PENCARIAN JALUR TERPENDEK UNTUK ROBOT M*ICROMOUSE* DENGAN MENGGUNAKAN ALGORITMA *BACKTRACKING*

Rusmini[1,2], Setiawardhana, M. Iqbal Nugraha
[1]Jurusan Teknik Mekatronika
Politeknik Elektronika Negeri Surabaya-Institut Teknologi Sepuluh Nopember
Jln raya ITS Sukolilo, surabaya 60111, Tel: (031) 5947280, Fax: (031)594 6114
[2]Jurusan Teknik Elektronika
Politeknik Manufaktur Negeri Bangka Belitung
Jln. Timah Raya Kawasan Industri Air KantungSungailiat-Bangka 33211, Tel: 0717-93586, 95252, Fax. 0717-93585
nde_mini@yahoo.com
setiadhana@gmail.com
iqbal_nugrah@yahoo.com

.

Abstrak

Robot Micromouse adalah robot cerdas yang dapat bergerak bebas di dalam sebuah labirin (maze) tanpa menyentuh objek sekitarnya, robot mengetahui ke arah mana harus bergerak, berapa derajat harus berputar jika menemui jalan buntu pada area labirin. Robot micromouse ini termasuk kedalam jenis Robot Mobile yaitu *Autonomous Mobile Robot* dimana pengendalian gerakan dari robot yang berdasarkan program kemudi yang diberikan sehingga seolah -olah robot tersebut bergerak sendiri. Arah pergerakan mobile robot ini ditentukan ketika ada respon terhadap obyek di depan, kanan dan kiri. Robot ini dibuat dengan mikrokontroller ATMEGA 8535 sebagai pengendali, sensor inframerah GP2D12 untuk mendeteksi adanya tembok atau tidak adanya tembok dan driver motor untuk menggerakkan motor sebagai aktuator. Robot ini menggunakan algoritma *backtracking* untuk mencari jalan terpendek dalam sebuah labirin ke tempat yang dituju.

Kata kunci : robot *micromouse*, algoritma *backtracking*, mikrokontroller ATMEGA 8535.

I. PENDAHULUAN

Algoritma *backtracking* (runut balik) merupakan salah satu metode yang digunakan untuk mencari jalan keluar dari suatu labirin, dengan metode ini robot *micromouse* dapat menjelajahi labirin dan menemukan jalan terpendek menuju *finish.* Algoritma *backtracking* pada dasarnya mencari semua kemungkinan solusi yang dibuat dalam bentuk pohon terlebih dahulu baru kemudian pohon tersebut di jelajahi (*explore)* secara *DFS (Depth Field Search)* sampai ditemukan solusi yang layak. Dalam mencari jalan keluar dari suatu labirin dimana *field* yang dibentuk dapat direpresentasikan dalam bentuk biner dan pada setiap petak maksimal terdapat 4 kemungkinan, yaitu : atas, kanan, bawah dan kiri. Untuk masalah ini biasanya solusi pertama yang ditemukan bukanlah solusi yang optimal sehingga untuk mendapatkan hasil yang optimal dibutuhkan pencarian terhadap semua kemungkinan solusi.

II. TINJAUAN PUSTAKA

A. Robot *Micromouse*

Robot *Micromouse* adalah robot cerdas yang dapat bergerak bebas di dalam sebuah labirin *(maze)* tanpa menyentuh obyek sekitarnya, robot mengetahui ke arah mana harus bergerak, berapa derajat harus berputar jika menemui jalan buntu pada area labirin. Robot micromouse ini termasuk ke dalam jenis robot *mobile* yaitu *Autonomous Mobile Robot*. *Autonomous Mobile Robot* adalah pengendalian gerakan dari robot yang berdasarkan program kemudi yang diberikan sehingga seolah -olah robot tersebut bergerak sendiri. Robot *micromouse* ini akan berjalan dalam labirin yang dimulai dari posisi start pada bagian sudut labirin. Kemudian robot akan bergerak menuju finish yang berada di tengah labirin. Setelah mencapai *finish* maka robot akan kembali ke posisi start.

B. Algoritma *Backtracking*

Algoritma *backtracking* pertama kali diperkenalkan oleh D.H. Lehmer pada tahun 1950. Dalam perkembangannya beberapa ahli seperti RJ Walker, Golomb, dan Baumert menyajikan uraian umum tentang *backtracking* dan penerapannya dalam berbagai persoalan dan aplikasi. Runut-balik merupakan perbaikan dari algoritma *brute-force*, secara sistematis mencari solusi persoalan di antara semua kemungkinan yang ada. Hanya pencarian yang mengarah ke solusi saja yang dikembangkan, sehingga waktu pencarian dapat dihemat. Runut-balik lebih alami dinyatakan dalam algoritma rekursif. Algoritma *backtracking* (runut balik)

ISBN: 978-979-8689-14-7

merupakan salah satu metode pemecahan masalah yang termasuk dalam strategi yang berbasis pencarian pada ruang status. Algoritma *backtracking* melakukan pencarian solusi persoalan secara sistematis pada semua kemungkinan solusi yang ada. Oleh karena algoritma ini berbasis pada algoritma *Depth-First Search (DFS)*, maka pencarian solusi dilakukan dengan menelusuri suatu struktur berbentuk pohon berakar secara preorder. Proses ini dicirikan dengan ekspansi simpul terdalam lebih dahulu sampai tidak ditemukan lagi suksesor dari suatu simpul.

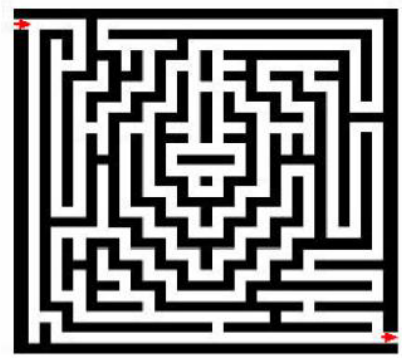
2.1.1 Prinsip Pencarian Solusi dengan Metode Runut-Balik

Langkah-langkah pencarian solusi pada pohon ruang status yang dibangun secara dinamis :

1. Solusi dicari dengan membentuk lintasan dari akar ke daun. Aturan pembentukan yang dipakai adalah mengikuti metode pencarian mendalam (DFS). Simpulsimpul yang sudah dilahirkan dinamakan **simpul hidup** (*live node*). Simpul hidup yang sedang diperluas dinamakan **simpul-E** (*Expand-node*). Simpul dinomori dari atas ke bawah sesuai dengan urutan kelahirannya.
2. Tiap kali simpul-E diperluas, lintasan yang dibangun olehnya bertambah panjang. Jika lintasan yang sedang dibentuk tidak mengarah ke solusi maka simpul-E tersebut "dibunuh" sehingga menjadi **simpul mati** (*dead node*). Fungsi yang digunakan untuk membunuh simpul-E adalah dengan menerapkan **fungsi pembatas** (*bounding function*). Simpul yang sudah mati tidak akan pernah diperluas lagi.
3. Jika pembentukan lintasan berakhir dengan simpul mati, maka proses pencarian diteruskan dengan membangkitkan simpul anak yang lainnya. Bila tidak ada lagi simpul anak yang dapat dibangkitkan, maka pencarian solusi dilanjutkan dengan melakukan runutbalik ke simpul hidup terdekat (simpul orang tua). Selanjutnya simpul ini menjadi simpul-E yang baru. Lintasan baru dibangun kembali sampai lintasan tersebut membentuk solusi.
4. Pencarian dihentikan bila kita telah menemukan solusi atau tidak ada lagi simpul hidup untuk runut-balik.

2.1.2 Penerapan Algoritma *Backtracking* dalam menyelesaikan Maze Labirin

Algoritma yang tepat untuk menemukan jalan keluar dari dalam labirin adalah algoritma runut-balik. Dengan algoritma ini, kita mencoba sebuah lintasan hingga menemui jalan buntu, lalu jejaki, (retrace) langkah
sebelumnya sampai kita menemukan lintasan yang lain, lalu ulangi lagi lintasan tersebut. Pada akhirnya kita akan menemukan lintasan yang mengarah ke pintu keluar, atau mencoba semua lintasan dan memutuskan tidak terdapat solusinya. Untuk menggambarkan algoritma runut-balik secara rinci, bagi lintasan menjadi sederetan langkah. Sebuah langkah terdiri dari pergerakan satu unit sel pada arah tertentu. Arah yang mungkin: ke atas (*up*), ke bawah (*down*), ke kiri (*left*), ke kanan (*right*). Pada gambar 1 adalah contoh sebuah labirin.

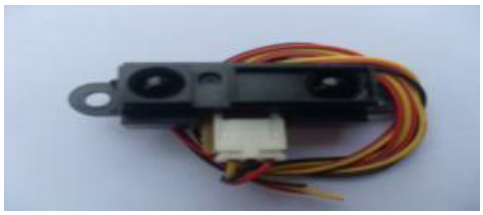


Gambar 1 Contoh Sebuah Labirin

Sumber : (Andika Pratama, "Analisis Penerapan Algoritma *Backtracking* Pada Pencarian Jalan Keluar di Dalam Labirin", Makalah IF2251, Bandung, 2007. Hal 2)

C. Sensor Jarak SHARP GP2D12

Sharp GP2D12 adalah sensor jarak analog yang menggunakan infrared untuk mendeteksi jarak antara 10 cm sampai 80 cm. GP2D12 mengeluarkan output voltase non-linear dalam hubungannya dengan jarak objek dari sensor dan menggunakan *interface analog to digital converter* (ADC).



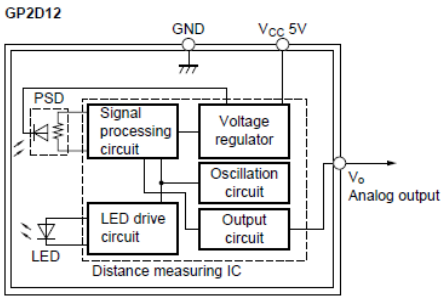
Gambar 2 Bentuk Fisik GP2D12

Beberapa karakteristik dari sensor jarak GP2D12 adalah :

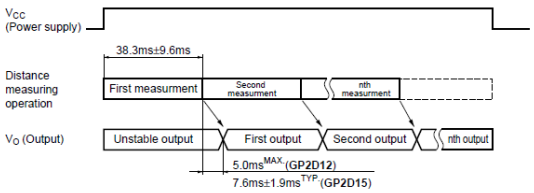
1. *Power supply* 4,5 - 5,5 Volt.
2. *Output* berupa tegangan analog yang berkisar antara 0,4 – 2,5 Volt.

3. Pembacaan jarak tidak begitu dipengaruhi oleh warna obyek yang diukur
4. Dapat mendeteksi obyek dengan jarak berkisar antara 8 cm – 80 cm
5. Tidak membutuhkan rangkain kontrol eksternal
6. Tidak begitu dipengaruhi oleh kondisi pencahayaan ruangan

Prinsip kerja sensor inframerah ini dalam mengukur jarak berbeda dengan sensor ultrasonik yang menggunakan waktu pantul gelombang bunyi karena waktu pantul cahaya jelas terlalu singkat untuk dapat diukur. Cahaya inframerah dengan frekuensi 40 kHz dipancarkan dan hasil pantulannya diterima oleh susunan detektor inframerah. Sudut pantulan sinar inframerah akan berubah sesuai jarak sensor dan obyek.

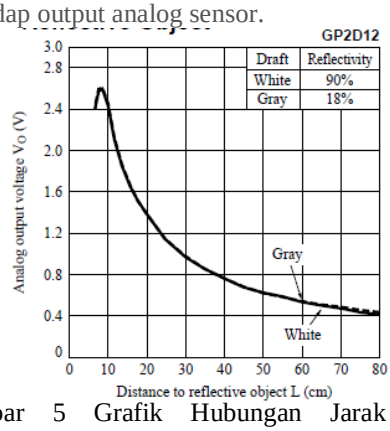


Gambar 3 Internal Blok Diagram GP2D12



Gambar 4 *Timing Chart* GP2D12

Berikut hubungan antara jarak deteksi objek terhadap output analog sensor.



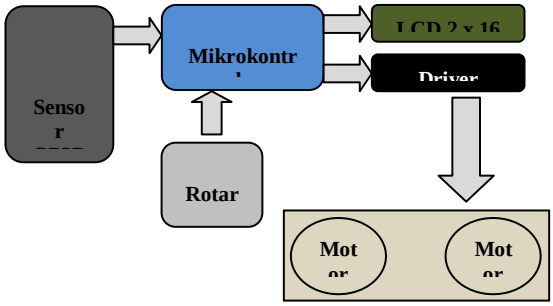
Gambar 5 Grafik Hubungan Jarak dengan Tegangan *Output*

III. PERANCANGAN SISTEM

A. Konfigurasi Sistem

Secara umum, penelitian ini terdiri dari perancangan dan pembuatan robot *micromouse* serta perancangan dan pembuatan algoritma *backtracking* untuk menemukan jalan terpendek dari suatu labirin.

Dari segi mekanik, robot ini memilii empat buah roda yaitu dua buah roda di belakang yang berfungsi sebagai penggerak, satu buah roda di depan sebagai roda bebas dan satu buah roda ditengah yang berfungsi sebagai *rotary encoder*.



Gambar 6. Rancangan Sistem

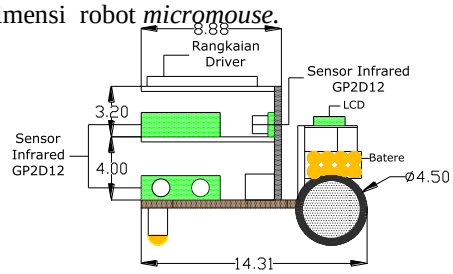
B. Perancangan dan Pembuatan Perangkat Keras

Pada perancangan perangkat keras untuk robot micromouse ini, terbagi dalam dua hal pokok yang harus dikerjakan :

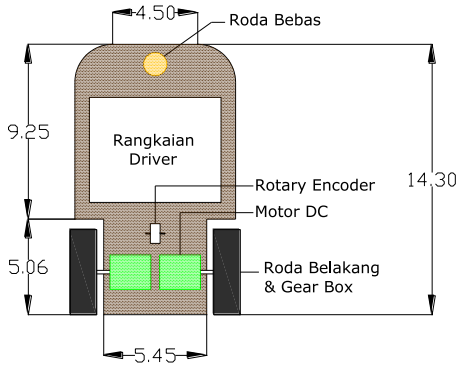
1. Konstruksi mekanik robot, yang meliputi *body* robot, motor DC dan roda.
2. Rangkaian elektronika, yang meliputi rangkaian sistem minimum ATMega 8535, rangkaian sensor inframerah, rangkain driver motor dengan menggunakan L293D, *rotary encoder* dan LCD.

C. Perancangan dan Pembuatan Konstruksi Mekanik Robot

Konstruksi mekanik robot ini meliputi *body robot* yang dibuat dengan menggunakan papan PCB *double layer* dengan ketebalan sebesar 2mm. Pada bagian penggerak, motor yang digunakan adalah motor DC yang dilengkapi dengan *gearbox* metal. Gambar 7 menunjukkan dimensi robot *micromouse*.



(a) Tampak Samping

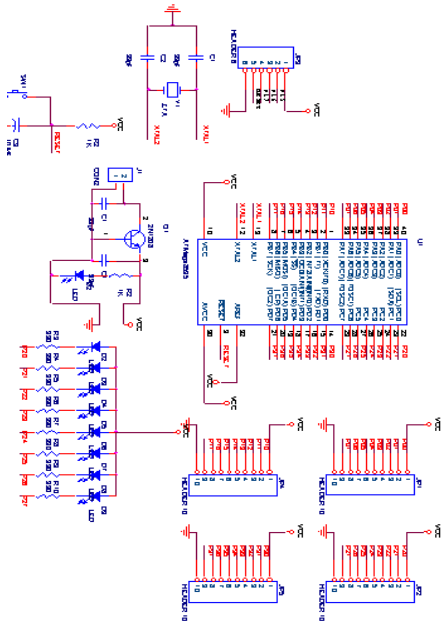


(b) Tampak Atas

Gambar 7 Dimensi Robot

D. Perancangan dan Pembuatan Mikrokontroler

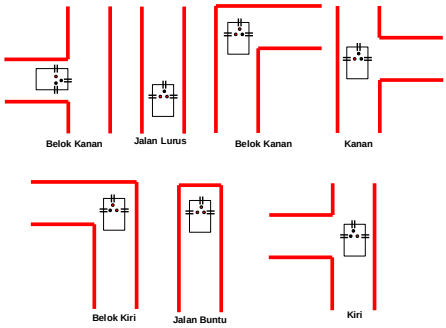
Dalam membuat rangkaian mikrokontroler diperlukan pemahaman mengenai sistem minimum dari mikrokontroler yang akan dirancang itu sendiri. Sistem rangkaian yang dirancang diusahakan menggunakan rangkaian yang seringkas mungkin dan dengan pengkabelan yang baik, karena biasanya rangkaian tersebut bekerja dengan frekuensi yang relatif tinggi, sehingga peka terhadap *noise* dari luar. Gambar 8 merupakan sistem minimum yang digunakan untuk robot *micromouse.* Sistem minimum ini bekerja pada tegangan 5 V yang diperoleh dari IC *regulator* 7805. Sistem minimum ini juga dilengkapi delapan buah *LED* yang dihubungkan ke PORT C dan dua buah interupsi eksternal.



Gambar 8. Rangkaian Sistem Minimum

E. Perancangan dan Pembuatan Sensor

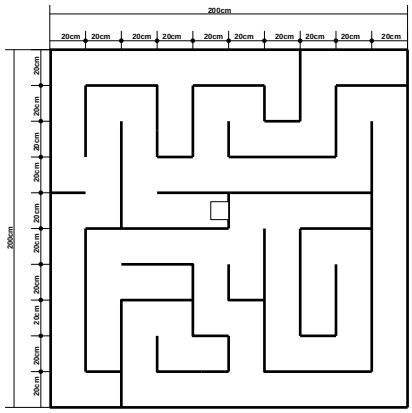
Pada penelitian ini, digunakan tiga buah sensor *infrared* GP2D12 sebagai pendeteksi ada atau tidaknya dinding. Ketiga sensor itu diletakkan satu buah di sebelah kiri, satu buah di sebelah kanan dan satu buah di depan.



Gambar 9. Konfigurasi Sensor pada Robot

F. Perancangan Algoritma *Backtracking*

Rancangan labirin yang dibuat adalah tampak seperti pada gambar 3.9. ukuran labirin adalah 200cm x 200cm. Labirin tersebut dibagi menjadi 100 kotak, sehingga ukuran masing-masing kotak adalah 20cm x 20cm. Berikut adalah rancangan labirin yang dibuat.



Gambar 10 Labirin ukuran 200cm x 200cm

Warna dinding adalah putih dan *background* berwarna hitam. Untuk posisi *start* bisa diletakkan pada keempat sudut dinding labirin. Sedangkan posisi *finish* berada ditengah labirin. Berdasarkan bentuk labirin tersebut, hanya ada satu jalan keluar dari *start* menuju *finish* sehingga robot harus menemukan jalan keluar tersebut dengan cara melakukan pencarian atau menelusuri seluruh labirin (*search mode)* dan kembali menuju *start* dengan menggunakan jalur terpendek hasil dari pemetaan yang telah dilakukan pada mode pencarian. Mode ini bisa dikatakan *return mode.*

Prinsip Kerja *Search Mode*

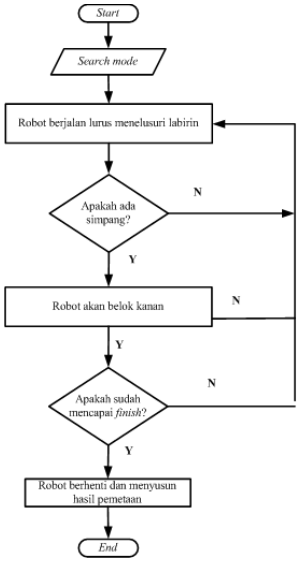
Mode pencarian *(Search mode)* adalah proses untuk menemukan posisi *finish* dengan cara

menelusuri labirin. Pada proses ini, robot akan bergerak secara berurutan dengan cara mengutamakan belok kanan bila menemukan persimpangan. Bila didalam perjalanan robot tidak menemukan belok kanan maka robot akan lebih memilih jalan lurus. Dan bila robot tidak menemukan juga jalan lurus maka robot akan belok kiri.

Mekanisme mode pencarian (*search mode)*

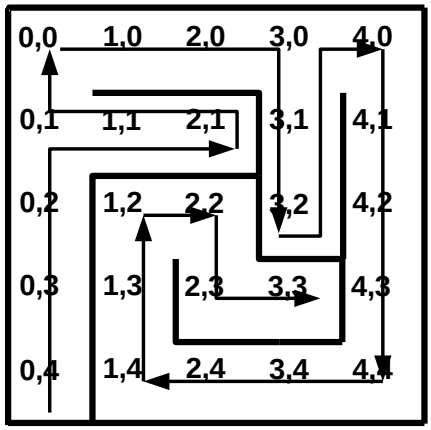
Ketika robot sudah dijalankan, maka robot akan berjalan menelusuri labirin dengan prioritas belok kanan apabila bertemu simpang. Jika bertemu dengan jalan buntu, maka robot berputar 180 derajat dan jalan buntu tersebut diberi tanda 0. Kemudian robot akan mencoba arah yang lain pada simpang yang sama sebelumnya. Ini dilakukan terus sampai posisi *finish* tercapai. Jika robot sudah mencapai *finish*, robot akan kembali menuju *start* dengan jalur terpendek.

Pada proses ini, robot bisa dikatakan dalam tahap *learning.* Robot pada tahap ini akan memberikan kode-kode setiap persimpangan yang dijumpainya. Kode-kode ini kemudian disimpan dalam *memory* mikrokontroler. Kode-kode yang tersusun dari awal *start* sampai dengan *finish* ini akan menjadi data-data yang penting bagi robot untuk memutuskan jalan keluar yang diharapkan. Diagram alir (*flowchart*) mekanisme *mode* ini dapat dilihat pada gambar 11. Pada mode ini, dimulai dari *start* kemudian robot akan melakukan pencarian jalan dan berjalan lurus. Bila ada persimpangan robot lebih mengutamakan belok kanan dan jika tidak ada persimpangan maka robot akan tetap jalan lurus. Dalam perjalanan robot akan terus mencari tempat tujuan atau *finish*. Bila robot sudah menemukan titik yang diinginkan maka robot akan berhenti dan melakukan pemetaan hasil dari proses pencarian jalan tadi.



Gambar 11 Diagram Alir Mode Pencarian

Pada gambar diatas dapat dilustrasikan proses pencarian jalurnya sebagai berkut : robot lebih mengutamakan belok kanan bila menemukan persimpangan. Gambar 12 menunjukkan ilustrasi pergerakan robot dalam mode pencarian.

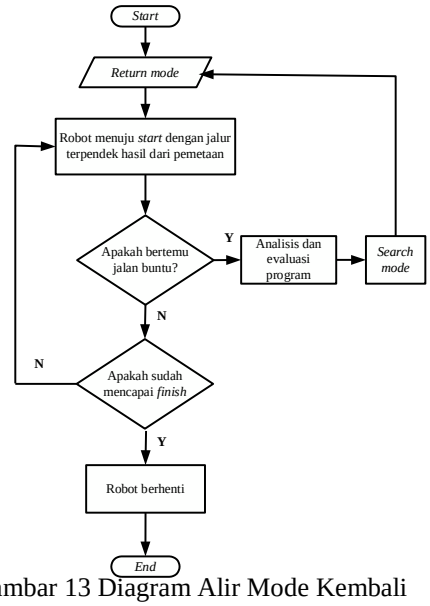


Gambar 12. Ilustrasi Pergerakan Robot pada Mode Pencarian

Pada gambar diatas, misalkan *start* pada titik (0,4) dan *finish* (3,3). Pertama robot akan bergerak ke atas atau luru dari titik (0,4) sampai dengan titik (0,1). Pada titik (0,1) robot menemukan persimpangan dan robot akan menuju titik (1,1) karena disini robot lebih mengutamkan belok kanan. Robot akan bergerak terus hingga menuju *finish* yaitu titik (3,3). Bila menemukan jalan buntu robot akan berputar 180 derajat dan pada jalan buntu tersebut ditandai angka 0.

Prinsip Kerja Mode Kembali (*Return Mode)*

Return mode adalah proses robot berjalan kembali dari *finish* menuju *start* dengan jalur terpendeknya. Pada *return mode*, robot diharapkan sudah berjalan dengan kecepatan yang lebih dibandingkan dengan *search mode.* Ketika *return mode* dijalankan, apakah robot bisa mencapai *start* dengan jalur terpendek?Jika proses ini belum berhasil, maka terjadi kesalahan pada proses pemetaan. Dengan demikian, proses akan diulang pada *search mode* dan dievaluasi sampai tidak terjadi kesalahan. Diagram alir (*Flowchart) repeat mode* dapat dilihat pada gambar 13. Prinsip dasar algoritma *backtracking* ini adalah bahwasanya robot akan mengutamakan belok kanan apabila menemukan persimpangan dibanding lurus atau belok kiri. Bila tidak ada belok kanan maka robot cenderung untuk jalan lurus.



Gambar 13 Diagram Alir Mode Kembali

Pada mode ini, robot tidak akan melewati jalan yang sudah dilewati pada saat mode pencariaan. Artinya, robot tidak akan melewati jalan yang buntu lagi, karena untuk jalan buntu sudah ditandai angka 0 dan robot tidak bisa melewati jalan itu lagi.

IV. HASIL PENGUJIAN

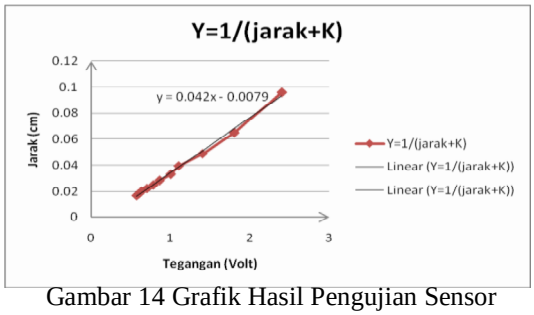
Pada bagian pengujian, telah dilakukan pengujian terhadap sensor, motor, kontroler dan algoritma untuk mengetahui karakteristik, kestabilan dan kesesuaian target yang diinginkan.

Hasil pengujian sensor *infrared*

Tabel 1 Pengujian Sensor

			k=0.42
Jarak	**X=Vout**	**ADC**	**Y=1/(jarak+K)**
10	2.4	121	0.09596929
15	1.8	82	0.064850843
20	1.4	67	0.048971596
25	1.1	55	0.039339103
30	1	45	0.03287311
35	0.86	40	0.028232637
40	0.78	38	0.024740228
45	0.7	35	0.022016733
50	0.63	31	0.019833399
55	0.6	29	0.018044027
60	0.57	27	0.016550811

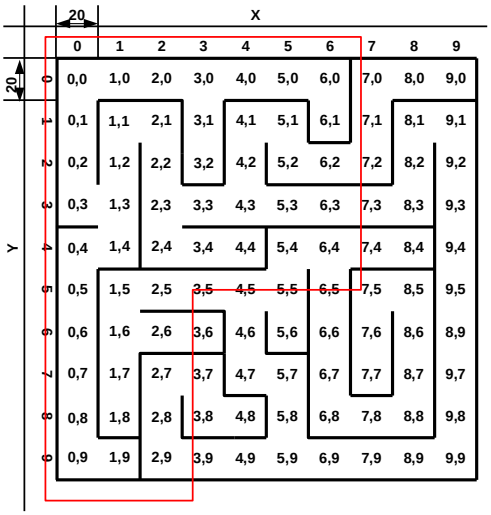
Dari tabel diatas, dapat dikonversikan ke jarak yang sesungguhnya. Gambar 14 menunjukan perubahan tegangan terhadap nilai jarak.



Gambar 14 Grafik Hasil Pengujian Sensor

Pengujian Algoritma *Backtracking*

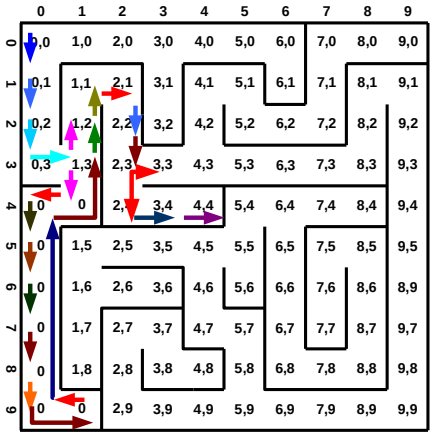
Dalam pengujian algoritma ini, robot dicoba untuk melakukan tugas yang sebenarnya. Tujuannya adalah apakah algoritma yang digunakan bisa diterapkan atau tidak.



Gambar 15 Blok Pengujian *Backtracking*

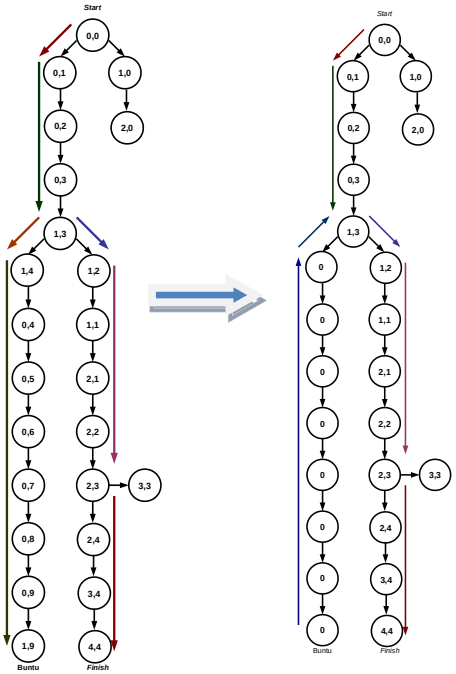
Pada gambar 15 misalkan robot diletakkan pada posisi *start* (0,0) dan *finish* terletak pada titik koordinat (4,4). Algoritma *backtracking* bekerja sebagai berikut. Pada proses pencarian, robot akan bergerak mulai dari titik (0,0). Apabila robot menemui persimpangan maka robot akan lebih mengutamakan belok kanan. Setiap robot bertemu jalan buntu, maka robot akan kembali ke titik koordinat sebelumnya dan kotak yang merupakan jalan buntu akan ditandai dengan angka '0'. Sedangkan yang lain tetap.

Pada blok tersebut, robot harus dapat menemukan titik yang diinginkan (*finish)* yaitu kotak dengan titik koordinat (4,4) dan kembali ke posisi *start* yaitu kotak dengan titik koordinat (0,0) dengan menggunakan jalur terpendek. Berikut adalah gambar 16 yang menunjukkan pergerakan robot secara keseluruhan.



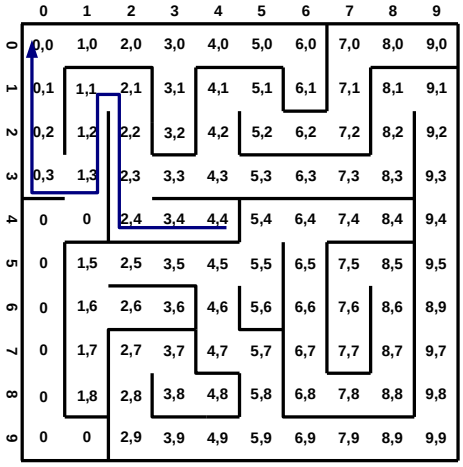
Gambar 16 Arah Pergerakan Robot

Dapat dilihat bahwa robot melakukan pencarian jalan dengan semua kemungkinan yang ada sampai menuju *finish.* Setelah mengetahui arah pergerakan robot, maka algoritma ini dibuat dalam bentuk pohon solusi. Gambar pohon solusi dapat dilihat pada gambar 17



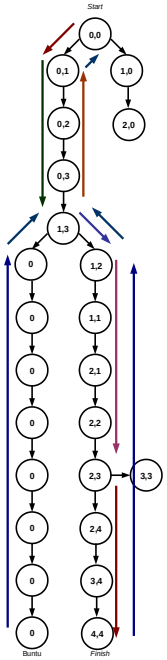
Gambar 17 Pohon Solusi

Setelah robot mencapai *finish* maka robot akan kembali ke posisi *start* yaitu titik koordinat (0,0) dengan menggunakan jalur terpendek hasil dari pencarian saat robot menuju *finish.* Robot tidak aka mengulangi jalan yang sama apabila sudah diketahui bahwa jalan tersebut adalah jalan buntu. Jalan buntu sudah ditandai dengan angka '0'. Berikut gambar 18 menunujukkan pergerakan robot kembali ke *start.*



Gambar 18 Arah Pergerakan Robot Kembali dengan Jalur Terpendek

Dari gambar diatas dapat dilihat bahwa robot akan kembali ke titik *start* yaitu titik (0,0) dimulai dari titik (4,4) kemudian robot bergerak menuju titik (3,4). Dari titik (3,4) robot bergerak menuju titik (2,4) lalu ke titik (2,3) kemudian menuju titik (2,2). Kemudian robot bergerak dari titik (2,2) menuju titik (2,1) dan bergerak lagi menuju titik (1,1) kemudian menuju titik (1,2) dan (1,3). Pada titik (1,3) robot langsung bergerak menuju titik (0,3) karena titik (1,4) adalah jalan buntu. Dari titik (0,3) robot bergerak ke titik (0,2) kemudian ke titik (0,1) dan akhirnya robot bergerak ke titik (0,0). Pada titik (0,0) robot akan berhenti. Setelah mengetahui arah pergerakan kembalinya robot dengan jalur terpendek dapat dilihat gambar 19 yang menunjukan pohon solusi robot kembaili menuju *start.*



Gambar 19 Pohon Solusi Robot Kembali

Berdasarkan gambar diatas diperoleh hasil sebagai berikut :
Hasil pencarian : (0,0), (0,1), (0,2), (0,3), (1,3), (1,4), (0,4), (0,5), (0,6), (0,7), (0,8), (0,9), (0,8), (0,7), (0,6), (0,5), (0,4), (1,4), (1,3), (1,2), (1,1), (2,1), (2,2), (2,3), (2,4), (3,4), (4,4).
Jalur terpendek : (4,4), (3,4), (2,4), (2,3), (2,2), (2,1), (1,1), (1,2), (1,3), (0,3), (0,2), (0,1), (0,0).

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Kesimpulan yang dapat diambil dalam penelitian ini bahwa robot *micromouse* yang dibuat belum dapat menjelajahi semua labirin dengan baik sehingga robot tidak dapat melakukan pemetaan dengan baik. Ini disebabkan oleh *body* robot yang terlalu besar.

B. Saran

Ada beberapa saran untuk pengembangan robot *micromouse* ini dimasa mendatang, yaitu sebagai berikut :

1. Ukuran robot harus lebih kecil dari lorong labirin. Karena apabila ukuran robot terlalu besar maka robot tidak bisa berputar pada saat menemukan jalan buntu.
2. Penggunaan sensor SHARP GP2D12 perlu dikalibarasi ulang untuk mendapatkan nilai jarak yang sebenarnya.

VI. DAFTAR PUSTAKA

[1] Andika Pratama, "Analisis Penerapan Algoritma *Backtracking* Pada Pencarian Jalan Keluar di Dalam Labirin", Makalah IF2251, Bandung, 2007. Hal 2)

[2] Anita, Sur Syafidtri. "Robot Micromouse dengan Menggunakan Algoritma Depth-First Search", 2010.

[3] Mishra, Swati. *Maze Solving Algorithm for Micromouse*. IEEE International Conference on Signal Image Technology and internet Based Systems :2008.

[4] Adi, Agung Nugroho. *Mekatronika*. Yogyakarta. Graha Ilmu:2010

[5] Heryanto, M. Ary, Adi P, Wisnu. *Pemograman Bahasa C untuk Mikrokontroler ATMEGA 8535*.Yogyakarta. Andi:2008.

[6] Pitowarno, Endra. (2006). *Robotika: Desain, Kontrol, dan Kecerdasan Buatan.* ANDI. Yogyakarta.

[7] *http://www.orientalmotor.com.products/pdfs/A_OM/AcRevIntr0.pdf*

[8] *http://www.budiman-ok.co.cc*

[9] Datasheet IC L293D

[10] Datasheet ATmega 8535

[11] Datasheet GP2D12