ALGORITMA FLOODFILL UNTUK MENENTUKAN TITIK KOORDINAT MAZE MAPPING PADA ROBOT LINEFOLLOWER

Ary Sulistyo Utomo
Program Studi Teknik Elektromedik
Akademi Teknik Elektro Medik
Email: ary.utomo@gmail.com

ABSTRAK

Robot line follower (RLF) adalah robot yang dapat berjalan mengikuti suatu *maze* yang berupa garis secara otomatis. *RLF* dapat digunakan untuk aplikasi pengiriman barang dari suatu tempat awal ke tempat tujuan dengan cepat, tepat dan akurat. Untuk menyelesaikan permasalahan tersebut dibutuhkan suatu algoritma yang digunakan untuk mengetahui posisi koordinat robot sehingga robot dapat diketahui pergerakannya. Pergerakan robot berawal dari *start* dan mecapai titik *finish* yang telah di tentukan. Pada penelitian ini menggunakan algoritma *floodfill* untuk mengetahui posisi koordinat RLF. Pengujian dilakukan dengan cara menjalankan RLF dari titik *start* menuju ke titik *finish*. Area yang digunakan berukuran 200 x 200 cm mempunyai tebal garis lintasan 2 cm dengan jarak terdekat pada setiap simpangnya adalah 40 cm Warna garis adalah putih dan *background* berwarna hitam. Hasil penelitian menunjukkan bahwa kestabilan RLF menyusuri garis lintasan dicapai pada nilai pengaturan PID Kp=30, Ki=8 dan KD=100. Hasil dari koordinat pada saat robot bergerak di simpan pada memori eeprom dari mikrokontroller dan di tampilkan pada LCD 2x16. Koordinat *start* dimulai dengan (0,0) dengan *finish* (2,2) pada *maze* yang telah ditentukan.

Kata kunci: *Robot line follower, floodfill, maze mapping, PID.*

ABSTRACT

Line follower robot (RLF) is a robot that can walk to follow a maze that is a line automatically. RLF can be used for application delivery from a starting point to a destination quickly and accurately. To resolve these issues required an algorithm used to determine the position coordinates of the robot so that the robot can be known movement. The movement of the robot begins mecapai start and finish point that has been set. In this study, using an algorithm to determine the position coordinates floodfill RLF. Testing is done by running the RLF from the start point to get to the finish point. Area used measuring 200 x 200 cm has a trajectory line 2 cm thick with the closest distance at any Adverse is 40 cm line color is white and black background. The results showed that the stability of the RLF down the line trajectory achieved in the setting value PID Kp = 30, Ki = 8 and KD = 100. Results of the coordinates when the robot moves in the store in eeprom memory of the microcontroller and displayed on the LCD. Starting with the start coordinates (0,0) to finish (2.2) at a predetermined maze.

Keywords: *Robot line follower, floodfill, maze mapping, PID.*

1. PENDAHULUAN

Diera *modern* sekarang ini perkembangan Ilmu Pengetahuan dan Teknologi (IPTEK) bidang robotika sangat pesat. Keakuratan, ketepatan dan efisiensi dalam menyelesaikan permasalahan yang rumit menjadi tantangan untuk dihadapi. Salah satunya adalah robot yang dapat menyelesaikan suatu tugas tertentu. Misalnya robot dapat mengantarkan barang ke tempat tujuan dan kembali ke tempat semula. Aplikasi robot seperti ini digunakan pada pengiriman barang yang terdapat di gudang, pengantar makanan di restoran dsb. Robot akan berjalan secara otomatis mengirimkan barang menuju ke tempat penyimpanan yang ada di gudang. Keakuratan, ketepatan dan efisiensi kerja agar robot dapat melakukan tugasnya sangat diperlukan. Dengan permasalahan tersebut maka diperlukan algoritma yang dapat membuat robot agar mempunyai kemampuan keakuratan, ketepatan dan efisiensi kerja.

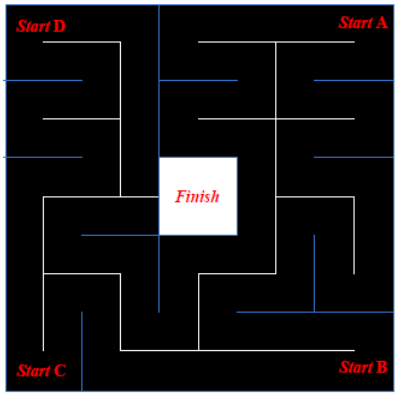
Penelitan sebelumnya yang dilakukan oleh Hendriawan, A ;& Akbar, R dari PENS yaitu mencari jalur terpendek menggunakan algoritma djikstra's [1], Sakib, Shadman pada tahun 2014 menjelaskan algoritma yang dipakai untuk menyelesaikan jalur terpendek [2], Mishra, S. melakukan perhitungan untuk menentukan jalur terpendek pada kopetesi robot tikus [3]. Pada penelitan sebelumnya yang telah saya lakukan yaitu perbandingan algoritma floodfill dan djikstra's pada *maze mapping* untuk robot *line follower* [4]. Yadav, Verma dan Mahanta,S 2012 melakukan penelitian tentang penyelesaian jalur terpendek pada perlombaan robot micro mouse [5]. Yanto, Febi & Welly, Irma pada 2015 melakukan penelitian tentang penyelesaian jalur pada *line follower* apabila terdapat jalur *loop*, lancip dan melengkung [6].

Pada penelitian ini untuk menyelesaikan permasalahan tersebut akan menggunakan algoritma *floodfill* mengetahui posisi titik koordinat *start* dan *finish* robot, serta dapat mencari jalur terpendek yang telah dilalui oleh robot. Pencarian jalur terpendek tersebut supaya kerja robot menjadi *efisien*. Dengan *efisien* tersebut maka robot diharapkan dapat menyelesaikan masalah dengan waktu lebih cepat dan kerja robot lebih ringan. Robot akan berjalan dengan menelusuri *maze* dari titik *start* yang telah ditentukan menuju ke titik *finish*, setelah robot sampai dititik *finish* robot akan kembali ke titik *start* dengan jalur terpendek. Informasi titik koordinat robot tersebut supaya kinerja robot dapat dipantau posisi *start* dan *finish*. Apabila robot berhenti karena sesuatu hal, maka robot dapat segera diketahui permasalahannya. Titik koordinat *start* dan *finish* dapat diketahui robot sehingga membuat kerja robot menjadi lebih akurat.

2. METODOLOGI PENELITIAN

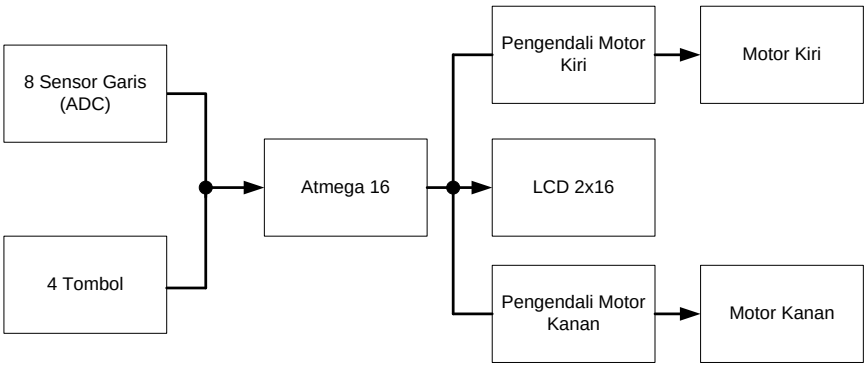
Robot linefollower akan menyelesaikan jaringan jalan yang telah ditentunkan. Jaringan jalan ditunjukan pada Gambar 1. Robot *linefollower* berjalan mulai dari titik *start* yang berada di ujung garis, setelah itu mencarii titik *finish*. Garis yang bewarna putih merupakan jalan yang akan dilalui. Garis berwarna biru diibaratkan dinding. Kotak putih adalah titik *finish*. Setiap pergerakan robot akan menampilkan titik koordinat pada lcd 2x16. Titik koordinat robot diasumsikan (0,0) sehingga apabila robot sukses mencari titik *finish* maka titik *finish* tersebut berkoordinat (2,2).

Rancangan *line maze* yang dibuat adalah tampak seperti pada Gambar 1. Lapangan yang digunakan berukuran 200 x 200 cm. Tebal garis adalah 2cm dengan jarak terdekat pada setiap simpangnya adalah 40 cm. Warna garis adalah putih dan *background* berwarna hitam. Untuk posisi *start* diletakkan pada ujung garis yaitu *start*A, *start*B, *start*C, dan *start*D. Sedangkan posisi *finish* ditandai dengan sebidang kotak berwarna putih berukuran 20 x 20cm. Berdasarkan bentuk dari *line maze* tersebut, maka hanya ada satu jalan keluar saja dari start menuju *finish* sehingga robot harus menemukan jalan keluar tersebut dengan cara mencari (*search mode*) terlebih dahulu. pada *mode* ini koordinat disusun berdasarkan pergerakannya. Jika sudah, berikutnya robot akan kembali dari *finish* menuju start dengan jalur terpendek (return *mode*). Pada mode ini koordinat finish telah didapatkan.



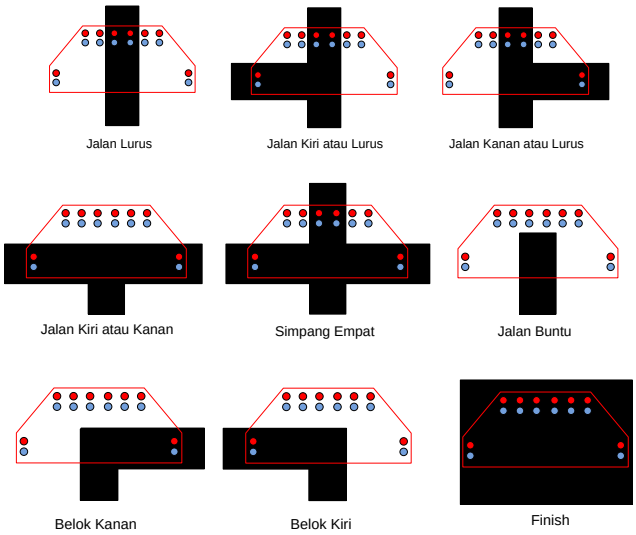
Gambar 1. Lintasan *maze* robot [5]

Perancangan *hardware* dalam penelitian ini terdapat beberapa bagian yaitu sensor, *mikrokontroler*, dan aktuator yang ditunjukan pada Gambar 2 diagram blok sistem. Sensor yang digunakan yaitu *photodioda* yang dapat mengeluarkan tegangan berbeda sesuai dengan intensitas cahaya yang diterima. *Mikrokontroller* digunakan sebagai otak dari robot, dengan memasukkan algoritma dalam bentuk program. Aktuator robot yaitu menggunakan motor DC yang dapat menggerakan robot sesuai dengan perintah.



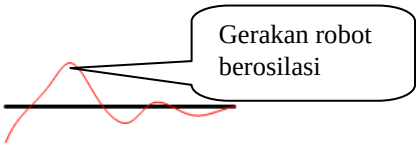
Gambar 2. Diagram Blok Sistem

Sensor garis pada penelitian ini menggunakan delapan buah sensor *photodiode* . Dengan penempatan yaitu, enam buah diletakkan di tengah dan dua buah lagi di sebelah kiri dan kanan namun dengan posisi lebih ke belakang. Jarak antara sensor yang engah dengan yang disebelah kiri dan kanan adalah 2,5 cm. Konfigurasi seperti ini dilakukan dengan tujuan untuk membedakan antara pembacaan persimpangan dan finish. Keterangan lebih jelas dalam penempatan dan cara pembacaan simpang ada pada gambar 3.



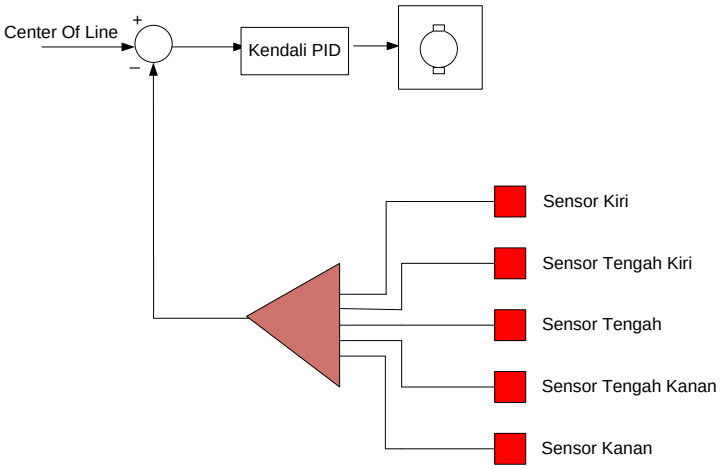
Gambar 3. Konfigurasi Sensor

Perancangan kendali pada penelitian ini digunakan sebagai acuan robot agar dapat bergerak mengikuti garis dengan posisi ditengah garis dengan gerak osilasi seminimal mungkin. Pengaturan gerak robot ditunjukan pada gambar 4.



Gambar 4. Pola gerakan robot *berosilasi*

Kontrol untuk mengendalikan kecepatan motor DC pada robot digunakan *kontroler PID*. Kontroler ini merupakan kombinasi antara kontrol P, I dan D. Dengan menggabungkan tiga kontroler tersebut, maka akan diperoleh luaran yang cukup ideal dari yang diharapkan dapat meredam gerak *osilasi* robot.



Gambar 5. Blok diagram kendali *PID* pada robot

Kontroler PID digital merupakan bentuk lain dari *kontroler PID* yang diprogram dan dijalankan menggunakan komputer atau mikrokontroler. Untuk dapat mengimplementasikan *PID* digital di komputer atau mikrokontroler, maka *kontroler PID analog* harus diubah terlebih dahulu ke bentuk digital [7]. Penurunan *kontroler*

PID digital dapat dilihat pada Persamaan 2.1 sampai dengan Persamaan 2.4. Bentuk persamaan matematis dari *kontroler PID* adalah sebagai berikut.

$$u(t) = K_p\, e(t) + K_i \int_0^t e\,(t)\,dt + K_d \frac{de(t)}{dt} \quad (1)$$

Dimana $K_i = \frac{1}{\tau_i}$ dan $K_d = \tau_d$ bentuk integral dan diferensial dapat ditulis dalam bentuk diskrit seperti pada Persamaan 2 dan Persamaan 3

$$\int_0^t e(t)\,dt \approx T \sum_0^k e(k) \quad (2)$$

$$\frac{de(t)}{dt} \approx \frac{e_k - e_{k-1}}{T} \quad (3)$$

Sehingga diperoleh dalam bentuk kontroler PID diskrit ialah sebagai berikut:

$$u(k) = K_p\, e_k + K_i\, T \sum_0^k e_k + \frac{1}{T} K_d (e_k - e_{k-1}) \quad (4)$$

$$u = K_p\, x\, error + K_i x (error + lasr\ error) + \frac{K_d}{Ts}\, x\, (error - last\ error) \quad (5)$$

Dimana :

Kp adalah konstanta proportional, Ki adalah Konstanta Integral, Kd adalah konstanta diferensial, *Errror* adalah nilai kesalahan, *Last error* adalah jumlah kesalahan sebelumnya, Ts adalah waktu sampling

Dalam penggunaannya, posisi sensor terhadap garis mengartikan error yang terjadi. Berikut adalah gambaran posisi sensor beserta nilai errornya.

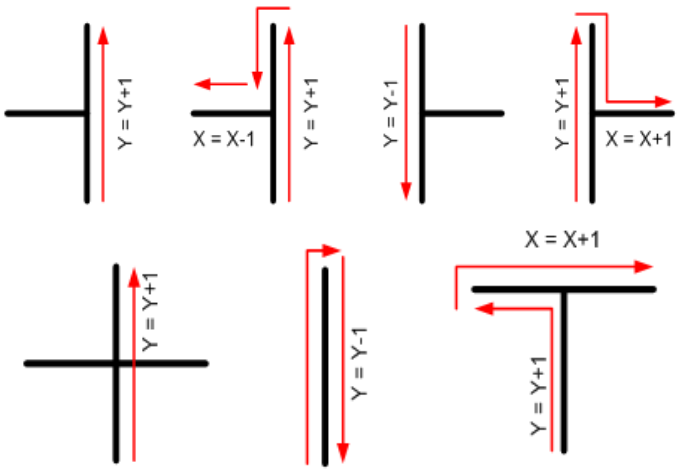
01000000 = 4
00100000 = 2
00010000 = 0
00001000 = 0
00000100 = -2
00000010 = -4
01100000 = 3
00110000 = 1
00011000 = 0 (set point = target position)
00001100 = -1
00000110 = -3

Dalam aplikasinya, maka peran dari kontroler ini dapat diterapkan dalam program dengan formulasi seperti berikut:

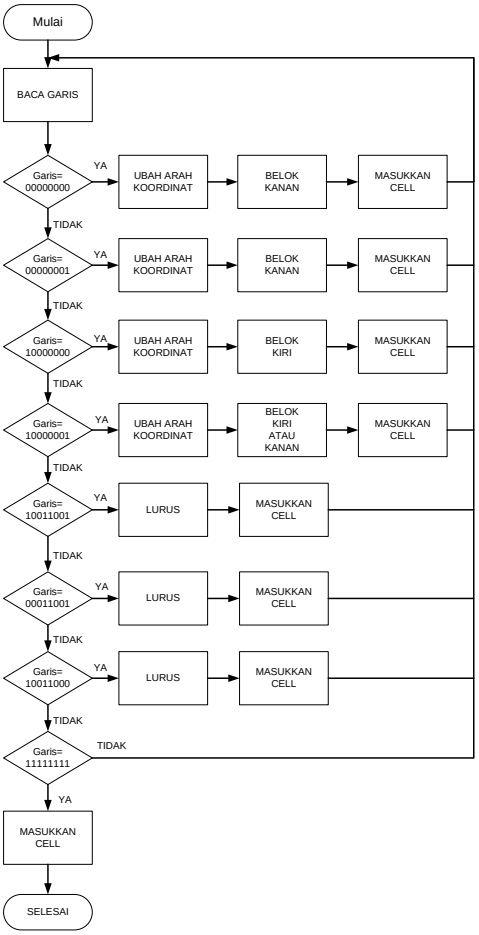
Pwm_ka = Setkecepatan + (Kp.error + Ki.a_error + Kd . d_error)
Pwm_ki = Setkecepatan – (Kp.error + Ki.a_error + Kd . d_error)

Ket: Setkecepatan adalah nilai pwm yang diinginkan pada saat *error* = 0

Perancangan pemetaan koordinat pergerakan robot seperti gambar 6. Pada pemetaan ini setiap RLF penemukan percabangan maka akan memperbarui titik koordinatnya yang di simpan dalam memori. *Flowchart* pergerakan RLF ditunjukan pada gambar 7.



Gambar 6. Pemetaan koordinat



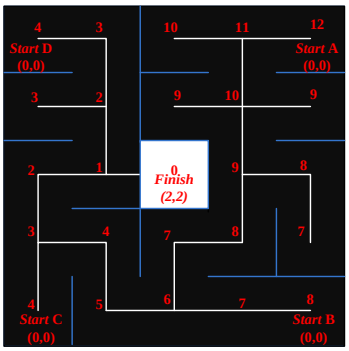
Gambar 7. Flowchart pergerakan RLF

3. HASIL PENELITIAN DAN PEMBAHASAN

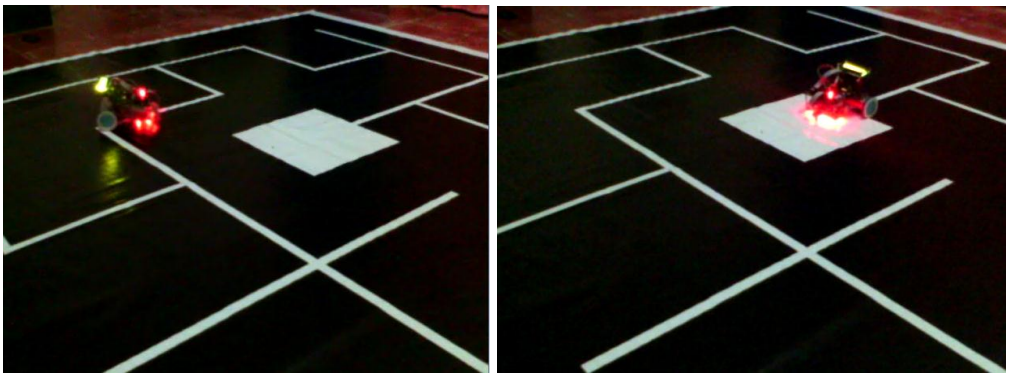
3.1 KENDALI ROBOT

Pengujian pada kendai robot *PID*, dimana percobaan dilakukan pada suatu sistem dengan Kd = 2, ki = 0 , dan pengaturan Kp . *Gain Proposional* dinaikkan secara perlahan sampai batas kestabilan, dimana sistem mulai mengalami osilasi. Berdasarkan data yang diperoleh sistem mulai stabil pada nilai Kp antara 20 s/d 50. Nilai dari kontrol P semakin besar nilainya maka semakin cepat sistem mengejar nilai *set point* dan nilai terlalu kecil semakin lambat mengejar *set poin*, tetapi jika nilai Kp terlalu besar dapat mengakibatkan tidak stabilnya sistem. Apabila diperlukan gerak robot yang halus maka nilai dari Kd diperbesar dan penambahan sedikit Ki. Nilai Kd semakin besar semakin memperhalus jalannya robot, tetapi jika terlalu besar maka gerak robot semakin lambat. Dari hasil percobaan robot didapatkan Kp=30, Ki=100 dan Kd=8. Pada pengaturan tersebut respon stabil dihasilkan ±0.3 detik.

3.2 PENGUJIAN ALGORITMA FOODFILL



Pada pengujian *start* di titik D diperoleh hasil waktu tempuh pencarian tercepat 7 detik dan waktu tempuh kembali 7 detik. Waktu rata-rata pencarian sebesar 7,3 detik dan kembali 7,2 detik. Jarak pada saat pencarian yaitu 160 cm sedangkan jarak terpendek diperoleh 160 cm. Pada pengujian *start* di titik C diperoleh hasil waktu tempuh pencarian tercepat 5 detik dan waktu tempuh kembali 5 detik. Waktu rata-rata pencarian sebesar 6,4 detik dan kembali 6 detik. Pada pengujian *start* di titik B diperoleh hasil waktu tempuh pencarian tercepat 16 detik dan waktu tempuh kembali 17 detik. Waktu rata-rata pencarian sebesar 18,4 detik dan kembali 18 detik. Pada pengujian *start* di titik A diperoleh hasil waktu tempuh pencarian tercepat 33 detik dan waktu tempuh kembali 30 detik. Waktu rata-rata pencarian sebesar 35,7 detik dan kembali 31,5 detik. Jarak pada saat pencarian yaitu 560 cm sedangkan jarak terpendek diperoleh 480 cm. Perbedaan waktu pencarian dan kembali sedikit dikarenakan jarak keduanya hanya sedikit perbedaannya yaitu 80 cm. Pada pengujian semuanya titik start pada koordinat pada (0,0) dan *finish* (2,2).



Gambar 8. Prototipe Robot Line Follower dan Maze

4. KESIMPULAN

a. Nilai keluaran pembacaan ADC pada background hitam rata-rata adalah sebesar 4,8, sedangkan pada garis putih rata-rata sebesar 32,25 sehingga dapat menghasilkan rata-rata nilai tengah (*Threshold*) sebesar 18,5. Dapat disimpulkan bahwa kinerja dari sensor telah bekerja dengan baik pada arena yang telah ditentukan.
b. Nilai Kp=30, KD=100, dan Ki=8 merupakan nilai pengaturan untuk pengendalian *robot line follower* agar dapat berjalan dengan stabil mengikuti jalur yang telah ditentukan. Waktu stabil didapatkan ±0.4 detik.
c. Pada algoritma *floodfill* dapat digunakan untuk mengetahui jarak terpendek serta dapat mengetahui letak posisi koordinat robot karena setiap pergerak robot berdasarkan nilai dari masing masing *cell.*

DAFTAR PUSTAKA

[1] Hendriawan, A ;& Akbar, R. Penyelesaian Jalur Terpendek dengan menggunakan Algoritma Maze Mapping Pada Line Maze. Jurusan Teknik Elektronika, Politeknik Elektronika Negeri Surabaya Kampus PENS-ITS Sukolilo, Surabaya

[2] Sakib, Shadman 2014. "*Maze solving Algorithm for line following robot and derivation of linear path distance from nonlinear path*" IEEE International Conference on Signal Image Technology and Internet Based Systems..

[3] Mishra, S. 2008. *Maze Solving Algorithm for Micro Mouse*. IEEE International Conference on Signal Image Technology and Internet Based Systems.

[4] Utomo, Ary Sulistyo & Prasetyowati, Sri Arttini Dwi & Arifin, Bustanul, 2015. "PERBANDINGAN ALGORITMA FLOODFILL DAN DJIKSTRA'S PADA MAZE MAPPING UNTUK ROBOT LINE FOLLOWER" Prosiding Seminar Nasional Sains Dan Teknologi Fakultas Teknik ISBN : 978-602-99334-4-4, Juni 2015

[5] Yadav,S ;& Verma,K.K ;& Mahanta,S. 2012. *The Maze Problem Solved by Micro mouse*. International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-1, Issue-4, April 2012

[6] Yanto, Febi & Welly, Irma. 2015. "Analisa dan Perbaikan Algoritma Line Maze Solving Untuk Jalur Loop, Lancip, dan Lengkung pada Robot Line Follower (LFR)". Jurnal CoreIT, Vol.1, No.2, Desember 2015 ISSN: 2460-738X

[7] Hagglund.T and Astrom.K. (1995). P*ID Controllers: Theory, Design, and Tuning 2nd Ed. Instrument Society of America 67 Alexander*