

ROBOT MICROMOUSE DENGAN MENGGUNAKAN ALGORITMA FLOOD-FILL

Hardi Rifki Al'Amin

Program Studi Teknik Elektro, Fakultas Teknologi Industri, Universitas Islam Indonesia
Jalan Kaliurang Km.14,5 Sleman, Yogyakarta 55501
Telp. (0274) 895007, 895287 Faks.(0274) 89 500 7 Ext. 131
E-mail : hardirifki.a@gmail.com

Abstrak

Robot Micromouse adalah sebuah robot cerdas yang dapat bergerak dengan bebas di dalam sebuah area labirin tanpa menyentuh objek sekitarnya, yang pada akhirnya robot mengetahui ke arah mana harus bergerak, berapa derajat harus berputar jika menemui jalan buntu pada area labirin tersebut, bekerja secara autonomus atau tanpa intervensi dari manusia. Robot Micromouse pada penelitian ini dirancang dengan sensor proximity yang dapat mendeteksi dinding dan rintangan lainnya pada labirin dengan cara mengembalikan nilai yang sebanding antara jarak robot dengan dinding kiri, dinding kanan, atau dinding di depan robot. Sensor encoder pada roda digunakan untuk mengetahui jarak yang dapat ditempuh robot micromouse dalam 1 kali putaran roda sehingga robot dapat mengetahui di sel labirin mana ia berada. Selanjutnya, data dari sensor ini akan disimpan oleh robot untuk pemetaan bentuk labirin. Algoritma Flood-Fill merupakan metode yang digunakan untuk menyelesaikan masalah pencarian rute terpendek pada labirin 5x5 dalam penelitian ini. Algoritma Flood-Fill melibatkan proses penomoran pada setiap sel dalam labirin dimana nomor-nomor ini merepresentasikan jarak setiap sel dengan sel tujuan. Sel tujuan yang ingin dicapai diberi nomor 0 dan sel-sel pada labirin yang memungkinkan untuk mencapai posisi tujuan, ditandai dengan cara $n+1$. Dengan algoritma ini, awalnya robot melakukan eksplorasi pada setiap sel pada labirin sehingga dihasilkan peta dari labirin tersebut, selanjutnya robot menelusuri sel-sel dengan nilai terkecil sesuai dengan peta yang telah dibuat dengan waktu yang lebih cepat dari waktu eksplorasi. Dari hasil pengujian eksplorasi pada beberapa lapangan, robot selalu berhasil menemukan GOAL dengan metode algoritma flood-fill ini, tetapi tidak semua lapangan yang peta jalur terpendeknya berhasil dibuat oleh robot yang mungkin disebabkan adanya informasi yang belum dimiliki robot ketika robot menemui beberapa kondisi baru di beberapa lapangan yang memerlukan tindakan khusus.

Kata kunci : Robot *Micromouse*, Algoritma *Flood-Fill*, Labirin, Jalur terpendek, Sensor *proximity*, Pemetaan, Encoder.

1. PENDAHULUAN

1.1. Latar Belakang Masalah

Robot adalah sebuah alat mekanik yang dapat melakukan tugas fisik, baik menggunakan pengawasan dan kontrol manusia, ataupun menggunakan program yang telah didefinisikan terlebih dulu (kecerdasan buatan). Robot berasal dari bahasa *Czech*, *Robota*, yang berarti pekerja.

Belakangan ini, berbagai bencana atau kecelakaan seperti kebakaran, bocornya gas beracun, kecelakaan di pertambangan atau gempa bumi seringkali terjadi. Para petugas penyelamat atau pemadam kebakaran, seringkali harus menempuh bahaya dan risiko untuk menyelamatkan dan mengevakuasi korban kebakaran atau bencana alam. Terkadang para petugas juga harus membayar mahal tugas mulia itu dengan nyawanya. Untuk dapat membantu dan menggantikan tugas manusia yang berat ini, dibutuhkan robot yang memiliki

kemampuan cerdas. Salah satu cara untuk dapat membuat sebuah robot yang cerdas adalah dengan cara mengimplementasikan algoritma kecerdasan buatan pada robot tersebut.

Robot *Micromouse* merupakan salah satu tipe robot yang memiliki kecerdasan buatan didalamnya. Akhir-akhir ini, banyak digelar pertandingan robot *micromouse* di berbagai belahan dunia, dan mulai hangat dibicarakan di Indonesia. Robot *Micromouse* adalah robot cerdas yang dapat bergerak bebas di dalam sebuah labirin (maze) tanpa menyentuh objek sekitarnya, dan robot akan mengetahui ke arah mana harus bergerak, berapa derajat harus berputar jika menemui jalan buntu pada area labirin (maze) tersebut.

1.2. Rumusan Masalah

Berdasarkan uraian latar belakang diatas, maka didapat rumusan masalah sebagai berikut:

- a. Bagaimana cara merancang robot mobil yang dapat bergerak bebas di dalam sebuah labirin (maze) tanpa menyentuh objek sekitarnya.
- b. Bagaimana cara merancang robot mobil yang dapat mengetahui ke arah mana harus bergerak dan berapa derajat harus berputar jika menemui jalan buntu pada area labirin (maze).
- c. Bagaimana merancang Metode Algoritma *Flood-Fill* yang handal pada robot.
- d. Bagaimana merancang robot yang ukurannya tidak lebih dari 16.8cm x 16.8cm.

1.3. Batasan Masalah

Batasan masalah penelitian ini adalah:

- a. Bahasa pemrograman yang digunakan adalah bahasa BASIC pada *compiler* BASCOM AVR versi 1.11.9.0.
- b. Mikrokontroler yang digunakan adalah ATMEGA32 dari AVR.
- c. Mengimplementasikan Algoritma *Flood-Fill* pada robot *micromouse* dalam pencarian rute dari sel awal menuju sel yang dituju (*GOAL*) dan melakukan pemetaan pada labirin (maze) 5x5.
- d. Bentuk lintasan dari labirin sudah ditentukan sebelumnya.

1.4. Tujuan Penelitian

Penelitian ini bertujuan untuk mengetahui bagaimana metode Algoritma *Flood-Fill* jika diimplementasikan pada masalah pencarian rute dan pemetaan pada sebuah labirin berukuran 5x5. Jika terus dikembangkan, sistem pada robot *micromouse* ini dapat digunakan untuk menciptakan robot yang dapat menggantikan pekerjaan manusia yang dapat membahayakan keselamatan, misalnya menyelamatkan korban yang terjebak didalam gedung yang terbakar, dan tempat-tempat yang mempunyai radiasi tinggi.

2. TINJAUAN PUSTAKA

2.1. Tinjauan Pustaka

Nur Syafidtri Anita, 2010, dari Fakultas Ilmu Komputer Jurusan Sistem Komputer Universitas Gunadarma di Jakarta, telah melakukan penelitian tentang robot *micromouse* dalam paper miliknya yang berjudul "Robot Micromouse Dengan Menggunakan Algoritma Depth-First Search". Pada paper ini, Nur Syafidtri Anita sebagai peneliti menggunakan

algoritma *Depth-First Search* untuk algoritma pemogramannya. Pada algoritma ini dipakai sistem seperti pohon akar, dimana robot memulai mencari dari node akar terus ke level ke-1 atau ke level yang lebih tinggi dari kiri ke kanan, kemudian robot bergerak menuju level selanjutnya, demikian pula dari kiri ke kanan hingga akhirnya ditemukan solusi. Peneliti menjabarkan bahwa keuntungan dari algoritma ini adalah robot tidak akan menemukan jalur buntu, tetapi memiliki kelemahan membutuhkan memori yang banyak, karena harus menyimpan semua node dalam satu pohon, juga membutuhkan waktu yang cukup lama untuk dapat menemukan solusi karena akan menguji n level untuk mendapatkan solusi pada level yang ke- $(n+1)$.

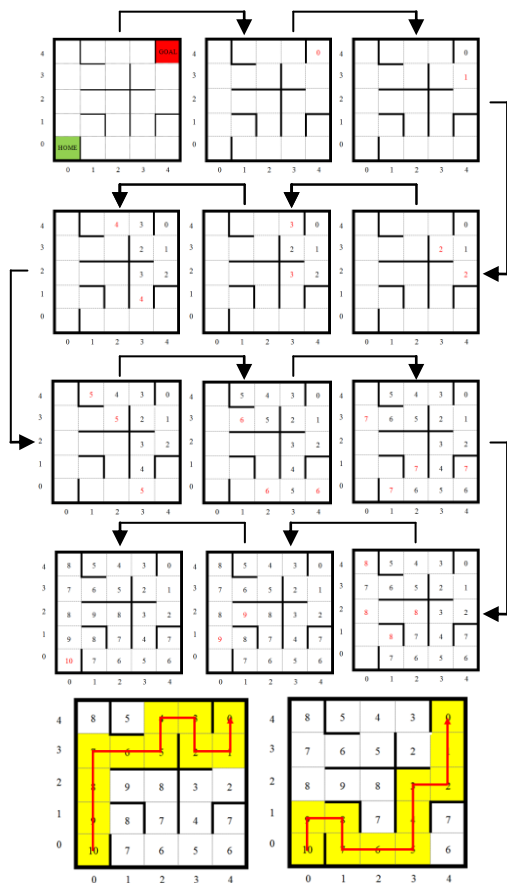
Untuk Algoritma *flood-fill*, telah dilakukan penelitian oleh Abdullah M N Rahman, Akhmad Hendriawan dan Reesa Akbar, 2010, dari Jurusan Teknik Elektronika Politeknik Elektronika Negeri Surabaya (PENS-ITS) dalam paper mereka yang berjudul "Penetapan Algoritma Flood Fill Untuk Menyelesaikan Maze Pada Line Follower Robot". Penelitian ini menggunakan Algoritma *Flood-Fill* dalam algoritma pemogramannya. Perbedaannya dengan penelitian yang penulis lakukan adalah dalam jenis robot yang digunakan dan medan yang dipakai. Robot yang digunakan adalah robot line follower dengan garis sebagai pengganti labirinnya, tentunya terdapat perbedaan pada robot dan sistem yang akan digunakan karena penulis menggunakan Robot *micromouse* yang berjalan menyusuri labirin yang berupa sekat-sekat dinding.

2.2. Algoritma *Flood-fill*

Secara umum algoritma *Flood-Fill* akan menentukan daerah-daerah yang terhubung dengan suatu simpul dalam array multi-dimensi. Algoritma ini sering digunakan dalam program editor gambar bitmap untuk mewarnai suatu daerah terbatas dengan warna tertentu (*boundary fill*). Algoritma ini dapat diadaptasi untuk menyelesaikan permasalahan labirin secara dinamis. Algoritma *Flood-Fill* dapat dianalogikan seperti membanjiri labirin dengan air yang banyak. Air akan terus mengalir hingga mencapai lantai tempat tujuan. Jalur yang dilalui oleh tetesan air pertama di tempat tujuan merupakan jalur terpendek untuk mencapai tempat tujuan tersebut.

Algoritma *Flood-Fill* ini melibatkan proses penomoran setiap sel dalam labirin dimana nomor-nomor ini merepresentasikan jarak setiap sel dengan sel tujuan. Sel tujuan yang ingin dicapai diberi nomor 0. Setiap maze labirin

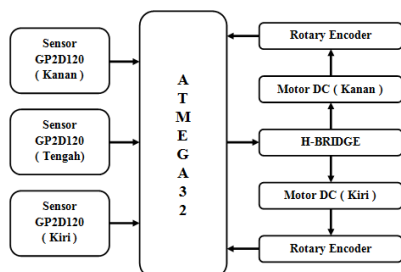
yang mempunyai dinding di 4 sisinya, diberi nomor -1. Sel – sel pada labirin yang memungkinkan untuk mencapai posisi tujuan, kita tandai dengan cara n+1 sampai semua sel telah ditandai. Jika robot berada pada pada sel dengan nomor 1, maka robot tersebut hanya berjarak 1 sel dari sel tujuan. Jika robot berada pada sel dengan nomor 3, maka robot tersebut berjarak 3 sel dari posisi sel tujuan.



Gambar 1 Visualisasi Algoritma *Flood-fill*

3. PERANCANGAN SISTEM

3.1. Blok Diagram



Gambar 2 Blok Diagram Robot Micromouse

Dari gambar 3.1 diatas dapat dilihat bahwa setiap blok mempunyai fungsi masing-masing

yang secara terpusat dikendalikan oleh mikrokontroler ATMEGA32. Adapun fungsi secara umum dari masing-masing bagian blok diagram dapat dijelaskan sebagai berikut :

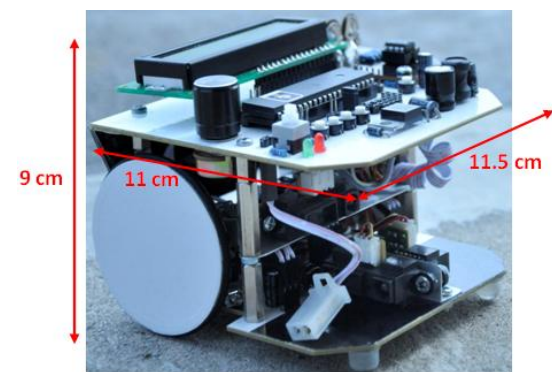
- a. Mikrokontroler ATMEGA32 berfungsi sebagai otak atau pengendali utama dari keseluruhan sistem yang ada pada robot.
- b. Sensor *Infrared* SHARP GP2D120 berfungsi sebagai pendeteksi jarak dinding labirin terhadap robot. Pada robot ini digunakan 3 buah sensor SHARP GP2D120 yang terletak pada sisi samping kanan, samping kiri, dan sisi depan robot.
- c. *Rotary Encoder* berfungsi untuk mengubah banyaknya langkah pada roda robot menjadi informasi berupa kode digital untuk diteruskan ke mikrokontroler ATMEGA32 sebagai masukan sehingga dapat diketahui sudah sejauh mana robot bergerak atau berjalan.
- d. *H-BRIDGE* berfungsi sebagai aktuator untuk mengontrol kecepatan motor DC yang digunakan sebagai penggerak roda robot.

3.2. Perancangan *Hardware*

3.2.1. Perancangan Mekanik Robot

Perancangan mekanik robot micromouse ini meliputi beberapa bagian, yaitu perancangan desain robot dengan ketentuan dimensi yang panjang dan lebarnya tidak boleh melebihi ukuran 16.8cm x 16.8cm atau ukuran 1 sel labirin.

Adapun dimensi dari robot micromouse ini adalah sebagai berikut :



Gambar 3 Robot Micromouse

3.2.2. Perancangan Elektronik Robot

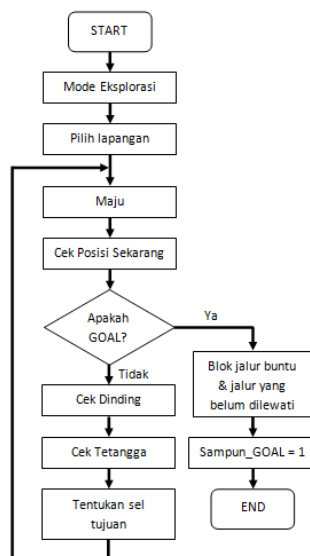
Dalam perancangannya, robot micromouse ini dibangun dari kumpulan beberapa rangkaian elektronik yang terbagi dalam beberapa bagian,

yaitu sebuah sistem minimum mikrokontroler ATMEGA32, rangkaian pencatu daya (*power supply*), rangkaian penampil LCD, rangkaian tombol, rangkaian H-BRIDGE, dan rangkaian penghasil pulsa pada *encoder* roda.

3.3. Perancangan Software

3.3.1. Perancangan Algoritma *Flood-fill* pada Pemilihan Jalur Terpendek dalam Mode Eksplorasi dari *HOME* menuju *GOAL*

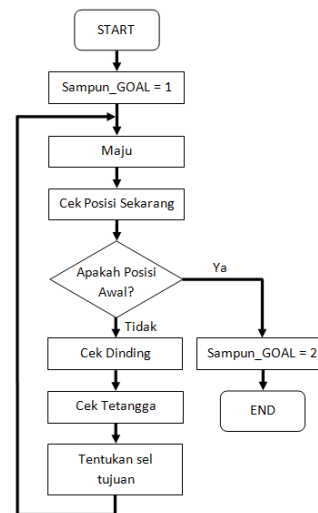
Penjelasan dari diagram alir proses Algoritma *flood-fill* menemukan *GOAL* ini dimulai dengan *START* yang menandakan awal mulainya program dan masuk ke dalam mode eksplorasi. Setelah itu dilakukan pemilihan lapangan guna mendapatkan nilai koordinat dari sel awal (*HOME*) lalu robot berjalan maju dari sel awal, setiap 24 langkah robot akan berhenti untuk mengecek apakah posisi robot sekarang bernilai 0 (*GOAL*) atau bukan. Jika *GOAL* robot akan melakukan blok terhadap sel yang belum pernah dilewati dan terhadap sel yang merupakan jalan buntu sehingga didapat jalur terpendek dari lapangan tersebut, dan robot juga akan mengisi nilai variabel *sampun_GOAL* dengan nilai 1 yang menandakan robot telah mencapai *GOAL* dan siap masuk ke mode pulang ke *HOME*. Tetapi, jika sel sekarang bukan merupakan *GOAL* maka robot akan melakukan cek dinding guna mendapatkan tetangga mana saja yang terbuka disekitar sel tempat robot berdiri sekarang sehingga bisa diketahui sel tetangga mana yang mempunyai nilai terkecil untuk kemudian dijadikan sebagai sel tujuan robot selanjutnya.



Gambar 4 *Flowchart* Algoritma *Flood-Fill* Pada Mode Eksplorasi

3.3.2. Perancangan *flowchart* Pemilihan Jalur dari *GOAL* menuju *HOME*

Penjelasan dari *flowchart* pada gambar 3.11 dimulai dengan *START* yang menandakan awal mulainya program. *Sampun_GOAL* bernilai 1 menandakan robot masuk kedalam mode pulang ke *HOME* dan menerapkan metode algoritma *flood-fill* mengikuti nilai sel yang lebih besar 1 dari sel sekarang hingga *HOME* ditemukan. Setiap 24 langkah robot akan berhenti untuk mengecek apakah posisi robot sekarang merupakan *HOME* atau bukan. Jika *HOME* maka robot akan mengisi nilai variabel *sampun_GOAL* dengan nilai 2 yang menandakan robot masuk ke mode hapalan menuju *GOAL*. Tetapi, jika sel sekarang bukan merupakan *HOME* maka robot akan melakukan cek dinding guna mendapatkan tetangga mana saja yang terbuka disekitar sel tempat robot berdiri sekarang sehingga bisa diketahui sel tetangga mana yang mempunyai nilai besar +1 untuk kemudian dijadikan sebagai sel tujuan robot selanjutnya.

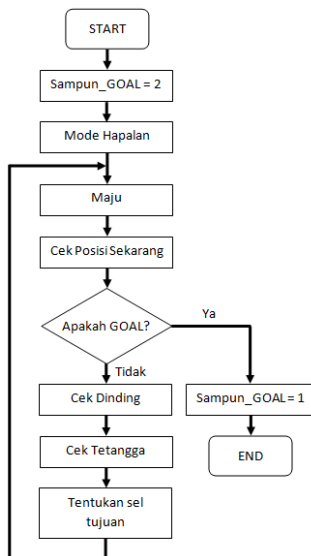


Gambar 5 *Flowchart* Algoritma *Flood-Fill* Pada Mode Pulang ke *HOME*

3.3.3. Perancangan *flowchart* Pemilihan Jalur pada Mode Hapalan

Proses yang terjadi pada mode hapalan sama dengan proses yang terjadi pada mode eksplorasi. Perbedaannya hanyalah pada mode hapalan, robot tidak perlu lagi mengeksplorasi karena peta jalur terpendek menuju *GOAL* sudah didapat sehingga robot hanya tinggal berhenti setiap 24 langkah (berpindah 1 sel) lalu berhenti untuk mengecek apakah posisi robot sekarang merupakan *GOAL* atau bukan. Jika *GOAL* maka robot akan kembali mengisi nilai variabel *sampun_GOAL* dengan nilai 1 yang

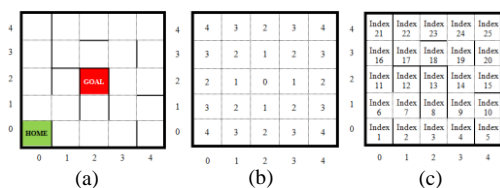
menandakan robot kembali masuk ke mode pulang ke HOME.



Gambar 6 Flowchart Algoritma Flood-Fill Pada Mode Hapalan

4. ANALISA DAN PEMBAHASAN

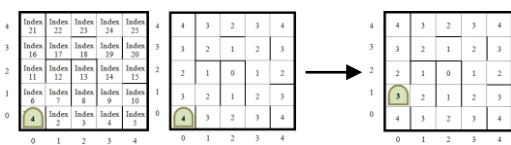
4.1. Pengujian Robot dalam Melakukan Eksplorasi Lapangan



Gambar 7 (a) Lapangan yang akan di eksplorasi, (b) Peta buta, (c) Nilai Index dari masing-masing sel.

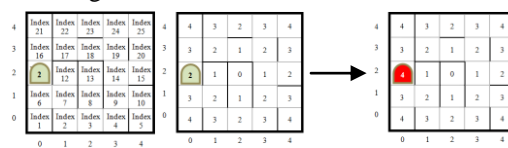
Diketahui robot akan mengawali eksplorasi dari koordinat (0,0), dan GOAL terletak pada koordinat (2,2). Menurut peraturan dari lomba robot micromouse, sebuah HOME harus memiliki 3 dinding di setiap sisinya, masing - masing di sisi kanan, sisi kiri, dan sisi belakang.

Karena hanya sel didepan robot yang terbuka, dan nilai sel didepan robot lebih kecil dari nilai sel sekarang, maka robot akan bergerak maju sejauh 1 sel menuju index 6 dan menjadikan nilai dari index 6 sebagai tetangga terkecil dari sel sekarang yaitu index 1.



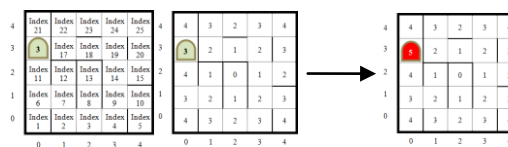
Gambar 8 Robot maju 1 sel ke depan

Variabel sampun pada index 1 diberi nilai +1 yang menandakan sel ini telah dilewati 1 kali. Diketahui di sisi kiri robot terdapat dinding, maka tetangga kiri diabaikan. Disisi kanan robot terbuka tetangga dengan nilai 2, dan didepan robot terbuka juga tetangga dengan nilai 2. Karena sel robot sekarang mempunyai nilai 3, dan masing - masing nilai dari tetangganya bernilai lebih kecil dan sama, maka robot lebih memilih untuk maju dengan alasan robot akan menghabiskan banyak waktu jika memilih sel yang ada dikannya karena harus berbelok dulu baru maju 1 sel. Robot maju 1 sel ke depan dan menjadikan sel depan sebagai tetangga terkecil dari sel tempat robot berada sekarang.



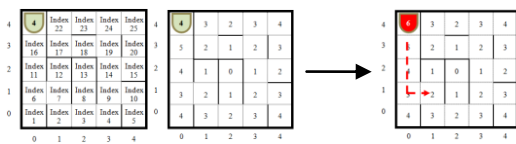
Gambar 9 Robot melakukan re-flood

Pada gambar 4.3, setelah robot sampai pada sel dengan Index 11, maka robot melakukan cek dinding dan mendapat informasi bahwa di sisi kanan dan sisi kirinya adalah dinding, maka sisi kanan dan kiri pun diabaikan. Sisi depan mempunyai index 16 dengan nilai 3, dan sisi belakang yang baru saja di tinggalkan mempunyai index 6 dan juga bernilai 3. Karena sel didepannya lebih besar, maka syarat dari metode algoritma flood-fill yang diterapkan yaitu robot hanya bisa bergerak maju ke sel yang nomornya lebih kecil dari sel sekarang tidak terpenuhi. Jika berada pada kondisi ini, maka robot harus melakukan re-flood guna mengubah peta buta yang ada sekarang dengan cara nilai sekarang = nilai sel tetangga depan + 1. Maka sel index 11 tempat sel sekarang yang awalnya bernilai 2 berubah menjadi 4 (3 + 1 = 4). Robot bergerak maju kedepan menuju sel dengan index 16 dan menjadikan Index 6 yang berada dibelakang sel sekarang sebagai tetangga terkecil dari index sel sekarang yaitu index 11. Variabel sampun pada index 11 diberi nilai +1 yang menandakan sel ini telah dilewati 1 kali.



Gambar 10 Robot kembali melakukan re-flood

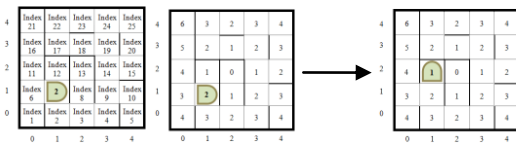
Setelah berpindah sejauh 1 sel, robot menemukan kondisi yang sama seperti pada sel sebelumnya, maka robot melakukan penomoran ulang (re-flood) kembali dan berjalan menuju sel dengan jarak terendah.



Gambar 11 Robot berputar dan re-flood

Seperti pada gambar 4.5, setelah robot berpindah 1 sel, robot menemukan jalan buntu dan berputar 180°, lalu robot menemukan kondisi yang sama seperti pada sel sebelumnya, maka robot melakukan penomoran ulang (re-flood) kembali dan berjalan menuju sel dengan jarak terendah.

Sekarang robot telah sampai pada yang mempunyai index 7 dengan nilai 2. Robot melakukan cek dinding dan mendapat informasi bahwa di sisi depan adalah dinding, maka sisi depan diabaikan. Sisi kanan mempunyai index 2 dengan nilai 3, sisi belakang mempunyai index 6 dengan nilai 3, sisi kiri mempunyai index 12 dengan nilai 1. Karena nilai index dari tetangga kiri lebih kecil maka robot berbelok 90° lalu maju ke index 12 dan menjadikan index 12 sebagai tetangga terkecil dari index 7.



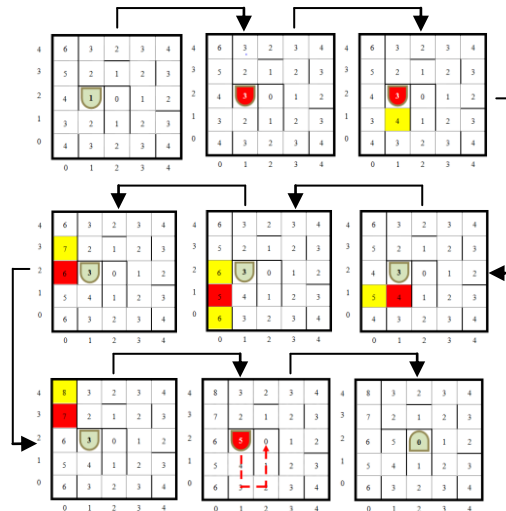
Gambar 12 Robot menuju sel di sisi kiri

Setelah sampai pada sel dengan Index 12, maka robot melakukan cek dinding dan mendapat informasi bahwa di 3 sisinya terdapat dinding atau robot menemukannya jalan buntu, maka robot langsung berputar 180°.

Setelah berputar, robot kembali melakukan cek dinding dan mendapat informasi bahwa di sisi kanan, sisi kiri dan sisi belakangnya adalah dinding, maka sisi kanan, sisi kiri dan sisi belakang pun diabaikan. Sisi depan mempunyai index 7 dengan nilai 2, sementara nilai index 12 pada sel sekarang adalah 1 atau lebih kecil dari tetangganya yang terbuka, maka robot kembali melakukan re-flood.

Tetapi perubahan mengakibatkan peta yang ada sementara menjadi rusak, hal ini karena index 12 merupakan tetangga terkecil dari index 7, dan syarat yang telah diberikan adalah suatu sel harus lebih besar 1 dari tetangga terkecilnya. Karena index 12 lebih besar dari index 7, maka nilai index 7 berubah menjadi **nilai tetangga terkecil + 1**, sehingga diperoleh nilai index 7 sebesar 4 ($3 + 1 = 4$). Dan hal ini juga berlaku untuk sel lainnya hingga tidak ada sel lagi yang nilai tetangga terkecilnya lebih besar 1. Setelah peta telah selesai diperbaiki, robot kembali

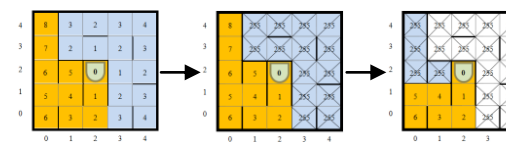
berpindah ke sel dengan nilai terkecil hingga sampai di sel tujuan (GOAL).



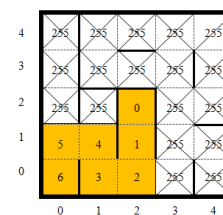
Gambar 13 Perbaiki peta dan perjalanan robot menuju GOAL dengan mengikuti sel dengan nilai terkecil.

Setelah sampai pada sel dengan Index 13, maka robot menemukan bahwa nilai index 13 adalah 0, maka GOAL telah ditemukan, tetapi setelah melakukan cek dinding robot mendapat informasi bahwa di 3 sisinya terdapat dinding atau robot menemukannya jalan buntu, maka robot langsung berputar 180°. Setelah berputar, karena GOAL telah ditemukan maka robot pun segera melakukan pemblokiran jalur pada peta.

Semua jalur yang belum dilalui dengan nilai sampun masih berisi 0 diisi dengan nilai 255. Selanjutnya sel yang merupakan jalan buntu juga diberi nilai 255 sehingga didapat peta jalur terpendek dari lapangan yang baru saja di eksplorasi oleh robot. Variabel Sampun_GOAL diisi dengan nilai 1, menandakan robot masuk pada mode perjalanan dari GOAL ke HOME.



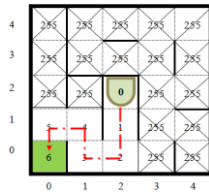
Gambar 14 Proses pemblokiran sel yang belum dilewati dan sel yang buntu



Gambar 15 Peta jalur terpendek hasil eksplorasi

4.2. Pengujian Robot dalam Mode Pulang

Saat robot melakukan perjalanan pulang dari *GOAL* ke *HOME* ini, robot hanya mengandalkan peta jalur terpendek yang telah dibuat setelah lapangan berhasil di eksplorasi, sehingga tidak akan ada proses *re-flood* pada mode ini. Robot akan menerapkan algoritma *flood-fill* dengan cara yang berbeda, yaitu akan mengikuti sel yang mempunyai nomor lebih besar 1 dari sel robot sekarang sampai robot berada di *HOME*.

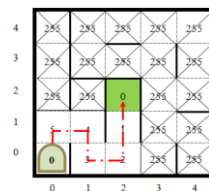


Gambar 16 Jalur perjalanan pulang dari *GOAL* ke *HOME*

4.3. Pengujian Robot dalam Mode Hapalan

Pada mode hapalan, robot akan menerapkan algoritma *flood-fill* dengan cara awal seperti saat eksplorasi dilakukan sampai robot berada di sel yang mempunyai nilai index = 0 (*GOAL*).

Pada mode hapalan ini, robot hanya mengandalkan peta jalur terpendek yang telah dibuat setelah lapangan berhasil di eksplorasi, sehingga tidak akan ada proses *re-flood* pada mode ini.



Gambar 17 Jalur perjalanan pulang dari *HOME* ke *GOAL* pada mode hapalan.

5. KESIMPULAN

Selama melakukan pengujian eksplorasi pada beberapa lapangan, robot selalu berhasil menemukan *GOAL* dengan metode algoritma *flood-fill* ini. Tetapi, tidak semua lapangan yang peta jalur terpendeknya berhasil dibuat oleh robot dari peta buta yang telah diberikan, hal ini terjadi mungkin disebabkan ada beberapa kondisi yang ditemui robot di beberapa lapangan yang perlu tindakan khusus. Karena terdapat parameter-parameter tertentu yang dilihat robot dalam membuat peta yang telah dieksplorasi menjadi sebuah peta dengan jalur terpendek.

DAFTAR PUSTAKA

- Abdullah M N Rahman, Akhmad Hendriawan dan Reesa Akbar, 2010. *Penerapan Algoritma Flood Fill Untuk Menyelesaikan Maze Pada Line Follower Robot*. Skripsi, tidak diterbitkan. Surabaya : Jurusan Teknik Elektronika PENS-ITS.
- Ananian, S. C., & Humphreys, Greg., 2003. *Theseus : A Maze-Solving Robot*. PAPER, published. Department of Electrical Engineering at Princeton University.
- Andrianto, Heri., 2008. *Pemrograman Mikrokontroler AVR Atmega16 Menggunakan Bahasa C (CodeVision AVR)*. Bandung : Penerbit INFORMATIKA. ISBN : 978-979-1153-41-6.
- Anita, Nur Syadfitri., 2010. *Robot Micromouse Dengan Menggunakan Algoritma Depth-First Search*. Skripsi, tidak diterbitkan. Depok : Fakultas Ilmu komputer Universitas Gunadharma.
- Bekti, Samudera Harapan., 2009. *Pencarian Shortest Path Dinamik Dengan Menggunakan Algoritma Bellman-Based Flood-Fill Dan Implementasinya Pada Robot Micromouse*. Makalah IF2091 Struktur Diskrit Tahun 2009. Bandung : Program Studi Teknik Informatika ITB.
- De, Tondra., & Hall, Drew., 2004. *A Detailed Design And Analysis Of Micromouse*. Honors Thesis I. University of Nevada, Las Vegas.
- Pitowarno, Endra., 2006. *Robotika Desain Kontrol, Dan Kecerdasan Buatan*. Yogyakarta : Penerbit Andi Offset Yogyakarta.
- Susilo, Deddy., 2010. *48 Jam Kupas Tuntas Mikrokontroler MCS51 & AVR*. Yogyakarta : Penerbit ANDI. ISBN : 978-979-29-1346-0.
- Willardson, D. M., 2001. *Analysis Of Micromouse Maze Solving Algorithms*. ECE 557 : Learning from Data, Spring 2001.
- www.micromouseinfo.com/introduction/mfloodfill.html
- www.micromouseonline.com/book/micromouse-book
- www.societyofrobots.com/member_tutorials/no_de/94