Optimalisasi Pencarian Jalur Terpendek *Mobile* Robot dengan Menggunakan Metode *Ant Colony Optimization* (ACO)

Ahmad Ihsan[1], Taufan Arif Adlie[2], Septia Harliansyah[3]

[1,3]Program Studi Informatika,
Fakultas Teknik,
Universitas Samudra, Langsa
[1]ahmadihsan@unsam.ac.id, [3]septiaharliansyahxrpl@gmail.com

[2]Program Studi Teknik Mesin,
Fakultas Teknik,
Universitas Samudra, Langsa
[2]taufanarif@unsam.ac.id

Ringkasan

Kemajuan pesat dalam teknologi robot saat ini telah menghadirkan perangkat mekanis yang mampu melaksanakan berbagai tugas fisik, baik di bawah pengawasan manusia maupun melalui program yang tertanam dalam prosesor. *Mobile robot* menonjol sebagai sistem mekanis yang mampu bergerak mandiri di lingkungan sekitarnya. *Paper* ini berfokus pada permasalahan penting dalam menemukan jalur terpendek yang sesuai dengan kondisi lingkungan bagi *mobile robot*. Penelitian ini bertujuan untuk mengoptimalkan pergerakan *mobile robot* dalam menemukan jalur terpendek dengan menerapkan metode *Ant Colony Optimization*. Melalui serangkaian percobaan, robot telah diuji sebanyak 20 kali, dengan 10 percobaan sebelum dan setelah penerapan optimalisasi. Hasil percobaan menunjukkan bahwa setelah pengoptimalan, robot mampu menemukan jalur terpendek, dengan contoh terbaik adalah jalur 2, yang memiliki panjang total 213 cm, nilai probabilitas 3,239, dan waktu tempuh antara 14 hingga 17 detik. Penelitian ini menegaskan bahwa penggunaan metode *Ant Colony Optimization* dapat berperan sebagai faktor penting dalam pengambilan keputusan terkait penentuan jalur terpendek dalam situasi percabangan jalan. *Mobile Robot* yang digunakan dalam penelitian ini, dikenal sebagai *Hummerbot*, dilengkapi dengan tiga sensor ultrasonik yang memungkinkannya untuk mendeteksi rintangan di sekitarnya. Penelitian ini memberikan pemahaman yang mendalam tentang penggunaan metode *Ant Colony Optimization* dalam menemukan jalur terpendek bagi *mobile robot*, membantu dalam mengatasi tantangan navigasi dalam kondisi lingkungan yang kompleks. Penerapan teknik-teknik ini membawa kontribusi signifikan dalam pengembangan robotika masa depan yang semakin canggih.

Kata Kunci: Mobile Robot, Jalur terpendek, Ant Colony Optimalization.

Abstract

Rapid advances in robotic technology have now provided mechanical devices capable of performing a variety of physical tasks, either under human supervision or through programs embedded in processors. Mobile robots stand out as mechanical systems capable of moving autonomously in their surrounding environment. This paper focuses on the

Received: 7 Juli 2023 Revised: 3 November 2023 Accepted: 5 November 2023

important problem of finding the shortest path that suits the environmental conditions for a mobile robot. This research aims to optimize the movement of mobile robots in finding the shortest path by applying the Ant Colony Optimization method. Through a series of experiments, the robot was tested 20 times, with 10 trials before and after implementing optimization. The experimental results show that after optimization, the robot is able to find the shortest path, with the best example being path 2 which has a total length of 213 cm, a probability value of 3.239, and a travel time of between 14 and 17 seconds. This research confirms that the use of the Ant Colony Optimization method can act as an important factor in decision making regarding determining the shortest path in road branch situations. The mobile robot used in this research is called Hummerbot, equipped with three ultrasonic sensors that allow it to detect obstacles in its surroundings. This research provides an in-depth understanding of the use of the Ant Colony Optimization method in finding the shortest path for mobile robots, helping to overcome navigation challenges in complex environmental conditions. The application of these techniques makes a significant contribution to the development of increasingly sophisticated future robotics.

Keywords: Mobile Robot, shortest path, Ant Colony Optimization.

1. Pendahuluan

Teknologi robot saat ini telah menunjukkan kemajuan yang luar biasa, mendorong pergeseran paradigma dalam berbagai industri dan lingkungan kerja. Robot, yang terdiri dari perangkat mekanis yang canggih, memiliki kemampuan untuk melaksanakan berbagai tugas fisik dengan presisi yang luar biasa, baik di bawah pengawasan manusia maupun melalui program yang telah disematkan di dalam prosesor. Keberadaan robot telah mengubah cara kita melihat proses otomatisasi dan interaksi manusia dengan teknologi di era modern [1].

Pentingnya peran robot dalam kehidupan sehari-hari semakin meningkat seiring dengan berkembangnya kebutuhan industri, medis, dan penelitian. Salah satu bentuk robot yang sangat penting dalam konteks ini adalah *mobile robot*. *Mobile robot* merupakan sistem mekanis yang memiliki kemampuan untuk bergerak secara mandiri di lingkungannya, dengan memanfaatkan sensor dan algoritma yang kompleks untuk mengatasi tantangan navigasi yang rumit [2].

Dalam mengemban tugasnya, *mobile robot* sering kali dihadapkan pada masalah pencarian jalur terpendek, yakni mencari rute tercepat dan teraman dari titik awal menuju titik tujuan, sambil menghindari berbagai rintangan atau hambatan yang mungkin muncul di sepanjang jalur tersebut. Pencarian jalur terpendek menjadi salah satu aspek kunci yang menentukan keberhasilan robot dalam menyelesaikan tugasnya dengan efisiensi yang maksimal. Namun, kompleksitas lingkungan dan berbagai parameter yang harus dipertimbangkan seringkali membuat pencarian jalur terpendek menjadi tantangan yang kompleks dan sulit dipecahkan [3].

Penelitian yang dilakukan dalam lingkup optimalisasi pencarian jalur terpendek *mobile robot* menggunakan metode *Ant Colony Optimization* (ACO) merupakan langkah penting dalam memahami dan meningkatkan kinerja robot dalam menghadapi tantangan navigasi tersebut. Penelitian ini bertujuan untuk mengoptimalkan kinerja *mobile robot* dengan cara menemukan jalur terpendek yang paling sesuai dengan kondisi lingkungan sekitar. Dengan menggunakan pendekatan ACO, diharapkan bahwa robot dapat menemukan jalur optimal secara efisien, menghemat waktu dan sumber daya dalam menjalankan tugas-tugasnya [4].

Dalam konteks ini, penggunaan metode *Ant Colony Optimization* menarik perhatian sebagai solusi yang potensial untuk mengatasi permasalahan pencarian jalur terpendek. Metode ini terinspirasi dari perilaku koloni semut dalam mencari jalur terpendek menuju sumber makanan. Konsep ini telah berhasil diadaptasi ke dalam konteks robotika untuk mengop-timalkan navigasi *mobile robot*, yang menandai langkah penting dalam pemahaman dan pengembangan teknologi robot yang lebih canggih dan efisien. Dengan menerapkan prinsip-prinsip yang terbukti berhasil dari algoritma ACO, diharapkan bahwa *mobile robot* dapat beroperasi dengan lebih efektif dan efisien di berbagai lingkungan kerja yang kompleks [5].

Dalam rangka mencapai tujuan optimalisasi pencarian jalur terpendek, penelitian ini akan fokus pada pengujian dan evaluasi kinerja *mobile robot* sebelum dan setelah penerapan metode ACO. Melalui serangkaian percobaan yang cermat, diharapkan bahwa penelitian ini akan memberikan wawasan yang mendalam tentang kemampuan dan potensi penerapan metode ACO dalam meningkatkan kemampuan navigasi *mobile robot*. Dengan demikian, penelitian ini diharapkan akan memberikan kontribusi yang signifikan dalam memperluas batas kemampuan teknologi robotika masa kini dan membuka peluang untuk pengembangan robotika masa depan yang semakin canggih dan adaptif [6].

2. Tinjauan Pustaka

2.1. Penelitian Sebelumnya

Penjelasan dari apa yang telah diteliti oleh Via Risqianti, Hasbi Yasin, dan Rukun Santoso menjelaskan bahwa penelitian mengenai algoritma ACO, terutama dalam aplikasinya untuk permasalahan TSP, telah dilakukan oleh Marco Dorigo (Belgia) dan Luca Maria. Berdasarkan *paper* yang ditulis mereka, *Ant Colony Optimization* (ACO) memiliki performa yang jauh lebih baik dibanding algoritma lain. Salah satu data di *paper* tersebut menunjukkan pada kasus TSP dengan 75 kota, ACO hanya membutuhkan simulasi tur sebanyak 3.480 kali untuk menemukan jalur tur terbaik, sedangkan *genetic algorithm* membutuhkan 80.000 kali simulasi tur untuk menemukan jalur tur terbaik, dan algoritma lain, seperti *evolutionary programming* (EP) dan *simulated annealing* (SA) bahkan membutuhkan jumlah simulasi tur yang lebih banyak lagi [7].

2.2. *Arduino Uno*

Papan *Arduino* adalah jenis papan elektronik yang populer saat ini, yang mengeksplorasi atau mengimplementasikan berbagai proyek elektronik dan termasuk pemrograman. Dari sekian banyak jenis *board Arduino* yang ada, *Arduino Uno* merupakan yang paling populer. Papan *Arduino Uno* dapat dilihat pada Gambar 1. Biaya yang murah dan mudah dipelajari menjadi salah satu kunci pengembangan papan tulis elektronik seukuran kartu kredit ini. *Arduino* sangat berguna dalam mempelajari aplikasi mikrokontroler untuk berbagai proyek terkait perangkat lunak.

Pin pada papan *Arduino* dapat dibagi menjadi dua kategori utama, yaitu *pin* digital dan *pin* analog. *Arduino Uno* berisi 14 *pin* digital dan enam *pin* analog. *Pin* digital adalah *pin* yang memiliki nilai digital 1 atau 0. *Pin* digital dapat bertindak sebagai *input* atau *output*. *Mode input* berarti *pin* harus dibaca, misalnya untuk membaca tombol saat ditekan *high* atau tidak ditekan *low*. Status keluar menunjukkan bahwa tumpukan hanya dapat

ditulis. Dengan kata lain, dalam *mode* yang dikeluarkan dapat mengatur nilai *pin* ke *HIGH* atau *LOW* [8].

Gambar 1. *Arduino Uno*

2.3. Sensor Ultrasonik HC-SR04

Sensor ultrasonik adalah sensor yang mengubah variabel fisik suara menjadi variabel listrik dan sebaliknya. Pengoperasian sensor ini berdasarkan prinsip pemantulan gelombang bunyi, sehingga dapat digunakan untuk menginterpretasikan keberadaan jarak suatu objek pada frekuensi tertentu [9]. Adapun bentuk dari sensor ultrasonik dapat dilihat pada Gambar 2.

Sensor ultrasonik tipe HCSR04 merupakan alat yang digunakan untuk mengukur jarak suatu objek. Area jarak terukur sekitar 2-450cm. Alat ini menggunakan dua *pin* digital untuk membaca komunikasi jarak jauh. Prinsip pengoperasian sensor ultrasonik ini adalah mengirimkan pulsa ultrasonik sekitar 40kHz kemudian dapat dipantulkan dan dihitung waktu responnya [10].

Gambar 2. Sensor Ultrasonik

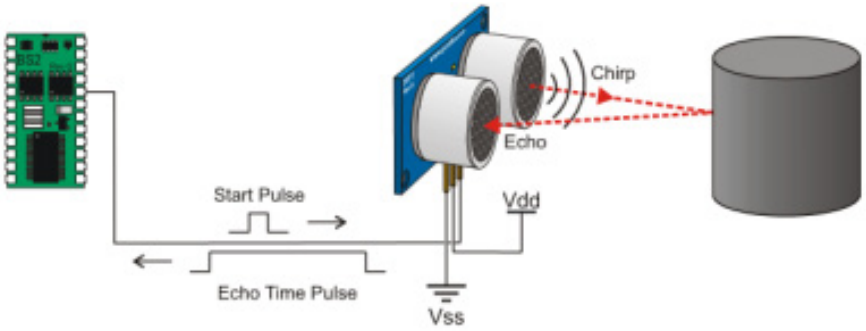
Jarak dihitung dengan menggunakan rumus:

$$Jarak = kecepatan\ suara * T/2 \quad (1)$$

Pembagi 2 diperlukan karena T adalah waktu yang diperlukan untuk menempuh dari sensor ke objek dan dari objek ke sensor. Dengan nilai kecepatan suara sebesar 343 m/s atau 343000 cm/s sehingga jarak dapat diperoleh dengan persamaan :

$$\text{Jarak} = 343000 * (T/10\text{-}6)/2 = 0{,}0343 * T/2 \text{ cm} \quad (2)$$

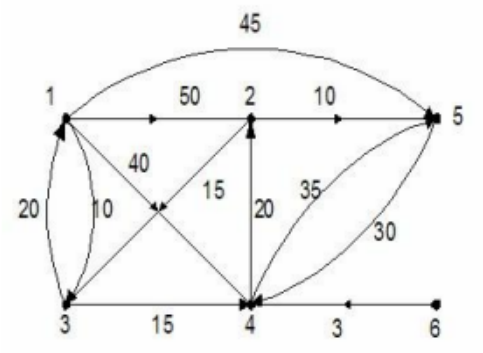
Ilustrasi sensor ultrasonik bekerja, mulai sinyal dikirimkan oleh *pin Trig* sampai diterima kembali oleh *pin Echo* dapat dilihat pada Gambar 3 [11].



Gambar 3. Ilustrasi Sensor Ultrasonik bekerja

2.4. Definisi Graf

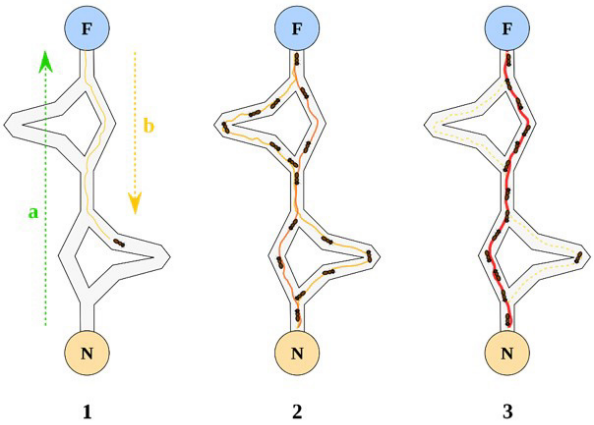
Graf adalah struktur diskrit yang terdiri dari adanya simpul (*vertex*) dan adanya sebuah sisi (*edge*), *graph* adalah pasangan himpunan (*V,E*) dimana *V* merupakan sebuah himpunan yang tidak kosong dari sebuah *vertex* dan *E* adalah himpunan sisi yang menghubungkan sepasang simpul dalam *graph* tersebut [12].



Gambar 4. Contoh Graf

2.5. *Ant Colony Optimization*

ACO adalah salah satu metode optimisasi heuristik yang mengadopsi sifat-sifat koloni semut dalam menemukan sumber makanan. ACO merupakan salah satu metode pencarian jalur terpendek yang tergolong dalam algoritma metaheuristik, yang artinya ACO menggunakan mekanisme pencarian yang terdiri dari beberapa tahapan dan bukan menggunakan algoritma pencarian yang pasti deterministik.



Gambar 5. Prinsip dasar ACO [14].

Secara informal, ACO bekerja sebagai berikut. Setiap semut mulai berputar melalui titik yang dipilih secara acak. Setiap semut memiliki titik awal yang berbeda. Berkali-kali mengunjungi titik-titik yang ada, satu per satu semut berniat untuk menghasilkan tur. Pemilihan poin didasarkan pada fungsi probabilitas yang disebut aturan transisi negara, dengan mempertimbangkan visibilitas terbalik jarak dan jumlah feromon yang terkandung dalam segmen ini [13]. Adapun prinsip dasar ACO dapat dilihat pada Gambar 5.

Faktor penting lainnya dari algoritma ACO adalah kadar feromon atau *Pheromone*. Feromon berfungsi sebagai energi yang mempengaruhi pemilihan jalur bercabang. Jumlah energy feromon berubah-ubah seiring banyaknya perjalanan atau pencarian yang terjadi [15]. Feromon ini adalah semacam sinyal untuk semut lain. Jalur yang pendek akan menyisakan sinyal yang lebih kuat. Semut berikutnya, ketika memutuskan jalan mana yang akan diambil, pada umumnya akan cenderung memilih untuk menempuh rute tersebut dengan sinyal yang terkuat, sehingga akan menghadapi jalur terpendek karena banyak semut yang akan melewati jalur tersebut [16].

Adapun langkah langkah dalam perhitungan ACO sebagai berikut :

1. Inisialisasi harga parameter-parameter algoritma:
   a. Intensitas jejak semut antar *node* dan perubahannya (τ_{ii}).
   b. Jarak antar node d_{ii}
   c. *Node* berangkat dan *node* tujuan.
   d. Tetapan pengendali intensitas jejak semut (α), nilai $\alpha \geq 0$.
   e. Tetapan pengendali intensitas visibilitas (β), nilai $\beta \geq 0$.
   f. Visibilitas antar *node* = $1/d_{ij}(nij)$.
2. Tentukan jarak antar *node*
3. Tentukan parameter *Alfa* (α), *Beta* (β), τ_{ij} awal, jarak antar *node*.
4. Tentukan Visibilitas antar *node* dengan menggunakan rumus:

$$(n_{ij}) = 1/d_{ij} \tag{3}$$

5. Tentukan penjumlahan dari visibilitas antar *node* yang telah dihitung dengan rumus:

$$\Sigma[\tau_{ik^1}]^{\alpha}.[n_{ik^1}]^{\beta} \tag{4}$$

6. Tentukan probabilitas dari *node* asal ke *node* berikutnya dapat dihitung dengan rumus:

$$p_{ij}^{k} = \frac{[\tau_{ij}]^{\alpha}.[n_{ij}]^{\beta}}{\Sigma[\tau_{ik^1}]^{\alpha}.[n_{ik^1}]^{\beta}} \tag{5}$$

7. Nilai probabilitas yang tertinggi merupakan jalur terdekat [17].

2.6. Kelebihan dan Kelemahan ACO

Algoritma ACO memiliki beberapa kelebihan, yaitu [18]:

1. Algoritma ACO menggunakan metode backtracking yang baik, sehingga dapat untuk mencapai solusi terbaik.
2. Algoritma ACO memiliki sistem kerjasama yang baik. Hal itu dapat ditunjukkan dengan efektivitas kerjasama antar koloni semut dalam mencari solusi dengan solusi terbaik.

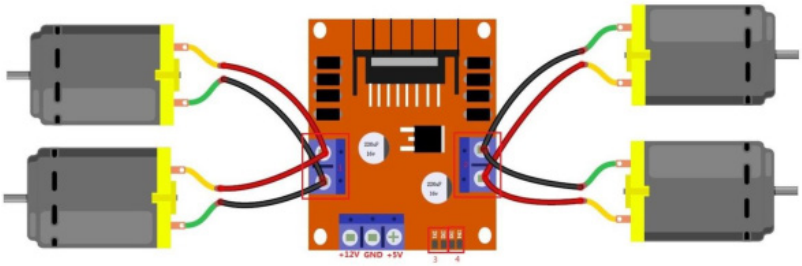
3. Menggunakan struktur yang lebih luas dalam algoritma ACO, yang dapat membantu untuk menemukan solusi yang dapat diterima pada tahap proses penelitian ini.
4. Algoritma ini dapat diterapkan pada versi yang sama untuk berbagai masalah optimisasi kombinatorial.
5. Algoritma dapat diterapkan untuk menyelesaikan masalah majemuk pengoptimalan lain, seperti QAP (*Quadratic Assignment Problem*) dan JSP (*Job Shop Scheduling Problem*), tanpa terlalu banyak perubahan.

Adapun kelemahan dari algoritma ini adalah tingkat kerumitan cukup tinggi, *runtime*-nya juga cukup lama [19].

3. Metode Penelitian

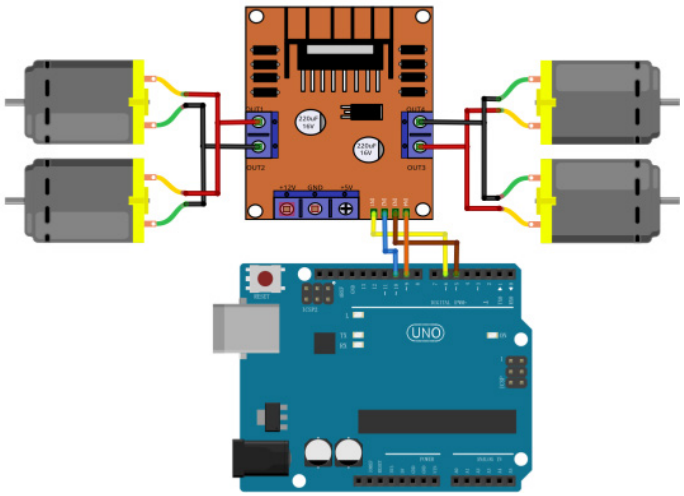
3.1. Perancangan *Mobile Robot*

Mobile robot dapat didefinisikan sebagai sistem mekanis yang dapat bergerak secara mandiri di lingkungannya. Rancangan *Mobile robot* dapat dilihat pada Gambar 6.

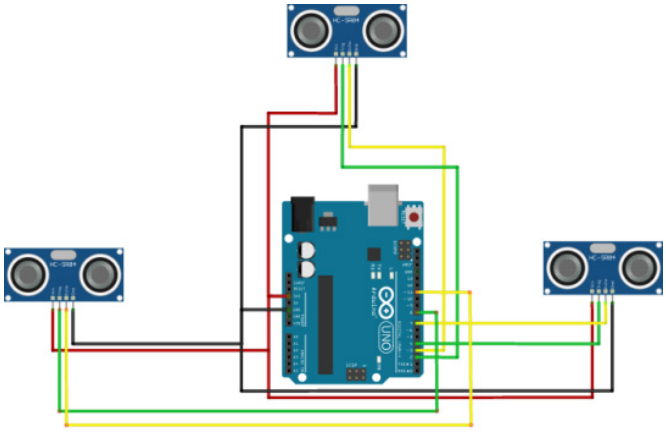


Gambar 6. Diagram skema kabel motor dinamo

Adapun rangkaian yang menghubungkan *motor drive board* ke *Arduino Uno* yang terdapat pada *N*1 pada *motor drive board* dimasukkan ke dalam *pin* 9, *N*2 ke *pin* 5, *N*3 ke *pin* 6, dan *N*4 ke *pin* 10, dapat dilihat pada Gambar 7.

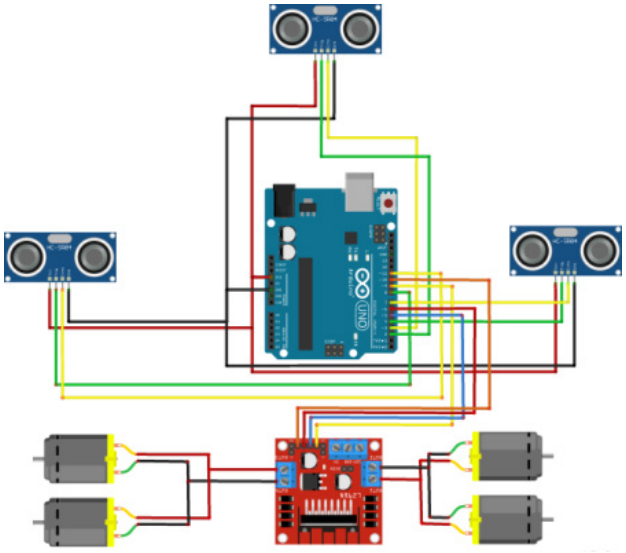


Gambar 7. Diagram skema *Motor Drive Board* dihubungkan ke *Arduino Uno*



Gambar 8. Diagram skema pengkabelan sensor ultrasonik

Diagram pengkabelan untuk sensor ultrasonik ditunjukkan pada Gambar 8. Pada metode ini digunakan tiga sensor ultrasonik yang berfungsi sebagai pengukuran jarak, mendeteksi halangan, dan sebagai penyeimbang robot. Untuk skema keseluruhannya dapat dilihat pada Gambar 9.



Gambar 9. Skema keseluruhan komponen

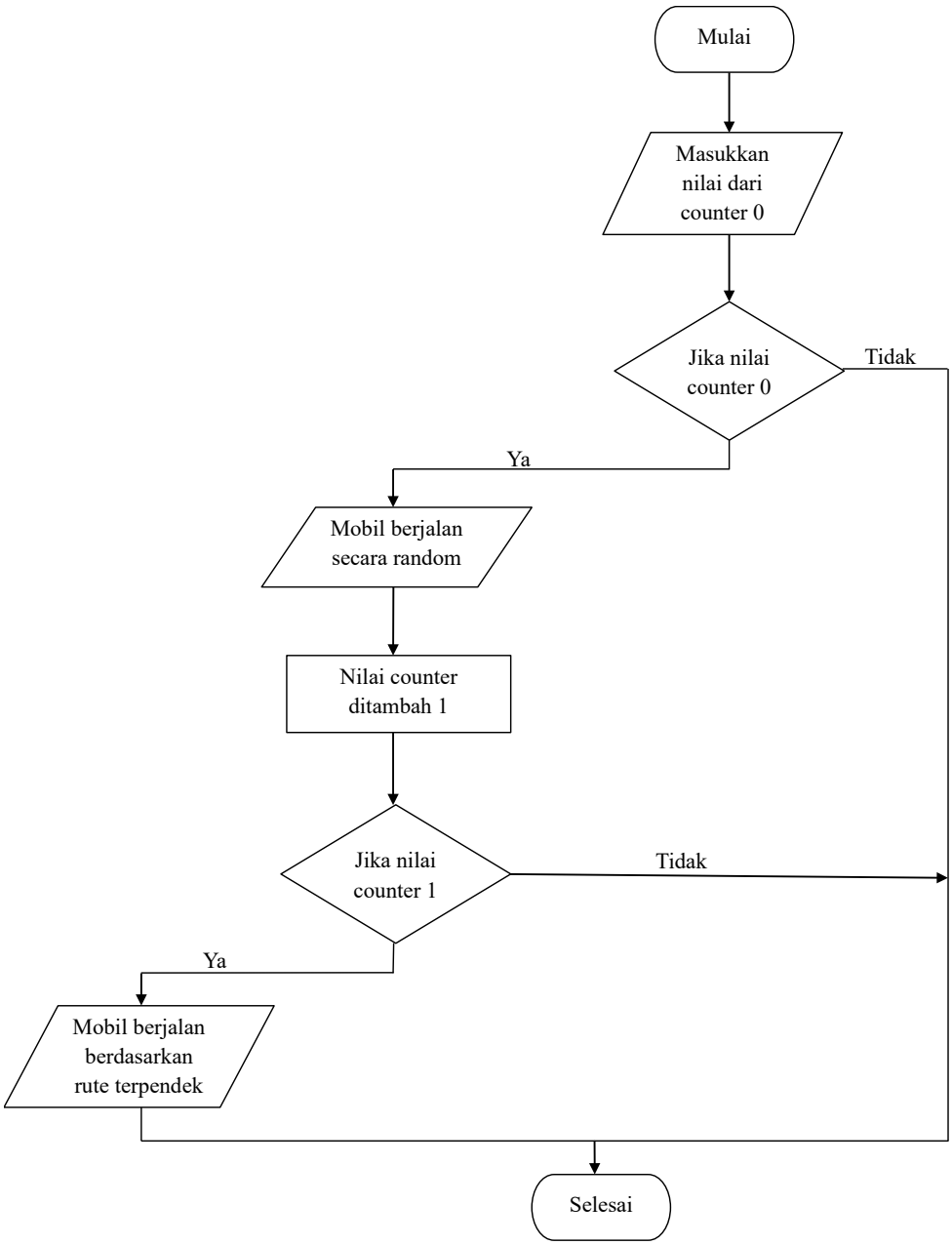
3.2. Perancangan Sistem

Alur *flowchart* dari *mobile* robot yang dibuat pada penelitian ini ditunjukkan pada Gambar 10. Berdasarkan dari *flowchart* tersebut, setelah mulai maka robot akan diberikan nilai *counter* 0, setelah itu robot akan dihadapkan pada dua pilihan, jika nilai *counter* 0 maka robot akan berjalan secara acak melalui tiga jalur yang telah ditentukan. Setelah robot berjalan hingga ke tujuan, maka nilai *counter* akan ditambah 1. Jika nilai *counter* tidak 0, maka program akan langsung keluar. Setelah robot berjalan hingga tujuan, maka robot akan dihadapkan pada dua pilihan lagi, jika nilai *counter* sama dengan 1, maka robot akan berjalan berdasarkan jalur terpendek, maka program selesai.

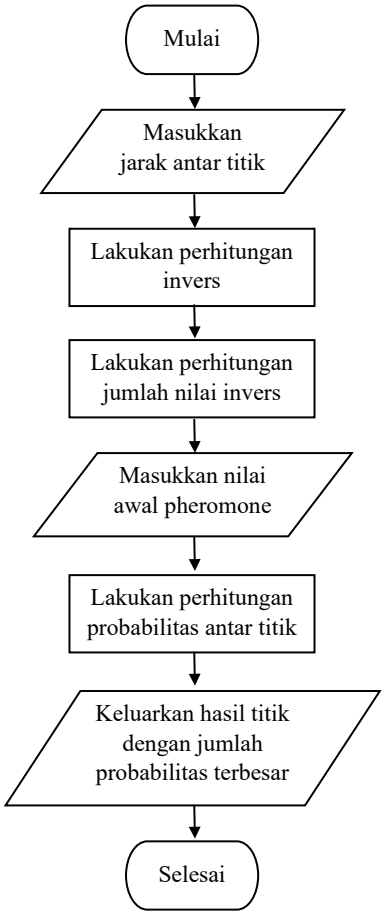
Flowchart dari metode *Ant Colony Optimizationm* dapat dilihat pada Gambar 11. Pada gambar tersebut terlihat langkah awal, yaitu mulai, setelah itu masuk ke *input* berupa jarak antar titik, setelah itu masuk ke proses, yaitu melakukan perhitungan *invert* untuk setiap titik. Setelah melakukan perhitungan *invert*, maka dilakukan proses perhitungan

probabilitas *invert*, setelah itu memasukkan nilai awal *pheromone*, setelah itu lakukan perhitungan probabilitas antar titik, selanjutnya keluarkan hasil tertinggi dari probabiitas antar titik, dan terakhir selesai.

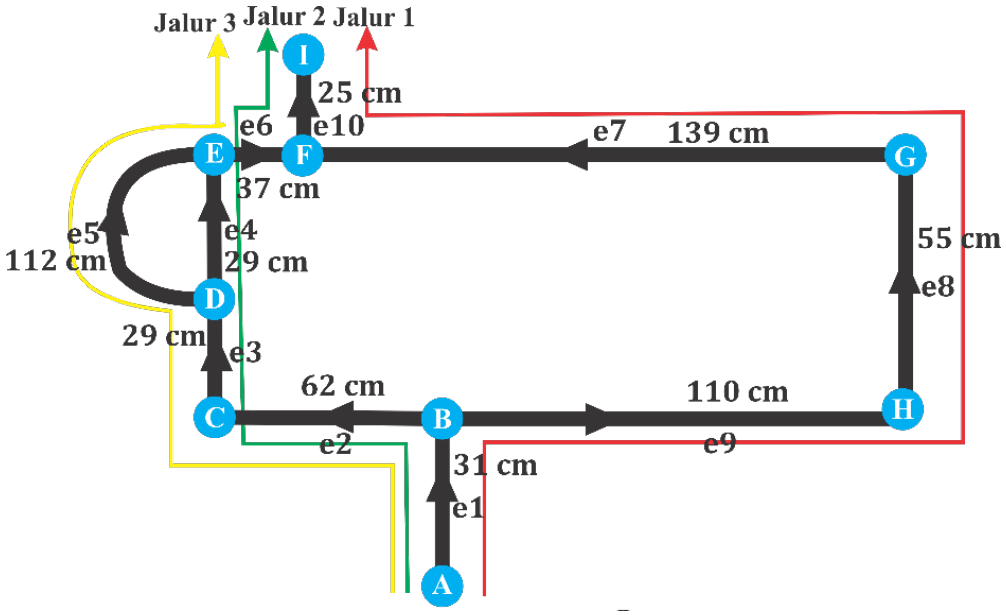
Bentuk rute *mobile robot* dalam versi graf dapat dilihat pada Gambar 12. Terdapat sembilan simpul dan 10 sisi pada graf tersebut. Simpul tersebut terdiri dari $V = \{A, B, C, D, E, F, G, H, I\}$ dan sisi pada graf tersebut terdiri dari $E = \{e1, e2, e3, e4, e5, e6, e7, e8, e9, e10\}$. Terdapat tiga jalur dengan tempat asal dan tujuan yang sama.



Gambar 10. *Flowchart* sistem dari *Mobile Robot*



Gambar 11. Flowchart Ant Colony Optimization



Gambar 12. Rute *Mobile* Robot dalam bentuk Graf

4. Hasil dan Pembahasan

4.1. Hasil Penelitian

Penentuan rekapan jarak disesuaikan dengan Gambar 12. Berdasarkan gambar tersebut, dihasilkan rekapan jarak per titik yang dapat dilihat pada Tabel 1. Pada tabel tersebut disajikan titik atau jalur yang dilalui, panjang jarak yang dilalui, waktu tempuh yang diperlukan untuk melalui jalur atau titik tersebut, serta probalilitas jarak atau peluang jarak yang didapat dari menggunakan rumus *Ant Colony Optimization* yang terdapat pada persamaan (5), dimana jika nilai dari probabilitas tersebut tinggi, maka jalur tersebut merupakan jalur terpendek untuk dilalui oleh robot tersebut.

Tabel 1. Hasil pengujian pengukuran jarak per titik

No.	Jalur	Jarak	Waktu Tempuh	Probabilitas Jarak
1	e1	31 cm	02,93 detik	1,000
2	e2	62 cm	03,22 detik	0,281
3	e3	29 cm	02,31 detik	0,681
4	e4	29 cm	0,68 detik	0,443
5	e5	112 cm	04,29 detik	0,155
6	e6	37 cm	01,72 detik	0,439
7	e7	139 cm	05,49 detik	0,071
8	e8	55 cm	02,71 detik	0,716
9	e9	110 cm	04,37 detik	0,333
10	e10	25 cm	01,90 detik	0,395

Tabel 2. Hasil pengujian pengukuran jarak per jalur

No	Jalur	Sisi Jalur	Panjang Lintasan	Waktu Tempuh	Probabilitas
1	Jalur Pertama	e1	31 cm	02,93detik	1,000
		e9	110 cm	04,37 detik	0,333
		e8	55 cm	02,71 detik	0,716
		e7	139 cm	05,49 detik	0,071
		e10	25 cm	01,90 detik	0,395
	Total		360 cm	17,4 detik	2,515
2	Jalur Kedua	e1	31 cm	02,93 detik	1,000
		e2	62 cm	03,22 detik	0,281
		e3	29 cm	02,31 detik	0,681
		e4	29 cm	0,63 detik	0,443
		e6	37 cm	01,72 detik	0,439
		e10	25 cm	01,90 detik	0,395
	Total		213 cm	13,02 detik	3,239
3	Jalur Ketiga	e1	31 cm	02,93 detik	1,000
		e2	62 cm	03,22 detik	0,281
		e3	29 cm	02,31 detik	0,681
		e5	112 cm	04,29 detik	0,115
		e6	37 cm	01,72 detik	0,439
		e10	25 cm	01,90 detik	0,395
	Total		296 cm	15,24 detik	2,911

Hasil pengujian pengukuran jarak per jalur pada *Mobile Robot* dapat dilihat pada Tabel 2. Pada tabel tersebut panjang lintasan jarak pada *Mobile Robot* didapat dengan cara menambahkan semua jarak per titik yang didapat pada Tabel 1 yang dilalui *Mobile Robot* pada masing masing jalur. Untuk total waktu tempuh didapat dengan cara

menambahkan semua waktu tempuh per titik yang didapat pada Tabel 1 yang dilalui *Mobile Robot* pada masing masing jalur. Untuk total probabilitas didapat dengan cara menambahkan semua probabilitas jarak per titik yang didapat pada Tabel 1 yang dilalui *Mobile Robot* pada masing masing jalur. Dari Tabel 2 jalur ke 2 yang memiliki total probabilitas paling tinggi daripada 2 jalur yang lainnya. Berdasarkan teori *Ant Colony Optimization* (ACO), nilai probabilitas tertinggi merupakan jalur terpendek yang dapat dilalui oleh *Mobile Robot*. Hal ini berarti bahwa jalur 2 merupakan jalur terpendek dengan nilai total probabilitas 3,239 dan waktu tempuh 13,46 detik.

Setelah dilakukan pengujian sebanyak 10 kali pengujian, maka didapatlah hasil pengujian seperti yang disajikan pada Tabel 3.

Tabel 3. Hasil pengujian *mobile robot*

No	Jalur	Panjang Lintasan	Waktu Tempuh	Total Probabilitas	Optimalisasi	Keterangan
1	Jalur Pertama	360 cm	22,44 detik	2,515	Sebelum Optimalisasi	Menuju Tujuan
2	Jalur Pertama	360 cm	22,03 detik	2,515	Sebelum Optimalisasi	Kembali Ke tempat asal
3	Jalur Kedua	213 cm	17,89 detik	3,239	Setelah Optimalisasi	Menuju Tujuan
4	Jalur Kedua	213 cm	15,00 detik	3,239	Setelah Optimalisasi	Kembali Ke tempat asal
5	Jalur Pertama	360 cm	24,37 detik	2,515	Sebelum Optimalisasi	Menuju Tujuan
6	Jalur Pertama	360 cm	21,00 detik	2,515	Sebelum Optimalisasi	Kembali Ke tempat asal
7	Jalur Kedua	213 cm	16,04 detik	3,239	Setelah Optimalisasi	Menuju Tujuan
8	Jalur Kedua	213 cm	17,05 detik	3,239	Setelah Optimalisasi	Kembali Ke tempat asal

4.2. Implementasi Rumus *Ant Colony Optimization* (ACO)

Langkah langkah perhitungan pada kasus pencarian jalur terpendek *Mobile Robot*:

1. Inisialisasi harga parameter-parameter algoritma:
   a. τ_{ij}= 0.01
   b. $\alpha = 1$
   c. $\beta = 1$
2. Menentukan jarak antar node (d_{ij})
   Untuk menentukan jarak antar *node* atau titik dari rute yang dilalui robot, maka diperlukan sensor ultrasonik yang berada pada bagian depan dari robot. Hasil dari pengukuran tersebut dapat dilihat pada Tabel 4.
3. Menentukan visibilitas antar *node*
   Untuk mendapatkan nilai visibilitas antar *node* atau titik dapat menggunakan rumus pada Persamaan (3). Nilai dari visibilitas antar *node* atau titik dapat dilihat pada Tabel 5.

Tabel 4. d_{ij} Jarak antar titik (*node*)

	A	B	C	D	E	E	F	G	H	I
A	0	31	0	0	0	0	0	0	0	0
B	31	0	62	0	0	0	0	0	110	0
C	0	62	0	29	0	0	0	0	0	0
D	0	0	29	0	29	112	0	0	0	0
E	0	0	0	29	0	0	37	0	0	0
E	0	0	0	112	0	0	37	0	0	0
F	0	0	0	0	37	37	0	139	0	25
G	0	0	0	0	0	0	139	0	55	0
H	0	110	0	0	0	0	0	55	0	0
I	0	0	0	0	0	0	25	0	0	0

Tabel 5. Nilai visibilitas antar titik

	A	B	C	D	E	E	F	G	H	I
A	0,000	0,032	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
B	0,032	0,000	0,016	0,000	0,000	0,000	0,000	0,000	0,009	0,000
C	0,000	0,016	0,000	0,034	0,000	0,000	0,000	0,000	0,000	0,000
D	0,000	0,000	0,034	0,000	0,034	0,009	0,000	0,000	0,000	0,000
E	0,000	0,000	0,000	0,034	0,000	0,000	0,027	0,000	0,000	0,000
E	0,000	0,000	0,000	0,009	0,000	0,000	0,027	0,000	0,000	0,000
F	0,000	0,000	0,000	0,000	0,027	0,027	0,000	0,007	0,000	0,040
G	0,000	0,000	0,000	0,000	0,000	0,000	0,007	0,000	0,018	0,000
H	0,000	0,009	0,000	0,000	0,000	0,000	0,000	0,018	0,000	0,000
I	0,000	0,000	0,000	0,000	0,000	0,000	0,040	0,000	0,000	0,000

4. Menentukan penjumlahan dari visibilitas antar node

   Untuk mendapatkan jumlah visibilitas antar *node* atau titik dapat menggunakan rumus pada persamaan (4). Jumlah dari visibilitas antar *node* atau titik dapat dilihat pada Tabel 6.

Tabel 6. Penjumlahan dari visibilitas

	A	B	C	D	E	E	F	G	H	I	Total
A	0,000	0,032	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
B	0,032	0,000	0,016	0,000	0,000	0,000	0,000	0,000	0,009	0,000	0,001
C	0,000	0,016	0,000	0,034	0,000	0,000	0,000	0,000	0,000	0,000	0,001
D	0,000	0,000	0,034	0,000	0,034	0,009	0,000	0,000	0,000	0,000	0,001
E	0,000	0,000	0,000	0,034	0,000	0,000	0,027	0,000	0,000	0,000	0,001
E	0,000	0,000	0,000	0,009	0,000	0,000	0,027	0,000	0,000	0,000	0,000
F	0,000	0,000	0,000	0,000	0,027	0,027	0,000	0,007	0,000	0,040	0,001
G	0,000	0,000	0,000	0,000	0,000	0,000	0,007	0,000	0,018	0,000	0,000
H	0,000	0,009	0,000	0,000	0,000	0,000	0,000	0,018	0,000	0,000	0,000
I	0,000	0,000	0,000	0,000	0,000	0,000	0,040	0,000	0,000	0,000	0,000

5. Menentukan probabilitas dari *node* asal ke *node* berikutnya

   Untuk menentukan probabilitas dari *node* asal ke *node* berikutnya dapat menggunakan rumus pada persamaan (5).

Tabel 7. Nilai probabilitas dari *node* asal ke *node* berikutnya

	A	B	C	D	E	E	F	G	H	I	Jalur terdekat
A	0,000	**1,000**	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	**AB**
B	0,561	0,000	**0,281**	0,000	0,000	0,000	0,000	0,000	0,158	0,000	**BC**
C	0,000	0,319	0,000	**0,681**	0,000	0,000	0,000	0,000	0,000	0,000	**CD**
D	0,000	0,000	0,443	0,000	**0,443**	0,115	0,000	0,000	0,000	0,000	**DE**
E	0,000	0,000	0,000	0,561	0,000	0,000	**0,439**	0,000	0,000	0,000	**EF**
E	0,000	0,000	0,000	0,248	0,000	0,000	**0,752**	0,000	0,000	0,000	**EF**
F	0,000	0,000	0,000	0,000	0,267	0,267	0,000	0,071	0,000	**0,395**	**FI**
G	0,000	0,000	0,000	0,000	0,000	0,000	0,284	0,000	0,716	0,000	GH
H	0,000	0,333	0,000	0,000	0,000	0,000	0,000	0,667	0,000	0,000	HG
I	0,000	0,000	0,000	0,000	0,000	0,000	1,000	0,000	0,000	0,000	IF

4.3. Perbandingan Metode Ant Colony Optimization dengan Algoritma Bellman-ford

Iterasi perbandingan metode *Ant Colony Optimization dengan* Algoritma *Bellman-ford* dalam kasus pencarian jalur terpendek dapat dilihat pada Tabel 8.

Tabel 8. Perbandingan Metode ACO dengan Algoritma *Bellman-ford*

Node (Titik)	ACO	Algoritma *Bellman-Ford*
A – B	1	31
B – C	0,281	93
B – H	0,158	141
C – D	0,681	112
G – H	0,716	196
D – E	0,443	151
D – E	0,115	234
E – F	0,439	188
E – F	0,439	271
F – G	0,071	335
F – I	0,395	360
F – I	0,395	213
F – I	0,395	296

Dari hasil iterasi yang terdapat pada Tabel 8, maka dapat disimpulkan jumlah probabilitas dalam algoritma ACO dan total jarak pada algoritma *Bellman-ford* dapat dilihat pada Tabel 9.

Tabel 9. Total Probabilitas ACO dan Jarak *Bellman-ford*

Jalur	ACO	*Bellman-ford*
A-B-H-G-F-I (Jalur 1)	2,515	360
A-B-C-D-E-F-I (Jalur 2)	3,239	213
A-B-C-D-E-F-I (Jalur 3)	2,911	296

Berdasarkan hasil pada Tabel 9, dimana metode *Ant Colony Optimization* dan Algoritma *Bellman-ford* menunjukkan bahwa jalur 2 merupakan jalur terpendek untuk lintasan pada Gambar 12.

4.4. Pembahasan Penelitian

Penelitian ini menggunakan *Mobile Robot*, yaitu *Hummerbot*. *Mobile Robot* tersebut dilengkapi dengan tiga buah sensor ultrasonik. Satu buah sensor berada di depan, dan dua sensor berada di sisi kiri dan kanan robot. Untuk sistem pergerakan robot, robot bergerak berdasarkan pembagian jalur pada robot, terdapat tiga jalur pada *track* tersebut

jalur 1 melalui jalur e1, e9, e8, e7, e10, jalur 2 melalui jalur e1, e2, e3, e4, e6, e10, jalur 3 melalui jalur e1, e2, e3, e5, e6, e10. Untuk pengukuran jarak antar titik digunakan sensor ultrasonik yang berada di bagian depan pada robot tersebut. Data jarak antar titik diukur dari jarak posisi robot berada tepat di setiap titik hingga terdapat rintangan di hadapan sensor ultrasonik tersebut.

Untuk metode ACO, setelah mendapatkan jarak antar titik dengan bantuan sensor ultrasonik, setiap nilai dari jarak antar titik titik tersebut ditentukan nilai visibilitas antar titik. Setelah divisibilitas, nilai jarak antar titik yang telah divisibilitas tersebut di jumlahkan setiap baris dalam tabel. Setelah mendapatkan jalur terpendek, robot tersebut diprogram dengan menggunakan bantuan dari fungsi *random* untuk menentukan jalur awal robot tersebut jalan sebelum dilakukan optimalisasi. Setelah robot selesai jalan ke jalur yang ditentukan oleh fungsi random tersebut, robot akan berjalan melewati jalur terpendek yang telah didapat dengan menggunakan metode ACO.

5. Kesimpulan

Dari penelitian ini, disimpulkan bahwa penggunaan metode ACO dalam optimalisasi pencarian jalur terpendek untuk *mobile robot* telah terbukti efektif. Serangkaian percobaan sebanyak 20 kali mengindikasikan bahwa setelah penerapan metode ini, robot mampu secara akurat menentukan jalur terpendek, dengan contoh terbaik yang ditemukan adalah jalur 2 dengan panjang total 213 cm, probabilitas 3,239, dan waktu tempuh sekitar 14 hingga 17 detik. Hasil ini menunjukkan bahwa ACO dapat berperan penting dalam pengambilan keputusan terkait navigasi di persimpangan jalan yang kompleks. Selain itu, ditegaskan bahwa Algoritma *Bellman-Ford* juga merupakan alternatif yang efektif dalam pencarian jalur terpendek, dengan kemampuannya menghasilkan total jarak dari titik awal hingga tujuan.

Daftar Pustaka

[1] A. Sadiyoko, "Industry 4.0 ancaman, tantangan atau kesempatan," *Oratio Dies XXIV FTI Unpar*, pp. 1–36, 2017.

[2] M. Farwati, I. T. Salsabila, K. R. Navira, T. Sutabri, "Analisa pengaruh teknologi artificial intelligence (AI) dalam kehidupan sehari-hari," *Jursima J. Sist. Inf. Manaj.*, 11, 1, pp. 39–45, 2023.

[3] Tarwaka, *Ergonomi Industri*, no. May. 2011.

[4] N. Setyawan, "Pengembangan adaptive particle swarm optimization (PSO) dan aplikasinya pada perencanaan jalur mobile robot dengan halangan dinamis," *Teknol. Elektro*, pp.1-14, 2017.

[5] V. R. Palilingan, "Multi-objective ACO (MO-ACO) algorithm untuk optimasi strategi mitigasi bencana", Tondano: Unimja Press, 2018.

[6] T. Rambey, A. N. Hasibuan, B. Prakoso, A. Y. Astuti, "Strategi manajemen perubahan hipmikindo dalam membangun sumberdaya technopreneur dengan mendirikan entrepreneur centres," *J. Bisnis, Logistik dan Supply Chain*, 1, 2, pp. 51–59, 2021. Doi: 10.55122/blogchain.v1i2.311.

[7] V. Risqiyanti, H. Yasin, R. Santoso, "Pencarian jalur terpendek menggunakan metode algoritma 'ant colony optimization' pada GUI Matlab (Studi kasus: PT Distriversa Buana Mas cabang Purwokerto)," *J. Gaussian*, 8, 2, pp. 272–284, 2019. Doi: 10.14710/j.gauss.v8i2.26671.

[8] A. Kadir, "Arduino & sensor: Tuntunan praktis mempelajari penggunaan sensor untuk aneka proyek elektronika berbasis Arduino," Yogyakarta: Andi (Anggota IKAPI), 2018.

[9] K. Fatmawati, E. Sabna, Y. Irawan, "Rancang bangun tempat sampah pintar menggunakan sensor jarak berbasis mikrokontroler Arduino," *Riau J. Comput. Sci.*, 6, 2, pp. 124–134, 2020.

[10] F.- Puspasari, I.- Fahrurrozi, T. P. Satya, G.- Setyawan, M. R. Al Fauzan, E. M. D. Admoko, "Sensor ultrasonik HCSR04 berbasis Arduino Due untuk sistem monitoring ketinggian," *J. Fis. dan Apl.*, 15, 2, pp. 36, 2019. Doi: 10.12962/j24604682.v15i2.4393.

[11] F. Teknik, P. Studi, T. Elektro, U. Widya, D. Klaten, "Komparasi sensor ultrasonik HC-SR04 dan JSN-SR04T," 10, 2, pp. 717–724, 2019.

[12] M. K. Harahap, N. Khairina, "Pencarian jalur terpendek dengan algoritma Dijkstra," *SinkrOn*, 2, 2, pp. 18, 2017. Doi: 10.33395/sinkron.v2i2.61.

[13] J. Matematika, F. Matematika, D. A. N. Ilmu, P. Alam, U. N. Semarang, "Ant colony optimization dalam penyelesaian travelling salesman problem," 2016.

[14] D. Udjulawa, S. Oktarina, "Penerapan algoritma ant colony optimization untuk pencarian rute terpendek lokasi wisata," *Klik - J. Ilmu Komput.*, 3, 1, pp. 26–33, 2022. Doi: 10.56869/klik.v3i1.326.

[15] V. Maniezzo, L. M. Gambardella, F. De Luigi, "5 . Ant colony optimization," pp. 1–21, 1991.

[16] M. Dorigo, K. Socha, "Ant colony optimization," *Handb. Approx. Algorithms Metaheuristics*, pp. 26-1-26–14, 2007. Doi: 10.1201/9781420010749.

[17] B. Triandi, "Penemuan jalur terpendek dengan algoritma ant colony," *CSRID Journal*, 4, 2, pp. 73-80, Juni 2012.

[18] I. A. Ashari, "Perbandingan performansi algoritma genetika dan algoritma ant colony optimization dalam optimasi penjadwalan mata kuliah," *Repos. Univ. Negeri Semarang*, pp. 1–80, 2016, [Online]. Available: https://lib.unnes.ac.id/28048/1/4611412015.pdf

[19] A. Setiawan, F. Andriyanto, L. S. Putro, N. P. T.P., U. Permana, "Perbandingan algoritma ant colony optimization, disjktra, tabu search, multiple ant colony system untuk vehicle routing problem dengan time window," 2012.