnya. Dari posisi start, secara keseluruhan robot bergerak maju hingga ada halangan di depannya, kemudian robot mengecek sisi mana yang tidak ada halangan. Selanjutnya robot akan berbelok kesisi yang kosong tersebut. Namun jika tidak ada sisi yang kosong maka robot akan berbalik arah dan bergerak hingga ada halangan didepannya. Jika menemui persimpangan robot akan berbelok ke kanan untuk persimpangan pertama, dan berbelok kekiri untuk persimpangan kedua, begitu seterusnya secara bergantian. Dari gambar terlihat adanya gerakan yang melengkung pada beberapa jalur, karena penutup ruangan menggunakan bahan berupa kardus. Bahan ini dapat meredam bunyi yang dilepas dari sensor jarak, sehingga jarak yang tertangkap oleh sensor dengan benda

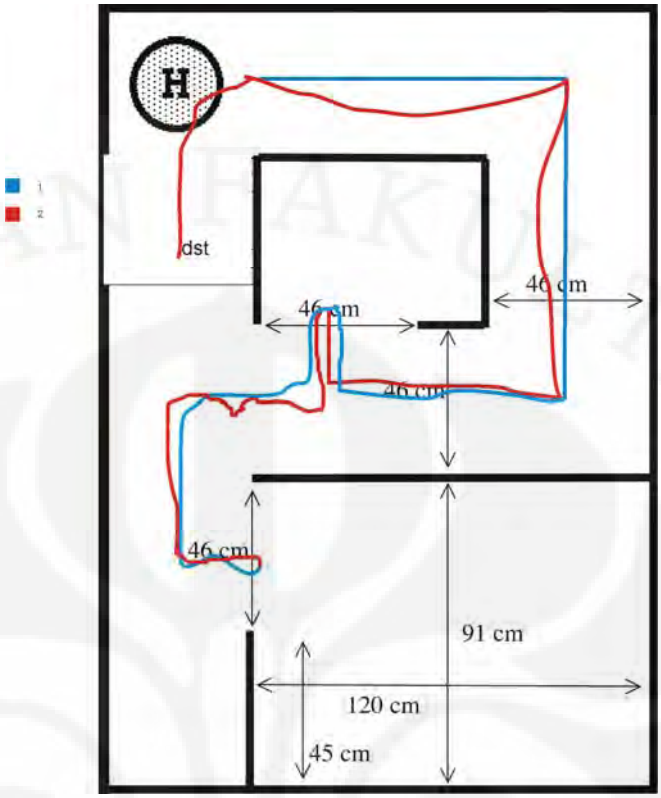
terlihat jauh. Namun dengan pengendali jaringan syaraf tiruan back propagation, robot dapat menangani kondisi ini sehingga robot dapat menghindari terjadinya tabrakan dengan dinding. Selain itu pada satu waktu saat robot menjalankan fungsi pergerakan menggunakan jaringan syaraf tiruan, pembacaan sensor mengalami kekacauan. Hal ini ditunjukkan dengan pengukuran jarak yang dilakukan sensor tidak tepat. Untuk mengatasi hal ini, pembacaan jarak yang dilakukan sensor dianggap benar jika sama dengan atau lebih kecil dari 50 cm, jika terbaca lebih maka dianggap 50 cm. Konstanta 50 cm didapat dari pengukuran jarak terjauh saat robot berada diantara kedua dinding labirin. Algoritma menemui persimpangan, dimana robot akan berbelok kekanan untuk persimpangan pertama dan kekiri untuk persimpangan kedua dan seterusnya, membuat robot dapat menjelajahi seluruh labirin dan dapat meminimalkan pergerakan robot pada daerah tertentu (pergerakan membentuk lingkaran atau berulang-ulang).



Gambar 4.17 Rute robot saat kedua ruangan tertutup

4.2.2 Ujicoba Pergerakan Robot Saat Kedua Ruangan Terbuka

Pengujian dilakukan untuk melihat pergerakan robot mengelilingi labirin sambil memasuki ruangan. Hasil pengujian berupa hasil pengamatan pergerakan robot yang dapat dilihat pada Gambar 4.18. Secara lengkap dapat dilihat dari CD yang terlampir. Posisi awal robot sejajar dengan garis pergerakan robot. Ujicoba dengan algoritma pertama akan menghasilkan kondisi seperti ujicoba yang pertama, karena robot akan terus bergerak maju hingga ada halangan didepannya. Oleh karena itu ujicoba ini menggunakan algoritma memasuki ruangan, seperti yang sudah dijelaskan di bab sebelumnya. Dari posisi start, robot bergerak lurus karena tidak ada sisi yang kosong atau ada halangan didepan. Robot berbelok ke kanan ketika menemukan ada halangan di depan dan di kiri. Selanjutnya robot bergerak lurus dan berbelok ke kanan saat menemukan ada halangan di depan dan di kiri. Kemudian robot bergerak maju dan berbelok ke kanan saat mendapati sisi kanan yang kosong. Kemudian robot maju kedepan selama satu detik dan sensor garis mendeteksi adanya garis, hal ini berarti robot sedang memasuki ruangan yang pertama. Selanjutnya robot berbalik arah dan keluar ruangan. Saat keluar ruangan robot menghadapi kondisi pesimpangan, maka robot akan bergerak kekanan dan maju hingga ada halangan didepannya. Kemudian robot menghadapi kondisi persimpangan lagi, robot berbelok ke kiri dan terus maju. Selama bergerak maju robot berbelok ke kiri karena menemukan ada sisi kiri yang kosong. Kemudian robot maju selama satu detik dan sensor garis mendeteksi garis putih dan robot kembali berbalik arah keluar ruangan. Saat keluar ruangan robot menghadapi kondisi pesimpangan, maka robot akan bergerak kekanan dan maju serta berbelok kekanan saat menemukan sisi kanan kosong. Begitu seterusnya robot akan bergerak masuk ke ruang pertama dan keluar dari ruang pertama dan berbelok kekiri karena ada persimpangan kemudian kembali ketitik awal dan robot bergerak kembali keruang pertama dan selanjutnya ke ruang kedua dan seterusnya keadaan ini akan berulang.

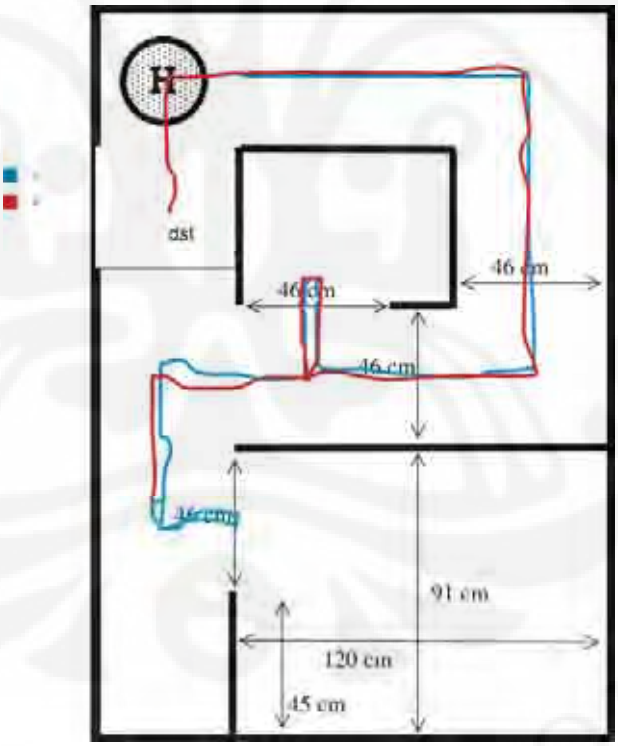


Gambar 4.18 Rute robot saat ruangan terbuka

4.2.3 Ujicoba Pergerakan Robot Saat Kedua Ruangan Terbuka dan Mencari Api

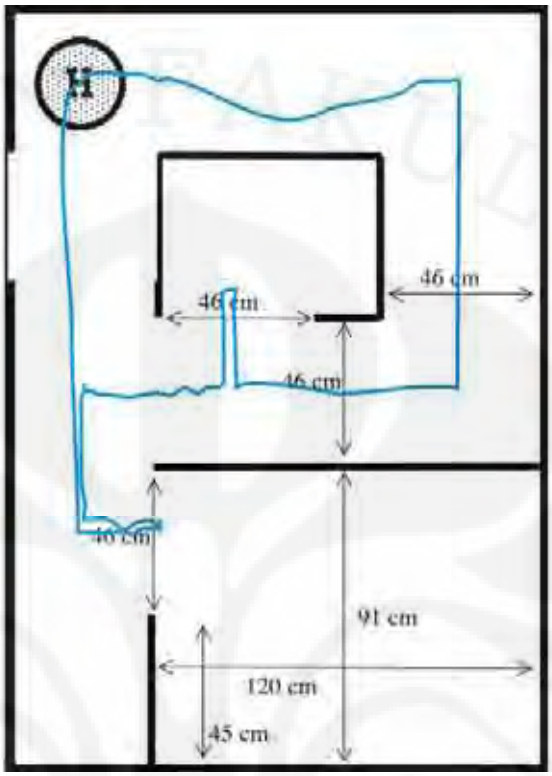
Pengujian dilakukan untuk melihat pergerakan robot mengelilingi labirin sambil memasuki ruangan. Hasil pengujian berupa hasil pengamatan pergerakan robot yang dapat dilihat pada Gambar 4.19. Secara lengkap dapat dilihat dari CD yang terlampir. Posisi awal robot sejajar dengan garis pergerakan robot. Ujicoba ini menggunakan algoritma memasuki ruangan dan ditambah fungsi mendeteksi api dan mematikannya, seperti yang sudah dijelaskan di bab sebelumnya. Dari posisi start, robot bergerak lurus karena tidak ada sisi yang kosong atau ada halangan didepan. Robot berbelok ke kanan saat ada halangan di depan dan kiri. Selanjutnya robot bergerak lurus dan berbelok ke kanan saat ada halangan di depan dan kiri. Kemudian robot bergerak maju dan selama pergerakan tersebut, robot menemukan sisi kanan yang kosong dan berbelok ke kanan. Kemudian robot maju kedepan selama satu detik dan sensor garis mendeteksi ada garis, berarti robot sedang memasuki ruangan yang pertama dan mulai memeriksa apakah ada api.

Selanjutnya robot berbalik arah dan keluar ruangan. Saat keluar ruangan robot menghadapi kondisi pesimpangan, maka robot akan bergerak kekanan. dan maju hingga ada halangan didepannya. Kemudian robot menghadapi kondisi persimpangan lagi, robot berbelok kesebelah kiri dan terus maju. Selama bergerak maju robot menemukan sisi kiri yang kosong sehingga Robot berbelok ke kiri. Kemudian maju selama satu detik dan sensor garis mendeteksi garis putih kemudian robot mulai memeriksa apakah ada api dan robot kembali berbalik arah keluar ruangan. Saat keluar ruangan robot menghadapi kondisi pesimpangan, maka robot akan bergerak kekanan. dan maju serta berbelok saat disisi kanan ada sisi yang kosong. Begitu seterusnya robot akan bergerak masuk keruang pertama dan keluar dari ruang pertama dan berbelok kekiri karena ada persimpangan kemudian kembali ketitik awal dan robot bergerak kembali keruang pertama dan selanjutnya keruang kedua dan seterusnya keadaan ini akan berulang.



Gambar 4.19 Rute robot saat ruangan terbuka dan mencari serta mematikan api

4.2.4 Ujicoba Keseluruhan Fungsi Robot



Gambar 4.20 Rute robot saat mencari, mematikan api dan kembali ke home

Pengujian dilakukan untuk melihat pergerakan robot mengelilingi labirin sambil memasuki ruangan. Hasil pengujian berupa hasil pengamatan pergerakan robot yang dapat dilihat pada Gambar 4.20. Secara lengkap dapat dilihat dari CD yang terlampir. Posisi awal robot sejajar dengan garis pergerakan robot. Ujicoba ini menggunakan algoritma keempat (kembali ke home), seperti yang sudah dijelaskan di bab sebelumnya. Dari posisi start, Robot bergerak lurus karena tidak ada sisi yang kosong atau ada halangan didepan. Robot berbelok ke kanan saat ada halangan di depan dan kiri. Selanjutnya robot bergerak lurus dan berbelok ke kanan saat ada halangan di depan dan kiri. Kemudian robot bergerak maju dan berbelok ke kanan saat ada sisi kanan yang kosong. Kemudian robot maju kedepan selama satu detik dan sensor garis mendeteksi ada garis, berarti robot sedang memasuki ruangan yang pertama dan mulai memeriksa apakah ada api. Variabel ruangan bertambah 1. Selanjutnya robot berbalik arah dan keluar ruangan. Saat keluar ruangan robot menghadapi kondisi pesimpangan, maka robot akan bergerak kekanan. dan maju hingga ada halangan didepannya. Kemudian

robot menghadapi kondisi persimpangan lagi, robot berbelok kesebelah kiri dan terus maju. Selama bergerak maju, robot mendeteksi ada sisi kiri yang kosong sehingga robot berbelok ke kiri. Kemudian maju selama satu detik dan sensor garis mendeteksi garis putih kemudian robot mulai memeriksa apakah ada api. Variabel ruangan bertambah menjadi 2. Karena Variabel Ruangan = 2 maka robot kembali berbalik arah keluar ruangan dan kembali bergerak lurus menuju home.

BAB V
KESIMPULAN

Hasil perancangan, ujicoba dan analisa memberikan beberapa kesimpulan sebagai berikut:

1. Pergerakan *mobile fire fighting robot* menggunakan jaringan syaraf tiruan sebagai pengendali pergerakan robot di dalam labirin dapat direalisasikan dengan menggunakan mikrokontroler AVR dari Atmel Corporation yang berjenis ATMEGA32.
2. Konfigurasi sensor jarak (PING) sebanyak 5 buah yang berbentuk setengah lingkaran dapat meng-*cover* daerah sekitar robot.
3. Pelatihan jaringan syaraf tiruan pada mobile fire fighting robot cukup menggunakan jumlah pasangan data input dan output sebanyak 9 buah saja, dengan jumlah iterasi maksimal 100 epoh.
4. Jumlah node minimal pembentukan jaringan pada *hidden layer* untuk pergerakan robot pada skripsi ini dengan *4 input* dan 2 *output* adalah 15 buah.
5. Algoritma menemui persimpangan, dimana robot akan berbelok kekanan untuk persimpangan pertama dan kekiri untuk persimpangan kedua dan seterusnya, membuat robot dapat menjelajahi seluruh labirin dan dapat meminimalkan pergerakan robot pada daerah tertentu (pergerakan membentuk lingkaran atau berulang-ulang).
6. Bahan-bahan yang dapat meredam bunyi dapat membuat pembacaan jarak oleh sensor PING salah, dimana jarak yang terukur lebih jauh dibandingkan jarak yang sebenarnya.
7. Pembacaan jarak yang salah oleh sensor ping selama proses pergerakan robot mengunakan jaringan syaraf tiruan dapat dilakukan dengan cara: pembacaan jarak yang dilakukan sensor dianggap benar jika sama dengan atau lebih kecil dari 50 cm, jika terbaca lebih dari 50 cm maka dianggap jarak yang terukur adalah 50 cm.

DAFTAR ACUAN

[1] Muis, Saludin, ”Teknik Jaringan Syaraf Tiruan”, halaman 3.

[2] Hermawan, Arief, ”Jaringan Syaraf Tiruan Teori Dan Aplikasi”, halaman 3-7,11-12,37-38,49-57.

[3] Kusumadewi, Sri, ”Membangun Jaringan Syaraf Tiruan menggunakan MATLAB & EXCEL LINK”,50-51,94-112.

[4] Fausett, Laurene, ”Fundamental of Neural Networks (Architectures, Algorithms, and Applications)”, halaman 294-296.

DAFTAR PUSTAKA

Acroname Easier Robotics. UVTron. Acroname Inc. September 2004.

Atmel AVR ATmega32 8-bit microcontroller with 32K Bytes In-System Programmable Flash, Atmel Corporation, 2003.

Fausett, Laurene, ”Fundamental of Neural Networks (Architectures, Algorithms, and Applications)”, New Jersey: Prentice-Hall.1994

Hermawan, Arief, ”Jaringan Syaraf Tiruan Teori Dan Aplikasi”,Andi Offset, 2006.

Ignatius Denny Wicaksono,”Pengenalan Suara dengan Jaringan Syaraf Tiruan Menggunakan DSP Starter Kit TMS320C6713 ” Skripsi, Program Sarjana Fakultas Teknik UI, Depok, 2007.

Kusumadewi, Sri, ”Membangun Jaringan Syaraf Tiruan menggunakan MATLAB & EXCEL LINK”,Graha Ilmu, 2004.

Matlab help, neural network toolbox.

Muis, Saludin, ”Teknik Jaringan Syaraf Tiruan”,Graha Ilmu, 2006.

PING))) ™ Ultrasonic Distance Sensor. Parallax Inc. 2006.

Ridho Alpha Kusuma,”Rancang Bangun Mobile Robot Otonomi Menggunakan Logika Fuzzy untuk Pengendali Pergerakan dan Graph untuk Navigasi dalam Lingkungan Statis” Skripsi, Program Sarjana Fakultas Teknik UI, Depok, 2005.

Suyanto, ”Artificial Intelligence”, Informatika, 2007.

Yan Zhou, Dawn Wilkins, Robert P. Cook.” Neural Network Control for A Fire-Fighting Robot”. Dept. of Computer and Information Science,University of Mississippi, Weir 302, University, MS 38677

LAMPIRAN 1

PERANGKAT KERAS

MOBILE FIRE FIGHTING ROBOT

Dalam robot pemadam api ini, dimana keseluruhan sistemnya berjalan secara otomatis, membutuhkan sensor-sensor yang mendukung. Robot ini mempunyai tugas untuk mencari api dan mematikan dengan menelusuri lorong-lorong. Beberapa sensor yang digunakan adalah sensor jarak, sensor api, sensor kecepatan atau sensor posisi.

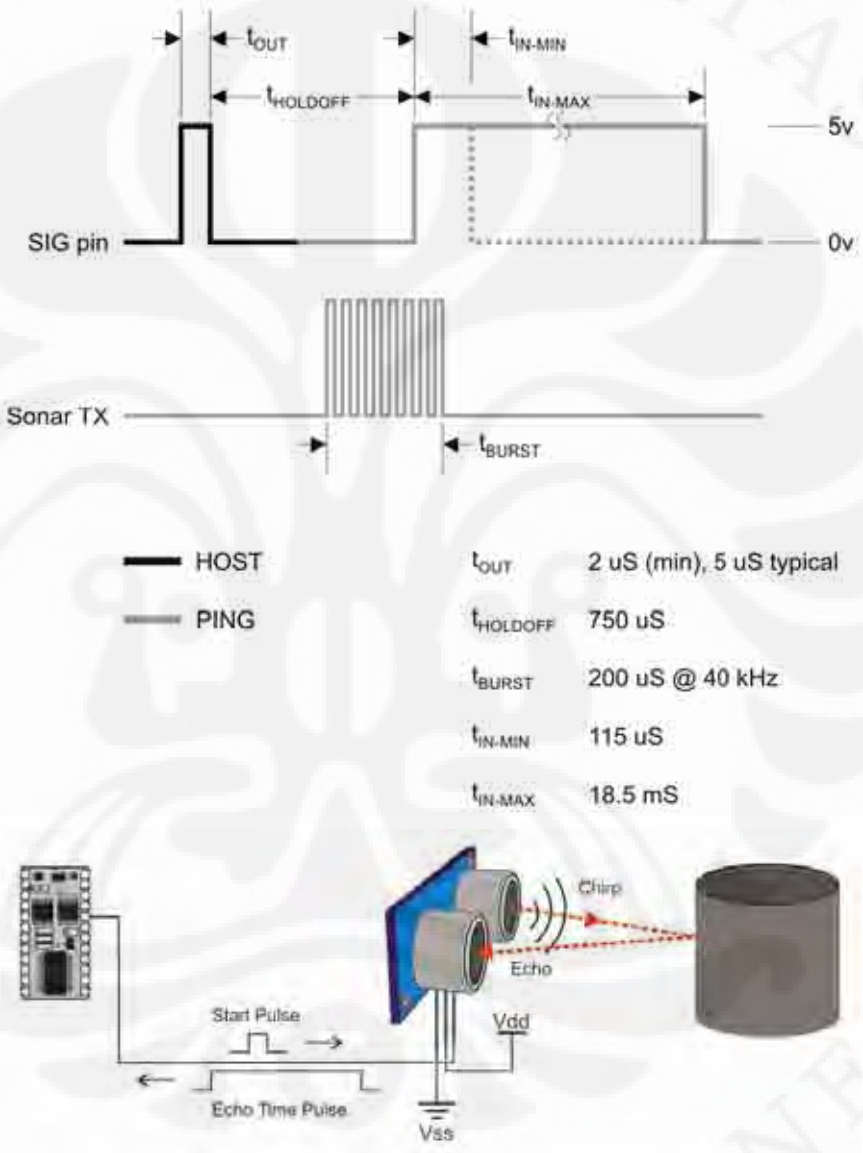
Sensor Jarak

Untuk mengukur jarak antara robot dengan dinding sebagai acuan pergerakan robot, maka digunakan sensor jarak / *ranger* yang menggunakan sinyal ultrasonik. Sensor yang digunakan adalah sensor jarak PING, seperti yang ditunjukkan pada Gambar 9.1 dibawah ini. Sensor PING terdiri dari 2 bagian yaitu speaker dan mikropon.

Gambar 9.1Sensor PING

Sensor PING mempunyai jangkauan pengukuran sebesar 3 centimeter hingga 3 meter.Cara kerja sensor PING [4] dapat di lihat pada Gambar 9.2 dibawah, dimana mikrokontroller mengirimkan sinyal output ke sensor PING agar sensor tersebut mengeluarkan bunyi ultrasonik sebesar 40 kHz melalui speaker.Selanjutnya mikrokontroler akan menerima input dari sensor PING, ketika sensor PING mendeteksi adanya echo melalui bagian mikropon. Hal ini akan mengubah nilai input mikrokontroler yang semula di set bernilai 1 (sinyal

high) menjadi nilai 0 (sinyal low). Dari sini kita dapat menghitung waktu yang dibutuhkan agar nilai input berubah dari 1 menjadi 0. Waktu ini kita gunakan untuk menghitung jarak suatu objek dari sensor menggunakan rumus untuk mengukur kedalaman laut yaitu: S=(V * T) / 2 dimana V= kecepatan bunyi di udara yaitu: 340 m/s.



Gambar 9.2 Cara kerja sensor PING

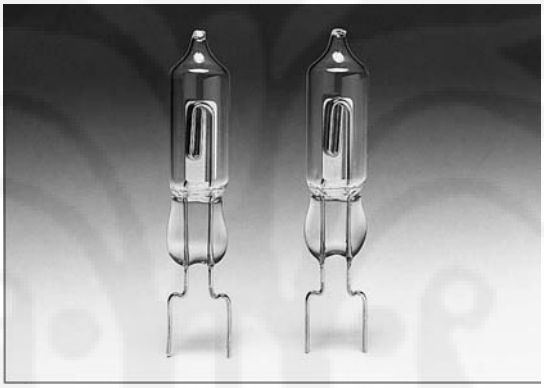
Sensor Api

Sensor Pendeteksi Api yang bernama UVTron mendeteksi adanya gelombang ultraviolet (panjang gelombang 185-260nm) [5] yang diemisikan oleh

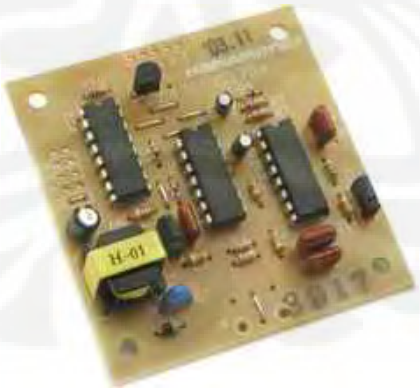
api. Sensor api yang digunakan merupakan produk dari Hamamatsu. Sensor api ini berguna untuk mendeteksi adanya api dalam suatu daerah atau tidak dan sebagai penentu arah api dengan sedikit modifikasi.

Karakteristik dari UVTron adalah dapat mendeteksi adanya api secara tepat dengan jarak sampai 5m, bahkan pantulan api dari dinding pun bisa dideteksi. UVTron akan menghasilkan pulsa sepanjang 10 mS bila terdapat api dan mempunyai interval 160mS, bila jarak sumber api semakin dekat maka interval akan semakin cepat.

UVTron ini terdiri dari 2 bagian yaitu tabung sensor ultraviolet dan rangkaian pengendali tabung sensor ultraviolet.



Gambar 9.3 Tabung sensor ultraviolet

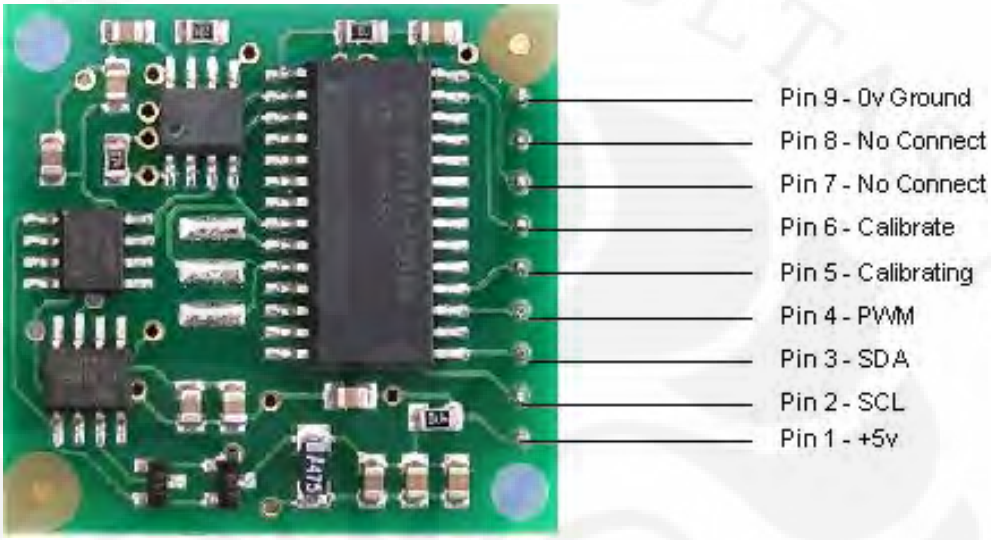


Gambar 9.4 Rangkaian pengendali UVTron

Sensor Kompas

Sensor kompas ini digunakan untuk menentukan arah suatu robot terhadap suatu titik referensi tertentu, sehingga dapat digunakan untuk proses berbelok arah

pada robot. Sensor ini menggunakan Philips KMZ51 yang merupakan sensor medan magnet, yang cukup sensitif untuk mendeteksi medan magnet bumi. Penggunaan kompas ini dapat dilakukan dengan 2 metode yaitu PWM dan I2C. Pada skripsi ini digunakan metode PWM.



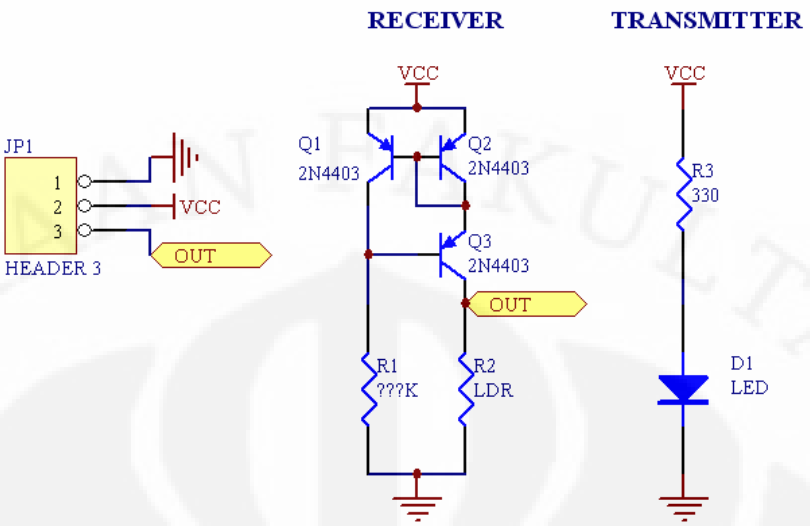
Gambar 9.5 Kompas Devantech

Sensor Garis Putih

Sensor garis putih menggunakan LDR (*Light Dependent Resistor*) sehingga berfungsi seperti sensor cahaya. Sensor terdiri dari 2 bagian, yaitu pemancar berupa LED berwarna putih dan bagian penerima berupa LDR. LDR yang menangkap cahaya pantulan dari LED sehingga nilai resistansinya berubah-berubah, bila semakin terang maka nilai hambatannya semakin kecil.

Konfigurasi LDR adalah dengan pembagi tegangan tetapi respon yang diharapkan sangat lambat dan histeresis terlalu besar lalu ditemukan dengan menggunakan konfigurasi cermin arus (*current mirror*). Konfigurasi cermin arus yang digunakan adalah konfigurasi Wilson (*Wilson Current Mirror*).

Dimana rangkaian cermin arus menginjeksikan sumber arus ke LDR yang dijadikan oleh beban.



Gambar 9.6 Skematik sensor garis dengan cermin arus Wilson

Terlihat pada Gambar 9.6 , bahwa yang dicari adalah nilai R1 berdasarkan dari nilai LDR pada saat gelap dan nilai V_{BE} dari transistor 2N4403. Nilai keluaran dari sensor ini antara gelap dan terang dibuat mempunyai *range* yang jauh, sehingga mempunyai *voltage swing* yang besar.

$$Ip = \frac{Vcc}{R_{LDR}} \quad \text{.....(8.1)}$$

dimana:

Vcc = tegangan rangkaian

Ip = arus yang akan diinjeksikan ke LDR

R_{LDR} = resistansi LDR pada saat gelap.

Sehingga dari arus injeksi (Ip) diatas dapat dicari nilai dari R1 yang sesuai dengan konfigurasi keluaran yang dinginkan.

$$R1 = \frac{(Vcc - V_{BE})}{Ip} \quad \text{.....(8.2)}$$

dimana:

Vcc = tegangan rangkaian

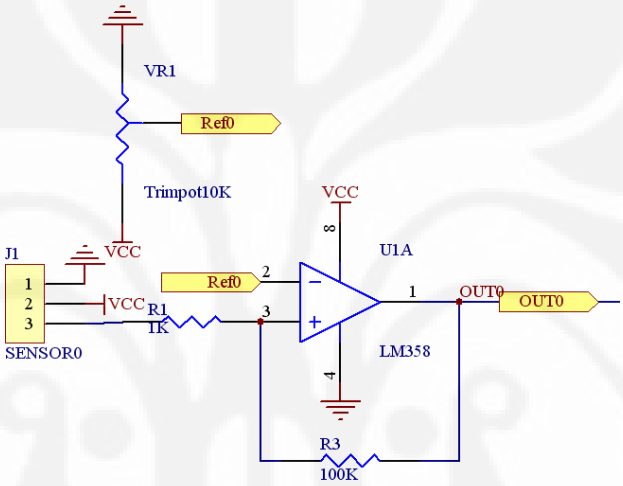
Ip = arus yang akan diinjeksikan ke LDR

V_{BE} = tegangan basis-emiter.

Keluaran dari sensor ini yang berupa tegangan yang akan dibandingkan di komparator analog yang akan menjadikannya suatu keluaran digital.

Komparator Analog

Rangkaian komparator analog berfungsi untuk membandingkan antara dua masukan analog yang berupa tegangan dan mempunyai keluaran berupa logika ”0” dan logika ”1”. Rangkaian komparator analog menggunakan OP-AMP sebagai komponen utamanya.



Gambar 9.7 Skematik dari komparator

Komparator membandingkan masukan antara titik *non-inverting* dan *inverting*, dimana:

1. Bila masukan *non-inverting* > masukan *inverting* maka keluarannya Vcc atau ”1”.
2. Bila masukan *non-inverting* ≤ masukan *inverting* maka keluarannya GND atau ”0”.

Rangkaian komparator yang dibuat untuk robot ini (Gambar 9.7) membandingkan nilai tegangan dari sensor garis yang kemudian dibandingkan dengan nilai referensi yang diatur. Selain membandingkan, rangkaian ini juga mempunyai konfigurasi *hysteresis* , yaitu R3 dan R1, yang berfungsi untuk memperkecil daerah *hysteresis* sehingga nilai logika lebih cepat dicapai.

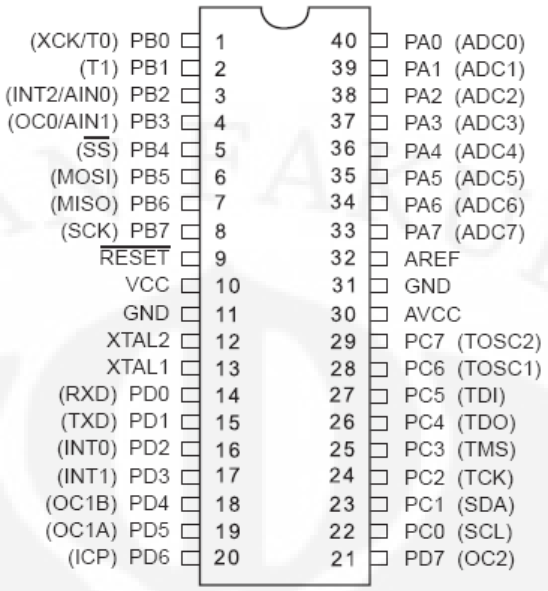
Mikrokontroller ATMEGA 32

Proses pengendalian robot ini menggunakan mikrokontroler 8-bit. Mikrokontroler yang digunakan adalah mikrokontroler seri AVR dari ATMEL, yaitu ATMEGA32. ATMEGA adalah CMOS mikrokontroler 8-bit daya rendah berdasar pada AVR yang ditingkatkan arsitektur RISC-nya. Dengan mengeksekusi perintah daya tinggi dalam putaran *clock* tunggal, ATMEGA 32 menerima hingga ujung mendekati 1 MIPS per MHz memungkinkan perancang sistem untuk mengoptimalkan penggunaan daya terhadap kecepatan pemrosesan.

Mikrokontroler ATMEGA32 sangat mencukupi sebagai pengendali utama dari robot ini, karena dalam robot ini ATMEGA32 menggunakan kecepatan 16 MIPS (*Millions Instructions Per Second*) yang sangat cepat dalam melakukan perhitungan, selain itu fitur-fitur dari ATMEGA32 sudah sangat mencukupi dalam aplikasi robot dan akuisisi data. Beberapa fitur dari ATMEGA32 adalah:

1. Memori Program (*Flash Memory*) 32 Kb.
2. EEPROM 512 Bytes.
3. SRAM 1Kb.(buat simpan variable)
4. Generator PWM sebanyak 4 *channel*.
5. ADC (*Analog-to-Digital Converter*) 10 bit 8 *channel*.
6. Dua buah *Timer/Counter* yang beresolusi 8 bit.
7. Satu buah *Timer/Counter* yang beresolusi 16 bit.
8. Empat puluh I/O paralel yang *bidirectional*.
9. Vektor *Interrupt* sebanyak 21 buah.
10. *External Interrupt* sebanyak 3 buah.
11. *Analog Comparator*.
12. U(S)ART, SPI, TWI memudahkan dalam proses multi-mikrokontroler.

Konfigurasi pin dari ATMEGA32 yang digunakan dalam robot pemadam api ini ditunjukkan pada Gambar 9.8.



Gambar 9.8 Konfigurasi pin ATMEGA32

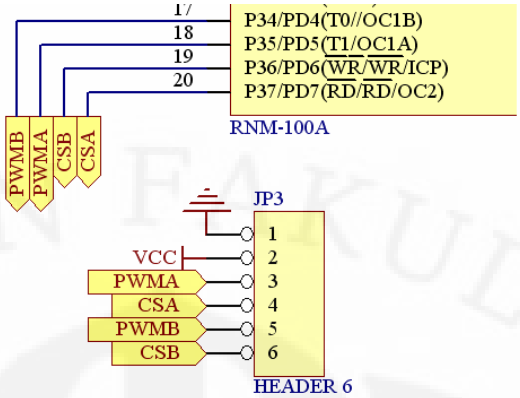
Generator Pulse Width Modulation

Generator PWM berfungsi untuk menghasilkan pulsa PWM yang digunakan untuk mengatur kecepatan putaran motor. PWM sendiri dihasilkan oleh ATMEGA32 di semua *Timer*.

Pada robot ini, pulsa PWM dihasilkan oleh Timer 1 yang mempunyai resolusi 16 bit, tetapi resolusi dari PWM sendiri menggunakan nilai 10 bit, atau mempunyai nilai maksimal 03FF. Pada ATMEGA32, ada beberapa jenis mode PWM tetapi yang digunakan adalah mode PWM "*Phase Correct PWM*" karena mode ini sangat sesuai dengan pengendalian motor [6].

Pengendalian pergerakan robot menggunakan sinyal PWM sebagai pengatur kecepatan putaran motor dan pembalik polaritas untuk mengatur pergerakan motor maju atau mundur untuk tiap motor. Konfigurasi sinyal PWM untuk pergerakan robot yang digunakan adalah:

1. PORTD.5 (OC1A) untuk sinyal PWM motor kanan.
2. PORTD.7 sebagai pembalik polaritas motor kanan.
3. PORTD.4 (OC1B) untuk sinyal PWM motor kiri.
4. PORTD.6 sebagai pembalik polaritas motor kiri.



Gambar 9.9 Skematik dari pengendali motor

Pada PWM konfigurasi mempunyai nilai frekuensi

$$f_{PWM} = \frac{f_{crystal}}{2.N.TOP} \quad \text{.....(8.3)}$$

dimana:

f_{PWM} = frekuensi PWM

$f_{crystal}$ = frekuensi kristal dimana digunakan 11,0592 MHz

N = nilai *prescaler*, dimana digunakan 64

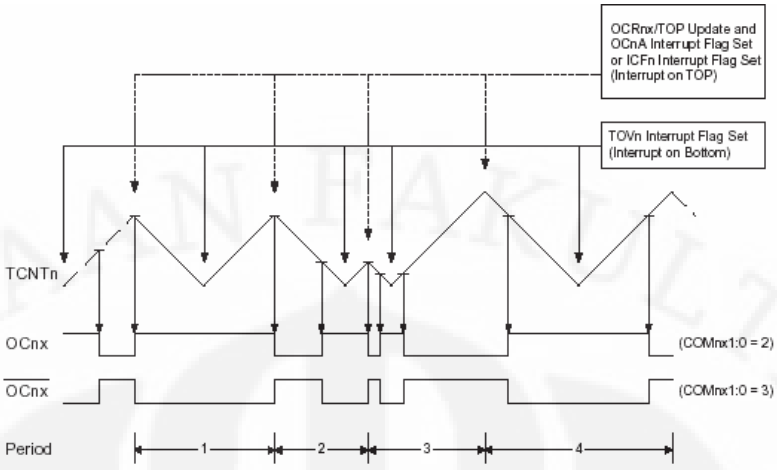
TOP = resolusi PWM, dimana 10 bit = 03FFH = 1023 desimal

Sehingga frekuensi yang digunakan adalah:

$$f_{PWM} = \frac{110592000}{2.64.1023}$$

f_{PWM} = 844,5 Hz.....**(8.4)**

Prinsip kerja pembangkitan sinyal PWM pada *Timer* 1 ATMEGA32 adalah perbandingan nilai antara register TCNT1 (*Timer Counter* 1) yang selalu mencacah dengan nilai OCR1 (*Output Compare Register* 1) yang dikehendaki, yang apabila nilainya sama maka akan menghasilkan pulsa “high” ataupun “low”, dapat dilihat pada Gambar 9.10.



Gambar 9.10 Timing diagram PWM

Berikut ini adalah tabel penggunaan pin-pin dari ATMEGA32 sebagai masukan dan keluaran dalam mengatur fungsi robot secara keseluruhan.

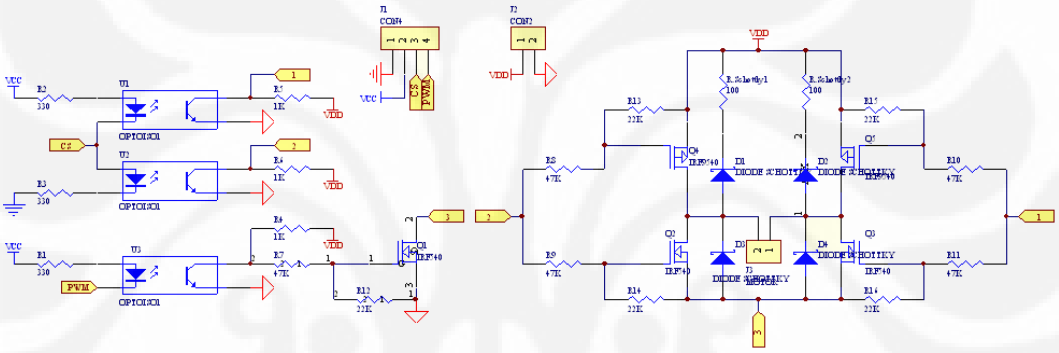
Tabel 9.1Daftar konfigurasi fungsi masukan dan keluaran dari ATMEGA32

NO	Konfigurasi	PORT	Fungsi Yang Digunakan
1	Sensor Kompas	PORTA.0	
2	Sensor jarak kiri belakang	PORTA.1	
3	Sensor jarak kiri depan	PORTA.2	
4	Sensor jarak kanan depan	PORTA.3	
5	Sensor jarak kanan belakang	PORTA.4	
6	Sensor jarak depan	PORTA.5	
7	Sensor jarak belakang	PORTA.6	
8	Motor Air	PORTC.7	
9	Sensor garis	PORTD.2	*External Interrupt* 0
10	Sensor Api (UVTron)	PORTD.3	*External Interrupt* 1
11	PWM motor kanan	PORTD.5	OC1A
12	Pembalik polaritas motor kanan	PORTD.7	
13	PWM motor kiri	PORTD.4	OC1B
14	Pembalik polaritas motor kiri	PORTD.6	

Pengendali Motor DC

Pengendalian motor DC menggunakan teknik PWM, hal ini dikarenakan teknik PWM mudah untuk dibuat, disipasi daya pada sumber arus kecil, serta linearitas kecepatan putaran. Pengendali motor DC pada robot ini dibagi menjadi 3 bagian penting, yaitu: Rangkaian Isolator Optik dan Pembalik Polaritas serta Rangkaian H-Bridge. Sinyal PWM yang digunakan untuk mengendalikan motor dihasilkan oleh mikrokontroler.

Setiap bagian-bagian tersebut terdiri dari 2 buah; satu buah untuk motor kiri dan satu buah untuk motor kanan. Secara lengkap rangkaian ditunjukkan pada Gambar 9.11dibawah ini



Gambar 9.11 Rangkaian isolator optik, pembalik polaritas dan H-Bridge

Rangkaian Isolator Optik

Rangkaian ini merupakan rangkaian yang memisahkan antara catu daya motor dengan catu daya digital; hal ini dikarenakan supaya catu daya motor tidak merusak catu daya digital. Rangkaian ini menggunakan *optocoupler* yang di dalamnya terdapat LED inframerah pada sisi catu daya digital dan fototransistor pada sisi catu daya motor.

Pada Gambar 9.11 dimana pada masukan JP1 terdapat sinyal PWM dari mikrokontroler, pada saat keluaran PWM bernilai “0” maka akan mencatu LED inframerah di *optocoupler* sehingga mengaktifkan fototransistor, sehingga fototransistor menjadi saturasi dan antara sisi kolektor dan emitter menjadi tersambung, sehingga nilai pada kolektor menjadi “0”. Sedangkan pada saat keluaran PWM bernilai “1” maka LED inframerah di *optocoupler* tidak akan

tercatu sehingga fototransistor tetap pada daerah *off* dan antara sisi kolektor dan emitter tidak tersambung, sehingga nilai pada kolektor menjadi $Vc = Vcc\_(I_{R4} * R_4)$. Konfigurasi ini disebut juga dengan "Active High".

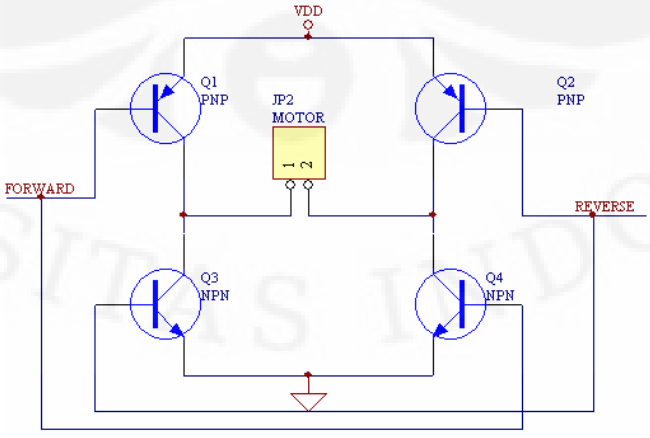
Rangkaian Pembalik Polaritas

Motor DC yang digunakan diharapkan untuk dapat berputar dua arah, sehingga masukan PWM harus diberikan ke 2 masukan dari H-Bridge harus bergantian antara sisi masukan untuk maju dan mundur; sehingga diperlukan adanya rangkaian pembalik polaritas. Rangkaian pembalik polaritas menggunakan *optocoupler* yang di dalamnya terdapat LED inframerah pada sisi catu daya digital dan fototransistor pada sisi catu daya motor. Pada gambar 3.12 diwakili dengan sinyal CS. Dimana cara kerjanya sama dengan sinyal PWM yang telah dijelaskan diatas.

Rangkaian H-Bridge

Untuk menggerakkan motor DC diperlukan sebuah pengendali atau *driver* yang sesuai dengan spesifikasi motor DC yang digunakan. Pengendali motor DC ini dapat mengatur kecepatan motor DC dengan metode PWM dan dapat memutar arah putaran motor DC.

Teknik yang lazim digunakan pada pengendali motor DC adalah dengan menggunakan konfigurasi H-Bridge. Prinsip H-Bridge adalah pencatuan dengan saklar, tetapi karena digunakan PWM maka digunakan saklar elektronik yaitu transistor.



Gambar 9.12 Rangkaian H-Bridge dasar

Pada Gambar 9.12 terlihat bahwa rangkaian H-Bridge terdiri dari 2 sisi, yaitu High Side berupa transistor PNP (Q1 dan Q2) dan Low Side berupa transistor NPN (Q3 dan Q4). Bila mencatu maju maka sisi "FORWARD" diberi nilai "HIGH" maka Q1 akan aktif dan mengalirkan VDD ke motor dan Q4 juga mengalirkan GND ke motor; begitupun sebaliknya.

Untuk meningkatkan realibilitas H-bridge, pada bagian ground pada Gambar 9.12 ditambah transistor yang berfungsi sebagai pengatur PWM menjadi seperti Gambar 9.11 Rangkaian isolator optik, pembalik polaritas dan H-Bridge. Sedangkan 4 transistor lainnya bertindak sebagai pengatur arah. Pada sisi keluaran ke motor harus dipasang dioda, yang disebut dioda Flyback, hal ini bertujuan supaya membuang EMF (*Electromagnetic Field*) yang dihasilkan dari motor yang berputar supaya tidak merusak Rangkaian H-bridge. Dioda yang digunakan adalah dioda dengan kecepatan tinggi, yang biasanya digunakan Dioda Schottky.

Motor DC Sebagai Penggerak

Robot ini menggunakan motor DC sebagai penggerak, motor DC yang digunakan adalah motor pabrikan MAXON dengan tipe A MAX 32 15 watt + spur gearhead GS38,0,10Nm,6:1, dengan catu daya maksimal adalah 48 volt. Pengendalian kecepatan dilakukan oleh mikrokontroler yang menghasilkan PWM dan dikendalikan melalui rangkaian H-Bridge.

Spesifikasi motor DC adalah [7]:

1. Kecepatan putaran : 5860 rpm.
2. Tegangan maksimal : 48 volt.
3. Torsi Maksimum : 117 mNm/A.
4. Arus tanpa beban : 74 mA.
5. Arus Maksimum : 3610 mA.

Motor DC Sebagai Penyemprot Air

Motor yang digunakan merupakan motor DC yang biasa digunakan sebagai penggerak *wiper* pada mobil. Motor DC ini memerlukan catu daya 12 volt.

LAMPIRAN 2

VIDEO MOBILE FIRE FIGHTING ROBOT

Lampiran ini berisi CD mengenai video demo mobile fire fighting robot yang dibuat pada skripsi ini.

[1] Muis, Saludin, ”Teknik Jaringan Syaraf Tiruan”, halaman 3.
[2]Hermawan, Arief, ”Jaringan Syaraf Tiruan Teori Dan Aplikasi”,3-7,11-12,37-38,49-57.
[3]Kusumadewi, Sri, ”Membangun Jaringan Syaraf Tiruan menggunakan MATLAB & EXCEL [LINK”,50-51,94-112.
[4]pING
[5]uvtron
[6]Atmel AVR ATmega16 8-bit microcontroller with 16K Bytes In-System Programmable Flash, Atmel Corporation, 2003. Hal.101-102
[7]maxon

LAMPIRAN 1

PERANGKAT KERAS

MOBILE FIRE FIGHTING ROBOT

Dalam robot pemadam api ini, dimana keseluruhan sistemnya berjalan secara otomatis, membutuhkan sensor-sensor yang mendukung. Robot ini mempunyai tugas untuk mencari api dan mematikan dengan menelusuri lorong-lorong. Beberapa sensor yang digunakan adalah sensor jarak, sensor api, sensor kecepatan atau sensor posisi.

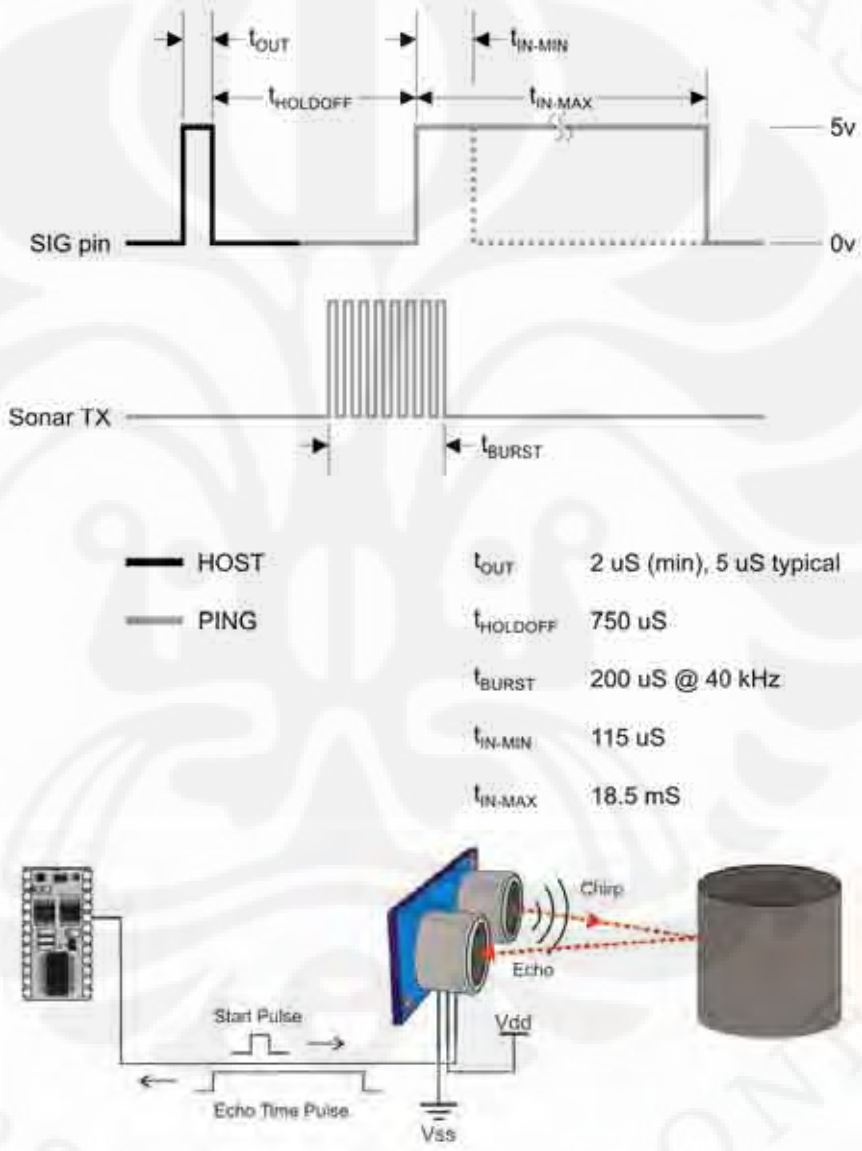
Sensor Jarak

Untuk mengukur jarak antara robot dengan dinding sebagai acuan pergerakan robot, maka digunakan sensor jarak / *ranger* yang menggunakan sinyal ultrasonik. Sensor yang digunakan adalah sensor jarak PING, seperti yang ditunjukkan pada Gambar 2.1 dibawah ini. Sensor PING terdiri dari 2 bagian yaitu speaker dan mikropon.

Gambar 2.1Sensor PING

Sensor PING mempunyai jangkauan pengukuran sebesar 3 centimeter hingga 3 meter.Cara kerja sensor PING [1] dapat di lihat pada Gambar 2.2 dibawah, dimana mikrokontroller mengirimkan sinyal output ke sensor PING agar sensor tersebut mengeluarkan bunyi ultrasonik sebesar 40 kHz melalui speaker.Selanjutnya mikrokontroler akan menerima input dari sensor PING, ketika sensor PING mendeteksi adanya echo melalui bagian mikropon. Hal ini

akan mengubah nilai input mikrokontroler yang semula di set bernilai 1 (sinyal high) menjadi nilai 0 (sinyal low). Dari sini kita dapat menghitung waktu yang dibutuhkan agar nilai input berubah dari 1 menjadi 0. Waktu ini kita gunakan untuk menghitung jarak suatu objek dari sensor menggunakan rumus untuk mengukur kedalaman laut yaitu: S=(V * T) / 2 dimana V= kecepatan bunyi di udara yaitu: 340 m/s.



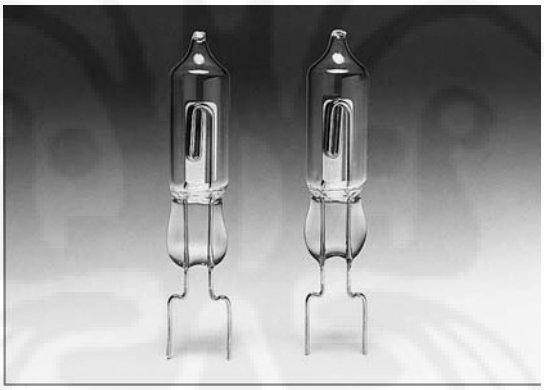
Gambar 2.2 Cara kerja sensor PING

Sensor Api

Sensor Pendeteksi Api yang bernama UVTron mendeteksi adanya gelombang ultraviolet (panjang gelombang 185-260nm) [2] yang diemisikan oleh api. Sensor api yang digunakan merupakan produk dari Hamamatsu. Sensor api ini berguna untuk mendeteksi adanya api dalam suatu daerah atau tidak dan sebagai penentu arah api dengan sedikit modifikasi.

Karakteristik dari UVTron adalah dapat mendeteksi adanya api secara tepat dengan jarak sampai 5m, bahkan pantulan api dari dinding pun bisa dideteksi. UVTron akan menghasilkan pulsa sepanjang 10 mS bila terdapat api dan mempunyai interval 160mS, bila jarak sumber api semakin dekat maka interval akan semakin cepat.

UVTron ini terdiri dari 2 bagian yaitu tabung sensor ultraviolet dan rangkaian pengendali tabung sensor ultraviolet.

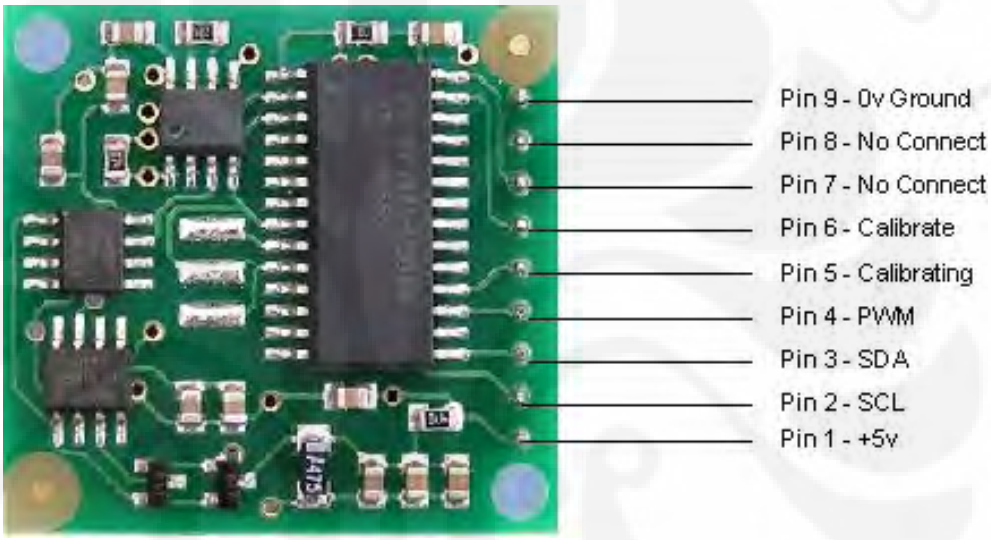


Gambar 2.3 Tabung sensor ultraviolet

Gambar 2.4 Rangkaian pengendali UVTron

Sensor Kompas

Sensor kompas ini digunakan untuk menentukan arah suatu robot terhadap suatu titik referensi tertentu, sehingga dapat digunakan untuk proses berbelok arah pada robot. Sensor ini menggunakan Philips KMZ51 yang merupakan sensor medan magnet, yang cukup sensitif untuk mendeteksi medan magnet bumi. Penggunaan kompas ini dapat dilakukan dengan 2 metode yaitu PWM dan I2C. Pada skripsi ini digunakan metode PWM.



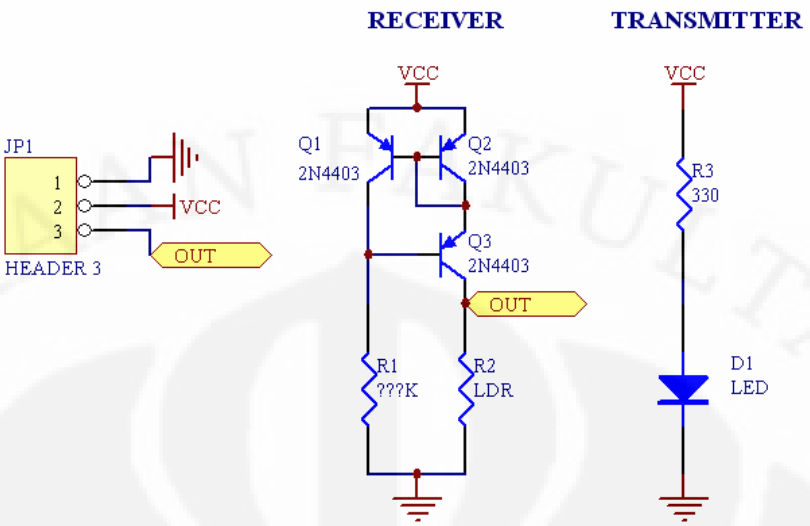
Gambar 2.5 Kompas Devantech

Sensor Garis Putih

Sensor garis putih menggunakan LDR (*Light Dependent Resistor*) sehingga berfungsi seperti sensor cahaya. Sensor terdiri dari 2 bagian, yaitu pemancar berupa LED berwarna putih dan bagian penerima berupa LDR. LDR yang menangkap cahaya pantulan dari LED sehingga nilai resistansinya berubah-berubah, bila semakin terang maka nilai hambatannya semakin kecil.

Konfigurasi LDR adalah dengan pembagi tegangan tetapi respon yang diharapkan sangat lambat dan histeresis terlalu besar lalu ditemukan dengan menggunakan konfigurasi cermin arus (*current mirror*). Konfigurasi cermin arus yang digunakan adalah konfigurasi Wilson (*Wilson Current Mirror*).

Dimana rangkaian cermin arus menginjeksikan sumber arus ke LDR yang dijadikan oleh beban.



Gambar 2.6 Skematik sensor garis dengan cermin arus Wilson

Terlihat pada Gambar 2.6 , bahwa yang dicari adalah nilai R1 berdasarkan dari nilai LDR pada saat gelap dan nilai V_{BE} dari transistor 2N4403. Nilai keluaran dari sensor ini antara gelap dan terang dibuat mempunyai *range* yang jauh, sehingga mempunyai *voltage swing* yang besar.

$$Ip = \frac{Vcc}{R_{LDR}} \quad \text{.....(8.1)}$$

dimana:

Vcc = tegangan rangkaian

Ip = arus yang akan diinjeksikan ke LDR

R_{LDR} = resistansi LDR pada saat gelap.

Sehingga dari arus injeksi (Ip) diatas dapat dicari nilai dari R1 yang sesuai dengan konfigurasi keluaran yang dinginkan.

$$R1 = \frac{(Vcc - V_{BE})}{Ip} \quad \text{.....(8.2)}$$

dimana:

Vcc = tegangan rangkaian

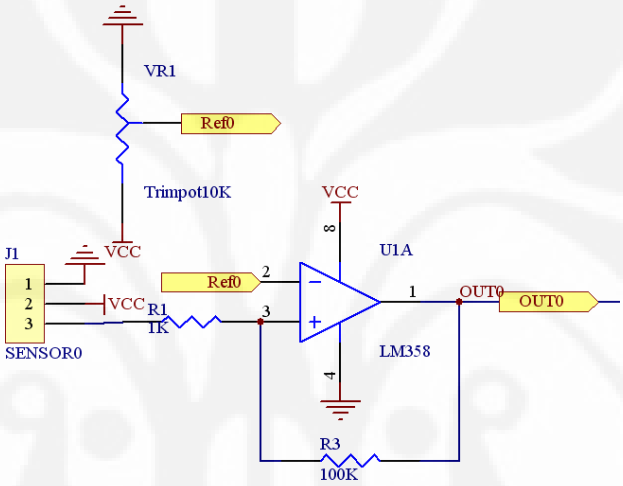
Ip = arus yang akan diinjeksikan ke LDR

V_{BE} = tegangan basis-emiter.

Keluaran dari sensor ini yang berupa tegangan yang akan dibandingkan di komparator analog yang akan menjadikannya suatu keluaran digital.

Komparator Analog

Rangkaian komparator analog berfungsi untuk membandingkan antara dua masukan analog yang berupa tegangan dan mempunyai keluaran berupa logika ”0” dan logika ”1”. Rangkaian komparator analog menggunakan OP-AMP sebagai komponen utamanya.



Gambar 2.7 Skematik dari komparator

Komparator membandingkan masukan antara titik *non-inverting* dan *inverting*, dimana:

1. Bila masukan *non-inverting* > masukan *inverting* maka keluarannya Vcc atau ”1”.
2. Bila masukan *non-inverting* ≤ masukan *inverting* maka keluarannya GND atau ”0”.

Rangkaian komparator yang dibuat untuk robot ini (Gambar 2.7) membandingkan nilai tegangan dari sensor garis yang kemudian dibandingkan dengan nilai referensi yang diatur. Selain membandingkan, rangkaian ini juga mempunyai konfigurasi *hysteresis* , yaitu R3 dan R1, yang berfungsi untuk memperkecil daerah *hysteresis* sehingga nilai logika lebih cepat dicapai.

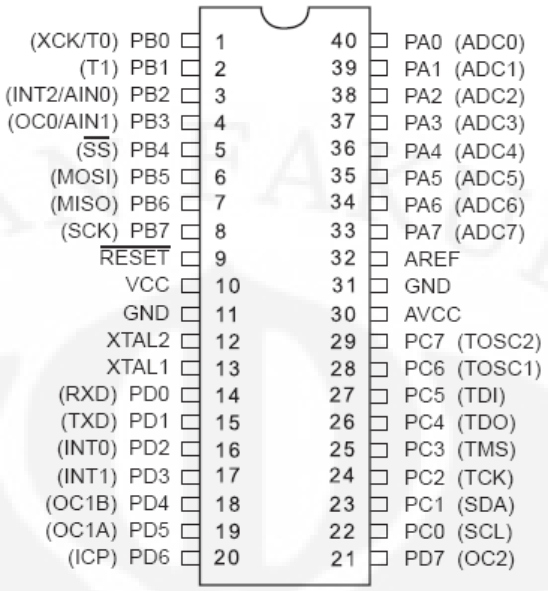
Mikrokontroller ATMEGA 32

Proses pengendalian robot ini menggunakan mikrokontroler 8-bit. Mikrokontroler yang digunakan adalah mikrokontroler seri AVR dari ATMEL, yaitu ATMEGA32. ATMEGA adalah CMOS mikrokontroler 8-bit daya rendah berdasar pada AVR yang ditingkatkan arsitektur RISC-nya. Dengan mengeksekusi perintah daya tinggi dalam putaran *clock* tunggal, ATMEGA 32 menerima hingga ujung mendekati 1 MIPS per MHz memungkinkan perancang sistem untuk mengoptimalkan penggunaan daya terhadap kecepatan pemrosesan.

Mikrokontroler ATMEGA32 sangat mencukupi sebagai pengendali utama dari robot ini, karena dalam robot ini ATMEGA32 menggunakan kecepatan 16 MIPS (*Millions Instructions Per Second*) yang sangat cepat dalam melakukan perhitungan, selain itu fitur-fitur dari ATMEGA32 sudah sangat mencukupi dalam aplikasi robot dan akuisisi data. Beberapa fitur dari ATMEGA32 adalah:

1. Memori Program (*Flash Memory*) 32 Kb.
2. EEPROM 512 Bytes.
3. SRAM 1Kb.(buat simpan variable)
4. Generator PWM sebanyak 4 *channel*.
5. ADC (*Analog-to-Digital Converter*) 10 bit 8 *channel*.
6. Dua buah *Timer/Counter* yang beresolusi 8 bit.
7. Satu buah *Timer/Counter* yang beresolusi 16 bit.
8. Empat puluh I/O paralel yang *bidirectional*.
9. Vektor *Interrupt* sebanyak 21 buah.
10. *External Interrupt* sebanyak 3 buah.
11. *Analog Comparator*.
12. U(S)ART, SPI, TWI memudahkan dalam proses multi-mikrokontroler.

Konfigurasi pin dari ATMEGA32 yang digunakan dalam robot pemadam api ini ditunjukkan pada Gambar 2.8.



Gambar 2.8 Konfigurasi pin ATMEGA32

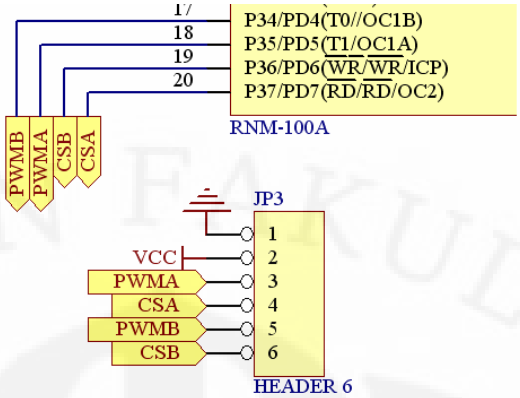
Generator Pulse Width Modulation

Generator PWM berfungsi untuk menghasilkan pulsa PWM yang digunakan untuk mengatur kecepatan putaran motor. PWM sendiri dihasilkan oleh ATMEGA32 di semua *Timer*.

Pada robot ini, pulsa PWM dihasilkan oleh Timer 1 yang mempunyai resolusi 16 bit, tetapi resolusi dari PWM sendiri menggunakan nilai 10 bit, atau mempunyai nilai maksimal 03FF. Pada ATMEGA32, ada beberapa jenis mode PWM tetapi yang digunakan adalah mode PWM "*Phase Correct PWM*" karena mode ini sangat sesuai dengan pengendalian motor [3].

Pengendalian pergerakan robot menggunakan sinyal PWM sebagai pengatur kecepatan putaran motor dan pembalik polaritas untuk mengatur pergerakan motor maju atau mundur untuk tiap motor. Konfigurasi sinyal PWM untuk pergerakan robot yang digunakan adalah:

1. PORTD.5 (OC1A) untuk sinyal PWM motor kanan.
2. PORTD.7 sebagai pembalik polaritas motor kanan.
3. PORTD.4 (OC1B) untuk sinyal PWM motor kiri.
4. PORTD.6 sebagai pembalik polaritas motor kiri.



Gambar 2.9 Skematik dari pengendali motor

Pada PWM konfigurasi mempunyai nilai frekuensi

$$f_{PWM} = \frac{f_{crystal}}{2.N.TOP} \quad \text{.....(8.3)}$$

dimana:

f_{PWM} = frekuensi PWM

$f_{crystal}$ = frekuensi kristal dimana digunakan 11,0592 MHz

N = nilai *prescaler*, dimana digunakan 64

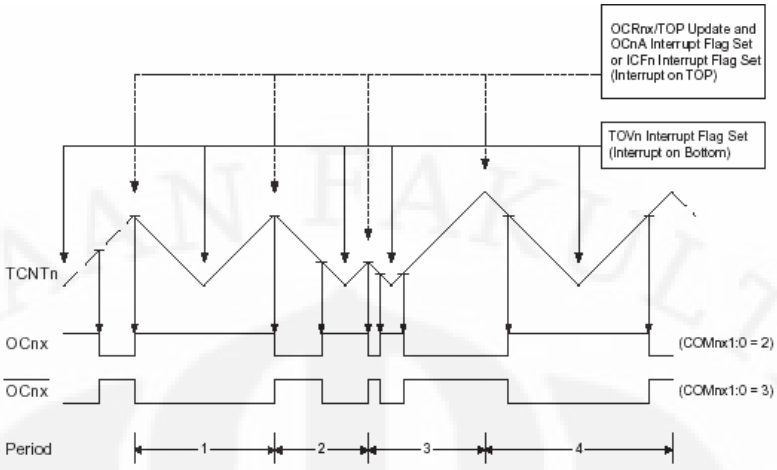
TOP = resolusi PWM, dimana 10 bit = 03FFH = 1023 desimal

Sehingga frekuensi yang digunakan adalah:

$$f_{PWM} = \frac{110592000}{2.64.1023}$$

f_{PWM} = 844,5 Hz.....**(8.4)**

Prinsip kerja pembangkitan sinyal PWM pada *Timer* 1 ATMEGA32 adalah perbandingan nilai antara register TCNT1 (*Timer Counter* 1) yang selalu mencacah dengan nilai OCR1 (*Output Compare Register* 1) yang dikehendaki, yang apabila nilainya sama maka akan menghasilkan pulsa “high” ataupun “low”, dapat dilihat pada Gambar 2.10.



Gambar 2.10 Timing diagram PWM

Berikut ini adalah tabel penggunaan pin-pin dari ATMEGA32 sebagai masukan dan keluaran dalam mengatur fungsi robot secara keseluruhan.

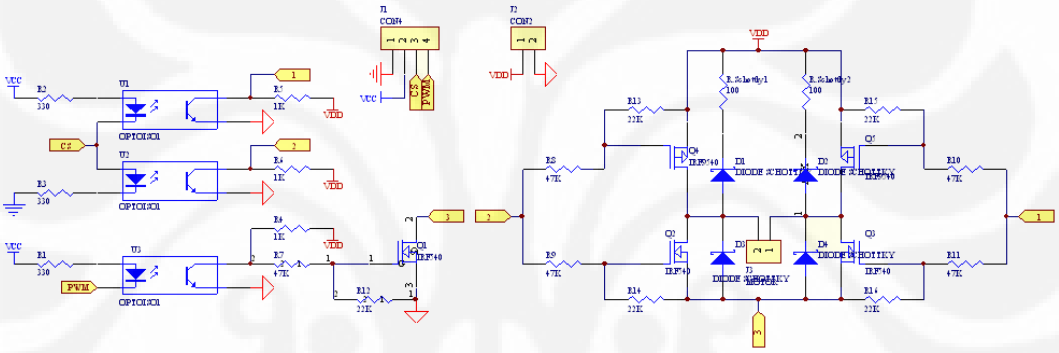
Tabel 2.1Daftar konfigurasi fungsi masukan dan keluaran dari ATMEGA32

NO	Konfigurasi	PORT	Fungsi Yang Digunakan
1	Sensor Kompas	PORTA.0	
2	Sensor jarak kiri belakang	PORTA.1	
3	Sensor jarak kiri depan	PORTA.2	
4	Sensor jarak kanan depan	PORTA.3	
5	Sensor jarak kanan belakang	PORTA.4	
6	Sensor jarak depan	PORTA.5	
7	Sensor jarak belakang	PORTA.6	
8	Motor Air	PORTC.7	
9	Sensor garis	PORTD.2	*External Interrupt* 0
10	Sensor Api (UVTron)	PORTD.3	*External Interrupt* 1
11	PWM motor kanan	PORTD.5	OC1A
12	Pembalik polaritas motor kanan	PORTD.7	
13	PWM motor kiri	PORTD.4	OC1B
14	Pembalik polaritas motor kiri	PORTD.6	

Pengendali Motor DC

Pengendalian motor DC menggunakan teknik PWM, hal ini dikarenakan teknik PWM mudah untuk dibuat, disipasi daya pada sumber arus kecil, serta linearitas kecepatan putaran. Pengendali motor DC pada robot ini dibagi menjadi 3 bagian penting, yaitu: Rangkaian Isolator Optik dan Pembalik Polaritas serta Rangkaian H-Bridge. Sinyal PWM yang digunakan untuk mengendalikan motor dihasilkan oleh mikrokontroler.

Setiap bagian-bagian tersebut terdiri dari 2 buah; satu buah untuk motor kiri dan satu buah untuk motor kanan. Secara lengkap rangkaian ditunjukkan pada Gambar 2.11dibawah ini



Gambar 2.11 Rangkaian isolator optik, pembalik polaritas dan H-Bridge

Rangkaian Isolator Optik

Rangkaian ini merupakan rangkaian yang memisahkan antara catu daya motor dengan catu daya digital; hal ini dikarenakan supaya catu daya motor tidak merusak catu daya digital. Rangkaian ini menggunakan *optocoupler* yang di dalamnya terdapat LED inframerah pada sisi catu daya digital dan fototransistor pada sisi catu daya motor.

Pada Gambar 2.11 dimana pada masukan JP1 terdapat sinyal PWM dari mikrokontroler, pada saat keluaran PWM bernilai “0” maka akan mencatu LED inframerah di *optocoupler* sehingga mengaktifkan fototransistor, sehingga fototransistor menjadi saturasi dan antara sisi kolektor dan emitter menjadi tersambung, sehingga nilai pada kolektor menjadi “0”. Sedangkan pada saat keluaran PWM bernilai “1” maka LED inframerah di *optocoupler* tidak akan

tercatu sehingga fototransistor tetap pada daerah *off* dan antara sisi kolektor dan emitter tidak tersambung, sehingga nilai pada kolektor menjadi $Vc = Vcc\_(I_{R4} * R_4)$. Konfigurasi ini disebut juga dengan "Active High".

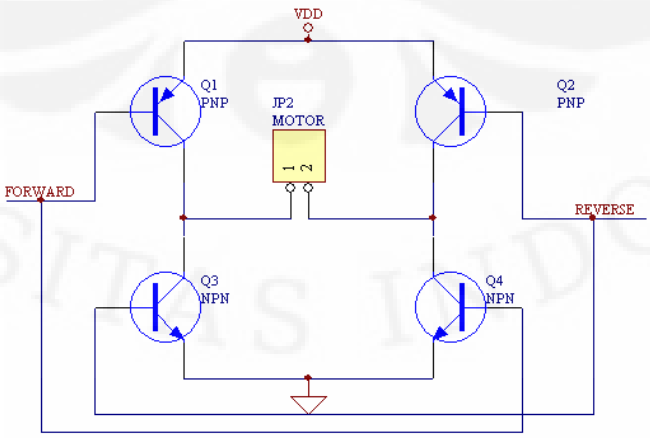
Rangkaian Pembalik Polaritas

Motor DC yang digunakan diharapkan untuk dapat berputar dua arah, sehingga masukan PWM harus diberikan ke 2 masukan dari H-Bridge harus bergantian antara sisi masukan untuk maju dan mundur; sehingga diperlukan adanya rangkaian pembalik polaritas. Rangkaian pembalik polaritas menggunakan *optocoupler* yang di dalamnya terdapat LED inframerah pada sisi catu daya digital dan fototransistor pada sisi catu daya motor. Pada gambar 3.12 diwakili dengan sinyal CS. Dimana cara kerjanya sama dengan sinyal PWM yang telah dijelaskan diatas.

Rangkaian H-Bridge

Untuk menggerakkan motor DC diperlukan sebuah pengendali atau *driver* yang sesuai dengan spesifikasi motor DC yang digunakan. Pengendali motor DC ini dapat mengatur kecepatan motor DC dengan metode PWM dan dapat memutar arah putaran motor DC.

Teknik yang lazim digunakan pada pengendali motor DC adalah dengan menggunakan konfigurasi H-Bridge. Prinsip H-Bridge adalah pencatuan dengan saklar, tetapi karena digunakan PWM maka digunakan saklar elektronik yaitu transistor.



Gambar 2.12 Rangkaian H-Bridge dasar

Pada Gambar 2.12 terlihat bahwa rangkaian H-Bridge terdiri dari 2 sisi, yaitu High Side berupa transistor PNP (Q1 dan Q2) dan Low Side berupa transistor NPN (Q3 dan Q4). Bila mencatu maju maka sisi "FORWARD" diberi nilai "HIGH" maka Q1 akan aktif dan mengalirkan VDD ke motor dan Q4 juga mengalirkan GND ke motor; begitupun sebaliknya.

Untuk meningkatkan realibilitas H-bridge, pada bagian ground pada Gambar 2.12 ditambah transistor yang berfungsi sebagai pengatur PWM menjadi seperti Gambar 2.11 Rangkaian isolator optik, pembalik polaritas dan H-Bridge. Sedangkan 4 transistor lainnya bertindak sebagai pengatur arah. Pada sisi keluaran ke motor harus dipasang dioda, yang disebut dioda Flyback, hal ini bertujuan supaya membuang EMF (*Electromagnetic Field*) yang dihasilkan dari motor yang berputar supaya tidak merusak Rangkaian H-bridge. Dioda yang digunakan adalah dioda dengan kecepatan tinggi, yang biasanya digunakan Dioda Schottky.

Motor DC Sebagai Penggerak

Robot ini menggunakan motor DC sebagai penggerak, motor DC yang digunakan adalah motor pabrikan MAXON dengan tipe A MAX 32 15 watt + spur gearhead GS38,0,10Nm,6:1, dengan catu daya maksimal adalah 48 volt. Pengendalian kecepatan dilakukan oleh mikrokontroler yang menghasilkan PWM dan dikendalikan melalui rangkaian H-Bridge.

Spesifikasi motor DC adalah [4]:

1. Kecepatan putaran : 5860 rpm.
2. Tegangan maksimal : 48 volt.
3. Torsi Maksimum : 117 mNm/A.
4. Arus tanpa beban : 74 mA.
5. Arus Maksimum : 3610 mA.

Motor DC Sebagai Penyemprot Air

Motor yang digunakan merupakan motor DC yang biasa digunakan sebagai penggerak *wiper* pada mobil. Motor DC ini memerlukan catu daya 12 volt.

LAMPIRAN 2

VIDEO MOBILE FIRE FIGHTING ROBOT

Lampiran ini berisi CD mengenai video demo mobile fire fighting robot yang dibuat pada skripsi ini.

[1]pING
[2]uvtron
[3]Atmel AVR ATmega16 8-bit microcontroller with 16K Bytes In-System Programmable Flash, Atmel Corporation, 2003. Hal.101-102
[4]maxon