PERANCANGAN *LINE MAZE SOLVING ROBOT* DENGAN ALGORITMA *SHORT PATH FINDER*

Alfian Maarif[1], Wahyudi Budi Pramono[2], Dwi Ana Ratna Wati[2]
[1]Penulis, Mahasiswa Jurusan Teknik Elektro UII
[2]Dosen Pembimbing, Staf Pengajar di Jurusan Teknik Elektro UII
Teknik Elektro
Fakultas Teknologi Industri Universitas Islam Indonesia
Jalan Kaliurang KM 14,5 Sleman Yogyakarta 55584
Email: alfianmaarif@gmail.com

ABSTRAKS

Kemajuan teknologi membuat perkembangan di berbagai bidang. Kemajuan di bidang robotika adalah berkembangnya berbagai jenis robot. Salah satu jenis robot adalah line maze solving robot yang berfungsi untuk menemukan jalur terpendek labirin. Bagaimana menemukan jalur terpendek merupakan permasalahan dalam labirin. Permasalahan ini diselesaikan menggunakan Algoritma Short Path Finder. Algoritma ini terdiri atas tiga mode yaitu mode search, mode short path dan mode return and path. Mode search berfungsi menemukan finish, mode short path untuk menemukan lintasan terpendek dan mode return and path untuk kembali ke start dengan jalur terpendek. Sensor yang digunakan robot adalah sensor garis, yang berfungsi mendeteksi jalur yang dilalui robot. Sensor garis merupakan masukan untuk pengolah data. Pengolah data adalah Mikrokontroler ATmega32. Bagian keluaran dan penggerak robot adalah driver motor tipe H-bridge dan motor DC. Berdasarkan hasil percobaan, sensor garis mudah terpengaruh cahaya luar. Oleh karena itu, peredam cahaya perlu diberikan di bagian samping sensor garis untuk mengurangi cahaya luar. Robot mengalami kesalahan pembacaan saat kondisi lepas sensor garis depan karena robot tidak dapat meluruskan posisinya terhadap lintasan. Algoritma Short Path Finder dapat diterapkan pada jenis lintasan bertipe maju dan belum dapat diterapkan pada lintasan bertipe melingkar.

Kata Kunci: *Line maze solving robot, Short Path Finder.*

I. PENDAHULUAN

Kemajuan teknologi membuat perkembangan di berbagai bidang. Perkembangan di bidang teknologi, khususnya di bidang robotika adalah berkembangnya berbagai jenis robot dengan fungsi dan aplikasi masing-masing. Salah jenis robot adalah *line maze solving robot* yang berfungsi untuk menemukan jalur terpendek dari *maze* atau labirin.

Terdapat dua jenis *maze* dalam bidang robotika yaitu *wall maze* dan *line maze*. *Wall maze* merupakan labirin yang dibangun dari dinding tanpa atap. Sedangkan *line maze* merupakan labirin yang dibangun dari garis.

Permasalahan yang timbul dalam *line maze* (labirin garis) adalah bagaimana mendapatkan jalur terpendek dari *maze* (labirin). Pada penelitian ini, permasalahan pada *line maze* diselesaikan dengan menggunakan Algoritma *Short Path Finder*. Algoritma ini mencari posisi *finish* dan kembali ke *start* dengan jalur terpendek. Algoritma ini merupakan hasil modifikasi dari algoritma yang telah ada sebelumnya yaitu Algoritma *Maze Mapping*, *Flood Fill* dan *Pledge*.

Rumusan masalah dalam penelitian ini adalah bagaimana merancang *line maze solving robot*? dan bagaimana tingkat keberhasilan Algoritma *Short Path Finder* yang diterapkan pada *line maze solving robot* dalam pencarian jalur terpendek.

Tujuan dari penelitian ini adalah merancang *line maze solving robot*, menerapkan Algoritma *Short Path Finder* pada *line maze solving robot* dan mengetahui tingkat keberhasilan Algoritma *Short Path Finder*.

II. TINJAUAN PUSTAKA

Penelitian yang terkait tentang *line maze solving robot* dilakukan oleh Abdullah M N Rahman tentang "Penerapan Algoritma *Flood Fill* untuk Menyelesaikan *Maze* pada *Line Follower Robot*", M. Iqbal Nugraha tentang "Penerapan Algoritma *Maze Mapping* untuk menyelesaikan *Maze* pada *Line Tracer*" dan Arif Darmawan tentang "Penerapan Algoritma *Pledge* untuk Menyelesaikan *Maze* pada *Line Follower*".

Line maze solving robot adalah suatu robot yang digunakan untuk menyelesaikan suatu labirin garis (Vannoy, 2009). *Line maze solving robot* dirancang dari *line follower robot* dan *micro mouse*. Implementasi dari *line maze solving robot* adalah suatu robot yang digunakan untuk menelusuri, menyimpan lintasan labirin garis dan menemukan jalur terpendek.

A. Algoritma *Short Path Finder*

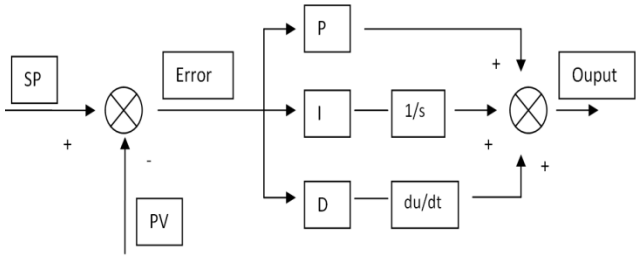
Algoritma *Short Path Finder* adalah algoritma pencarian jalur terpendek *line maze*. Algoritma ini memiliki tiga mode yaitu mode *search*, *short path* dan *return and path*. Algoritma ini dibangun berdasarkan pada algoritma pencarian yang telah ada. Algoritma tersebut adalah Algoritma *Wall Follower*, *Maze Mapping*, *Flood Fill* dan *Pledge*. Pada tabel 1 ditampilkan komponen Algoritma *Short Path Finder* dan jenis algoritma yang digunakan.

Tabel 1 Komponen Algoritma *Short Path Finder*

Algoritma Short Path Finder	
Komponen	**Algoritma**
Pencarian *Finish*	*Wall Follower*
Penyederhanaan	*Maze Mapping*
Path Lintasan	*Flood Fill*
Lintasan khusus	*Pledge*

B. Pengendali PID

Pengendali PID adalah pengendali yang terdiri atas tiga pengendali yaitu pengendali P, I dan D. Gambar 1 merupakan diagram blok dari pengendali PID.



Gambar 1 Blok diagram Pengendali PID

Setiap kelebihan dan kekurangan dari masing-masing pengendali P, I dan D dapat saling ditutupi dengan menggabungkan menjadi pengendali PID. Elemen pengendali P, I dan D secara keseluruhan bertujuan untuk mempercepat reaksi sebuah sistem, menghilangkan osilasi dan kestabilan sistem. Persamaan pengendali PID ditunjukkan pada persamaan 1 dan persamaan dalam bentuk diskrit ditunjukkan pada persamaan 2.

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

$$u_k = K_p e_k + K_i \sum_0^k e_k T + \frac{1}{T} K_d (e_k - e_{k-1}) \quad (2)$$

Karakteristik penambahan pengendali P (Kp), I (Ki) dan D (Kd) ditunjukkan pada tabel 2.

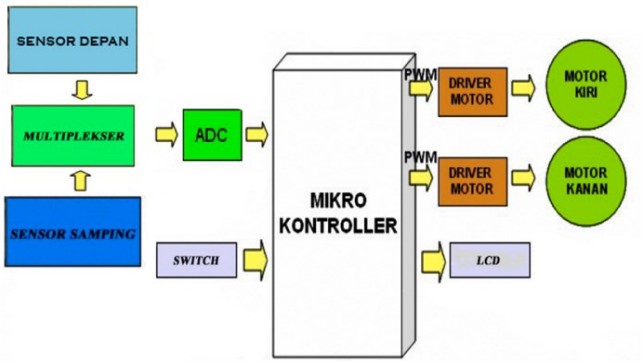
Tabel 2 Karakteristik penambahan nilai Pengendali P, I dan D

Karakteristik	P	I	D
Waktu naik	Turun	Turun	Perubahan kecil
Overshoot	Naik	Naik	Turun
Waktu turun	Perubahan Kecil	Naik	Turun
Kesalahan keadaan tunak	Turun	Hilang	Perubahan kecil

III. PERANCANGAN SISTEM

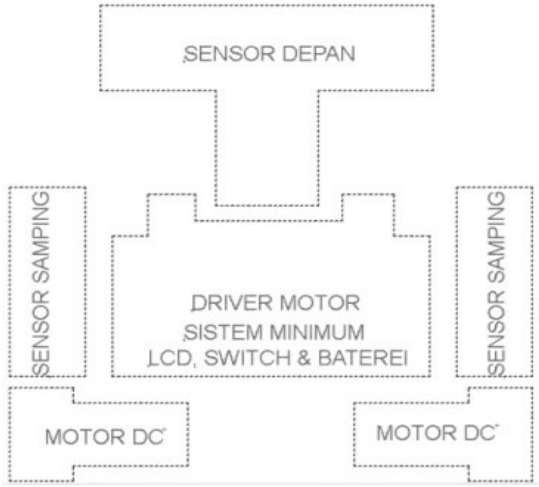
A. Perancangan Sistem dan *Hardware*

Secara umum konfigurasi sistem robot terdiri dari *input*, pengendali dan *output*. Dari sisi *input* adalah sensor garis, pengendali adalah Mikrokontroler ATmega32 dan *output* adalah *driver* motor dan motor DC. Blok diagram sistem dari *line maze solving robot* ditunjukkan pada gambar 2.



Gambar 2 Blok diagram sistem robot

Desain dari *line maze solving robot* yang dibuat ditunjukkan pada gambar 3.

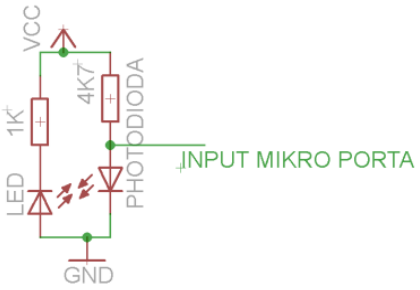


Gambar 3 Desain *Line maze solving robot*

Rangkaian pembangun robot terdiri atas sistem minimum ATmega 32, sensor garis dan *driver* motor. Pada sistem minimum ATmega 32 terdapat rangkaian sebagai berikut:

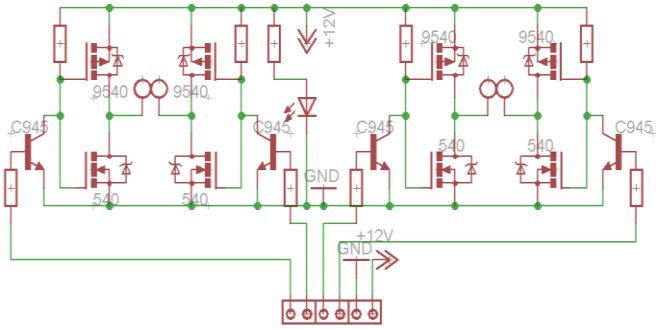
1. *Clock* Generator
2. *Power Supply*
3. Tombol *Reset* dan Tombol Konfigurasi
4. Sistem Monitoring

Sensor garis menggunakan LED sebagai pemancar cahaya (*transmitter*) dan Fotodioda sebagai penerima cahaya (*receiver*). Skematik dari sensor garis ditunjukkan pada gambar 4.



Gambar 4 Skematik Sensor Garis

Rangkaian utama dari *driver* motor terdiri atas *Mosfet* IRF 9540, *mosfet* IRF 540 dan *transistor* C945. Rangkaian *driver* motor ditunjukkan pada gambar 5.

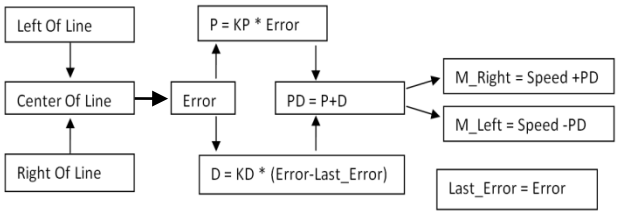


Gambar 5 Rangkaian *driver* motor

B. Perancangan *Software*

1. Perancangan Pengendali Kecepatan Motor

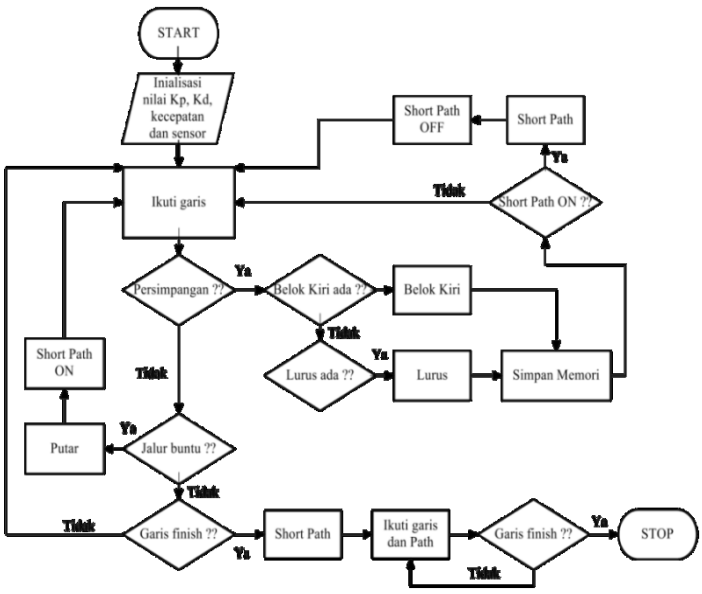
Perancangan pengendali kecepatan motor berdasarkan pada pengendali PID. Pengendali ini bukan merupakan pengendali PID murni melainkan hasil modifikasi menjadi pengendali kecepatan motor. Pengendali kecepatan motor berfungsi untuk membuat pergerakan robot dapat mengikuti garis dan menghasilkan pergerakan yang stabil. Secara garis besar urutan pengendalian kecepatan motor robot ditunjukkan pada gambar 7.



Gambar 7 Blok diagram pengendali kecepatan motor

2. Perancangan *Algoritma Short Path Finder*

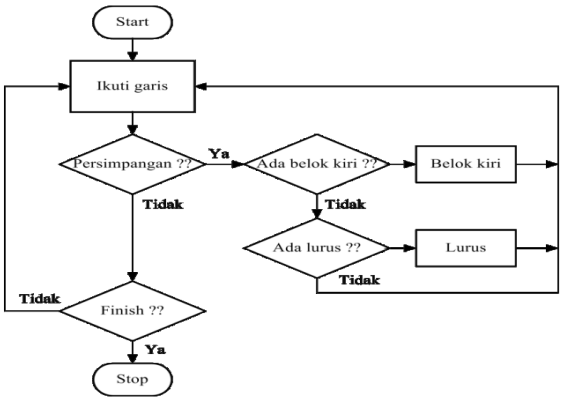
Algoritma *Short Path Finder* terdiri atas 3 mode yaitu mode *search,* mode *short path* dan mode *return and path. Flowchart* dari Algoritma *Short Path Finder* ditunjukkan pada gambar 8.



Gambar 8 *Flowchart* Algoritma *Short Path Finder*

Mode *Search*

Dalam mode ini robot akan melakukan pencarian *finish* dengan metode logika kanan atau logika kiri. Pada perancangan ini robot menggunakan logika kiri sebagai mode pencarian. Robot akan terus menerus melakukan belokan yang menjadi prioritasnya yaitu belok kiri. Robot akan memilih belok kiri jika menemui persimpangan dan memilih lurus sebagai prioritas kedua jika tidak ada belok kiri. Mode *search* diambil dari Algoritma *Wall Follower* dan *Left/Right Hand Rule*. Urutan prioritasnya ditunjukkan pada gambar 9.



Gambar 9 Urutan prioritas logika kiri

Mode *Short Path*

Mode yang kedua adalah mode untuk menyederhanakan lintasan yang dilalui robot dan mendapatkan jalur terpendek. Mode *short path* dipanggil saat robot menemui jalan buntu. Dengan menghilangkan jalur yang terdapat jalan buntu tersebut maka robot akan mendapatkan jalur terpendek dari lintasan.

Dengan membangkitkan suatu nilai tertentu terhadap pergerakan robot apabila menemui persimpangan dan jalan buntu maka akan dapat dilakukan penyederhanaan lintasan. Tabel 3 merupakan tabel pemberian angka pergerakan robot jika menemui persimpangan dan jalan buntu.

Tabel 3 Pemberian nilai pada gerakan robot.

Gerakan	**Inialisasi Gerakan**
Belok Kiri	2
Jalan Lurus	3
Belok Kanan	4
Putar Kanan	5

Setiap gerakan robot disimpan dalam memori tertentu dan dipanggil jika menemui jalan buntu. Gerakan robot ini nantinya disederhanakan untuk mendapatkan jalur terpendek. Penyederhanaan gerakan robot terdapat pada tabel 4.

Tabel 4 Penyederhanaan lintasan

Gerakan	**Short Path**
2-5-2	3
2-5-3	4
3-5-2	4
3-5-2-5-2 4-5-2	5
2-5-2-5-3 3-5-3	5

Hasil dari mode ini adalah didapatkan jalur terpendek lintasan yang dilewati robot. Jalur terpendek ini digunakan sebagai jalur terpendek untuk kembali menuju ke *start*.

Mode *return and path*

Mode *return and path* adalah mode untuk memanggil jalur terpendek yang telah didapat dan menerapkannya pada lintasan untuk kembali ke *start*. *Path* dilakukan jika robot menemui persimpangan. *Path* didapatkan dari penyederhanaan lintasan yang dilalui oleh robot pada saat melakukan pencarian *finish*. *Path* merupakan gerakan tertentu yang harus diambil oleh robot jika menemui persimpangan. Gerakan tersebut adalah belok kiri, belok kanan dan lurus. Gerakan tersebut disimpan dalam memori mikrokontroler dalam bentuk kode-kode angka. Kode-kode angka tersebut ditunjukkan pada tabel 5.

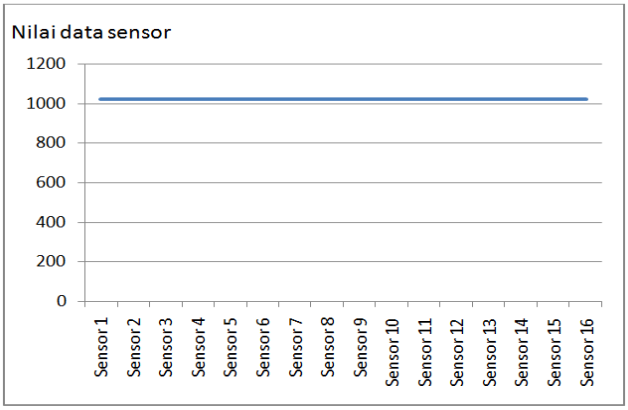
Tabel 5 Kode-kode pada *path*

Gerakan	**Kode *path***
Belok Kiri	2
Jalan Lurus	3
Belok Kanan	4

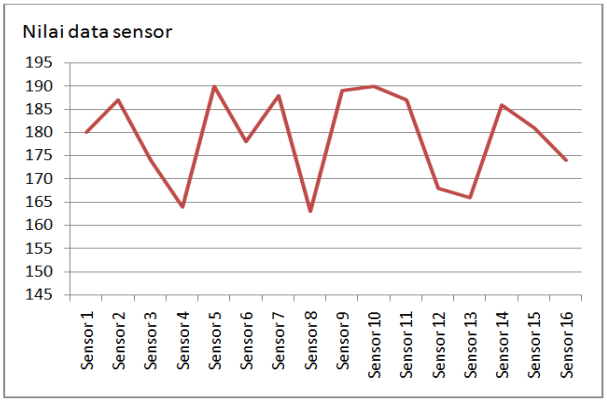
IV. PENGUJIAN DAN ANALISIS

A. Pengujian dan Analisis Sensor Garis

Hasil pengukuran data ADC pada lintasan garis warna hitam dan putih ditampilkan pada grafik gambar 10 dan 11.



Gambar 10 Grafik pembacaan sensor pada garis warna hitam



Gambar 11 Grafik pembacaan sensor pada garis warna putih

Pengambilan data ADC dilakukan pada kondisi ruang tertutup dengan penerangan lampu normal. Dari gambar grafik data pengujian, jarak pembacaan sensor pada warna lintasan hitam dan putih memiliki beda data rata-rata 800 angka. Jarak pembacaan yang lebar dapat menambah akurasi sensor dalam menentukan garis warna putih dan hitam.

Tingkat kehandalan sensor garis untuk membedakan warna lintasan hitam dan putih sangat dipengaruhi cahaya luar karena menggunakan sensor cahaya. Semakin sedikit cahaya dari luar maka semakin baik kinerja dari sensor cahaya. Peredam cahaya diberikan pada bagian samping sensor untuk membatasi jumlah cahaya luar yang masuk. Oleh karena itu, sensor garis hanya menerima cahaya dari bawah robot yang berasal dari cahaya pantulan lintasan.

B. Pengujian dan Analisis Driver Motor

Hasil pengujian *driver* motor ditampilkan pada tabel 6 dan 7.

Tabel 6 Hasil pengujian arah putaran motor.

***Direction* A**	***Direction* B**	**Gerakan Motor**
0	0	Tidak Bergerak (mengerem)
0	1	Maju
1	0	Mundur
1	1	Tidak Bergerak (mengerem)

Tabel 7 Hasil pengujian kecepatan putaran motor.

PWM (*Range* 0-255 Tegangan 12 Volt)	**RPM**
50	294
100	588
150	882
200	1176
250	1470

Berdasarkan hasil pengujian pada tabel 4.1 dan 4.2, *driver* motor dapat berfungsi dengan baik karena dapat mengatur arah putaran dan kecepatan motor sesuai perintah yang diberikan. *Driver* motor tipe *H-bridge* dengan *mosfet* jenis IRF540 dan IRF9540 dapat memberikan respon pergerakan motor yang cepat pada robot. Oleh karena itu, robot cepat dalam memberikan respon terhadap kondisi jalur pada lintasan.

C. Pengujian dan Analisis Sistem Minimum Mikrokontroler ATmega32

Hasil pengujian sistem minimum ditunjukkan pada tabel 8 dan hasil pengujian level tegangan ditunjukkan pada tabel 9.

Tabel 8 Hasil pengujian Sistem Minimum.

Pengujian	**Hasil**
LCD	Dapat Berfungsi dengan Baik
Tombol Konfigurasi	Dapat Berfungsi dengan Baik
Tombol *Reset*	Dapat Berfungsi dengan Baik
Port A	Dapat Berfungsi dengan Baik
Port B	Dapat Berfungsi dengan Baik
Port C	Dapat Berfungsi dengan Baik
Port D	Dapat Berfungsi dengan Baik
PWM	Dapat Berfungsi dengan Baik
ADC	Dapat Berfungsi dengan Baik
Port Downloader	Dapat Berfungsi dengan Baik

Tabel 9 Hasil pengukuran level tegangan.

Pengecekan Tegangan	**Tegangan (Volt)**
Baterai	12,6
Mikrokontroler	4,95
Sensor depan dan samping	4,95
Driver motor	12,3
LCD	4,95

Berdasarkan hasil pengujian, Sistem Minimum Mikrokontroler ATmega32 dapat berfungsi secara normal pada semua fitur yang digunakan. Sistem minimum juga dapat menjalankan sistem monitoring dan tombol dengan baik. *Power supply* dapat memberikan tegangan yang stabil dan dapat menyuplai tegangan ke semua rangkaian. Pada pengujian level tegangan terjadi *drop* tegangan sebesar 0,3 volt antara tegangan baterei dan *driver* motor. *Drop* tegangan yang terjadi disebabkan

komponen dioda yang memiliki sifat menurunkan tegangan sebesar 0,3 volt.

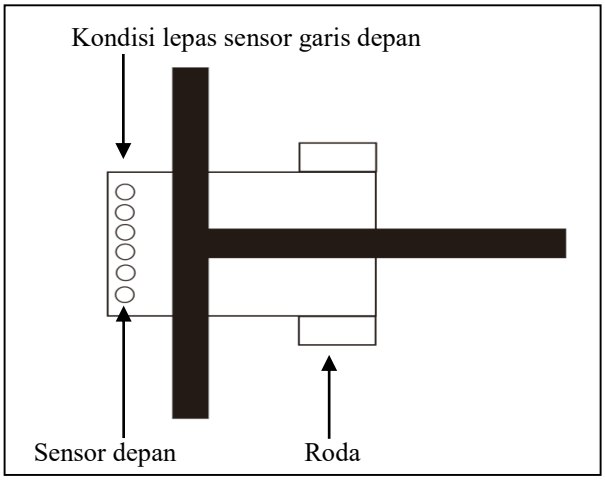
D. Pengujian dan Analisis Pengendali Kecepatan Motor

Gerakan yang sering dilakukan robot meliputi maju lurus (*line following*), belok kanan 90°, belok kiri 90° dan putar kanan 180°. Komponen yang terpenting dalam melakukan gerakan tersebut adalah sensor garis. Sensor garis menentukan kapan robot harus melakukan gerakan dan menghentikan gerakan. Pengujian dilakukan pada semua kondisi lintasan yang akan dilewati robot *line maze*. Hasil pengujian ditunjukkan pada tabel 10.

Tabel 10 Pengujian berbagai macam kondisi lintasan.

Jenis kondisi lintasan	Hasil
Belokan 90°	Keberhasilan 100 %
Perempatan	Keberhasilan 100 %
Pertigaan tipe 1	Keberhasilan 100 %
Pertigaan tipe 2	Keberhasilan 80 %
Pertigaan tipe 3	Keberhasilan 100 %
Pertigaan tipe 4	Keberhasilan 100 %
Garis buntu	Keberhasilan 100 %

Dari hasil pengujian didapatkan hasil bahwa robot sedikit mengalami kendala saat menjumpai pertigaan tipe 2 (Gambar 4.3). Hal ini disebabkan tidak ada garis depan untuk membuat posisi robot lurus. Posisi robot yang tidak lurus membuat sensor samping tidak mendeteksi garis secara bersamaan dan tidak mendeteksi adanya pertigaan. Kondisi lepas sensor garis depan ditunjukkan pada gambar 11.



Gambar 11 Kondisi robot saat keadaan lepas sensor depan

Pengendali mempengaruhi pergerakan robot untuk mengikuti jalur hitam pada lintasan. Pengendali berfungsi dalam meluruskan kondisi robot terhadap jalur pada lintasan dengan cara mengatur putaran motor kiri dan kanan.

Menambah nilai KP membuat robot mengalami osilasi tetapi mempercepat respon terhadap perubahan kondisi lintasan. Penambahan nilai KD dapat mengurangi osilasi tetapi membuat robot bergerak patah. Untuk nilai KP dan KD yang terbaik adalah pada angka KP = 5 dan KD = 15 pada kondisi kecepatan 100 dari 255. Nilai ini didapat dari hasil percobaan di lintasan secara berulang dengan merubah nilai KP, KD dan kecepatan sampai mendapatkan pergerakan yang paling bagus.

E. Pengujian dan Analisis Algoritma Short Path Finder

Hasil pengujian Algoritma *Short Path Finder* ditunjukkan pada tabel 11 untuk lintasan tipe maju dan tabel 12 untuk lintasan tipe melingkar.

Tabel 11 Hasil pengujian Algoritma *Short Path Finder* pada Lintasan bertipe lurus

Lintasan	Kesalahan	Keberhasilan
1	-	Sukses kembali
2	-	Sukses kembali
3	-	Sukses kembali
4	-	Sukses kembali
5	-	Sukses kembali
6	-	Sukses kembali
7	-	Sukses kembali

Tabel 12 Hasil pengujian Algoritma *Short Path Finder* pada Lintasan bertipe melingkar

Lintasan	Kesalahan	Keberhasilan
1	-	Sukses kembali
2	-	Sukses kembali

Berdasarkan hasil pengujian pada lintasan berarah maju, robot dapat melalui semua kondisi lintasan dan berhasil mendapatkan jalur terpendek lintasan. Pada kondisi lintasan jenis melingkar robot dapat kembali ke *finish* tetapi bukan merupakan jalur terpendek. Hal ini terjadi dikarenakan robot hanya menyederhanakan lintasan yang dilaluinya. Selain itu, robot juga menggunakan metode logika kiri dalam mencari *finish* sehingga pasti akan mengambil belok kiri yang menjadi prioritasnya.

Algoritma *Short Path Finder* yang diterapkan pada robot umumnya dapat

melakukan pencarian dan menemukan jalur terpendek dari *finish* ke *start* dengan baik. Kesalahan umumnya terjadi pada kondisi pertigaan tipe 2 yaitu kondisi lepas sensor garis depan. Hal tersebut membuat robot salah dalam menentukan jenis lintasan dan membuat salah dalam menentukan jalur terpendek. Untuk mengatasi hal tersebut dilakukan penambahan sensor samping menjadi tiga buah masing-masing pada samping kanan dan kiri. Selain itu, dilakukan pembacaan sensor secara berulang untuk menambah akurasi pembacaan garis.

V. KESIMPULAN

Berdasarkan hasil penelitian, diperoleh kesimpulan sebagai berikut:

1. Hasil penerapan algoritma untuk sistem pencarian jalur terpendek, optimal diterapkan pada lintasan bertipe maju dengan tingkat keberhasilan 100%.
2. Algoritma *Short Path Finder* tidak bisa diterapkan pada jenis lintasan melingkar.
3. Perlu pemasangan peredam cahaya pada sensor garis karena mudah terpengaruh cahaya luar dan untuk menambah akurasi dalam menentukan jenis lintasan.
4. Sensor garis yang bagus dengan tingkat akurasi yang bagus adalah apabila jarak data nilai ADC antara data warna hitam dan putih semakin lebar atau jauh.
5. Perlu mikroprosesor yang lebih cepat dan memori yang lebih besar untuk mampu menyelesaikan *maze* yang besar dan jarak antar jalur *maze* yang pendek.

DAFTAR PUSTAKA

Bekti Harapan, Samudra, 2009. "*Pencarian Shortest Path Dinamik dengan Algoritma Bellman-Based Flood Fill dan Implementasinya pada Robot Micromouse*", Skripsi, Sekolah Teknik Elektro dan Informatika (STEI), Institut Teknologi Bandung (ITB), Bandung.

Braunl, Thomas, 2008. "*Embended Robotics: Mobile Robot Design and Application with Embedded System"*. Perth: Springer.

Darmawan, Arif, 2010. "*Penerapan Algoritma Pledge untuk Menyelesaikan Maze pada Line Follower Robot",* Skripsi, Politeknik Elektronika Negeri Surabaya (PENS), Institut Teknologi Sepuluh November (ITS), Surabaya.

Fahmizal, 2011. "*Implementasi Sistem Navigasi Behavior Based dan Kontroler PID pada Manuver Robot Maze",* Skripsi, Fakultas Teknologi Industri (FTI), Institut Teknologi Sepuluh November (ITS), Surabaya.

Hartanto, Thomas Wahyu Dwi., Y. Wahyu Agung Prasetyo, 2002. *"Analisis dan Desain Sistem Kontrol dengan Matlab"*. Yogyakarta: Andi.

Iqbal, Muhammad, 2009. "*Penerapan Algoritma Maze Mapping untuk Menyelesaikan Maze pada Line Follower Robot",* Skripsi, Politeknik Elektronika Negeri Surabaya (PENS), Institut Teknologi Sepuluh November (ITS), Surabaya.

McCabe, Patrick. "*How To Have a Robot Maze Solve".* Diakses pada 13 Februari 2014. Dari http://www.patrickmccabemakes.com/PatrickMccabeMakes/Mazesolving.html

Philips, Charles L., Harbor, Royce D., Widodo RJ, 1996. *"Sistem Kontrol : Dasar-dasar".* Jakarta: PT Prehanllindo.

Rahman, Abdullah, 2010. "*Penerapan Algoritma Flood Fill untuk Menyelesaikan Maze pada Line Follower Robot",* Skripsi, Politeknik Elektronika Negeri Surabaya (PENS), Institut Teknologi Sepuluh November (ITS), Surabaya.

Sigit, Riyanto, 2007. "*Robotika, Sensor & Aktuator"*. Surabaya: Graha Ilmu.

Vannoy II, Richard T, 2009. *Design a Line Maze Solving Robot: Teaching a Robot to Solve a Line Maze.* Diakses 09 Februari 2014. Dari http://www.richardvannoy.info/line-maze.php

_____.*Datasheet ATmega32*.

_____.*Datasheet IC 4052*.