TUGAS AKHIR – TE 141599

NAVIGASI *MOBILE ROBOT NONHOLONOMIC* MENGGUNAKAN *FUZZY – ANT COLONY SYSTEM*

Dimas Bintang Pamungkas
NRP. 2213100158

Dosen Pembimbing
Dr. Trihastuti Agustinah, ST., MT
Ir. Rusdhianto Effendi AK., MT

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2017

FINAL PROJECT – TE 141599

NONHOLONOMIC MOBILE ROBOT NAVIGATION USING FUZZY – ANT COLONY SYSTEM

Dimas Bintang Pamungkas
NRP. 2213100158

Supervisor
Dr. Trihastuti Agustinah, ST., MT
Ir. Rusdhianto Effendi AK., MT

DEPARTEMENT OF ELECTRICAL ENGINEERING
Faculty of Electrical Technology
Sepuluh Nopember Insitute of Technology
Surabaya 2017

PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul **"Navigasi *Mobile Robot* Nonholonomic menggunakan *Fuzzy-Ant Colony System*"** adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Juli 2017

Dimas Bintang P.
NRP. 2213 100 158

[Halaman ini sengaja dikosongkan]

NAVIGASI *MOBILE ROBOT NONHOLONOMIC*
MENGGUNAKAN *FUZZY – ANT COLONY SYSTEM*

TUGAS AKHIR

Diajukan untuk Memenuhi Sebagian Persyaratan Untuk
Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Pengaturan
Departemen Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Trihastuti Agustinah, S.T., M.T.	Ir. Soedhianto Effendi A.K., M.T.
NIP. 196808121994032001	NIP. 195204241985021001

DEPARTEMEN TEKNIK ELEKTRO

SURABAYA
JULI, 2017

Navigasi *Mobile Robot Nonholonomic* menggunakan *Fuzzy-Ant Colony System*

Dimas Bintang Pamungkas
2213100158

Dosen Pembimbing 1 : Dr. Trihastuti Agustinah, ST., MT.
NIP : 196808121994032001
Dosen Pembimbing 2 : Ir. Rusdhianto Effendi AK., MT.
NIP : 195704241985021001

ABSTRAK

Navigasi robot merupakan penentuan kedudukan (*position*) dan arah perjalanan baik di medan sebenarnya atau di peta. Navigasi atau perencanaan jalur mempunyai peranan penting dalam teknologi *mobile robot*. Secara sederhana, melakukan perencanaan jalur yaitu menghasilkan titik-titik yang dapat dilewati (*feasible)* melalui informasi yang telah ada yaitu peta lingkungan robot.

Pemodelan lingkungan adalah langkah pertama untuk *path planning*. Untuk lingkungan yang tidak diketahui, pemodelan dilakukan dengan informasi yang diperoleh dari sensor. Pada lingkungan yang nyata, *noise* dan keterbatasan akurasi dari sensor mengarah ke masalah dasar dalam *modeling*, yaitu kecepatan dan akurasi, yang mempengaruhi navigasi langsung secara *real time*. Untuk mengatasi masalah tersebut diperlukan *Fuzzy Description Environment* yang terdiri atas model *fuzzy* dari informasi yang didapat di lingkungan sekitar robot. Dari model tersebut akan menjadi acuan *path planner* yang menjadikan referensi pada *path planning*. Dari pemodelan *map environtment* akan diproses oleh *ant colony* sehingga didapatkan jalur menuju titik akhir.

Dari hasil simulasi program didapatkan titik-titik *waypoint* dengan jarak terpendek yaitu 363,2724 satuan piksel.

Kata Kunci : Perencanaan Jalur, *Fuzzy*, *Ant Colony*, *Mobile Robot* Nonholonomik

[Halaman ini sengaja dikosongkan]

Nonholonomic Mobile Robot Navigation using Fuzzy-Ant Colony System

Dimas Bintang Pamungkas
2213100158

Supervisor 1 : Dr. Trihastuti Agustinah, ST., MT.
ID : 196808121994032001
Supervisor 2 : Ir. Rusdhianto Effendi AK., MT.
ID : 195704241985021001

ABSTRACT

Navigation is the process or activity of accurately ascertaining one's position and planning and following a route. Path planning or navigation plays an important role in mobile robot technology. path planning produce the waypoints that can be passed (feasible) through the existing information is a robot environment map.

. Environmental modeling is the first step for path planning. For unknown environments, the modeling is done with information generated from the sensor. In the real environment, noise and limitations of accuracy of the sensor lead to the basic problem in modeling. To overcome these problems required Fuzzy Description Environment consisting of fuzzy models of information obtained in the environment around the robot.

From the model will be a map, and map which becomes the reference of path planning. From modeling the enivirontment map will be processed by ant colony so there is a path to the end point.

From the simulation result of the program with shortest path distance is 363.2724 pixel unit.

Keywords - Path Planning, Fuzzy, Ant Colony, Nonholonomic Mobile Robot.

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji syukur kehadiran Allah SWT, Dialah Tuhan pemilik segala ilmu, yang memberikan pemahaman kepada semua makhluk atas ilmu-Nya dan berkat karunia-Nya lah sehingga penelitian Tugas Akhir ini dapat terlaksanakan dengan baik.

Tugas Akhir ini disusun untuk memenuhi syarat menyelesaikan pendidikan Strata-1 pada Bidang Studi Sistem Pengaturan, Fakultas Teknik Elektro, Institut Teknologi Sepuluh Nopember Surabaya. Dengan judul Tugas Akhir :

"*Navigasi *Mobile Robot Nonholonomic* menggunakan *Fuzzy-Ant Colony System"

Dalam Pelaksanaan Tugas Akhir maupun penyusunan buku ini penulis telah dibantu oleh banyak pihak, dan pada kesempatan ini, penulis haturkan ucapan terima kasih kepada :

1. Kedua orang tua penulis yang telah membesarkan dan mendidik anak-anaknya dengan doa dan ridho mereka
2. Kedua Dosen Pembimbing penulis, ibu Dr. Trihastuti Agustinah, ST., MT. Dan bapak Ir. Rusdhianto Effendi AK., MT. yang memberikan arahan dan bimbingan selama penelitian Tugas Akhir
3. Teman-teman seperjuangan Teknik Sistem Pengaturan yang memberikan inspirasi, dukungan serta motivasi kepada penulis
4. Khairurizal Alfathdiyanto, ST. Sebagai alumni yang memberikan bantuan atas pendapat akan Tugas Akhir ini

Penulis menyadari akan adanya kekurangan dalam penulisan Tugas Akhir ini; oleh karena itu saran serta masukan akan sangat dibutuhkan dan dapat diterima demi kebermanfaatan Tugas Akhir ini. Semoga penulisan buku ini dapat memberikan sumbangsih kepada teknologi dan kebaikan dimasa depan.

Surabaya, 05 Juni 2017

Penulis

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

TABLE OF CONTENT

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

BAB 1
PENDAHULUAN

Pada bab ini akan dibahas mengenai hal-hal yang mandahului tugas akhir. Hal-hal tersebut meliputi latar belakang, permasalahan, tujuan penelitian, metodologi sistematika penulisan dan relevansi.

1.1 Latar Belakang

Dalam beberapa dekade terakhir robot merupakan salah satu fokus dalam sains. Terlebih penggunaan robot sebagai pengganti manusia dalam hal eksplorasi dan mobilisasi daerah yang tidak mudah dilalui manusia. Sebagai dasar dari penggunaan robot itu, mobilisasi robot merupakan hal yang sangat penting dimana dalam hal menghindari halangan dan pengambilan keputusannya adalah hal penting. Untuk menghindari rintangan terlebih untuk lingkungan yang tidak diketahui dapat digunakan beberapa metode salah satunya dengan menggunakan *Fuzzy* untuk memodelkan lingkungan sekitar dan *Ant Colony System(ACS)* untuk mencari jalur optimal untuk menghindari rintangan.

Untuk lingkungan tidak diketahui dalam rana kerja robot mobil merupakan tantangan yang sangat sulit terlebih ketika dihadapi dengan lingkungan yang dinamis sehingga rintangan setiap kali perlu menjadi tinjauan untuk kerja robot mobil.

Pemodelan lingkungan adalah langkah pertama untuk *path planning*. Untuk lingkungan yang tidak diketahui, pemodelan dilakukan dengan informasi yang diperoleh dari sensor dan biasanya, lingkungan yang nyata, *noise* dan keterbatasan akurasi dari sensor mengarah ke masalah dasar dalam *modeling*, yaitu kecepatan dan akurasi, yang mempengaruhi navigasi langsung secara *real time*. Untuk mengatasi masalah tersebut diperlukan *Fuzzy Description Environment* yang terdiri atas model *fuzzy* dari informasi yang didapat di lingkungan sekitar robot.

Ant colony system (ACS) yang dikembangkan oleh M.Dorigo et al. Titik awal dari algoritma ini adalah untuk mengoptimalkan jalan dengan mensimulasikan fenomena semut mencari makan. ACS memiliki keuntungan dari *co-processing* dan *parallel processing* maka navigasi yang agak rumit dapat dicapai dengan algoritma sederhana.

Algoritma koloni semut biasa digunakan dalam penyelesaian optimal seperti *scheduling* dan *routing*. Namun pada penelitian Tugas

Akhir ini, koloni semut digunakan sebagai penentuan jalur atau *path planning* pada navigasi robot.

Hasil yang diberikan algoritma berupa titik jalur yang bergantung pada lingkungan sekitar yang dideskripsikan sensor secara terus menerus, sehingga dapat bekerja pada lingkungan dinamik juga.

1.2 Rumusan Masalah

Mobile robot diharapkan bisa menghindari semua rintangan yang ada di dalam sebuah lingkungan baik rintangan yang statis maupun dinamis. Oleh karena itu, diperlukan sebuah kontroler yang bisa mengatur gerak *mobile robot* sekaligus menghindari berbagai rintangan tersebut. Pada Tugas Akhir ini, akan dibuat sebuah algoritma *Ant Colony System* (*ACS)* sehingga bisa terjadi proses *learning* yang diharapkan akan memiliki performansi yang lebih baik dari sebelumnya.

1.3 Batasan

Dalam penelitian Tugas Akhir ini memiliki batasan-batasan sebagai berikut:

1. Simulasi bekerja menggunakan peta/*map* buatan sesuai dengan keadaan sebenarnya.
2. Simulasi menggunakan sensor yang ideal dengan presisi tinggi, dengan masukan jarak pada tiap arah robot.
3. Simulasi menghasilkan keluaran berupa titik yang dapat dituju oleh robot berupa koordinat yang sama pada peta.
4. Implementasi dilakukan pada bidang datar, sedangkan tanjakan bisa menjadi halangan jika terdeteksi pada sensor.
5. Navigasi dilakukan dalam keadaan statis tanpa adanya kejadian dinamis

1.4 Tujuan

Tujuan dari pelaksanaan Tugas Akhir ini adalah menghasilkan:

1. Menghasilkan navigasi *mobile robot* menggunakan *Ant Colony System* yang dipadukan dengan *Fuzzy Environment Modelling.*
2. Mendapatkan *waypoint* terpendek menggunakan algoritma tersebut.
3. Mendapatkan parameter *Ant Colony* yang terbaik dari setiap percobaan simulasi.

1.5 Metodologi

Metodologi yang digunakan pada penelitian Tugas Akhir ini adalah sebagai berikut:

1. **Studi Literatur**
   Dengan mengumpulkan data dan literatur dari jurnal dan buku, akan didapatkan pendapat, metode, dan pemodelan
   dengan permasalahan yang mirip dengan penelitian Tugas Akhir yang ditulis ini.
2. **Identifikasi Masalah**
   Melakukan identifikasi melalui batasan yang telah ditentukan dan membandingkan dengan keadaan yang sebenarnya.
3. **Perancangan Algoritma**
   Pada tahap ini dilakukan perancangan sebuah algoritma untuk perencanaan jalur dengan Algoritma koloni semut
4. **Simulasi**
   Pada tahap ini hasil perancangan akan diuji dengan cara mensimulasikan pada MATLAB, sehingga dapat dilakukan pengamatan dan analisa terhadap keluaran sistem.
5. **Implementasi**
   Dengan melakukan implementasi perangkat lunak pada alat yang ada yang berupa memberi masukan kepada simulasi robot. Spesifikasi robot dan hasil keluaran dari *software* yang disesuaikan dengan pemrograman robot
6. **Penulisan Buku Tugas Akhir**
   Setelah didapatkan hasil dari simulasi dan implementasi, akan dituliskan hasil dari penelitian Tugas Akhir juga analisa terhadap hasil dan metode yang digunakan, serta saran agar penelitian pada metode ini agar berkembang

1.6 Sistematika

Pembahasan Tugas Akhir ini dibagi menjadi lima bab dengan sistematika sebagai berikut :

BAB I : Pendahuluan
Bab ini meliputi latar belakang, permasalahan, tujuan penelitian, metodologi dan sistematika penulisan.

BAB II : Teori Penunjang

Bab ini membahas tinjauan pustaka yang membantu penelitian, diantaranya teori *mobile robot*, pemodelan pergerakan nonholonomik, *path planning*, *fuzzy* dan algoritma koloni semut

BAB III : Perancangaan Sistem

Bab ini dijelaskan mengenai identifikasi permasalahan yang akan dirancang, perancangan simulasi, dan implementasi

BAB IV : Hasil dan Analisa

Bab ini memuat hasil simulasi dan implementasi perencanaan jalur dengan program yang telah dibuat.

BAB V : Penutup

Bab ini berisi kesimpulan dan saran dari hasil penelitian yang telah dilakukan.

1.7 Relevansi

Hasil yang diperoleh dari Tugas Akhir ini diharapkan menjadi referensi pengembangan teknologi dan perbandingan metode *path planning* yang ada pada robot nonholonomik, khususnya dengan metode koloni semut di masa mendatang.

BAB 2
TEORI PENUNJANG

Pada bagian ini berisi tentang teori yang menunjang penelitian Tugas Akhir yang berupa pemahaman dasar, baik karakteristik robot, teori perencanaan lintasan dan navigasi, maupun algoritma cerdas seperti *fuzzy* dan algoritma koloni semut. Seluruh teori yang digunakan disajikan untuk memperkuat argumen penulis dalam melakukan penelitian.

2.1 Mobile Robot

Mobile robot atau robot bergerak merupakan teknologi yang berkembang pesat di dunia ini. Mobile robot merupakan robot yang memiliki kegunaan yaitu berpindah posisi dari satu titik ke titik yang lain, bergerak di sekitar lingkungannya. [1]

Untuk dapat membuat sebuah robot mobile minimal diperlukan pengetahuan tentang mikrokontroler dan sensor-sensor elektronik. Sensor digunakan untuk mengenali lingkungan pada robot sehingga robot dapat menghindari halangan yang ada di sekitar. Untuk robot bersensor biasanya berperan sebagai AMR atau *autonomous mobile robot*, selain itu juga ada robot menggunakan sensor tapi tanpa perlu kontrol cerdas untuk menentukan jalur jalannya yaitu AGV atau *autonomous guided vehicle* dimana robot ini menggunakan sensor untuk mengikuti jalur yang telah ditentukan berupa garis terang gelap (*line follower robot)*, dinding (*wall follower robot*), ataupun cahaya. Ada juga robot yang tanpa menggunakan sensor, namun digerakkan menggunakan kontroller yang dijalankan oleh manusia, robot ini sekarang sudah banyak dijual dipasarkan berupa mainan menggunakan remot kontrol.

Sensor yang digunakan pada robot juga ada banyak macam, baik laser, sonar, kamera, dan juga gps. Pada robot-robot canggih juga terdapat sensor kemiringan tanah agar tidak terjatuh dalam melewati jalur-jalur tertentu.

Mobile robot kebanyakan memiliki alat gerak berupa roda, namun juga ada yang berupa kaki serangga, maupun *caterpillar track* seperti pada gambar berikut. [1]

Gambar 2.1 *Mobile Robot* Menggunakan *Caterpillar Track* [1]

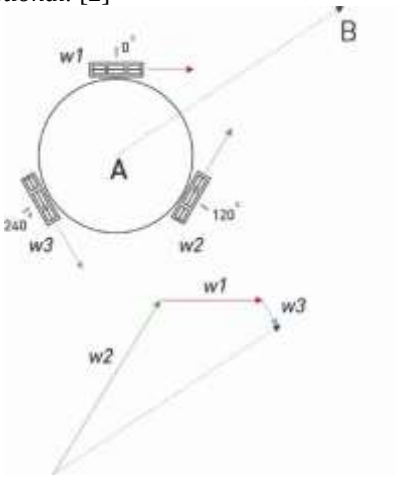
Gambar 2.2 *Mobile Robot* Berkaki [1]

Gambar 2.3 *Mobile Robot Beroda* [1]

2.2 Pergerakan Robot

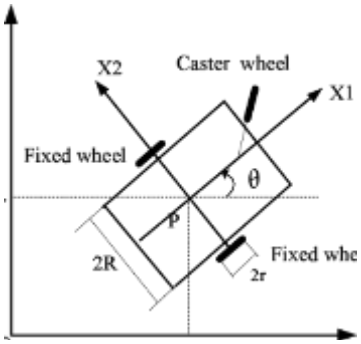
Pergerakan *mobile robot* dapat digolongkan menjadi dua yaitu nonholonomik dan holonomik. Dengan melihat perbedaannya kita dapat memberikan persamaan matematis pada sistem robot.

Sebuah *mobile robot* dikatakan pergerakan nonholonomik jika derajat yang dikontrol sama dengan derajat kebebasan. Dalam hal ini mobile robot dapat bergerak ke segala arah (*omni-directional)* tanpa harus mengubah arah sudut robot (badan robot). Robot tersebut dapat bergerak ke kedua sumbu (X dan Y) yang membuat robot tersebut bergerak ke arah tersebut. Robot ini bergerak menggunakan roda kastor atau bisa juga roda *omni-directional.* [2]



Gambar 2.4 Pergerakan *Mobile Robot* Secara Holonomik [2]

Sedangkan untuk robot nonholonomik Arah gerak nonholonomik dibatasi oleh efek sentuhan arah roda yang menempel di atas permukaan. Robot tidak bisa bergerak bebas ke arah samping robot. Sehingga arah robot harus diubah terlebih dahulu sebelum berpindah karena arah gerak mengikuti arah hadap roda. Robot dengan sifat gerak nonholonomik mempunyai keterbatasan dalam arah gerak. Fungsi geometri tertentu yang berhubungan dengan orientasi harus dipenuhi untuk mendapatkan gerak yang sesuai Biasanya robot nonholonomik menggunakan dua roda di kanan dan kiri robot secara paralel dengan satu roda kastor sebagai penyeimbang (seperti pada Gambar 2.3). Setiap roda digerakkan dengan motor yang independen.



Gambar 2.5 Pergerakan *Mobile Robot* Secara Nonholonomik [2]

2.3 Matriks Transformasi

Untuk memindahkan suatu titik atau bangun pada bidang dapat dilakukan dengan menggunakan transformasi. Transformasi geometri adalah bagian dari geometri yang melakukan perubahan dari titik awal ke titik yang telah ditransformasi. Terdapat macam-macam transformasi linear, untuk kasus ini digunakan matriks translasi dan rotasi sebagai bantuan matematis pada kinematika robot.

$$T(x) = \boldsymbol{A}x \tag{2.1}$$

Dimana matriks $\boldsymbol{A}$ merupakan matriks transformasi $m \times n$ dan $T(x)$ merupakan hasil transformasi dari titik x.

2.3.1 Rotasi

Untuk merotasi sebuah titik (x,y) dengan sudut θ dengan searah jarum jam maka menggunakan fungsi.

$$x' = x\cos\theta - y\sin\theta \tag{2.2}$$

$$y' = x\sin\theta + y\cos\theta \tag{2.3}$$

Maka matriks transformasinya

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.4}$$

2.3.2 Translasi

Untuk menggeser titik (x,y) ke arah (dx, dy) menggunakan fungsi

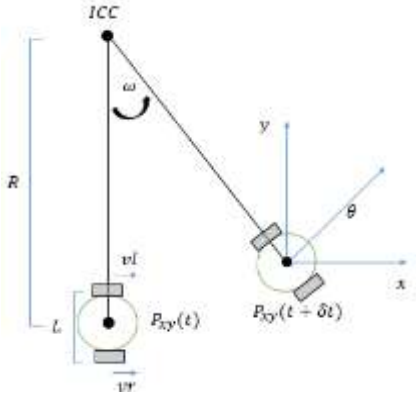
$$x' = x + dx \quad (2.5)$$
$$y' = y + dy \quad (2.6)$$

Maka matriks transformasinya

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix} \quad (2.7)$$

2.4 *Differential Drive Motor*

Sebuah *differential drive robot* memiliki dua roda yang bisa bergerak secara independen. Perbedaan atau kesamaan kecepatan kedua roda menentukan apakah robot tersebut akan bergerak rotasi atau lurus. [3]



Gambar 2.6 Parameter *Differential Drive Robot [3]*

Dimana L merupakan jarak antara dua roda kanan dan kiri. vr dan vl merupakan kecepatan translasi roda kanan dan kiri berturut-turut. Dan R merupakan jarak yang diukur dari ICC (*Instaneous Center of Curvature*) yang merupakan titik robot berotasi dengan titik tengah robot (titik tengah antara dua roda). ICC terbentuk seketika jika robot melakukan pembelokan sehingga membentuk kurva, dan ICC merupakan sumbu dari kurva tersebut. [3]

Dimisalkan x,y merupakan koordinat dari robot dan θ merupakan sudut yang terbentuk pada posisi robot setelah rentang waktu δt. Maka titik ICC dapat diperoleh:

$$ICC_{xy} = [x - Rsin(\theta)\ , y + Rcos(\theta)] \quad (2.8)$$

Dengan ω merupakan perubahan rotasi robot terhadap ICC tiap waktu. Maka ketika $vl \neq vr$, ω akan sebanding dengan vl dan vr.

$$\omega\left(R + \frac{L}{2}\right) = vr \tag{2.9}$$

$$\omega\left(R - \frac{L}{2}\right) = vl \tag{2.10}$$

Jika Persamaan (2.9) dan (2.10) digabungkan maka didapat Persamaan (2.11) dan (2.12)

$$R = \frac{L}{2}\frac{vr + vl}{vr - vl} \tag{2.11}$$

$$\omega = \frac{vr - vl}{l} \tag{2.12}$$

Dari persamaan-persamaan diatas dapat ditarik beberapa hal, antara lain: [3]

1. Jika $vl = vr$, maka akan diperoleh gerak lurus linear ke depan. R bernilai tak terhingga dan tidak ada gerak rotasi yang mengakibatkan ω bernilai 0
2. Jika $vl = -vr$, maka R bernilai 0, dan akan diperoleh rotasi pada titik poros tengah robot $(L/2)$ atau berputar di tempat.
3. Jika $vl = 0$, maka roda kiri akan menjadi poros putar sehingga akan diperoleh $R = L/2$ dan gerak rotasi ke kiri begitu juga ketika $vl = 0$ maka akan menimbulkan gerak rotasi ke kanan

2.5 Forward Kinematics [3]

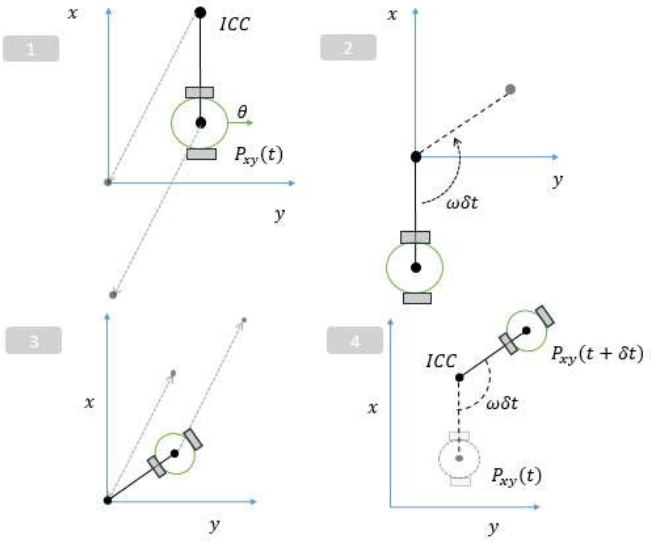
Forward kinematics merupakan suatu metode analisa kinematika yang digunakan untuk mencari posisi (dalam hal ini robot) dengan masukan fungsi berupa sudut. Dengan ini *Forward Kinematics* digunakan untuk mencari *end effector* dalam hal ini berarti posisi robot setelah melakukan perjalanan.

Pada kasus ini kita mencari posisi $P_{xy}(t + \delta t)$ dengan diketahui posisi awal baik koordinat robot $P_{xy}(t)$=(x,y) dan sudut hadap robot(θ). Terlihat pada Gambar 2.6 dimana posisi $P_{xy}(t)$ akan di rotasi dengan titik pusat ICC.

Berikut proses yang dilakukan dalam transformasi koordinat robot.

1. Mentranslasi titik ICC ke titik pusat koordinat [0,0] begitu juga titik $P_{xy}(t)$ yang juga di translasi. $x - ICC_x, y - ICC_y$
2. Merotasi titik $P_{xy}(t)$ yang telah ditranslasi sejauh $\omega\delta t$ dimana ω merupakan kecepatan sudut dan δt merupakan lama waktu robot berpindah. Untuk sudut θ juga akan bergerak sejauh $\omega\delta t$.
3. Lalu mengembalikan posisi ICC dari [0,0] ke posisi semula. Dengan menambahkan titik dengan ICC_{xy}. $x - ICC_x, y - ICC_y$.

Dengan menggunakan aturan transformasi maka didapatkan persamaan untuk mendapatkan $x', y', dan\ \theta'$.



Gambar 2.7 Proses *Forward Kinematics*

2.6 Inverse Kinematics [3]

Inverse kinematics merupakan metode untuk mendapatkan nilai, baik itu x, y atau θ pada tiap waktu. Pada *mobile robot*, kita memberikan masukan kecepatan roda kanan dan kiri maka kita akan dapatkan *state* robot pada saat waktu t dengan menggunakan persamaan kecepatan roda *differential drive* pada Persamaan 2.9 – 2.12.

$$\theta(t) = \frac{1}{L}\int_0^t \big(vr(t) - vl(t)\big)dt \tag{2.13}$$

$$x(t) = \int_0^t \frac{\big(vr(t) + vl(t)\big)}{2}\cos[\theta(t)]\,dt \tag{2.14}$$

$$y(t) = \int_0^t \frac{\big(vr(t) + vl(t)\big)}{2}\sin[\theta(t)]\,dt \tag{2.15}$$

2.7 Path Planning [4]

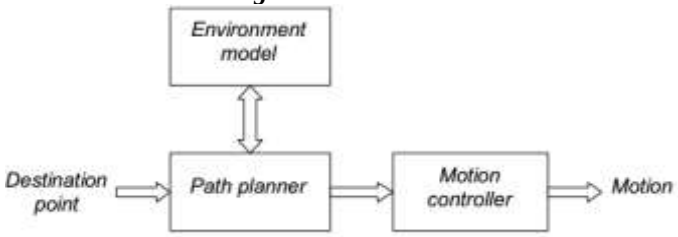
Path planning merupakan proses dimana robot menentukan jalur yang akan dilewati sehingga didapatkan titik yang merupakan titik tujuan. Dengan ini robot dapat berpindah ke titik tujuan dengan jarak terpendek dan tanpa terjadi tabrakan ke halangan yang ada di lingkungan sekitar.

Selain itu *path planning* juga dapat digunakan untuk meminimumkan belokan(*turning*), pengereman(*braking),* dan energi.

Path planning dapat dilakukan dalam dua keadaan, *statis* dan *dinamis*. Keadaan statis yaitu ketika lingkungan sekitar sudah diketahui oleh robot, dan tidak berubah sampai robot mengikuti *path* yang sudah dibentuk. Sedangkan keadaan dinamis, ketika robot mengetahui keadaan lingkungan sekitar namun terdapat halangan yang berubah-ubah/ berpindah secara tiba-tiba. Ataupun ketika robot tidak mengetahui keadaan lingkungan sama sekali namun sensor akan menangkap halangan jika ada pada jangkauannya.

Metode dalam *path planning* bergantung pada keadaan tersebut sehingga tiap keadaan memerlukan perlakuan yang berbeda.

2.7.1 *Global Path Planning*

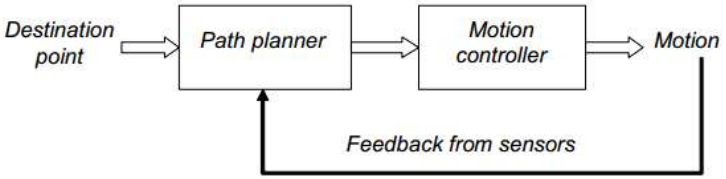


Gambar 2.8 Blok Diagram *Global Path Planning*

Jika telah diketahui pemodelan lingkungan sekitar robot maka *path planning* dapat dilakukan dengan meninjau pemodelan lingkungan secara langsung tanpa meninjau kemungkinan halangan secara dinamis. Oleh dari itu *global path planning* disebut dengan *path planning* secara *offline* karena diperlukan pengetahuan akan lingkungan robot keseluruhan. *Global path planning* dapat dilakukan dengan *obstacle* statis, dan *obstacle* yang tidak dimodelkan sebelumnya tidak dapat ditinjau untuk dihindari.

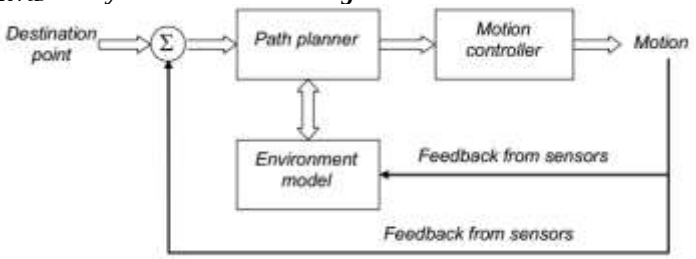
2.7.2 *Local Path Planning*

Local path planning dapat dilakukan tanpa mengetahui lingkungan sekitar robot. Sensor akan merasakan halangan dan tugas dari *path planner* yaitu memberikan titik ke kontroller yang sudah didapatkan sensor. *Local path planning* harus bekerja dengan cepat sehingga tabrakan dapat dihindari, dikarenakan robot memiliki jangkauan kerja bergantung pada sensor robot.



Gambar 2.9 Blok Diagram *Local Path Planning*

2.7.3 *Dynamic Path Planning*



Gambar 2.10 Blok Diagram *Dynamic Path Planning*

Dynamic Path Planning merupakan gabungan dari Local Path Planning dan *Global Path Planning*. Metode ini bekerja dengan memproses lingkungan yang telah ada untuk dilakukan *path planning* secara offline dan sensor akan memberikan kemampuan robot untuk menghindari halangan yang dinamik. Selain itu sensor akan memperbarui model lingkungan yang telah ada. Sehingga dapat diproses oleh *path planner*.

Dynamic path planning cocok dilakukan saat robot mengetahui keadaan global dari lingkungannya namun terjadi perubahan karena adanya *dynamic obstacle*.

2.8 *Fuzzy Model Environment* [5]

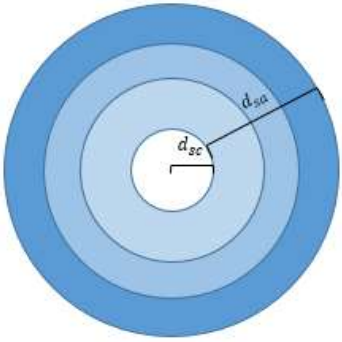
Logika *fuzzy* merupakan logika dimana penyelesaiannya bukan bernilai boolean namun bernilai kontinu dan samar bergantung pada derajat keanggotaan yang diperoleh. Semisal logika jarak yang bukan hanya jauh dan dekat, namun bernilai dekat, cukup dekat, normal, cukup jauh, jauh. Penentuan keanggotaan dan derajat kebenaran bergantung pada kita sebagai pemberi logika *fuzzy*.

Dalam hal ini *fuzzy* digunakan untuk memodelkan lingkungan robot yang berupa halangan (*obstacle*) yang digunakan sebagai parameter *path* yang akan dicari oleh program *Ant Colony*.

Untuk memodelkan *fuzzy* didapat dari parameter *obstacle* yang didapat jaraknya tiap waktu ($O_t^1, O_t^2, O_t^3, \dots, O_t^q$}terhadap posisi robot saat waktu t ($P_R(x_i, y_i)$). Didapatkan jarak $D(P_t, P_t^{oi}) = \sqrt{(x_t - x_t^{oi})^2 + (y_t - y_t^{oi})}$ dimana P_t^{oi} adalah *state* dari *obstacle* ke-i(i=1,2,..,q)

Membership function didapatkan dari hubungan antara jarak, radius *obstacle* jarak minimum aksi yang ditentukan dan jarak maksimum aksi yang kita tentukan.

$$\mu_{p_t}(p_t, p_t^{oi}) = \begin{cases} 1, & if\ D(p_t, p_t^{oi}) \le R^{oi} + d_{sc} \\ 1 - \dfrac{D(p_t, p_t^{oi}) - d_{sc}}{d_{sa}}, & if\ d_{sc} < D(p_t, p_t^{oi}) \le R^{oi} + d_{sa} \\ 0, & lainnya \end{cases} \quad (2.15)$$



Gambar 2.11 *Membership Function* dari Lingkungan Robot [5]

Seperti terlihat pada Gambar 2.11, dsc merupakan area aman dari robot, sehingga robot dikatakan aman/tidak menabrak jika tidak ada *obstacle* yang ada di area d_{sc}. Area kerja robot maksimal yaitu pada radius $d_{sc} + d_{sa}$ yang merupakan radius sensor maksimal.

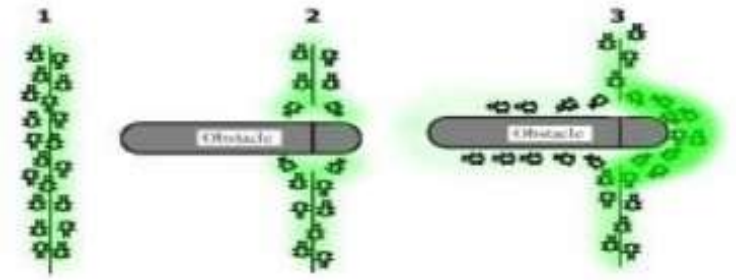
Dengan memodelkan setiap *obstacle* statis dan dinamis yang ada pada sekitar robot kita dapat memberikan keputusan pada robot dengan memberikan nilai *membership* pada lingkungan yang dapat di lalui oleh robot. Dari nilai *membership* ini, akan memberikan pengaruh pada nilai pembebanan(η) yang akan dilakukan pada proses *Ant Colony*.

2.9 Algoritma Koloni Semut (*Ant Colony Algorithm)*

Ant Colony System (ACS) adalah suatu metode penyelesaian masalah optimasi yang berupa kumpulan beberapa algoritma yang menggunakan teknik probabilistik dan perilaku koloni semut dalam mencari makanan. Konsep ACS pertama kali diperkanalkan melalui algoritma *Ant System* (AS) pada tahun 1992 oleh Marco Dorigo dalam disertasinya. [6]

Ant Colony dari perilaku koloni semut yang dikenal sebagai sistem semut. Secara alamiah koloni semut mampu menemukan rute terpendek dalam perjalanan dari sarang menuju ke sumber makanan dan kembali lagi, pada saat semut mencari makan semut akan berjalan secara acak, dan setelah menemukan makanan, semut akan kembali dan semut meninggalkan sebuah informasi yang disebut feromon untuk koloni semut yang lain, di tempat yang dilaluinya dan menandai rute tersebut

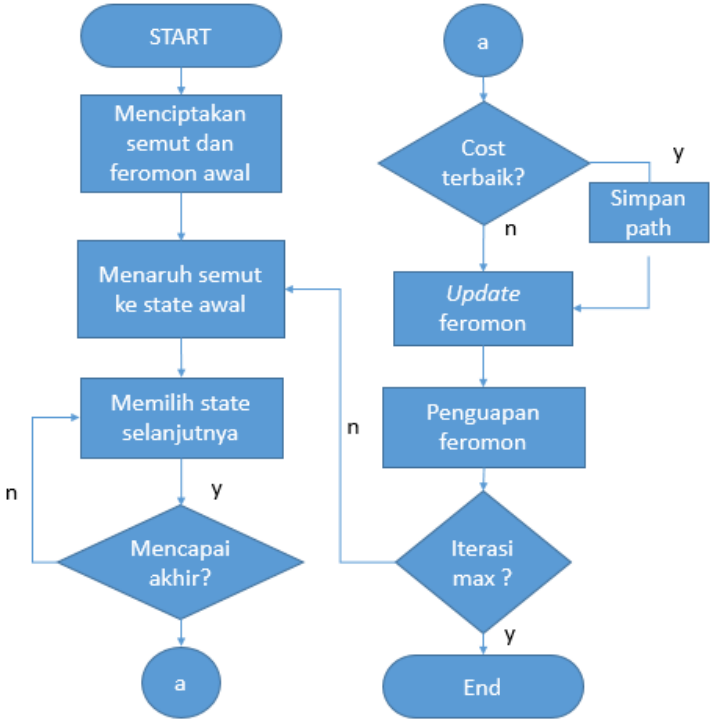
sehingga kemungkinan besar semut lain tidak akan menjelajahi secara acak namun cenderung berjalan melewati rute tersebut. Feromon digunakan sebagai komunikasi antar semut pada saat membangun rute. Rute yang memiliki feromon yang lebih tebal akan lebih cenderung digunakan sebagai semut lainnya. Dalam berjalannya waktu feromon akan menguap sehingga mengurangi daya tarik semut lain untuk melewati jalur tersebut.



Gambar 2.12 Jalur Feromon Semut Akan Menuju Jalur Terpendek [7]

Semisal adanya *obstacle* yang menghalangi jalur robot secara tiba-tiba maka feromon akan di-*update* sehingga membentuk jalur baru dimana jalur yang kaya akan feromon cenderung pendek dan probabilitasnya lebih banyak semut yang melewati. Jalur informasi feromon dapat digunakan untuk pencarian *waypoint* dengan perubahan lingkungan/rintangan. [7]

Untuk melakukan proses *Ant Colony System* ada beberapa tahapan yang harus dilakukan. Seperti yang terlihat pada diagram alur pada Gambar 2.13. Namun diagram secara umum dapat dimodifikasi sesuai kebutuhan. Semisal pada perulangan yang tidak berdasar iterasi, namun menggunakan kriteria lain sebagai penghentian program, seperti toleransi nilai optimal yang telah sesuai atau konvergensi nilai optimal yang telah dicapai.



Gambar 2.13 Diagram Alur *Ant Colony System*

2.9.1 *Pheromone*

Feromon merupakan alat komunikasi yang digunakan oleh semut untuk memberikan informasi dimana letak sumber makanan itu berada. Feromon akan menarik semut lain untuk cenderung melewati jalur yang terdapat feromon. Feromon dilambangkan oleh τ. τ berupa matriks $n \times n$ dimana n merupakan jumlah titik yang dapat menjadi solusi. Dalam kasus ini n merupakan index seluruh koordinat yang ada pada peta. $\tau_{a,b}$ berarti nilai feromon dari titik $a = x_a, y_a$ yang menuju keadaan titik $b = x_b, y_b$. Nilai feromon akan mempengaruhi pemilihan titik pada semut dengan meninjau proporsi feromon (α), pembebanan atau *attractiveness* (η), dan juga proporsi pembebanan(β).

Semut akan berinteraksi pada feromon yang ada pada titik yang *feasible*, maksudnya adalah semut tidak akan memilih titik pada halangan

karena feromon pada titik yang berupa halangan akan bernilai nol. Semut juga berinteraksi pada feromon yang terdapat pada iterasi sebelumnya bukan pada feromon pada tiap semut yang disebar pada iterasi yang sama.

2.9.2 Pemilihan Titik

Pemilihan titik dilakukan oleh setiap semut yang disebar yang semut tersebut akan memilih titik $P_{ij} \in RE$ dimana RE merupakan koordinat *feasible* dari titik P_i yang juga memiliki nilai feromon $\tau_{ij} = 0$, untuk titik yang *unfeasible* maka nilai feromon $\tau_{ij} = 0$.

Untuk pemilihan titik, digunakan *state transition rule* yang dirumuskan sebagai berikut. [5]

$$j = \begin{cases} \underset{J}{\arg\max}\{(\tau_{il}(n)\left[\eta_{il}(P_i)\right]^{\beta} & ,jika\ q \le q_0, \mu_j(t) = 0, l \in |Z| \\ & ,jika\ q > q_0 \end{cases} \quad (2.16)$$

Pemilihan *state* memiliki dua aturan yaitu eksploitasi dan eksplorasi. Dimana aturan tersebut dipilih berdasarkan nilai q yang merupakan nilai acak normalisasi bernilai kurang dari $q_0$yang merupakan parameter yang diberikan bernilai $0 \le q_0 \le 1$. Pada aturan eksploitasi titik yang dipilih merupakan titik dengan feromon tertinggi dari keseluruhan titik dengan ketentuan memiliki nilai $\mu_j(t) = 0$ yang berarti titik tersebut jauh dari halangan di sekitarnya.Keanggotaan semua titik yang dapat dilalui direpresentasikan sebagai Z.

Untuk aturan eksplorasi dilakukan dengan meninjau *feasible point* yang belum terjangkau sebelumnya dari titik P_i. Dengan J memiliki rumus sebagai berikut. [5]

$$P_{ij}^{k}(n) = \frac{\left[\tau_{iJ}(n)\right]^{\alpha}\left[\eta_{iJ}(P_i)\right]^{\beta}}{\sum\left[\tau_{iJ}(n)\right]^{\alpha}\left[\eta_{iJ}(P_i)\right]^{\beta}} \quad (2.17)$$

$$J = |Z|$$

$P_{ij}^{k}(n)$ merupakan probabilitas transisi dari titik P_i ke P_j yang akan dipilih melalui *roulette wheel.* Nilai α merupakan parameter penguat pada bagian feromon, sedangkan nilai β merupakan parameter penguat pada bagian beban. Beban atau *attractiveness* (η) ditentukan oleh

kecenderungan untuk memilih titik tersebut. Dalam kasus ini, *attractiveness* ditentukan oleh *fuzzy environment model* dan juga *polar histogram.*

Probabilitas transisi juga menentukan kecenderungan akan mengikuti feromon atau melakukan eksplorasi.

2.9.3 *Pheromones Evaporation*

Penguapan feromon merupakan proses yang sangat penting dalam pembentukan algoritma, ini dikarenakan proses ini mempengaruhi kinerja dari algoritma. Semisal semut berjalan menuju sumber makanan dengan jalur yang dilaluinya mempunyai jarak yang jauh sehingga memiliki proses waktu yang lama untuk mencapainya, semakin lama waktu semut melakukan perjalanan, maka semakin lama penguapan feromon terjadi. Maka semut lain akan mencari rute lain karena feromon sebelumnya sudah menguap. Akibat dari semut yang mencari rute lain, maka akan menghindari hasil yang konvergen ke solusi optimal lokal. Yang berarti kita masih dapat mencari jalur yang lebih pendek karena keacakan dari semut lain. Penguapan feromon dilambangkan secara matematis ρ dengan nilai rentang $0 < \rho < 1$.

Pada proses *path planning*, robot akan menyebarkan semut ke lingkungan yang telah dimodelkan. Semut tersebut akan berjalan dari sarang semut *(ant nest)* yang merupakan titik awal robot menuju sumber makanan *(food source)* yang merupakan titik tujuan dari robot.

Pada simulasi penguapan feromon dilakukan bersamaan dengan pemutakhiran feromon lokal dan pemutakhiran feromon secara global. Secara sederhana penguapan feromon dilakukan menggunakan persamaan berikut. [8]

$$\tau(n+1) = (1-\rho).\tau(n) \qquad (2.18)$$

2.9.4 *Pheromones Update* [5]

Feromon akan berubah nilai jika titik P_{ij} telah dilewati oleh semut. Akan dilakukan penguapan feromon dan juga penambahan feromon pada titik tersebut. Untuk melakukan pembaruan feromon dapat dilakukan bersamaan antara penambahan feromon dari semut dan penguapan feromon. Namun juga dapat dilakukan bergantian.

Untuk setiap rute yang telah mencapai titik akhir, maka tiap *node* yang ada dalam *path* akan dibarukan. Berikut rumus pembaruan feromon pada titik P_i ke P_j. [5]

$$\tau_{ij}(n+1) = \tau_{ij}(n) + \rho\Delta\tau_{ij} \quad (2.19)$$

Persamaan diatas merupakan pembaruan feromon secara umum tanpa penguapan feromon yang terjadi (penguapan feromon dilakukan sebelum atau sesudah penambahan feromon). Untuk pembaruan lokal dilakukan ketika semut melalui jalur P_{ij} dengan $\Delta\tau_{ij}$ bernilai sama dengan nilai awal τ_0.

Untuk pembaruan global, setelah semua semut sampai ke titik tujuan, semut yang mempunyai *cost* terbaik atau jarak terpendek pada *stage* ini atau terbaik pada seluruh *stage* ini akan dibarui feromonnya dengan $\Delta\tau_{ij}$ sebagai berikut.

$$\Delta\tau_{ij} = c_1 \times \Delta\tau'_{ij} + c_2 \times \Delta\tau''_{ij}$$

$$\Delta\tau'_{ij} = \frac{Q}{l_{best}}$$

$$\Delta\tau''_{ij} = \frac{Q}{g_{best}} \quad (2.19)$$

Dimana l_{best} merupakan *cost* terbaik pada *stage* ini (*local*) dan g_{best} merupakan *cost* terbaik pada seluruh *stage*/iterasi (*global*). Untuk c_1 & c_2 merupakan parameter kontribusi *local optimal* dan *global optimal.*

2.10 Polar Histogram [9]

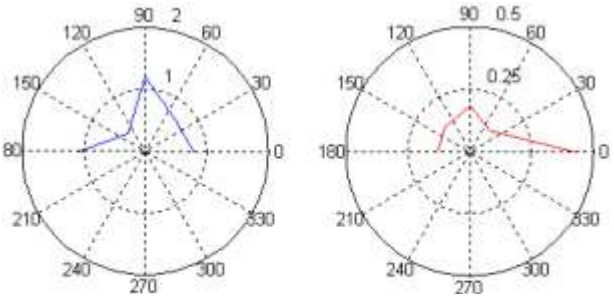
Polar histogram merupakan metode yang dapat digunakan untuk menggambarkan grafik dengan bentuk grafik berupa polar(lingkaran). Karena bentuknya polar, metode ini biasa digunakan sebagai pemodelan ataupun sistem penghindaran tabrakan *(collision avoidance).* Dimana besaran dari setiap derajatnya merupakan jarak *obstacle* terhadap titik robot.

Dengan mengubah jarak menjadi *density*/kepadatan halangan kita dapat mengkontrol robot untuk menghindari halangan. Jika kepadatan *obstacle* sedikit maka kita mengarahkan robot ke arah yang sedikit halangannya.

Untuk membuat *polar histogram* kita memerlukan data sensor dari segala arah, untuk simulasi yang ideal kita hanya memodelkan kepadatan tiap derajat dengan tiap derajat sensor dibuat independen dan tidak mempengaruhi sensor kanan kirinya. Untuk robot yang hanya memiliki

sensor yang sedikit dan di bagian tertentu saja (didepan), nilai *density* tiap sudut akan diperhitungkan dengan meninjau besaran di kanan dan kiri.

Contohnya pada robot Qbot yang memiliki 5 buah sensor infrared di sudut 0, 45, 90, 135, dan 180 derajat akan di hitung POD *(polar obstacle density)* yang menggunakan data jarak yang diperoleh dari sensor inframerah.



(a) Data Jarak Sensor inframerah (b) Normalisasi *Polar Obstacle Density*

Gambar 2.14 Plot Histogram dari Data Jarak dan Kepadatan *Obstacle*

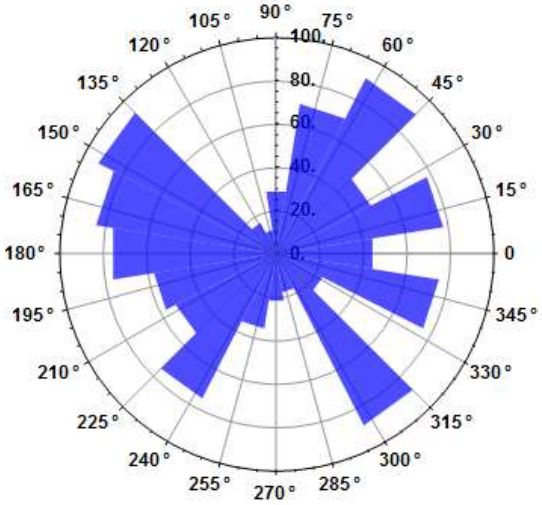
Gambar 2.14(a) merupakan data jarak yang didapat pada sensor inframerah pada Qbot. Dan gambar 2.13(b) merupakan *polar obstacle density* yang dapat dihitung dengan rumus:

$$POD_i = 0{,}1667 \times f_{norm}^2(d_{i-1}) + 0{,}6667 \times f_{norm}^2(d_i) + 1667 \times f_{norm}^2(d_{i+1})$$

$$f_{norm}^2 = 1 - \frac{\min(d_{th}, d)}{d_{th}} \quad (2.20)$$

Dimana $i = 1, \ldots, 5$ menunjukkan sensor ke-i yang diarahkan ke $45 \times (i-1)$ derajat. Jarak infra merah maksimum didefinisikan oleh d_{th}. Untuk d_0 dan $d_6$diberi nilai 0, dan $d_{th} = 1{,}5$ m. Setelah didapatkan normalisasi POD robot akan bergerak menuju ke *density* terkecil. Jika arah depan robot, kepadatannya besar maka robot akan berhenti dan berputar ke sampai kepadatan halangan di arah yang teraman yaitu 45^o(sisi robot).

Pada simulasi *polar histogram* dapat dilakukan untuk memberikan weight berdasar *density* untuk diberikan ke pembebanan pada algoritma *Ant Colony*. Karena *polar density* digunakan pada simulasi tidak dibatasi oleh kemampuan sensor maka kita dapat memberikan polar density yang ideal ke seluruh arah 0-360 derajat.



Gambar 2.15 *Polar Histogram* yang Digunakan Pada Simulasi

2.11 *Pure Pursuit Controller*

Merupakan kontroler geometrik untuk mengikuti kumpulan *waypoint*. Dengan memberikan set dari *waypoint* kontroller akan menghitung kecepatan linear dan angular untuk diberikan kepada differential drive robot untuk didapatkan kinematikanya.

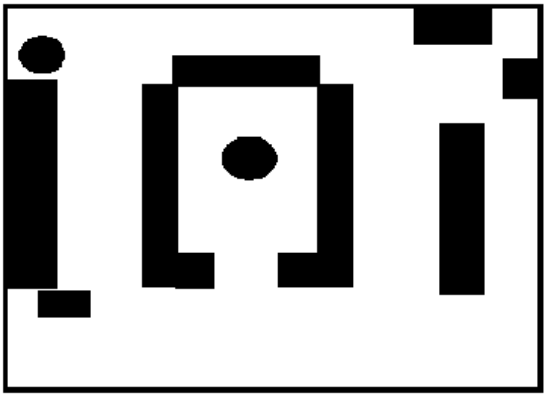
Dengan memberikan masukan spesifikasi kontroler berupa *waypoint*, kecepatan maksimal angular dan linear yang diinginkan dan jarak *look ahead* (parameter jarak robot ke *path* yang dituju) dan keadaan robot sekarang (koordinat x, y, dan θ) akan didapatkan kecepatan angular dan linear yang diperlukan untuk simulasi *differential drive robot.*

BAB 3
PERANCANGAN SISTEM

Pada bab ini akan dibahas tahapan dalam merancang alur algoritma *path planning* sebuah *mobile robot* berbasis *Ant Colony system* yang meliputi identifikasi masalah yang ada, perancangan algoritma *path planning*, perancangan algoritma *Ant Colony* dan perancangan simulasi.

3.1 Identifikasi Permasalahan

Pada tugas akhir ini, permasalahan yang akan dibahas adalah permasalahan *path planning* yang bertujuan menemukan sebuah *path* terpendek antara dua titik yang ditentukan. Dengan menggunakan grafik peta yang merepresentasikan keadaan lingkungan robot secara nyata dengan perbandingan antara simulasi dengan nyata yaitu 1 *pixel* = 2cm, simulasi akan bekerja mencari *path* yang menghubungkan antara dua titik tanpa tertabrak halangan. Peta yang dibuat berformat .png tanpa menentukan ketinggian dari halangan.



Gambar 3.1 Peta Lingkungan Pada Simulasi

Setelah diproses dengan *software* MATLAB, gambar akan diberi koordinat, titik hitam akan bernilai 1, dan area putih merupakan yang dapat dilalui oleh robot.

Peta yang dibuat merupakan gambar berformat *png* dengan resolusi 275×379 piksel. Setelah diproses, dibentuk *dataset* berupa *array*

berupa *logical* dengan menyatakan koordinat yang memiliki halangan atau tidak.

Pemodelan tersebut kemudian dilakukan pemetaan ulang dimana peta yang ada, dipisahkan antara halangan dan lantai.

Setelah didapatkan hasil yang sesuai, dalam hal ini yaitu mencari titik *waypoint* yang memiliki jarak yang terkecil, hasil tersebut digunakan untuk menjalankan simulasi robot *path following* yang terdapat pada MATLAB.

Simulasi robot tersebut merupakan simulasi *mobile robot* berjenis *Differential Drive Robot* yang sudah ada dalam bawaan MATLAB R2015b keatas. Simulasi ini hanya meninjau kinematika robot saja dengan *controller pure pursuit.*

Robot yang digunakan dalam simulasi merupakan penggambaran *differential drive robot* dengan spesifikasi dimensi sebagai berikut.

Tabel 3-1 Spesifikasi Fisik Simulasi Robot

Deskripsi	**Nilai**	**Satuan**
Diameter *mobile robot*	40	*cm*
Kecepatan maksimum dari *mobile robot*	0,5	*m/s*
Diameter roda	10	*cm*
Kecepatan anguler maksimum dari *mobile robot*	10	*rad/s*

Spesifikasi diatas akan digunakan untuk memberikan parameter simulasi *differential drive robot.*

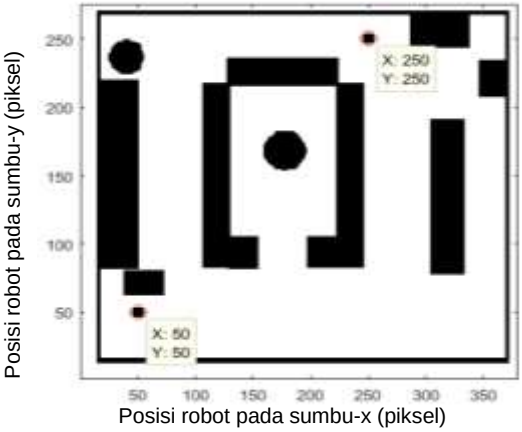
3.2 Perancangan Algoritma *Path Planning*

Diawali dengan mendeskripsikan lingkungan kerja seperti peta berupa halangan dan jalan yang dapat dilewati robot, dan memberikan titik awal serta tujuan. Lalu memodelkannya dan memasukkan sebagai sistem struktur yang dapat dipanggil tiap saat. Setelah itu memasukkan nilai parameter dan model yang diperlukan *Ant Colony.* Dan didapatkan *path* dengan jarak yang optimal. Proses *path planning* digambarkan pada diagram alir berikut

Gambar 3.2 Diagram Alir *Path Planning*

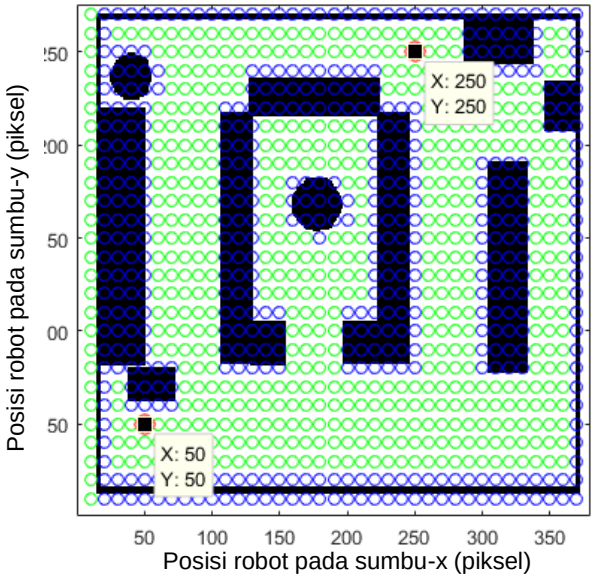
3.2.1 Deskripsi Lingkungan Kerja

Setelah memuat gambar sebagai peta, halangan berupa titik hitam tiap koordinat akan digunakan sebagai model *environment*. Titik koordinat awal dan tujuan robot dapat kita tentukan. Berikut contoh pada titik awal yang ditentukan [50 50] dan titik akhir [250 250].



Gambar 3.3 Pemberian *Starting Point* dan *End Point*

Memodelkan peta adalah langkah berikutnya dimana peta yang berupa kumpulan logika berukuran 275x379 akan diubah dengan ukuran yang sama namun peta tersebut disegmenkan menjadi koordinat yang membatasi titik kerja yang mungkin. Sehingga hasil *waypoint* akan bernilai kelipatan 10. Ini dilakukan untuk mengurangi titik kerja pada *Ant Colony* sehingga mempercepat *running time* pada program. Selain itu proses *grid* atau pengkisian ini akan memberikan hasil yang titiknya tidak berada pada halangan, dan mencegah hasil *waypoint* yang membuat robot terjadi tabrakan. Hasil dari fungsi *gridmap* berupa pengelompokan titik yang dapat dilalui (*possible point*) dan halangan yang ada (*obstacle*). Gambar berikut menunjukkan koordinat yang memungkinkan untuk dilalui robot.



Gambar 3.4 Pengkisian Peta untuk Titik-titik yang Dapat Dilalui dan Halangan

Selain titik yang dapat dilalui yang dilakukan pengkisian, tetapi halangan yang ada juga dikisikan. Lalu keduanya akan dimasukkan

kedalam variabel struktur yang akan diproses lebih lanjut. Berikut variabel struktur dan definisinya

Tabel 3-2 *Variable Structure* Pemodelan Peta

Nama variabel	**Definisi**
Model.pos	Titik koordinat yang dapat dilalui Robot
Model.obs	Titik pengkisian berupa halangan
Model.total	Gabungan dari model.pos dan model.obs, berisi seluruh koordinat kerja yang berlaku.
Model.sp	Titik koordinat *starting point*
Model.ep	Titik koordinat *end point*

Variabel model.total digunakan sebagai indeks koordinat. Ini bertujuan untuk mempermudah dalam proses *Ant Colony*, dikarenakan proses ini menggunakan *input* dengan satu indeks saja tidak bisa menggunakan koordinat yang memiliki dua indeks yaitu x dan y.

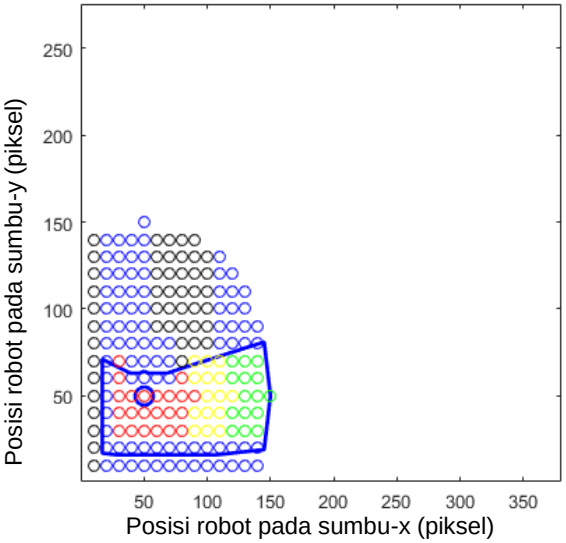
Semua model tersebut kecuali model.total akan diubah menjadi indeks. Sedangkan untuk memanggil kembali koordinat maka kita dapat memanggil variabel model.total(indeks).

3.2.2 Pemodelan Lingkungan Kerja

Pemodelan lingkungan kerja pada algoritma *path planning* berupa nilai *reachability* tiap titik yang ada pada radius tertentu terhadap titik yang sekarang dicapai dengan halangan, apakah titik *reach* tersebut memiliki terhalang oleh *obstacle* jika ditarik garis lurus ke titik awal. Reachability akan menentukan besar kemungkinan titik itu terpilih untuk jalur berikutnya atau tidak. Karena jika *reach* tidak memenuhi, maka peluang tersebut akan menjadi nol.

Pemodelan ini digunakan untuk memberikan nilai pada η yang merupakan pembebanan/*attractiveness* dari probabilitas pemilihan titik pada Ant Colony .

Pemodelan dilakukan diawal setiap *waypoint* yang dihasilkan semut. Proses ini merupakan proses yang juga melibatkan melibatkan dunia luar karena disetiap titik, kita mengetahui adanya halangan melalui sensor. Dan titik yang dipilih juga bergantung pada sensor yang melihat keadaan *real*. Oleh karena itu pemodelan lingkungan *fuzzy* dapat digunakan sebagai acuan *path planning* secara global ataupun secara lokal.



Gambar 3.5 Pemilihan Titik yang *Feasible*

Pada Gambar 3.5 merupakan proses untuk mencari titik yang *feasible* yang berdasarkan sensor jarak yang ada. Titik yang akan dipilih pada *Ant Colony* harus berada pada area berwarna hijau, kuning, atau merah. Dimana warna tersebut merupakan pembebanan yang akan diberikan sebagai probabilitas pemilihan *state* pada *Ant Colony* .

Area hitam pada gambar meruupakan area yang dapat dilalui oleh robot namun tidak dapat dilewati karena terhalang oleh *obstacle* berwarna biru.

3.3 Perancangan Algoritma Koloni Semut

Algoritma koloni semut merupakan teknik pencarian yang meniru fenomena semut yang mencari makan. Langkah awal yang dilakukan untuk merancang algoritma *path planning* berbasis koloni semut ini diawali dengan mencari titik yang memungkinkan dilalui oleh robot. Titik akan diubah menjadi index supaya mudah dalam menentukan *cost* maupun saat melakukan pencarian acak menggunakan *Roulette Wheel*. Setelah itu dilakukan pengambilan titik yang memungkinkan menggunakan *Roulette wheel selection*. Sehingga semut-semut akan berisi *titik path*. Setelah itu dilakukan evaluasi *cost* dimana semut yang

memiliki nilai *path* yang baik baik itu jarak maupun sudut belok yang dibuat, maka variabel dari semut tersebut akan disimpan. Tahap selanjutnya yaitu melakukan penguapan feromon antar titik yang dilalui semut, feromon tersebut akan menguap jika dilewati oleh semut. Penguapan terjadi agar titik tersebut memiliki kemungkinan yang lebih sedikit sehingga tidak dapat dipilih kembali saat iterasi berikutnya agar tidak terjadi *dead end*(keadaan dimana *waypoint* yang didapat mencapai keadaan yang konvergen di optimal lokal). Langkah tersebut dilakukan sampai iterasi berhenti. Sehingga didapatkan nilai optimal *path planning*.

3.3.1 Inisialisasi Parameter

Inisialisasi parameter merupakan proses dimana kita menentukan parameter-parameter yang bekerja pada *Ant Colony* dimana parameter tersebut berperan penting dalam pemilihan *state* yang mempengaruhi hasil *path planning* yang dibahas pada Bab 4.

Parameter yang diubah-ubah yaitu jumlah iterasi, jumlah semut, $\alpha, \beta, \rho, dan\ \tau_0$. Jumlah iterasi akan mempengaruhi lama dari proses *path planning* berlangsung, namun juga mempengaruhi hasil yang akan diperoleh karena memiliki banyak kesempatan untuk algoritma menemukan *shortest path*.

Jumlah semut berpengaruh pada waktu *running time* juga sama seperti jumlah iterasi. Berbeda dengan jumlah iterasi, semut tidak akan mempercepat konvergensi hasil karena jumlah semut tidak mempengaruhi jumlah terjadinya *update* feromon, sehingga tidak terjadi penguatan ataupun penguapan feromon yang mengakibatkan hasil konvergen lebih cepat.

Alpha merupakan parameter penguat feromon. Namun juga dapat melemahkan feromon tersebut untuk menarik semut. Jika alpha bernilai lebih dari satu, maka jika feromon yang ada bernilai kurang dari 1, α akan melemahkan fungsi feromon untuk dipilih dan cenderung ke feromon yang bernilai lebih dari satu. Karena akan menguatkan feromon yang lebih besar. Jadi alpha akan membuat feromon kuat jadi semakin kuat dan yang lemah menjadi sangat lemah.

Beta merupakan parameter penguat untuk beban. Jadi beban yang kita berikan melalui *polar histogram* akan memperkuat pengaruhnya. Karena simulasi yang dibuat memiliki dua tipe pembebanan, yaitu *fuzzy membership* dan histogram, maka β yang digunakan juga ada dua. Yaitu untuk *fuzzy*, dan histogram.

Rho merupakan parameter penguapan feromon. Semakin besar maka nilai feromon akan cepat turun dan mengakibatkan konvergensi lokal yang cepat.

Nilai $\tau_o$merupakan nilai feromon awal pada *Ant Colony*. Nilai τ_0 akan mempengaruhi penguapan sekaligus penambahan feromon. Ini dikarenakan pada pembaruan feromon lokal, nilai τ_o akan ditambahkan dengan proporsi ρ.

3.3.2 Karakteristik Semut

Semut yang disebarkan pada tiap iterasi membawa informasi mengenai hasil dari *path planning*. Pada simulasi ini semut membawa 3 informasi, yaitu *cost, point,* dan indeks. *Point* merupakan hasil *path* secara eksplisit. Dimana setiap semut yang disebar mencatat titik mana saja dia berjalan mencari makan. Kadang semut yang tidak mencapai sumber makanan/titik akhir dalam beberapa *waypoint* akan dihentikan supaya waktu simulasi jadi lebih cepat. Semut yang tidak mencapai sumber makanan pasti performanya lebih buruk maka lebih baik dihentikan.

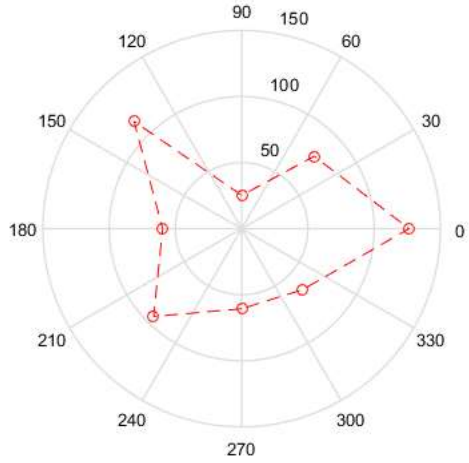
Cost merupakan kualitas dari jarak yang dilalui semut. Pada simulasi ini, nilai *cost* diberikan bernilai jarak yang dilalui semut ketika sampai pada sumber makanan dengan jumlah titik singgah kurang dari 10. Lebih dari itu maka semut akan dihentikan dan nilai *cost* diberikan sangat besar, sehingga jalur yang dilalui semut tersebut menjadi kecil.

Indeks merupakan konversi dari *path* yang dibawa semut ke indeks berdasarkan model.total. Ini dilakukan untuk mempermudah dalam proses pembaruan feromon. Karena set feromon yang berupa indeks, bukan koordinat.

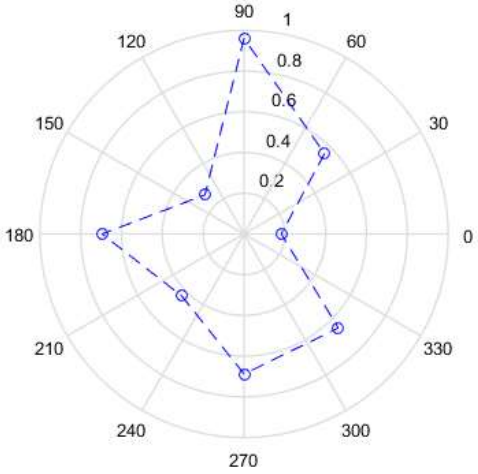
3.3.3 Pembebanan Feromon.

Untuk aturan pembebanan feromon yaitu menggunakan hasil dari *fuzzy environment model* dan juga *polar histogram.* Aturan yang dibuat disini yaitu untuk *fuzzy environment model.* Dengan memodelkan jarak

rata-rata tiap derajat dengan fungsi *membership fuzzy* terhadap jarak akan didapatkan nilai pembebanan sebesar [0 1] (pada persamaan 2.15).



Gambar 3.6 *Polar Histogram* dari jarak Halangan Sekitar Robot



Gambar 3.7 Nilai Pembebanan yang Didapatkan dari *Membership Fuzzy*

3.3.4 Pemilihan *State*

Node yang akan dipilih merupakan titik-titik yang berada disekitar robot. Dengan syarat pembebanan dan konsentrasi feromon titik di robot ke *state* yang akan dipilih, maka akan didapatkan probabilitas tiap *node* yang akan dipilih. Nilai probabilitas diberikan pada tiap *node* merupakan probabilitas kumulatif yang telah dinormalisasi.

Cara memilih *state* dengan probabilitas kumulatif yaitu dengan melakukan pemilihan dengan *roulette wheel*. Tiap titik yang akan dipilih merupakan titik yang tak terhalang oleh *obstacle*. Himpunan titik didapatkan dari penggunaan *polar histogram*. Dimana titik yang berada pada jangkauan sensor yang membentuk histogram akan dipilih. Sedangkan yang terhalang *obstacle* akan secara otomatis tidak dipilih karena tidak berada pada himpunan titik *feasible*.

BAB 4
HASIL DAN ANALISA

Pada bab ini akan dibahas hasil dan analisa yang diperoleh setelah menjalankan simulasi *path planning* dengan perubahan parameter. Dan dianalisa tentang hasil *shortest path*. Setiap hasil yang ada memiliki gambaran jalur yang telah dipilih pada 20 *state* terakhir.

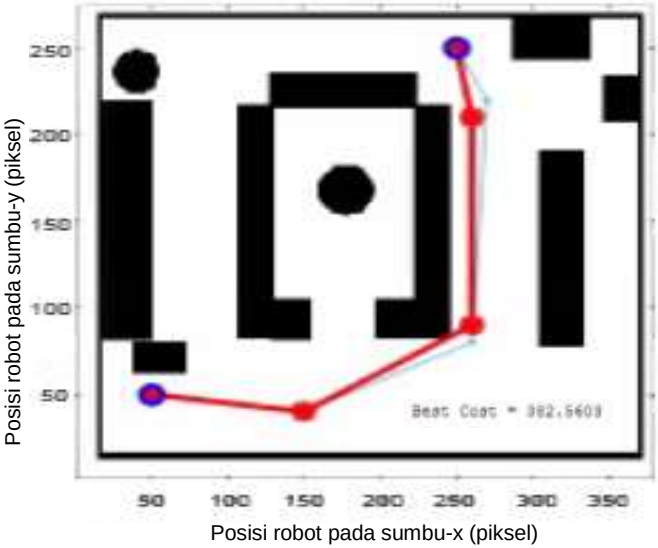
Setiap simulasi yang dilakukan dengan iterasi sebanyak 100 kali dengan 10 semut dan ρ=0,05

4.1 Hasil Simulasi dengan Parameter Alpha

Dengan mengubah nilai parameter alpha kita lihat bagaimana berkas-berkas feromon yang tersisa terhadap hasil yang ada.

4.1.1 Simulasi dengan Parameter α=2 dan $\tau_0 = 2$

Berikut yang dihasilkan dengan mengganti parameter alpha. Untuk parameter β=1, dan ρ=0,05



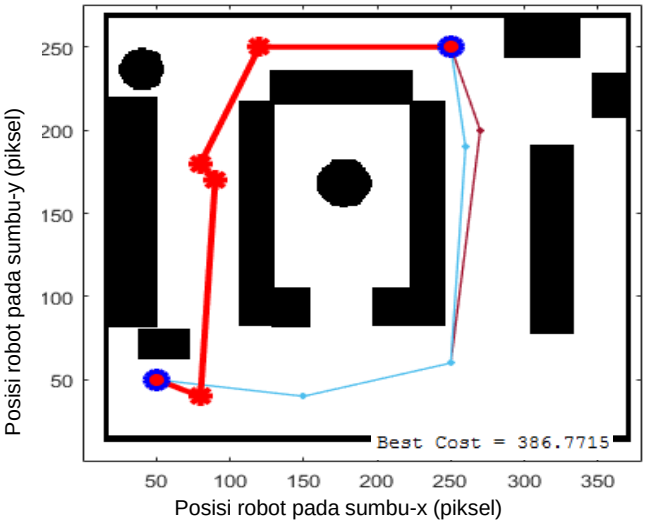
Gambar 4.1 Hasil dengan α=2 dan τ_0= 2

Dari Gambar 4.1 terlihat jalur terbaik berwarna merah memiliki jarak sebesar 392,5603 satuan piksel. Namun terlihat bahwa ada satu jalur feromon yang ada. Ini membuktikan bahwa jalur feromon yang memiliki konsentrasi feromon terbanyak bukan berarti *path* terpendek.

Selain itu terjadinya konvergensi ke optimum lokal terlihat pada iterasi ke 68 dimana hanya menyisakan satu jalur feromon terkuat.

4.1.2 Simulasi dengan Parameter α=2 dan $\tau_0 = 1$

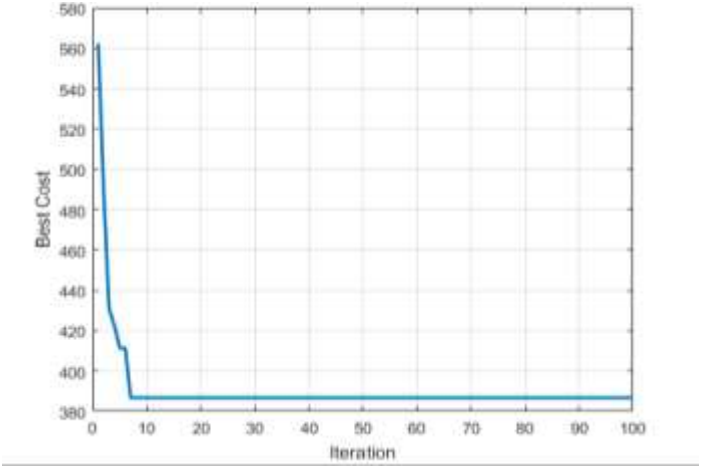
Berikut yang dihasilkan dengan mengganti parameter alpha. Untuk parameter β=1, dan ρ=0.05



Gambar 4.2 Hasil dengan α=2 dan τ_0= 1

Pada iterasi yang sama dengan τ_0 yang berbeda didapatkan hasil seperti pada Gambar 4.2 dimana ada dua berkas feromon yang masih tersisa 2 berkas. Dan bertahan sampai iterasi ke 76. Ini menandakan bahwa nilai τ_0 juga berpengaruh pada konvergensi hasil.

Begitu juga pada Gambar 4.3 yang merupakan nilai jarak terpendek yang tersimpan. Tidak terjadi penurunan cost lagi semenjak terakhir pada iterasi ke 7. Maka sehingga dapat disimpulkan bahwa nilai alpha mempengaruhi tingkat eksplorasi yang dilakukan oleh pembebanan.



Gambar 4.3 Nilai *Cost* Terbaik Tiap Iterasi untuk $\rho = 1$ dan $\tau_0 = 1$

Seperti yang dibahas pada teori sebelumnya parameter alpha menentukan penguatan terhadap feromon yang ada pada saat pemilihan *state*. Dengan parameter alpha yang tinggi maka akan mempercepat konvergensi ke *local optimum*, karena lebih memberikan kesempatan ke jalur feromon lama daripada eksplorasi menggunakan *attractiveness.*

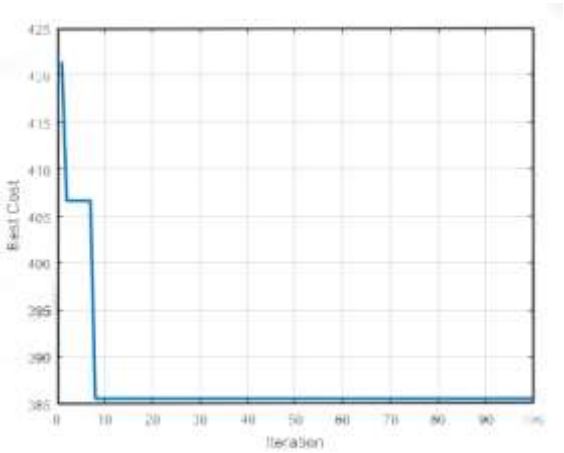
4.2 Hasil dan Analisa untuk Peningkatan pada Parameter Beta

Parameter beta merupakan parameter penguat untuk *attractiveness* dimana didapat dari pemodelan *fuzzy* dan pembebanan *polar histogram.* Untuk beban *membership fuzzy* memiliki rentang [0 1] dan diberikan penguatan β_1. Sedangkan untuk aturan *polar histogram* diberikan weight sebagai berikut, dengan penguatan β_2.

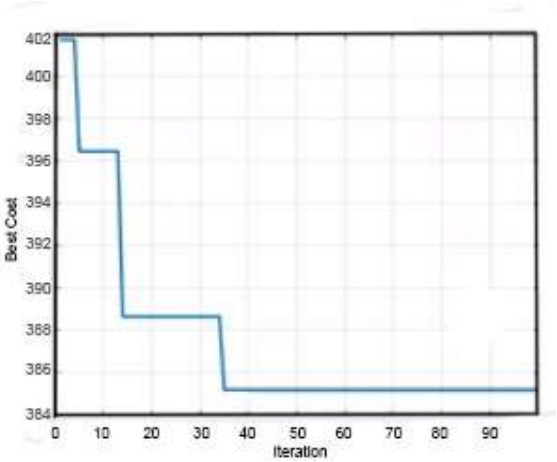
Tabel 4-1 *Weight* Berdasarkan Gambar 3.5

Warna	**Jarak**	***Weight***
Merah	10-50	1
Kuning	50-100	2
Hijau	100-150	3

Berikut hasil *cost* dari jarak terpendek tiap iterasi dengan parameter $\beta_1 = 2, \beta_2 = 1$ akan dibandingkan dengan pengaturan β_1=1 β_2=2.



Gambar 4.4 Nilai *Cost* Terbaik Tiap Iterasi untuk $\beta_1 = 2, \beta_2 = 1$ dan τ_0=1



Gambar 4.5 Nilai *Cost* Terbaik Tiap Iterasi untuk β_1 = 1, β_2 = 2 dan τ_0=1

Dapat dilihat pada Gambar 4.4 dan Gambar 4.5 bahwa tidak ada variasi perubahan nilai *cost* untuk Gambar 4.4 setelah iterasi ke 10, sedangkan untuk Gambar 4.5 terjadi penurunan sampai pada iterasi ke 30. Maka dapat disimpulkan bahwa nilai parameter β_1 akan menurunkan tingkat eksplorasi dengan *weight* jika $\tau_0 < 1$, dikarenakan simulasi akan cenderung menggunakan feromon sebagai acuan (eksploitasi) daripada melakukan eksplorasi menggunakan *weight*.

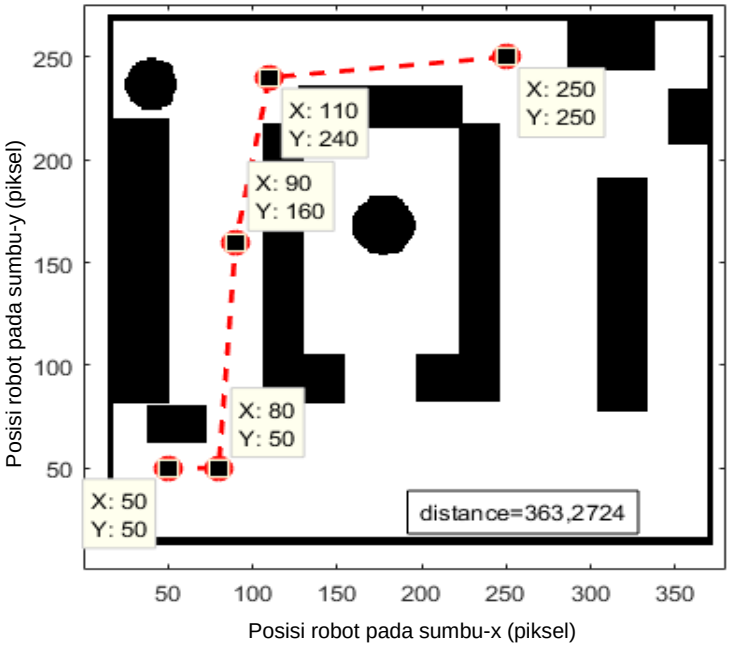
4.3 Hasil Terbaik Untuk Beberapa Eksperimen

Setelah mengubah parameter tiap kali menjalankan program, nilai dari *path* disimpan, termasuk nilai *cost* yang terbaik lalu ditentukan parameter mana yang didapati *shortest path*.

Tabel 4-2 Nilai *Cost* Untuk Tiap Eksperimen

No	α	β_1	β_2	τ_0	*Best Distance (pixel unit)*
1.	1	1	1	1	363,2724
2.	1	1	1	2	390,9210
3.	1	1	2	2	387,9379
4.	1	1	2	1	385,2259
5.	1	2	1	1	385,5593
6.	1	2	1	2	385,2259
7.	2	1	1	2	392,5603
8.	2	1	1	1	384,4717

Untuk rute terpendek didapatkan dengan parameter $\alpha = 1, \beta_1 = 1, \beta_2 = 1, \tau_0 = 1$ sebesar 363,2724 piksel dengan rute $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 50\ 80\ 90\ 110\ 250 \\ 50\ 50\ 160\ 240\ 250 \end{bmatrix}$.



Gambar 4.6 Hasil Simulasi Dengan Jarak Terpendek

Dengan penggunaan kisi yang sama, dengan melakukan perencanaan jalur secara manual, didapatkan titik-titik koordinat *waypoint:* $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 50\ 80\ 100\ 250 \\ 50\ 60\ 240\ 250 \end{bmatrix}$. Memiliki nilai jarak sebesar 363,0634 satuan piksel. Selisih dari keduanya sebesar 0,209 satuan piksel atau 0.42 cm dari perhitungan manual.

BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Algoritma ini mampu mendapatkan *path* yang memungkinkan untuk dilalui robot namun memiliki kelemahan tentang konvergensi hasil simulasi yang menyebabkan tidak mencapai titik optimal

Simulasi yang dilakukan dengan 8 kali percobaan dapat memperoleh hasil *path planning* dengan jarak terpendek sebesar 363,2724 dengan rute: (x;y)=[50 80 90 110 250;50 50 160 240 250]

Parameter *Ant Colony* sangat penting dan harus diperhatikan supaya memberikan hasil yang optimal, memberikan nilai alpha besar akan membuat hasil cepat konvergen namun tidak optimal.

5.2 Saran

Setelah melakukan penelitian ini, untuk pengembangan penelitian, disarankan beberapa hal, antara lain:

1. Mengembangkan *cost function* yang lebih baik dengan meninjau banyak titik dan energi yang dibutuhkan untuk melakukan belok.
2. Perlu meninjau waktu kompilasi program, karena program ini memerlukan *path planning* yang cepat agar dapat dicapai robot sebelum menabrak halangan.
3. Perbandingan dan penggabungan dengan metode lain sehingga bisa diperoleh solusi yang lebih efisien baik dari segi waktu, maupun komputasi.

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

[1] Gopalakrishnan. B, Tirunellayi. S dan Todkar. R, "Design and Development of an Autonomous Mobile Smart Vehicle: a Mechatronics Application," *Mechatronics 14* , p. 491–514, 2014.

[2] ..., "*Robot Locomotion, Holonomic and Non-Holonomic Drive*," RobotPlatform.com, 2010-2017. [Online]. Available: http://www.robotplatform.com/knowledge/Classification_of_Robots/Holonomic_and_Non-Holonomic_drive.html. [Diakses 04 Juni 2017].

[3] G. Dudek dan M. Jenkin, *"Computational Principles of Mobile Robotics,"* New York: Cambridge University Press, 2010.

[4] T. Lehtla, "*Introduction to Robotics,*" Tallinn, TTU, Dept. of Electrical Drives and Power Electronics, 2008, p. 84.

[5] B. Zeng, Y. Yang dan Y. Xu, "Mobile Robot Navigation in Unknown Dynamic Environment," dalam *Global Congress on Intelligent Systems*, Guang Zhou, 2009.

[6] M. Dorigo dan L. M. Gambardella, "Ant Colony System: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary,* vol. 1, no. 1, pp. 53-66, 1997.

[7] A. Reshamwala dan D. P. Vinchurkar, "Robot Path Planning using An Ant Colony," *(IJARAI) International Journal of Advanced Research in Artificial Intelligence,* vol. 2, p. 65, 2013.

[8] N. Habib, D. Purwanto dan A. Soeprijanto, "Mobile Robot Motion Planning by Point to Point Based on Modified Ant Colony Optimization and Voronoi Diagram," *International Seminar on Intelligent Technology and Its Application,* pp. 613-618, 2016.

[9] ..., "*User Manual Quanser Qbot,"* Ontario: Inovative Educate, 2012.

LAMPIRAN A. PROGRAM INTI *PATH PLANNING ANT COLONY*

```
%% Algoritma koloni semut untuk path planning
clc;
clear;
close all;
tic
%% Problem Definition
CostFunction=@(ant)evalCost(ant);% untuk
memanggil fungsi evaluasi cost
% posrobo=@()Robotsim2();
load('world.mat');
global world
h=ones(30,30);
h=plot(h)';
imagesc(~world); hold on
set(gca,'YDir','normal');
colormap(gray);
axis square;
hold off
plottingan=1;%gambar semut
[pos,obs]=gridmap2(world,0);
model.pos=pos;
model.obs=obs;
model.total=union(pos,obs,'rows');
model.sp=[50;50]; %Start point robot
model.ep=[250;250]; %end point robot
[~,model.isp]=ismember(model.sp',model.total,'ro
ws');
[~,model.esp]=ismember(model.ep',model.total,'ro
ws');
model.ind=ismember(model.total,model.pos,'rows')
;
model.iobs=find(model.ind==0);
model.ipos=find(model.ind==1);

% model berupa titik koordinat halangan maupun
titik yang memungkinkan pada
% robot. didapatkan dari hasil simulasi robot
```

```
%% Parameter untuk Algoritma
MaxIt=100;      % Maximum Iterations
nAnt=10;        % Number of Ants
Q=100;          % nilai penguatan update feromon
alpha=1;        % Phromone Exponential Weight
beta1=2;
beta2=1;        % Heuristic Exponential Weight
ro=0.05;       % Evaporation Rate


%% Initialization
%Pheromones
tau0=1;   % Initialization Pheromone
tau=tau0*ones(length(model.total));
tau(model.iobs,:)=0;
tau(:,model.iobs)=0;
BestCost=[];        % Holds Cost of Best
Solutions of Iteration
% Empty Ant
% Terdapat 3 point pada setiap individu semut:
% 1. point berupa koordinat yang merupakan hasil
% 2. indeks koordinat path
% 3. cost yang didapat pada satu semut
empty_ant.Point.x=[];
empty_ant.Point.y=[];
empty_ant.ind=[];
empty_ant.Cost=0;

% Ant Colony Matriks
ant=repmat(empty_ant,nAnt,1); %Membuat dimensi
matriks baru koloni semut sebanyak nAnt
% Best Ant
BestSol.Cost=inf; %nilai awal cost=tak hingga
now=[model.sp(1) model.sp(2)];
%% ACO Main Loop
for it=1:MaxIt
    % Move Ants
    for k=1:nAnt
        ant(k).Point.x=[];
```

```
        ant(k).Point.y=[];
        ant(k).ind=[];
        l=1;
        finish=1;
        %Tour starts at random
        while finish

            %% Setting next reachbility(!)


[laserx,lasery,pos1,pos2,pos3,finish,pf1,pf2,pf3
,pf4,pf5,pf6,pf7,pf8,pnow]=reach3(l,ant(k).Point
,model,0);
             [theta,rho]=cart2pol(laserx-
pnow(2),lasery-pnow(1));

[~,Pnow]=ismember(pnow,model.total,'rows');%meng
indekskan titik sekarang
        if finish==0
            break;
        end

[eta1,eta2]=weight(theta,rho,pos1,pos2,pos3,pf1,
pf2,pf3,pf4,pf5,pf6,pf7,pf8,length(model.total),
0);
                  %Set the roulette

P=(tau(Pnow,:).^alpha).*((eta1.^beta1).*(eta2.^b
eta2));%Evaluating probability
                  %of the next step
        P=P/sum(P);% Normized

index=RouletteWheelSelection2(P,model.total);%
Do the roulette
%
plot(model.pos(index,2),model.pos(index,1),'bo',
'linewidth',4)
             ant(k).Point.x=[ant(k).Point.x
model.total(index,2)];
```

```
            ant(k).Point.y=[ant(k).Point.y
model.total(index,1)];%Assign the point
            ant(k).ind=[ant(k).ind index];
        l=length(ant(k).Point.x)+1;
        if l>10
            break
        end

        end

        if l<=10
        poin=[model.sp(1) ant(k).Point.x
model.ep(1);model.sp(2) ant(k).Point.y
model.ep(2)];
        ant(k).Point.x=poin(1,:);
        ant(k).Point.y=poin(2,:);
        ant(k).ind=[model.isp ant(k).ind
model.esp];
        ant(k).Cost=CostFunction(poin);%Evaluate
cost of solution

        else
        poin=[model.sp(1) ant(k).Point.x
;model.sp(2) ant(k).Point.y ];
        ant(k).Cost=inf;
        end
    if plottingan==1
    h(nAnt*(it-1)+k)=plot(poin(1,:),poin(2,:),'-
*','linewidth',1,'markersize',3);
    if (nAnt*(it-1)+k)>20
    set(h((nAnt*(it-1)+k)-20),'Visible','off')
    end
        drawnow
    end

        %Selection of the best ant
        if ant(k).Cost<BestSol.Cost
            BestSol=ant(k);
            toc
```

```
        end

    end

    % Update Global Phromones
    for k=1:nAnt
        for i=1:length(ant(k).ind)-1

tau(ant(k).ind(i),ant(k).ind(i+1))=tau(ant(k).in
d(i),ant(k).ind(i+1))+Q/ant(k).Cost;%Update
pheromone
        end
    end

    % Evaporation
    tau=(1-ro)*tau;
    % Store Best Cost
    BestCost(it)=BestSol.Cost;
    % Show Iteration Information
    disp(['Iteration ' num2str(it) ': Best Cost
= ' num2str(BestCost(it))]);
end

%% Plotting cost dan path
figure (2);
plot(BestCost,'LineWidth',2);
xlabel('Iteration');
ylabel('Best Cost');
title('Best Cost at n^t^h Iteration');
grid on;
figure(1);
hold on;
plot([ model.sp(1) BestSol.Point.x
model.ep(1)],[ model.sp(2) BestSol.Point.y
model.ep(2)],'-
r*','linewidth',3,'markersize',10)
title('Best Path');
plot(model.sp(1),model.sp(2),'bo','markersize',1
0,'linewidth',3)
```

```
plot(model.ep(1),model.ep(2),'bo','markersize',1
0,'linewidth',3)
toc
```

LAMPIRAN B. PROGRAM *PATH FOLLOWING DIFFERENTIAL DRIVE ROBOT*

```
path=[BestSol.Point.x]';
path=[path [BestSol.Point.y]'];
figure
imagesc(~world); hold on
set(gca,'YDir','normal');
colormap(gray);
axis square;
plot([BestSol.Point.x],[BestSol.Point.y],'--
ro','linewidth',3,'markersize',10)
robotCurrentLocation = path(1,:);
robotGoal = path(end,:);
initialOrientation = 0;
robotCurrentPose = [robotCurrentLocation
initialOrientation];
robot = ddr(robotCurrentPose);
robot.Dt=0.1;
controller = robotics.PurePursuit;
controller.Waypoints = path;
controller.DesiredLinearVelocity = 5;
controller.LookaheadDistance=1;
controller.MaxAngularVelocity = 2;
goalRadius = 1;
distanceToGoal = norm(robotCurrentLocation -
robotGoal);
tic
while( distanceToGoal > goalRadius )
    % Compute the controller outputs, i.e., the
inputs to the robot
    [v, omega] = step(controller,
robot.CurrentPose);
    % Simulate the robot using the controller
outputs.
    drive(robot, v, omega)
```

```
    % Extract current location information
([X,Y]) from the current pose of the
    % robot
    robotCurrentLocation =
robot.CurrentPose(1:2);
    % Re-compute the distance to the goal
    distanceToGoal = norm(robotCurrentLocation -
robotGoal);
end
toc
```

RIWAYAT PENULIS

Penulis bernama Dimas Bintang Pamungkas dilahirkkan di kota Surabaya, 02 Agustus 1995. Karena kecintaan pada kota kelahiran, penulis mengenyam pendidikan di kota kelahiran, penulis tetap memutuskan melanjutkan pendidikan di kampus teknik kebanggaan kotanya yaitu Institut Teknologi Sepuluh Nopember Fakultas Teknik Elektro dengan konsentrasi dibidang studi Teknik Sistem Pengaturan.

Penulis merupakan anggota dari lab Sistem dan Sibernatika, mempelajari sistem navigasi *mobile robot.* Penulis dapat dihubungi melalui Surel : Dmatch.Dstar@gmail.com.

[Halaman ini sengaja dikosongkan]