REKAYASA
Journal of Science and Technology
https://journal.trunojoyo.ac.id/rekayasa

Rekayasa, 2025; 18(3): 297-313
ISSN: 0216-9495 (Print)
ISSN: 2502-5325 (Online)

Optimisasi Rute Penyelamatan Robot Mobil dalam Labirin: Algoritma Pencarian Jalur Terpendek

Sri Wahyuni [1]*, Muhammad Fuad[1], Khanif Khoirul Umam[1], Lailatus Sya'diyah[1], Deni Tri Laksono[2]
[1]Program Studi Teknik Mekatronika
[2]Program Studi Teknik Elektro
Fakultas Teknik Universitas Trunojoyo Madura
Kampus UTM Jl. Raya Telang No 02 Kamal Bangkalan 69162 Jawa Timur
*E-mail Korespondensi : s.wahyuni@trunojoyo.ac.id
DOI: https://doi.org/10.21107/rekayasa.v18i3.23810
Submitted August 30th 2024, Accepted May 16th 2025, Published December 7th 2025

Abstrak

Suatu area di mana terjadi kebakaran umumnya tertutup asap, udara panas dan berbahaya bagi tim pemadam untuk memasuki ruangan. Teknologi bidang robotika khususnya robot mobil sangat memungkinkan dijadikan sebagai penyelamat pengganti peran tim pemadam kebakaran. Tugas robot mobil penyelamat adalah menelusuri area secara menyeluruh dan selanjutnya pada perjalanan mengulang berikutnya robot mobil mampu menemukan jalur terpendek dengan membuang jalur yang menuju buntu. Kemampuan robot dalam melakukan pemetaan dan memilih jalur terpendek inilah yang menjadi tujuan penelitian. Lintasan berupa garis hitam bersimpang dengan sudut 90˚. Metode yang diterapkan untuk mencapai tujuan pada penelitian ini ialah *maze mapping algorithm left hand rule.* Metode *maze solving algorithm left hand rule* dipilih sebagai solusi untuk melakukan perencanaan jalur terpendek pada lingkungan berbentuk labirin dengan membuat kode-kode seperti L (*Left*), S (*Straigh*) dan U (*Dead End*). Setelah tersusun kode-kode dari proses pemetaan selanjutnya dilakukan penyederhanaan kode untuk robot menemukan jalan keluar dari labirin dengan jalur terpendek. Tingkat keberhasilan sebesar 90%, dari sepuluh kali pengujian menggunakan metode *maze mapping left hand rule* menghasilkan sembilan kali berhasil dan sekali gagal. Waktu *mapping* tercepat ditempuh dalam 23 detik dan rata-ratanya 62 detik, sedangkan *shortest time* 8 detik dan rata-ratanya 27 detik.

Kata Kunci: jalur terpendek, pemetaan labirin, photodioda, robot penyelamat

Abstract

An area where a fire occurs is generally covered in smoke, hot air and dangerous for firefighters to enter. Robotics technology, especially mobile robots, makes it possible to be used as a rescuer to replace the role of the firefighter team. The task of the rescue mobile robot is to thoroughly explore the area and then on the next iteration, the mobile robot is able to find the shortest path by eliminating paths that lead to dead ends. The robot's ability to map and choose the shortest path is the goal of this research. The path is a black line intersecting at a 90° angle. The method applied to achieve the objectives in this research is the maze mapping algorithm left hand rule. The maze solving algorithm left hand rule method was chosen as a solution to plan the shortest path in a maze-shaped environment by creating codes such as L (Left), S (Straigh) and U (Dead End). After the codes were compiled from the mapping process, the code was simplified so that the robot could find a way out of the maze with the shortest path. The success rate was 90%, out of ten tests using the maze mapping left hand rule method, resulting in nine successes and one failure. The fastest mapping time was 23 seconds and the average was 62 seconds, while the shortest time was 8 seconds and the average was 27 seconds.

Key words*:* shortest path, maze mapping, photodioda, rescuer robot

PENDAHULUAN

Dunia telah berevolusi sedemikian canggihnya, manusia berusaha menanamkan cara pikir otak yang cerdas dan adaptif ke berbagai mesin. Babak baru memasuki era yang sering disebut *industry* 4.0, di mana *machine learning* (Pamungkas *et al.*, 2024) dan *artificial intelligence* (Vartak, 2023) dikembangkan secara tak terbatas ke berbagai bidang. Tak hanya industri dengan mesin-mesin otomatisnya (Mukti , 2018; Sitepu & Putra, 2022; Pasaribu & Yurinanda, 2024), juga urusan penyelamatan dan pengamanan manusia seperti dinas pemadam kebakaran (Kaharsyah, 2023). Suatu area di mana terjadi kebakaran umumnya tertutup asap, udara panas dan berbahaya bagi tim pemadam (Sriyana, 2022) untuk memasuki ruangan. Reruntuhan atap bangunan terbakar dikuatirkan menimpa tim pemadam dan akhirnya menambah panjang daftar

korban yang seharusnya bisa dicegah (Safittri, 2020). Pada kondisi tersebut dibutuhkan penyelamatan secara cepat, efisiensi sumber daya (Farouq *et al.,* 2021), mengatasi hambatan fisik, keamanan robot dan operator serta pemetaan dan navigasi *real time* (Wahyuni, 2013; Fuad & Wahyuni, 2022; Wahyuni *et al.,* 2021; Wahyuni *et al.,* 2023). Teknologi bidang robotika khususnya robot mobil sangat memungkinkan dijadikan sebagai penyelamat pengganti peran tim pemadam kebakaran (Mohammed *et al.,* 2022) (Suhendar *et al.,* 2020). Robot mobil diterjunkan ke kawasan terjadinya kebakaran untuk menelusuri setiap lintasan dalam misi pencarian korban yang terjebak api. Dengan demikian, peran berbahaya dari tim pemadam kebakaran bisa diminimalkan. Salah satu tahapan menciptakan robot mobil penyelamat (Maulana *et al.,* 2022) adalah melakukan penelitian terkait bagaimana robot akan menelusuri area secara menyeluruh tanpa mengulang dan tersesat pada jalur terpendek (Prakash & Naganjaneyulu, 2021) .

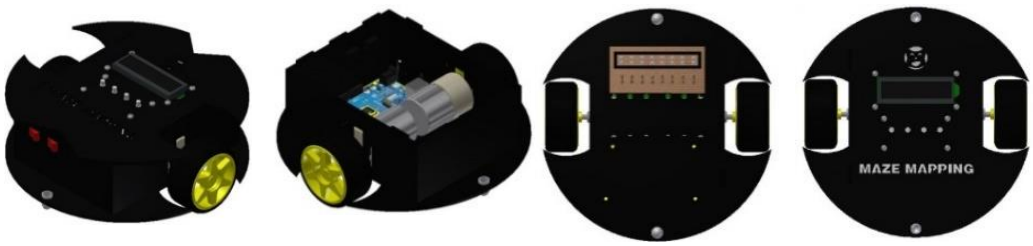
Teknik penelusuran ini telah banyak dijadikan salah satu topik pembahasan pada penelitian mobil robot, karena menelusuri jalur-jalur yang memiliki struktur rumit membutuhkan keakuratan posisi (Priambodo *et al.,* 2024). Kemampuan untuk mendeteksi sekitar dan berjalan sesuai aturan *(rule)* merupakan syarat utama dari pergerakan robot ini (Hendriawan & Akbar, 2010). Kemampuan tersebut dapat dicapai apabila sensor yang tertanam pada robot dapat terintegrasi baik dengan metode yang diterapkan pada sistem (Rufus *et al.,* 2024, Wulandari *et al.,* 2022), sehingga robot memiliki respon cepat terhadap perubahan lingkungan disekitarnya. Berdasarkan uraian permasalahan tersebut maka riset ini akan mendesain dan membuat sebuah robot yang memiliki kecerdasan buatan (Prakash & Naganjaneyulu, 2021) untuk melakukan penelusuran area secara *real time* yang disimulasikan dalam lingkungan labirin hitam putih. Dengan harapan menghasilkan sebuah robot yang memiliki kemampuan khusus untuk merekam suatu jalur yang dilewati (Kitsak *et al.,* 2023; Latif *et al.,* 2023). Kemudian, nantinya robot akan mengetahui letak jalan buntu dan menghindari jalur tersebut untuk kedua kalinya.

Agar dapat memudahkan pergerakan robot sewaktu melakukan penjelajahan medan *(tracking)*, robot dibekali dengan dua buah roda aktif yang bergerak secara sinergi dan dua roda dinamis yang bergerak bebas *(free wheels)*. Selain itu sistem harus mengetahui di mana letak posisi robot berada, sehingga akan mencegah terjadinya *cutting line* atau keluarnya robot dari lintasan. Maka dari itu robot dibekali dengan sensor atau transduser yang dapat memantulkan dan menerima gelombang cahaya. Sehingga pantulan dari sensor akan menghasilkan perbedaan nilai warna garis yang terbaca oleh robot. Selain itu sistem memiliki tujuan untuk melakukan penyisiran terhadap semua lokasi di dalam lintasan (Mohammed *et al.,* 2022, Zarkasi *et al.,* 2019) hingga mencapai titik yang diinginkan. Hal ini bertujuan agar robot dapat mengetahui jalur terpendek yang akan dilalui (Mohammed *et al.,* 2022; Yew-Hao *et al.,* 2021) setelah proses pembelajaran berlangsung. Untuk melakukan semua itu tentunya dibutuhkan algoritma (Alamri *et al.,* 2021) yang cocok untuk mengetahui tata cara agar robot dapat melakukan pengecekan terhadap semua lokasi lintasan. Metode *maze solving algorithm left hand rule* (Mohammed *et al*., 2022; Yew-Hao *et al.,* 2021) dipilih sebagai solusi untuk melakukan perencanaan jalur terpendek pada lingkungan berbentuk labirin.

METODE PENELITIAN

Rancangan Konstruksi Robot

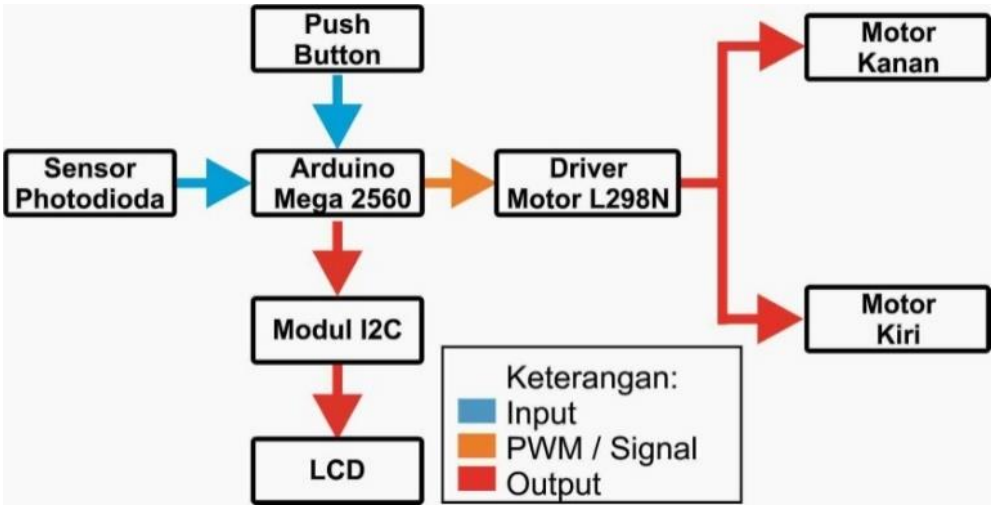
Rancangan robot mobil ditampilkan pada Gambar 1. Tahap awal pada penelitian yaitu membuat rancangan robot mobil yang dilengkapi dengan komponen antara lain : 16 sensor cahaya, 2 roda utama, 1 *gear box*, mikrokontroler, baterei dan lain-lain. Rangka mekanik robot terbuat dari akrilik hitam, berbentuk melingkar dengan diameter ± 20 cm.



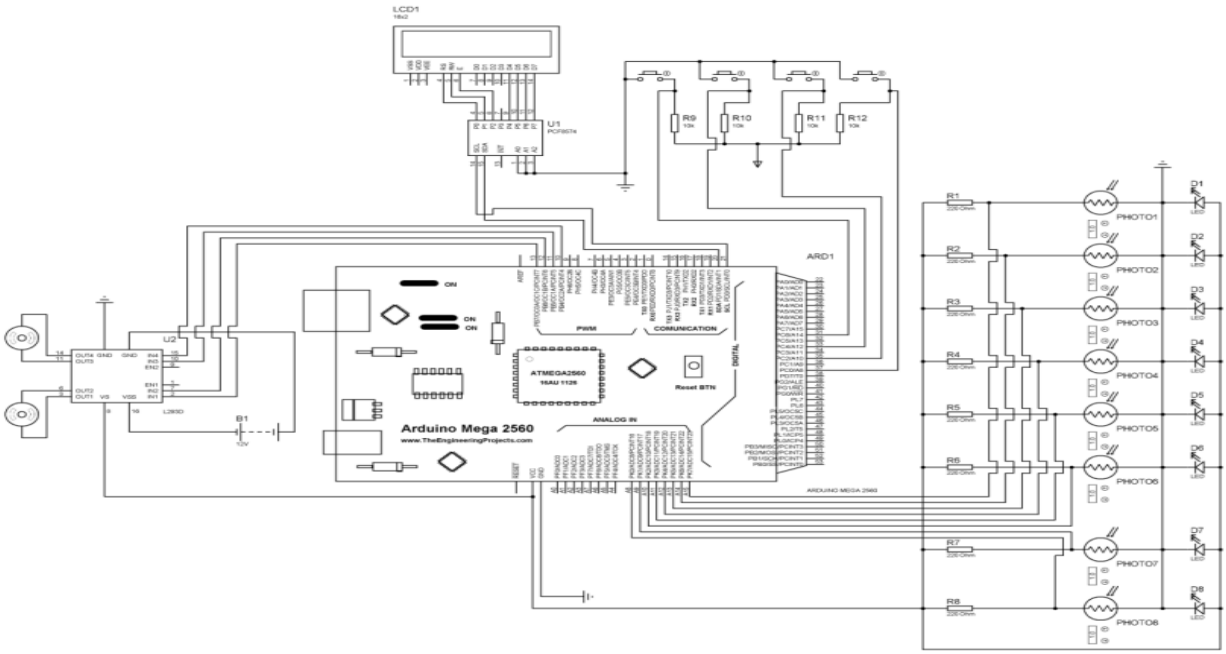
Gambar 1 . Desain robot mobil menggunakan model kinematika differential-steering

Rangkaian Elektronika

Terdapat beberapa komponen penyusun yang digunakan dalam perancangan elektronika ini diantaranya arduino mega 2560, sensor photodioda, *driver* motor dan motor DC *(direct current)* 12 volt (Prayoga & Sunggu, 2024). Misi utama dari rangkaian elektronika nantinya adalah membuat sistem dengan kerja otomatis. Gambar 5 adalah diagram elektronika dari robot mobil yang telah dibuat. Terdapat beberapa macam kode warna pada diagram blok. Kode warna biru merupakan *input* yang dihasilkan oleh dua komponen elektronika diantaranya *push button* dan sensor photodioda. *Push button* akan megirimkan sinyal berupa nilai biner antara 0 dan 1. Sedangkan sensor photodioda dapat mengirimkan sinyal berupa nilai ADC antara 1 hingga 1000 berdasarkan warna yang terbaca oleh sensor. Kode merah adalah *output* yang dihasilkan dari pengolahan data oleh arduino mega 2560 yang diterjemahkan oleh *driver* motor L293D dan disalurkan pada dua buah aktuator. Setelah membuat diagram blok sistem (Gambar 2) kemudian peneliti melakukan perancangan rangkaian elektronika secara virtual terlebih dahulu. Hasil perancangan dapat dilihat pada Gambar 3.



Gambar 2. Diagram Blok Sistem Mobile Robot



Gambar 3. Skema Rangkaian Elektronika Mobile Robot

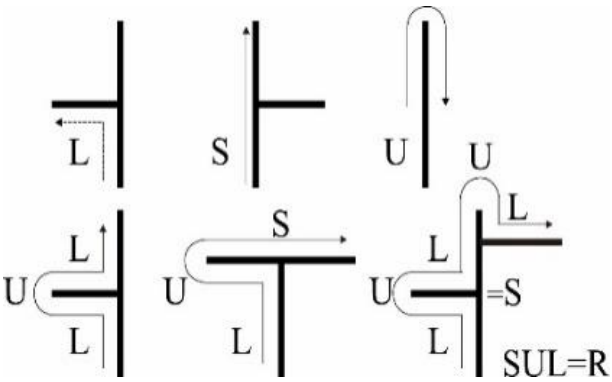
Rancangan Algoritma *Maze Mapping*

Metode ini bertugas untuk mengetahui posisi dan dapat menyimpan data sementara pada memori robot. Sehingga pada saat implementasi, sistem dapat belajar dan menganalisis data-data yang diperoleh selama pemetaan berlangsung. Selain itu sistem harus secara otomatis merumuskan dan menghasilkan kode yang memiliki jalur terpendek pada saat proses penjelajahan. Untuk penyimpan data, peneliti menggunakan fitur *array* pada arduino mega 2560 yang memiliki kapasitas maksimal yang dapat diatur oleh *user*. Pada saat sistem dioperasikan akan ada banyak kemungkinan yang terjadi. Misalkan robot bertemu persimpangan maka robot akan mengutamakan belok kiri. Dan apabila bertemu jalan buntu maka robot secara otomatis mengkodekan jalur tersebut dengan kode "U" kemudian robot akan berputar 1800. Proses inilah yang didefinisikan sebagai proses pemetaan atau pembelajaran. Kode-kode yang tersusun dari awal perjalanan (*start*) akan menjadi data penting bagi robot untuk mencapai tujuan yang diharapkan.

Pada saat proses pemetaan berlangsung robot dapat belajar dari jalur yang dilewatinya dan menyimpan data sementara kedalam memori dalam bentuk *array*. Proses ini merupakan pemetaan dari jalur yang telah dilewati menggunakan kode-kode khusus untuk disimpan dan diolah menjadi data terbaru (Hendriawan & Akbar, 2010). Sehingga dihasilkan sebuah formula untuk menuju titik tujuan melewati jalur terpendek. Kode-kode yang digunakan ialah sebagai berikut:

1. ***L*** adalah ***LEFT***. Pemberian kode ini apabila robot telah melakukan belok kiri karena terdapat persimpangan.
2. ***S*** adalah ***STRAIGHT*** atau jalan lurus. Kode ini diberikan apabila robot harus lebih memilih jalan lurus pada saat bertemu persimpangan.
3. ***U*** adalah ***DEAD END***. Berarti robot berjumpa dengan jalan buntu dan berjalan kembali ke persimpangan terakhir.

Setelah tersusun kode-kode dari proses pemetaan, kemudian dilakukan penyederhanaan atau konversi untuk menemukan jalan keluar dari labirin melalui jalan terpendek. Sehingga tersusun kode-kode baru sebagai jalan keluar menuju titik *finish*. Tips dari proses pemetaan menggunakan *maze mapping* ini apabila tidak ditemukan lagi kode "***U***" yang melambangkan jalan buntu/*dead end* dan robot telah menemukan kode "***D***" sebagai indikator titik *finish*, sebagaimana ditunjukkan pada Gambar 4 adalah contoh proses cara pemberian kode pada saat pemetaan berlangsung.



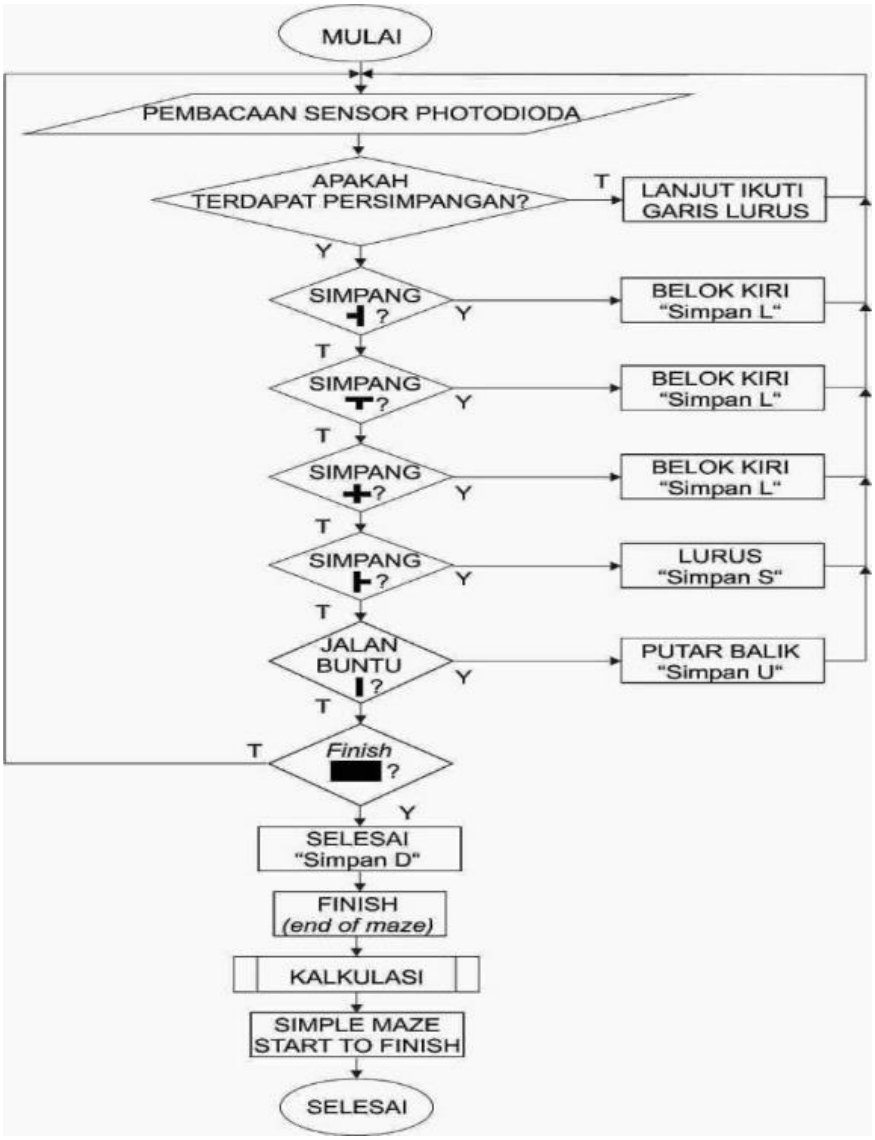
Gambar 4. Posisi Robot Pada Saat Pemberian Kode

Selesai pemetaan maka robot secara otomatis melakukan kalkulasi dari data yang tersimpan pada memori robot untuk mendapatkan jalur terpendek dalam menyelesaikan *line maze* pada penelusuran kedua kalinya. Penyederhanaan susunan kode-kode yang tersimpan diperlukan sebuah formulasi. Berikut ini merupakan formulasi konversi dari kalkulasi data yang tersimpan :

1. Konversi 1:
   L – U – L = S S – U – L = R L – U – S = R S – U – S = B

2. Konversi 2:
   S – B = U S – B – S = U L – B – S = R S – B – L = R R – B – L = U L – B – L = S
   L – U – R = B R – U – L = B L – B – R = B

Pada proses penyederhanaan dapat dilakukan dengan sekali konversi atau dua kali konversi. Langkah pertama dilakukan dengan aturan konversi 1, dan apabila hasil konversi 1 menemukan kode "**B**" maka akan dilakukan penyederhanaan kembali menggunakan aturan konversi 2. Gambar 5 merupakan *flowchart* sub proses kalkulasi data *maze mapping* menjadi jalur tercepat *(simple maze)*.



Gambar 5. Flowchart sub proses cek persimpangan

HASIL DAN PEMBAHASAN

Pembuatan Mekanik

Tahap perancangan selesai, selanjutnya peneliti melakukan pembuatan mekanik secara riil berdasarkan rancangan awal yang telah dibuat. Bahan yang digunakan sebagai kontruksi bodi robot pada penelitian ini adalah akrilik dengan ketebalan 3 mm. Penggunaan akrilik dengan tebal 3 mm bertujuan agar robot memiliki beban yang kecil, sehingga robot akan mudah melakukan manuver pada saat berjalan. Karena apabila beban robot berlebih maka akan memiliki dampak negatif berupa pergerakan yang lambat pada saat robot berjalan menelusuri lintasan. Hasil dari pembuatan mekanik *mobile robot* dapat dilihat pada Gambar 6.



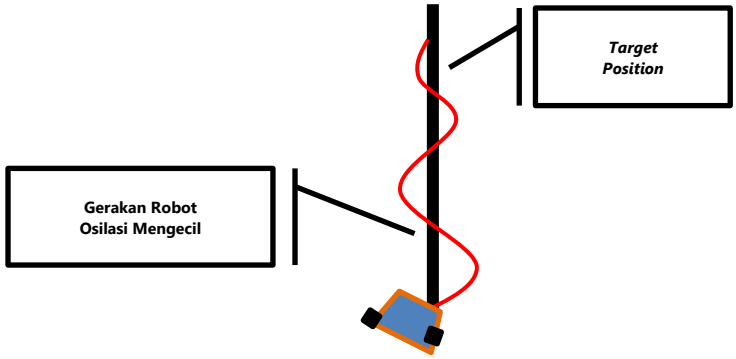
Gambar 6. Hasil pembuatan mekanik mobile robot

Perakitan Elektronika

Perakitan elektronika merupakan proses penggabungan komponen yang terpisah menjadi satu-kesatuan. Pada tahap ini peneliti menggunakan PCB dot matrik untuk menanamkan dan menyatukan seluruh komponen elektronika dengan timah dan solder. Beberapa komponen yang dirakit pada tahap ini diantaranya arduino mega 2560, motor DC, regulator 7905, sensor photodioda, resistor, saklar, sistem pengkabelan (*wire*), *driver* motor dan pin konektor. Untuk meminimalisir terjadinya *short* pada aliran listrik, digunakan pengaman berupa kabel bakar untuk membatasi antara kabel satu dan yang lainnya.

Penerapan Kontrol PID

Tujuan dari penerapan kontrol PID pada penelitian ini agar robot dapat menghasilkan mobilitas yang baik pada saat menelusuri *line maze* (Tanujaya et al., 2024). Diharapkan dengan penerapan metode ini robot selalu dapat memposisikan diri secara otomatis berada pada titik tengah dari garis. Luaran yang dihasilkan dari kontrol PID adalah nilai PWM untuk pengaturan kecepatan motor. Kontrol PID merupakan kombinasi kontrol P, I dan D (Amalia et al., 2023). Dengan menggabungkan kontrol tersebut akan menghasilkan kecepatan motor yang ideal (Fahmizal et al., 2024) dari yang diharapkan. Mekanisme kerja dari kontrol PID ini, pertama robot akan mengalami proses osilasi berdasarkan posisi robot terhadap garis. Sampai robot mencapai *target position* maka robot akan berjalan lurus hingga niai *error* kembali terjadi. Pola gerakan robot pada saat berosilasi dapat dilihat pada Gambar 7.



Gambar 7. Osilasi Pergerakan Robot

Dalam implementasinya, posisi sensor terhadap garis menginformasikan nilai *error* yang terjadi. *Error* memiliki bermacam-macam nilai yang bisa disebut dengan *present value (PV),* tergantung dari sensor mana yang membaca garis hitam. Nilai *present value* didapat dari hasil *tuning* pada saat melakukan pengujian. Nilai 1 adalah kondisi apabila sensor membaca hitam. Sedangkan nilai 0 adalah kondisi sensor membaca putih. Setelah didapat nilai maka selanjutnya diolah menggunakan aturan metode PID. Akan tetapi penelitian ini hanya menggunakan 2 variabel yaitu proporsional dan derivatif (PD). Langkah pertama yaitu peneliti menentukan konstanta proporsional dan derivatif yang akan digunakan untuk mengolah data kecepatan motor. Pada penelitian ini penentuan konstanta proporsional maupun derivatif dilakukan menggunakan aturan *tuning manual*. Artinya peneliti mencoba pergerakan robot secara *trial and error*. Setelah dilakukan percobaan didapat hasil yang baik dari pergerakan robot, dengan nilai konstanta

proporsional 0,6 dan konstanta derivatif 0,125. Nilai ini akan diolah kembali sehingga nantinya akan dihasilkan kecepatan PWM motor kanan dan kiri.

Selain melakukan uji coba secara langsung terhadap sistem, peneliti juga melakukan perhitungan secara manual untuk mengetahui dengan pasti proses dari aturan metode PD yang digunakan. Persamaan 1 hingga persamaan 5 merupakan perhitungan secara manual metode PD berdasarkan masukan dari sensor pada saat melakukan pengujian *mobile robot*:

$$Error = Set\ Point(SP) - Present\ Value(PV) \quad (1)$$

$$PD = KP \times Error + KD \times (Error - Last\ Error) \quad (2)$$

$$Last\ Error = Error \quad (3)$$

$$PWMKa = PWMref - PD \quad (4)$$

$$PWMKa = PWMref - PD \quad (5)$$

Pada tahap ini peneliti melakukan pengujian sebanyak sembilan kali menggunakan aturan proporsional dan derivatif. Pengujian 1 hingga pengujian 9 merupakan hasil perhitungan secara manual *output* kecepatan motor kanan (PWMka) dan kiri (PWMki). Pengujian ini menggunakan metode PD yang diuji cobakan berdasarkan posisi sensor ketika mendeteksi hitam. Pengujian kontrol proporsional derivatif berdasarkan posisi sensor menghasilkan kondisi belok kanan sangat tajam karena motor kiri memiliki kecepatan terpaut jauh lebih tinggi dibandingkan motor kanan selisih perbedaan kecepatan 17,5. Karena PV yang diperoleh berdasarkan posisi sensor adalah -10. Berdasarkan perhitungan secara matematis pada pengujian 1 hingga pengujian 9. Keseluruhan hasil nilai PWM disajikan pada Tabel 1.

Tabel 1. Hasil Pengujian Kontrol PD

Pengujian	Hasil PD	Output PWM Kiri	Output PWM Kanan	Selisih PWM
1	0	123	120	3
2	-2,175	120,825	122,175	1,35
3	-5,075	117,925	125,075	7,15
4	-5,8	117,2	125,8	8,6
5	-7,25	115,75	127,25	11,5
6	2,175	125,175	117,825	7,35
7	5,075	128,075	114,925	13,15
8	5,8	128,8	114,2	14,6
9	7,25	130,25	112,75	17,5

Penerapan metode proporsional derivatif pada robot *maze mapping* menghasilkan mobilitas yang baik pada saat berjalan menelusuri garis hitam. Robot berhasil berosilasi dengan baik hingga sensor tengah mendeteksi garis hitam. Implementasi metode PD ini memiliki peranan yang sangat penting bagi robot *line follower* yang memiliki tujuan untuk memetakan persimpangan. Karena dibutuhkan keakuratan dalam memetakan garis. Tentunya hal ini harus didukung oleh pergerakan robot yang stabil. Apabila hal ini tidak diperhatikan maka robot akan memiliki tingkat kesalahan yang besar.

Implementasi Metode *Maze Mapping*

Implementasi metode *maze mapping* dilakukan pada lintasan bergaris hitam dengan ukuran 2x2 meter persegi. Tebal garis yang digunakan pada penelitian ini memiliki ketebalan 1,5 cm. Ukuran tersebut disesuaikan dengan jarak antar sensor photodioda pada robot. Karena setelah melakukan uji coba pada penerapan metode proporsional derivatif diatas, tingkat ketebalan garis sangat mempengaruhi pembacaan sensor terhadap garis. Tentunya hal ini akan mengakibatkan tingkat kesalahan yang besar apabila sistem mulai melakukan pemetaan lintasan.

Pengujian metode ini dilakukan pada empat tipe lintasan yang berbeda. Lintasan dengan tingkat persimpangan yang mudah dan tingkat persimpangan yang relatif sulit. Hal ini dilakukan untuk menganalisis metode yang diterapkan apakah memiliki tingkat keberhasilan atau sebaliknya. Pada awal implementasi metode *maze mapping* ini, peneliti menggunakan delapan sensor photodioda, yang terletak

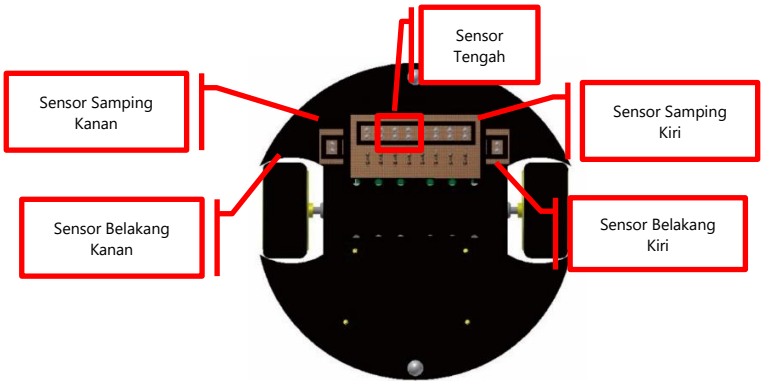
secara horizontal pada bagian depan robot. Kedelapan sensor tersebut digunakan untuk mendeteksi garis dan menyimpan data berupa persimpangan kedalam memori robot.

Pada pengujian pertama ini peneliti melakukan uji coba untuk menyimpan dan memberi inisialisasi terhadap masing-masing data. Kode L merupakan inisialisasi yang diberikan apabila robot mendeteksi belokan ke kiri. Kode R diberikan apabila sensor mendeteksi adanya belokan kekanan. Kode U merupakan inisialisasi apabila robot menemukan jalan buntu dengan indikator kedelapan sensor tidak mendeteksi adanya garis hitam.

Setelah dilakukan pengujian, ternyata terdapat beberapa kendala yang dialami, apabila menggunakan sensor yang memiliki letak secara horizontal. Kendala-kendala tersebut dijabarkan sebagai berikut:

1. *Mobile robot* tidak dapat membedakan antara lintasan yang berbelok ke kanan dengan persimpangan ke kanan
2. *Mobile robot* tidak dapat membedakan antara lintasan yang berbelok ke kiri dengan persimpangan ke kiri.
3. Robot tidak dapat membedakan antara garis lintasan dengan titik *finish*, karena titik *finish* dan garis pada lintasan memiliki warna yang sama yaitu hitam.

Berdasarkan beberapa kendala diatas akhirnya digunakan 10 sensor untuk pengujian yang kedua. Penambahan dua sensor ini diletakkan pada posisi belakang dari kedelapan sensor yang terletak horizontal. Dengan penambahan sensor ini robot akan dapat membedakan antara tikungan dan persimpangan. Gambar 8 merupakan tampilan dari robot setelah ditambahkan 2 sensor di posisi belakang.



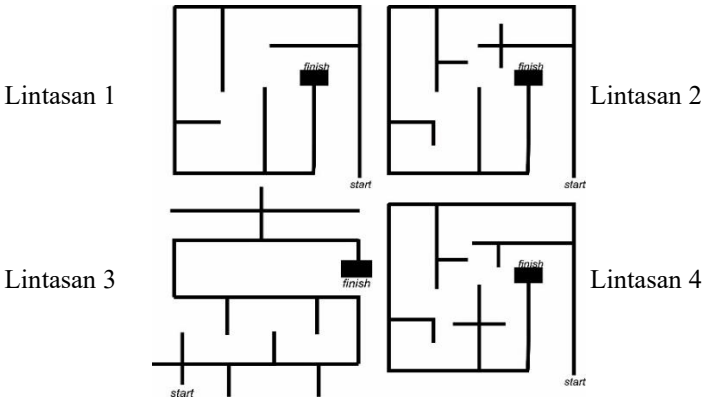
Gambar 8. Letak Sensor Photodioda pada Pengujian 2

Berikut ini merupakan beberapa kondisi yang digunakan untuk memetakan *maze* pada penelitian ini, dengan indikator sensor bernilai 1 adalah hitam dan sensor bernilai 0 adalah putih:

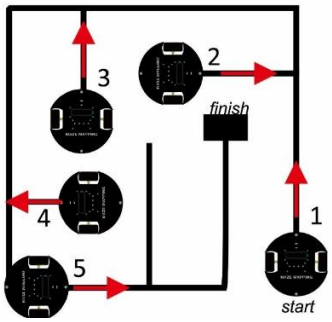
1. Jika sensor tengah = 1 dan sensor belakang kanan = 1 maka simpan S (kondisi persimpangan ke kanan).
2. Jika sensor tengah = 0 dan sensor belakang kanan = 1 maka simpan R (kondisi tikungan ke kanan).
3. Jika sensor samping kiri = 1 dan sensor samping kanan = 0 maka simpan L (simpang kiri ataupun tikungan kiri sama-sama = L)
4. Jika semua sensor = 0 maka simpan U (*dead end*)
5. Jika semua sensor = 1 maka robot berhenti (*finish*)

Pada proses penyimpanan data berupa node atau kode, peneliti menggunakan fitur *array path[];* pada program arduino. Penggunaan *array* ini bertujuan agar kode yang tersimpan memiliki letak berurutan sesuai dengan waktu pembacaan sensor. Selain itu penggunaan *array* berfungsi agar kode yang tersimpan tidak tertindih *(replace)* dari program sebelumya.

Setelah melewati proses implementasi program, selanjutnya menguji cobakan program yang telah dibuat untuk mengetahui tingkat keberhasilan suatu sistem dalam memecahkan suatu permasalahan. Terdapat beberapa macam lintasan atau *track* yang digunakan dalam penelitian ini. Tentunya dengan tingkat kerumitan dan jumlah persimpangan yang berbeda-beda. Gambar 9 merupakan variasi lintasan yang diuji cobakan pada *maze mapping mobile robot*. Gambar 10 merupakan variasi proses pengujian yang dilakukan oleh peneliti dalam pemetaan *maze* untuk pencarian jalur tercepat pada *mobile robot*.



Gambar 9. Variasi lintasan

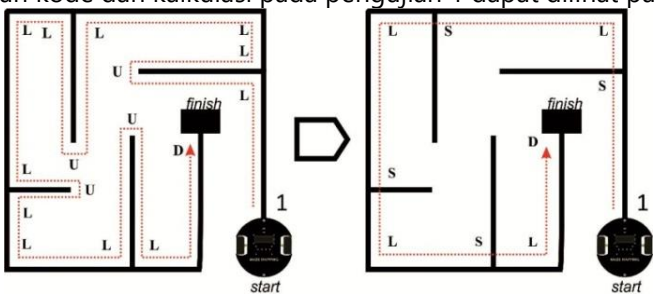


Gambar 10. Variasi Pengujian Pada Lintasan 1

Pengujian dilakukan sebanyak 10 kali pada posisi dan lintasan yang berbeda. Berikut ini merupakan proses pengujian dari implementasi algoritma *maze mapping mobile robot*:

1. Pengujian 1

Pengujian ini dilakukan pada posisi 1 seperti Gambar 11 dengan jumlah 4 simpang tiga. Hasil dari *mapping mobile robot* berdasarkan uji coba pada posisi 1 terhadap lintasan 1, waktu yang dibutuhkan oleh robot pada saat melakukan *mapping* lintasan pada pengujian 1 adalah 64 detik. Sedangkan waktu pada saat robot menelusuri lintasan untuk yang kedua kalinya pada proses *shortest path* hanya menghabiskan waktu selama 32 detik. Berdasarkan hasil tersebut menunjukkan bahwa proses penjelajahan lintasan untuk yang kedua kalinya menghabiskan waktu lebih cepat dari pada proses *mapping* dengan selisih waktu 42 detik. Hasil ini membuktikan bahwa pengujian 1 memiliki hasil yang baik untuk proses pencarian jalur tercepat. Proses pemberian kode dan kalkulasi pada pengujian 1 dapat dilihat pada Gambar 11.



Gambar 11. Mapping Maze dan Simple Maze pada Pengujian 1

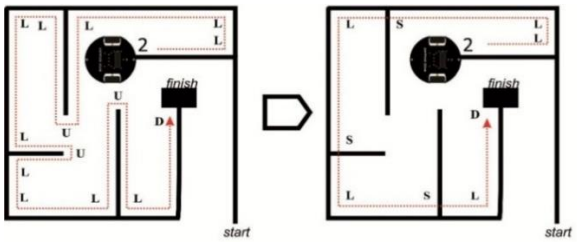
Kode hasil *mapping* pada pengujian 1:

L U L L L U L L L U L L L U L L D

S L S L S L S L D

2. Pengujian 2

Pengujian 2 titik *start* robot berada di posisi 2 seperti Gambar 12. Hasil yang diperoleh pada pengujian 2 robot dapat memetakan lintasan dengan baik hingga mencapai *finish*. Terdapat 15 kode hasil pemetaan saat *mapping* yang berhasil disimpan oleh robot. Pada pengujian 2 ini robot dapat mengkalkulasi dengan jumlah kode awal 15 kemudian dipersingkat dengan hasil kode *shortest path* sebanyak 9 kode. Waktu tempuh yang dibutuhkan oleh robot pada saat *mapping* adalah 65 detik. Sedangkan waktu yang dihabiskan robot pada saat *shortest path* adalah 33 detik. Hasil tersebut menunjukkan bahwa proses robot dalam menyelesaikan lintasan pada saat *shortest path* lebih cepat dibandingkan pada saat proses *mapping* berlangsung. Gambar 12 merupakan hasil dari *mapping mobile robot* berdasarkan uji coba pada posisi 2 terhadap lintasan 1.



Gambar 12. Mapping Maze dan Simple Maze pada Pengujian 2

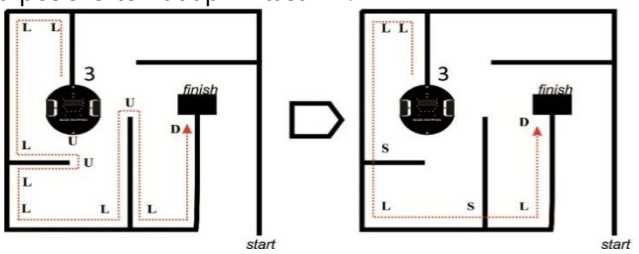
Kode hasil *mapping* pada pengujian 2:

L L L U L L L U L L L U L L D

L L S L S L S L D

3. Pengujian 3

Pengujian ini dilakukan di posisi 3 seperti Gambar 13. Pengujian ini memiliki jumlah persimpangan yang lebih sedikit dibandingkan dengan pengujian 1 dan 2. Pada pengujian 3 robot dapat memetakan lintasan dengan baik dengan jumlah kode awal yang berhasil disimpan sebanyak 11 kode dan dikalkukalisi menjadi 7 kode *shortest path*. Waktu yang dibutuhkan robot saat *mapping* pada pengujian 3 adalah 43 detik. Sedangkan waktu yang dibutuhkan kedua kalinya pada saat robot melakukan *shortest path* yaitu 25 detik. Hal ini menunjukkan bahwa waktu yang diperlukan robot dalam menyelesaikan lintasan untuk yang kedua kalinya memiliki waktu lebih cepat 18 detik, sehingga pada pengujian 3 ini robot tergolong berhasil dalam menyelesaikan misi jalur tercepat. Gambar 13 merupakan hasil dari *mapping mobile robot* berdasarkan uji coba pada posisi 3 terhadap lintasan 1.



Gambar 13. Mapping Maze dan Simple Maze pada Pengujian 3

Kode hasil *mapping* pada pengujian 3:

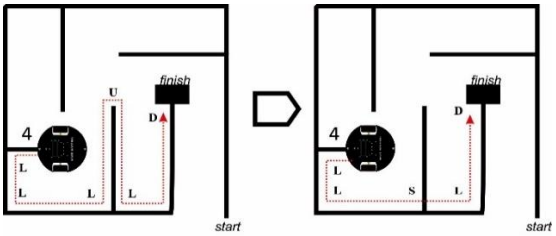
L L L U L L L U L L D

L L S L S L D

4. Pengujian 4

Pada pengujian ini titik *start* robot berada di posisi 4 seperti Gambar 14. Lintasan yang harus dipetakan oleh robot hanya memiliki dua persimpangan dan satu belokan. Pengujian 4 memiliki hasil yang baik karena robot dapat memetakan seluruh persimpangan yang ada. Kode awal yang tersimpan pada saat *mapping* sebanyak 7 kode dan berhasil dikalkulasi menjadi jalur tercepat sebanyak 5 kode. Waktu tempuh robot

pada saat melakukan *mapping* adalah 30 detik. Sedangkan pada saat robot menelusuri lintasan untuk kedua kalinya membutuhkan waktu 16 detik hingga mencapai *finish*. Gambar 14 merupakan hasil dari *mapping mobile robot* berdasarkan uji coba pada posisi 4 terhadap lintasan 1.



Gambar 14. Mapping Maze dan Simple Maze pada Pengujian 4

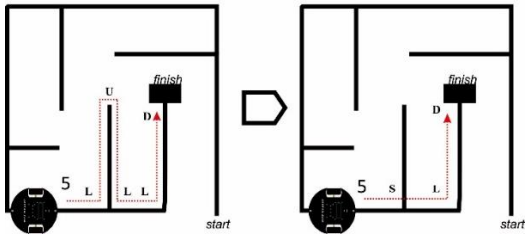
Kode hasil *mapping* pada pengujian 4:

L L L U L L D

L L S L D

5. Pengujian 5

Pada pengujian ini titik *start* robot berada di posisi 5 seperti Gambar 15. Pengujian 5 memiliki kategori persimpangan paling sedikit pada lintasan 1 dibandingkan pengujian 1 hingga 4. Langkah pada pengujian 5 tetap sama dengan pengujian sebelumnya yaitu robot melakukan *mapping* terlebih dahulu kemudian untuk penelusuran kedua kalinya robot diharapkan menghindari jalan buntu melewati jalur tercepat. Terdapat 5 kode yang berhasil dipetakan saat *mapping* berlangsung. Dari kelima kode tersebut robot dapat mengkalkulasi data secara otomatis menjadi kode lebih singkat yang berjumlah 3 kode. Waktu yang diperlukan robot untuk melakukan *mapping* pada pengujian 5 adalah 23 detik. Waktu tempuh robot untuk yang kedua kalinya lebih cepat dibandingkan dengan waktu tempuh pada saat proses *mapping* yaitu 8 detik. Selisih waktu antara proses saat *mapping* dan *shortest path* adalah 15 detik. Hasil ini menunjukkan bahwa pengujian 5 robot berhasil melewati jalur tercepat menuju *finish*. Gambar 15 merupakan hasil dari *mapping mobile robot* berdasarkan uji coba pada posisi 5 terhadap lintasan 1.



Gambar 15. Mapping Maze dan Simple Maze pada Pengujian 5

Kode hasil *mapping* pada pengujian 5:

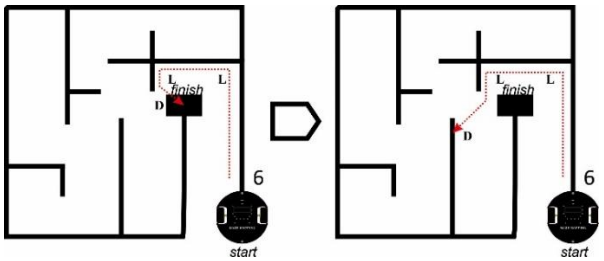
L U L L D

S L D

6. Pengujian 6

Pengujian ini dilakukan pada posisi 6 seperti Gambar 16. Lintasan 2 memiliki tingkat kerumitan yang lebih tinggi dibandingkan dengan lintasan 1. Peneliti menambahkan fitur simpang empat untuk menguji tingkat keberhasilan robot pada saat melakukan pemetaan. Hasil yang didapat pada pengujian 6 robot tidak dapat sampai hingga titik *finish* sesuai jalur lintasan. Terjadi kesalahan jalur pada saat robot berada di posisi garis yang memiliki jarak berdekatan dengan titik *finish* seperti pada Gambar 16. Sehingga pada saat robot bermanuver untuk berbalik secara tidak sengaja sensor photodioda mendeteksi adanya titik *finish*. Secara otomatis robot berhenti dan melakukan kalkulasi data. Faktor kerapatan jalur lintasan tersebut

menghasilkan kesalahan *mapping* dan robot terjadi *error* pada saat melakukan *shortest path*. Proses pemberian kode dan kalkulasi pada pengujian 6 dapat dilihat pada Gambar 16.

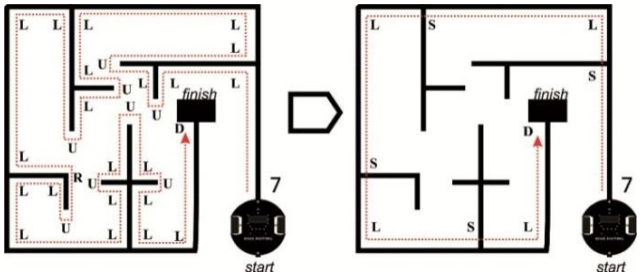


Gambar 16. Mapping Maze dan Simple Maze pada Pengujian 6

Kode hasil *mapping* pada pengujian 6:
L L D
L L D

7. Pengujian 7

Pada pengujian 7 robot diujicobakan pada lintasan 3. Lintasan ini memiliki jarak yang berjauhan antar masing-masing garis. Karena pada pengujian 6, jarak antar garis yang saling berdekatan memiliki dampak yang negatif terhadap proses *tracking* dan pengkalkulasian data. Lintasan 3 pada pengujian 7 memiliki lorong yang tergolong rumit karena peneliti menambahkan fitur persimpangan empat. Terdapat total 32 kode yang berhasil disimpan ke dalam memori robot dalam bentuk *array* dan berhasil dikalkulasi menjadi kode yang lebih singkat berjumlah 9 kode dengan lima kali konversi. Waktu yang dibutuhkan oleh robot dalam melakukan pemetaan pada pengujian 7 adalah 103 detik. Pada saat pengujian yang kedua kalinya robot hanya membutuhkan waktu 32 detik untuk mencapai *finish* dengan selisih waktu 72 detik lebih cepat. Proses penyimpanan kode dan proses kalkulasi data pada pengujian 7 dilihat pada Gambar 17.



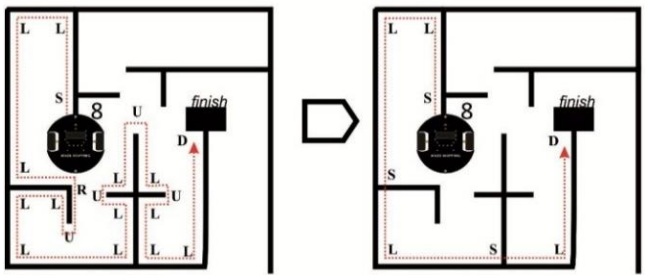
Gambar 17. Mapping Maze dan Simple Maze pada Pengujian 7

Kode hasil *mapping* pada pengujian 7:

LLULUSL	L	LLULUSL	L	LRULL	L	LLULULULL	L D
LSUSL	L	LSUSL	L	LBL	L	LSULULL	L D
LBL	L	LBL	L	S	L	LRULL	L D
S	L	S	L	S	L	LBL	L D
S	L	S	L	S	L	S	L D

8. Pengujian 8

Posisi *start* robot pada pengujian ini terletak di posisi 8 seperti pada Gambar 18. Lintasan yang digunakan tetap sama dengan pengujian 7 akan tetapi titik awal yang membedakan antar keduanya. Hasil yang didapat robot dapat memetakan 20 kode pada saat *mapping* berlangsung. Total kode tersebut robot dapat mengkalkulasi data secara otomatis menjadi kode lebih singkat yang berjumlah 8 kode. Waktu yang diperlukan robot untuk melakukan *mapping* pada pengujian 8 adalah 68 detik. Waktu tempuh robot untuk yang kedua kalinya lebih cepat dibandingkan dengan waktu tempuh pada saat proses *mapping* yaitu 35 detik. Selisih waktu antara proses saat *mapping* dan *shortest path* adalah 34 detik. Hasil ini menunjukan bahwa pada pengujian 8 robot berhasil melewati jalur tercepat menuju *finish*. Gambar 18 merupakan hasil dari *mapping mobile robot* berdasarkan uji coba posisi 8 pada lintasan 3.



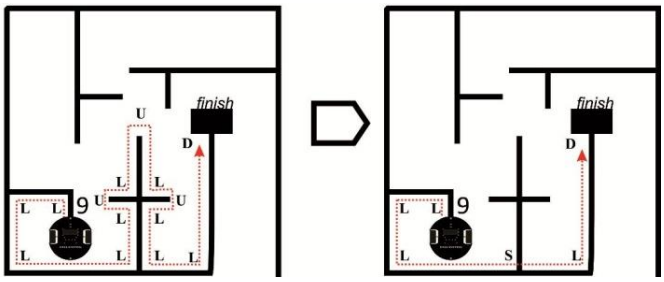
Gambar 18. Mapping Maze dan Simple Maze pada Pengujian 8

Kode hasil *mapping* pada pengujian 8:

S L L LRULL L LLULULULL L D
S L L LBL L LSULULL L D
S L L S L LRULL L D
S L L S L LBL L D
S L L S L S L D

9. Pengujian 9

Pada pengujian 19 ini peneliti tetap menggunakan lintasan 3 untuk melakukan pengujian *mobile robot maze mapping*. Namun posisi *start* robot diubah sesuai dengan Gambar 19. Pada pengujian 9 robot berhasil memetakan semua persimpangan yang dilewati dan mengkalkulasi menjadi kode tercepat. Terdapat 14 kode yang berhasil disimpan dalam bentuk *array*, kemudian robot mengkalkulasi menjadi kode yang lebih singkat menjadi 6 kode untuk jalan tercepat. Waktu yang dibutuhkan untuk melakukan pemetaan pada pengujian 9 adalah 48 detik dan waktu pada saat robot melakukan *shortest path* yaitu 29 detik dengan selisih waktu 25 detik lebih cepat dibandingkan pada proses *mapping*.



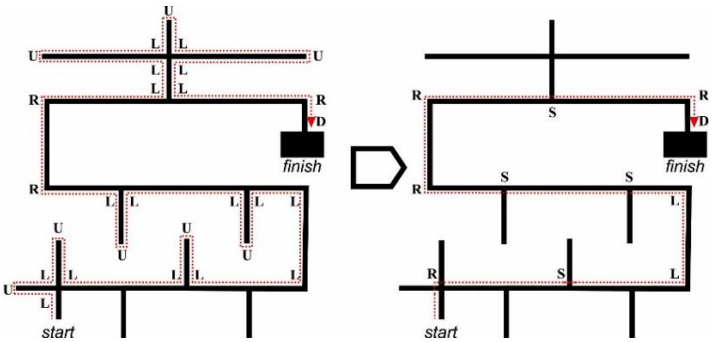
Gambar 19. Mapping Maze dan Simple Maze pada Pengujian 9

Kode hasil *mapping* pada pengujian 9:

L L L LLULULULL L D
L L L LSULULL L D
L L L LRULL L D
L L L LBL L D
L L L S L D

10. Pengujian 10

Lintasan 4 digunakan pada pengujian ini. Posisi awal robot diletakan di titik *start*. Selanjutnya robot melakukan *mapping* dengan kecepatan motor 220 dan menyimpan kode yang terbaca berdasarkan persimpangan dari lintasan. Pada pengujian 10 ini robot berhasil memetakan seluruh *maze* dari titik *start* hingga *finish*. Tahap kedua robot berhasil menelusuri lintasan 4 dengan melewati jalur tercepat dan menghindari jalan buntu. Hasil kode yang tersimpan dan proses kalkulasi data pada pengujian ini dapat dilihat pada Gambar 20.



Gambar 20. Mapping Maze dan Simple Maze pada Pengujian 10

Kode hasil *mapping* pada pengujian 10:

LULUL	LUL	L L	LUL	LUL	R	R	LLULULULL	R	D
SUL	S	L L	S	S	R	R	LSULULL	R	D
R	S	L L	S	S	R	R	LRULL	R	D
R	S	L L	S	S	R	R	LBL	R	D
R	S	L L	S	S	R	R	S	R	D

Tabel 2. Hasil Pengujian Maze Mapping dan Shortest Path

Pengujian	*Mapping* Lintasan	*Shortest Path*	*Mapping Time*	*Shortest Time*	Kendala
1	✓	✓	74 detik	32 detik	Cahaya matahari
2	✓	✓	65 detik	33 detik	Cahaya matahari
3	✓	✓	43 detik	25 detik	Cahaya matahari
4	✓	✓	30 detik	16 detik	Cahaya matahari
5	✓	✓	23 detik	8 detik	Cahaya matahari
6	X	X	X	X	Garis lintasan terlalu rapat
7	✓	✓	103 detik	32 detik	Struktur lintasan tidak rata
8	✓	✓	68 detik	35 detik	Struktur lintasan tidak rata
9	✓	✓	48 detik	29 detik	Struktur lintasan tidak rata
10	✓	✓	104 detik	33 detik	-
Rata-rata	✓	✓	62 detik	27 detik	-

Berdasarkan Tabel 2 peneliti melakukan pengujian sebanyak sepuluh kali pada lintasan yang berbeda-beda. Hasil yang didapat sembilan kali sistem dapat melakukan pemetaan lintasan dan berhasil melakukan *shortest path*. Sedangkan satu kali robot gagal melakukan *mapping* pada pengujian 6. Faktor ini dipengaruhi oleh kegagalan pada saat proses *mapping*, sehingga kalkulasi dan saat proses *shortest path* pasti mengalami kesalahan. Hasil yang didapat pada penelitian ini berdasarkan pengujian memiliki tingkat keberhasilan sebesar 90%. Hasil tersebut diperoleh dari rumus persentase keberasilan, yaitu jumlah keberhasilan dibagi jumlah maksimal pengujian dikali seratus persen.

KESIMPULAN

Setelah dilakukan pengujian secara bertahap pada *line follower mobile robot* dengan metode *maze mapping algorithm left hand rule*, dapat disimpulkan bahwa:

1. Pembacaan sensor photodioda yang digunakan pada penelitian ini memiliki tingkat sensitifitas yang tinggi dengan *range* antara 0 – 200 ketika membaca putih, dan 500 – 980 ketika membaca hitam. Akan tetapi jarak antara sensor dengan lantai sangat mempengaruhi terhadap tingkat kepekaan dan sensitivitas sensor ketika membaca warna. Hasil dari penelitian ini, jarak yang efektif antara sensor dengan lantai yaitu 5 mm.
2. Motor DC SAYAMA Tipe 127K94540K memiliki torsi yang cukup kuat, karena dapat menggerakan robot dengan berat 1 kg untuk bergerak maju dan berbelok. Akan tetapi apabila kecepatan PWM yang sama diinputkan pada motor, menghasilkan kecepatan yang tidak sama pada saat pengujian sesungguhnya. Hasil PWM kecepatan motor yang cocok berdasarkan pengujian yaitu motor kanan 120 dan motor kiri 123 untuk aplikasi PD dan 220 motor kanan dan 223 motor kiri untuk metode analog.
3. Penerapan metode *maze mapping algorithm left hand rule* pada penelitian ini dapat memetakan dan menyimpan data atau kode dalam bentuk *char* dan tersimpan dalam bentuk *array*. Kemudian robot berhasil mengkalkulasi data menjadi data jalur tercepat sebagaimana hasil dari pengujian 1-9 dimana pengujian ke 6 dihitung sebagai kegagalan. Waktu *mapping* tercepat ditempuh dalam 23 detik dan rata-ratanya 62 detik, sedangkan *shortest time* 8 detik dan rata-ratanya 27 detik yang masing-masing merupakan hasil pada pengujian 4.
4. Tingkat keberhasilan penelitian ini sebesar 90%, dari sepuluh kali pengujian menggunakan metode *maze mapping left hand rule* menghasilkan sembilan kali pengujian robot berhasil memetakan dan melewati jalan tercepat tanpa adanya jalan buntu. Sedangkan satu kali robot mengalami kegagalan dalam memetakan lintasan.

DAFTAR PUSTAKA

Alamri, S., Alamri, H., Alshehri, W., Alshehri, S., Alaklabi, A., & Alhmiedat, T. (2023). An Autonomous Maze-Solving Robotic System Based on an Enhanced Wall-Follower Approach. *Machines*, *11*(2). https://doi.org/10.3390/machines11020249

Alamri, S., Alshehri, S., Alshehri, W., Alamri, H., Alaklabi, A., & Alhmiedat, T. (2021). Autonomous Maze Solving Robotics: Algorithms and Systems. *International Journal of Mechanical Engineering and Robotics Research*, *10*(12), 668–675. https://doi.org/10.18178/ijmerr.10.12.668-675

Amalia, D., Nugraha, W., Sunardi, S., Rizko, R., Putri, J., & Pratama, R. A. (2023). Multisensor Fire Detection System Mobile Robot as a Learning Media in Firefighting Robotic Course. *Journal of Innovation in Educational and Cultural Research*, *4*(4), 682–690. https://doi.org/10.46843/jiecr.v4i4.1016

Arthalia Wulandari, I., Sukmasetyan, P., Muhammadiyah Magelang, U., Mayjen Bambang Soegeng, J., Mertoyudan, K., Magelang, K., & Tengah, J. (2022). Implementasi Algoritma Dijkstra untuk Menentukan Rute Terpendek Menuju Pelayanan Kesehatan. *Jurnal Ilmiah Sistem Informasi*, *1*(1), 30–37.

Christian Rufus, E., Rizkyaka Riyadi, R., Nugraha Hasibuan, D., Christian, E., Handrianus Pranatawijaya, V., & Nyaho Jl Yos Sudarso Palangka Raya, T. (2024). Penerapan Algoritma Dijkstra Dalam Menentukan Rute Terpendek Untuk Jasa Pengiriman Barang Di Palangka Raya. In *Jurnal Mahasiswa Teknik Informatika* (Vol. 8, Issue 3).

Fahmizal, F., Pratikno, M. S., Isnianto, H. N., Mayub, A., Maghfiroh, H., & Anugrah, P. (2024). Control and Navigation of Differential Drive Mobile Robot with PID and Hector SLAM: Simulation and Implementation. *Jurnal Ilmiah Teknik Elektro Komputer Dan Informatika*, *10*(3), 594–607. https://doi.org/10.26555/jiteki.v10i3.29428

Farouq, A. Al, Riansyah, Moch. I., & Habibi, A. (2021). View of Unmanned Aerial Vehicle Robot untuk Pemadam Api dengan Rancangan Alat Pemadam yang Efisien. *Jurnal Nasional Teknik Elektro*, *10*(1), 11–18. https://doi.org/https://doi.org/10.25077/jnte.v10n1.839.2021

Fuad, M., & Wahyuni, S. (2022). Path Planning and Smoothing in Maze Exploration Using Virtual Mobile Robot-Based Modified Probabilistic Road Map. *Proceeding - IEEE 8th Information Technology International Seminar, ITIS 2022*, 15–20. https://doi.org/10.1109/ITIS57155.2022.10010054

Hendriawan, A., & Akbar, R. (2010). AkhmadHendriawan_[C-D206-06] Penyelesaian Jalur Terpendek dengan menggunakan Algoritma Maze Mapping-IES2010. *The 12th Industrial Electronics Seminar 2010 (IES 2010)*, 91–94.

Kaharsyah, F. (2023). 2 nd Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI) 21 Maret 2023-Jakarta. *2nd Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI)*, *2*(1), 416–424. https://doi.org/10.12928/TELKOMNIKA.v20i2.19408

Kitsak, M., Ganin, A., Elmokashfi, A., Cui, H., Eisenberg, D. A., Alderson, D. L., Korkin, D., & Linkov, I. (2023). Finding shortest and nearly shortest path nodes in large substantially incomplete networks by hyperbolic mapping. *Nature Communications*, *14*(1). https://doi.org/10.1038/s41467-022-35181-w

Latif, E., Song, W., & Parasuraman, R. (2023). Communication-Efficient Reinforcement Learning in Swarm Robotic Networks for Maze Exploration. *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 1–6. https://doi.org/10.1109/INFOCOMWKSHPS57453.2023.10226167

M, S., Kamal, K., Rizal, M., & Syani, A. (2022). Rancang Bangun Robot Cerdas Pemadam Api Menggunakan Teknik Navigasi Wall Follower Berbasis Arduino. *ILTEK : Jurnal Teknologi*, *17*(02), 88–93. https://doi.org/10.47398/iltek.v17i02.16

Maulana, M. A., Novian, R., Nugraha, M. I., Made, I., Setiawan, A., Negeri, P. M., & Belitung, B. (2022). Autonomous Mobile Robot Dengan Menggunakan Metode Simultaneous Localization And Mapping Berbasis Lidar. *Prosiding Seminar Nasional Inovasi Teknologi Terapan (SNITT)*, 25–31.

Mohammed, I. M., Al-Dabagh, M. Z. N., Rashid, S. J., & Isa, N. A. M. (2022). Path discovering in maze area using mobile robot. *Telkomnika (Telecommunication Computing Electronics and Control)*, *20*(2), 416–425. https://doi.org/10.12928/TELKOMNIKA.v20i2.19408

Pamungkas, R. S., Schmidt, D., & Chen, C.-S. (2024). View of The Role of Machine Learning in Improving Robotic Perception and Decision Making. *International Transactions on Artificial Intelligence (ITALIC)*, *3*(1), 32–43. https://doi.org/https://doi.org/10.33050/italic.v3i1.661

Pasaribu, R. I. V., & Yurinanda, S. (2024). Optimasi Rute Terpendek Pemeliharaan Lisrik Dengan Algoritma Dijkstra Di PLN UIP Sumbagsel. *JISTech (Journal of Islamic Science and Technology)*, *9*(2), 240. https://doi.org/10.30829/jistech.v9i2.22446

Prakash, P., & Naganjaneyulu, V. (2021). Autonomous Line Maze Solver Using Artificial Intelligence. *ZKG International*, *6*(1), 103–108. www.zkginternational.com

Prayoga, S., & Sunggu, E. O. (2024). budiana,+(88-95)-8470-Manuscript-27609-1-15-20241209.v1+(10%). *Journal Of Applied Electrical Engineering*, *8*(2), 88–95.

Priambodo, A. S., Nasuha, A., & Dhewa, O. A. (2024). Integrated Implementation of Computer vision and PID Control for Dynamic Speed Control of Line follower Robot. *Telekontran : Jurnal Ilmiah Telekomunikasi, Kendali Dan Elektronika Terapan*, *12*(1), 81–93. https://doi.org/10.34010/telekontran.v12i1.13323

Ridwan Mukti, M., & Matematika dan Ilmu Pengetahuan Alam, F. (2018). Menentukan Rute Terpendek Dengan Menggunakan Algoritma Floyd-Warshall Dalam Pendistribusian Barang Pada PT. Rapy Ray Putratama. *KARISMATIKA*, *4*(1), 39–53.

Safittri, N. (2020). Rancang Bangun Robot Pemadam Api Secara Otomatis Menggunakan Sistem Pneumatik Berbasis Mikrokontroller ATMEGA32. *JURNAL TEKTRO*, *4*(1). https://doi.org/http://dx.doi.org/10.30811/tektro.v4i1.2746

Sitepu, Ra. N. B., & Putra, I. G. N. A. C. (2022). Penentuan Rute Terpendek Menggunakan Algoritma A Star (Studi Kasus: Distributor Barang). *Jurnal Nasional Teknologi Informasi dan Aplikasinya (JNATIA)*, *1*(1), 431–440.

Sriyana. (2022). *Rencana Kerja Tahun 2023*.

Suhendar -, B., Surahmat -, A., Fuady -, T., Jl Syeh Nawawi Albantani, J., & -Banten, S. (2020). Sensor Robot Pemadam Api Dengan Beberapa Mode Operasi Berbasis Teknologi Dtmf, Bluetooth Dan GSM. *Jurnal Sains & Teknologi*, *135*(2).

Tanujaya, M., Patrick, R., Sebastian, H., & Wahab, F. (2024). Penerapan Contour Detection dengan Pengontrol PID pada Robot Line Follower. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, *12*(4), 923. https://doi.org/10.26760/elkomika.v12i4.923

Vartak, V. U. (2023). *Maze Solving Robot Using A* Algorithm*.

Wahyuni, S. (2013). Algoritma Rekursif Untuk Pemetaan Jalur Autonomous Line Follower. *Prosiding Conference on Smart-Green Technology in Electrical and Information Systems (CSGTEIS)*, 287–294.

Wahyuni, S., Budiarto, H., & Jamilah, N. (2021). Mobile robot painted dashed lines as road markings. *IOP Conference Series: Materials Science and Engineering*, *1125*(1), 012080. https://doi.org/10.1088/1757-899x/1125/1/012080

Wahyuni, S., Fuad, M., Hilda, A. D. D., Umam, F., Latif, M., & Junaidi. (2023). Enhancing Mobile Robot Stability and Roaming Capability Using Fuzzy-PID Control Method. *2023 IEEE 9th Information Technology International Seminar (ITIS)*, *23*, 58–63.

Yew-Hao Hii, J., Lee, J.-V., Chuah, Y.-D., & Tunku Abdul Rahman, U. (2021). Optimization and Simulation of A Navigation Robot in Mazes. *International Journal of Engineering Research & Technology (IJERT)*, *10*(5), 222–227. www.ijert.org

Zarkasi, A., Ubaya, H., Deri Amanda, C., & Firsandaya, R. (2019). Implementation of ram based neural networks on maze mapping algorithms for wall follower robot. *Journal of Physics: Conference Series*, *1196*(1). https://doi.org/10.1088/1742-6596/1196/1/012043