Mapping dan Navigasi untuk Robot Pengantar Makanan di Restoran Berbasis ROS

Louise[1], Yohana Susanthi[2], Muliady[3]

Program Studi Teknik Elektro,
Fakultas Teknik,
Universitas Kristen Maranatha, Bandung
[1]1822030@eng.maranatha.edu, [2]yohana.susanthi@eng.maranatha.edu,
[3]muliady@eng.maranatha.edu

Abstrak

Perkembangan teknologi yang pesat memungkinkan pekerjaan manusia digantikan oleh robot sehingga dapat memberikan hasil lebih konsisten dan diharapkan meminimalisir kesalahan yang terjadi. Dengan menggunakan robot *turtlebot*2 dan sensor RP LiDAR yang dikontrol dengan ROS sebagai *framework* komunikasi antar proses, telah dirancang dan direalisasikan robot yang dapat melakukan *mapping*, *localization*, *path planning*, navigasi, serta *obstacle avoidance* untuk mengantarkan makanan di restoran. *Mapping* dan *localization* dapat dicapai dengan menggunakan algoritma SLAM dengan metoda *gmapping* dan *amcl*. Algoritma *path planning* menggunakan algoritma *Djikstra* untuk menghasilkan jalur yang paling pendek untuk mencapai tujuan. Robot dapat bernavigasi sesuai jalur tersebut sehingga dapat terwujud robot pengantar makanan yang otonom. Realisasi dari perancangan sistem ini menghasilkan robot yang dapat mengantarkan makanan ke meja tujuan dan kembali lagi ke dapur secara otomatis dengan waktu rata-rata untuk meja terdekat 51 detik dan untuk meja yang terjauh 124 detik dengan tingkat keberhasilan antara 60% sampai dengan 100%. Pada proses navigasi, robot dapat menghindari halangan-halangan yang terdeteksi oleh sensor RP LiDAR dan tidak terdapat perbedaan waktu yang berarti.

Kata kunci: *turtlebot*, RP LiDAR, ROS, *SLAM, path planning*

Abstract

In this modern era, more and more human jobs can be replaced by robots. One of the many robots developed is a robot to deliver food at restaurants. By using the turtlebot2 robot and RP LiDAR sensor controlled by ROS as a communication framework between processes, a robot that can perform mapping, localization, path planning, navigation and obstacle avoidance has been designed and realized to deliver foods in restaurants. Mapping and localization are achieved using the SLAM algorithm with gmapping and amcl methods. Path planning algorithm will be based on Djikstra algorithm to result in the shortest path to reach the destination. Robot will navigate according to the path to result in an autonomous food delivery robot. The realized robot can deliver food to the destination table and back to the kitchen automatically with an average time of 51 seconds for the nearest table and 124 seconds for the farthest table with a success rate between 60% to 100%. The robot can avoid obstacles detected by the RP LiDAR sensor and there is no significant time difference.

Keywords: turtlebot, RP LiDAR, ROS, SLAM, path planning

Received: 18 November 2022 Revised: 5 Februari 2023 Accepted: 14 Februari 2023

1. Pendahuluan

Layanan atau servis adalah sebuah tindakan yang dilakukan untuk memenuhi kebutuhan atau keinginan orang lain. Pelayanan yang optimal tidak hanya memenuhi kebutuhan dan keinginan tersebut, tetapi juga memberikan sebuah kepuasan kepada orang yang dilayani. Terdapat beberapa jenis pelayanan yang dapat dilakukan, yaitu: layanan dengan lisan, layanan dengan tulisan, dan layanan dengan perbuatan [1].

Pelayan adalah karyawan restoran yang memiliki tujuan dan tanggung jawab memberikan pelayanan akan kebutuhan makan dan minum. Contoh pelayanan yang dilakukan di restoran ini merupakan layanan yang dilakukan dengan perbuatan. Meskipun menghidangkan makanan dan minuman merupakan pekerjaan utama dari pelayan, servis yang diberikan pelayan harus membuat tamu merasa puas dan memiliki keinginan untuk kembali [2].

Robot adalah sebuah sistem yang dapat melakukan pekerjaan yang berulang secara terus-menerus dengan hasil keluaran yang stabil. Dengan perkembangan teknologi, terdapat banyak jenis robot yang dapat melakukan pekerjaan-pekerjaan sederhana, seperti mengangkat barang, hingga pekerjaan-pekerjaan kompleks, seperti membantu operasi medis. Salah satu jenis robot adalah *service robots*, yang dapat memberikan layanan dengan tingkat sosial dan emosional yang rendah [3].

Dengan menggunakan robot *turtlebot* berbasis *Yujin Kobuki* dan sensor RP LiDAR dapat dirancang dan direalisasikan sistem robot pengantar makanan. Sensor RP LiDAR merupakan sensor 2D yang menggunakan laser untuk mendeteksi benda-benda disekitarnya. Perancangan sistem menggunakan ROS sebagai *framework* komunikasi antar proses. Sistem terbagi atas beberapa proses, yaitu: *mapping*, *localization*, *path planning*, navigasi, dan *obstacle avoidance*. Proses *mapping* dan *localization* akan dicapai dengan menggunakan algoritma SLAM (*Simultaneous Localization and Mapping*). Proses *mapping* akan menangkap kondisi di lingkungan sekitar robot dan menyimpan hasil deteksi tersebut dalam bentuk peta. *Localization* dan *path planning*, yang menggunakan algoritma Djikstra digunakan untuk menghasilkan jalur untuk mencapai titik tujuan [4]. Robot dapat bernavigasi sesuai dengan jalur yang dihasilkan dengan fitur *obstacle avoidance* untuk menghindari halangan-halangan pada proses navigasi. Proses-proses ini akan menghasilkan robot pengantar makanan yang otonom.

2. Kajian Pustaka

Beberapa penelitian mengenai *mapping* dan *navigasi* menggunakan ROS dan *turtlebot* telah dilakukan. Diantaranya adalah *gmapping* yang menggunakan *laser scanner* dan ROS pada *turtlebot* [4]. Penelitian tersebut menggunakan sensor LiDAR (SICK TIM571) yang terpasang pada robot *turtlebot3* untuk melakukan *mapping* dengan algoritma *SLAM gmapping*. Data dari sensor dihubungkan ke SBC Odroid XU4 dan diproses lebih lanjut oleh ROS pada proses pemetaan. Penelitian lainnya adalah mengenai algoritma *path planning* dan navigasi pada robot otonom [5]. Pada penelitian ini digunakan simulasi *turtlebot3* dengan fitur ROS Gazebo. Sensor LiDAR digunakan untuk membantu pada sistem navigasi dan *obstacle avoidance* dengan penambahan sensor IMU untuk melakukan *tracking* pada posisi dan orientasi robot untuk menghasilkan navigasi yang akurat. Dengan menggunakan sensor LiDAR dan sensor IMU serta pemetaan dari ruang simulasi pada lingkungan Gazebo, dapat dilakukan navigasi dengan menggunakan ROS. Penggunaan ROS sangat menguntungkan dikarenakan sistem *node* yang memungkinkan

setiap *node* untuk bersifat independen yang berarti penambahan *node* baru tidak akan mengganggu *node* yang sedang berjalan. Selain itu, dapat digunakan pemrograman dengan bahasa C++ atau python untuk pengontrolan robot menggunakan ROS. Pada penelitian [6] dibahas mengenai *path planning* pada peta yang dibangun berdasarkan pada algoritma SLAM. Pada penelitian ini digunakan *gmapping* sebagai teknik *mapping*. *Gmapping* menggunakan partikel SLAM yang didapatkan dari data laser dari sensor dan data posisi serta orientasi (*pose*) dari robot. *Localization* pada penelitian ini menggunakan metoda amcl atau *adaptive monte carlo localization* sebagai algoritma untuk mengestimasi posisi dan orientasi robot. Dengan menggunakan *particle sample* yang diletakkan di seluruh atau sebagian peta setelah dilakukan estimasi, akan dilakukan deteksi pada lingkungan di sekitar robot dan sistem akan mencari kemiripan lingkungan sekitar dengan lingkungan pada peta untuk menentukan posisi dan orientasi. Jumlah partikel yang lebih banyak akan menghasilkan estimasi posisi dan orientasi yang lebih akurat. Pada bagian *path planning* digunakan algoritma Djikstra yang akan memberikan jalur dengan *cost* terendah pada peta yang dibagi menjadi *cell* dengan nilai *cost* pada setiap *cell* yang berbeda, tergantung pada ada atau tidak adanya halangan di sekitar *cell* tersebut. Jalur dengan *cost* terendah pada umumnya merupakan jalur terpendek yang bebas dari halangan. Pada penelitian ini dihasilkan peta yang dapat menunjukkan bagian dengan halangan, bagian tanpa halangan, dan bagian yang tidak diketahui serta hasil jalur dari *path planning*. Jalur yang dihasilkan dioptimisasi dengan menggunakan *k-Order Markov Optimization* (KOMO) untuk menghasilkan jalur yang lebih mulus tanpa belokan tajam.

Pada penelitian [11] dibahas mengenai navigasi dari *intelligent robot bobac* yang berbasis ROS dan sensor LiDAR. Proses mapping menggunakan algoritma SLAM dan navigasi robot menggunakan algoritma A* dan DWA. Pengujian *path planning*, *local obstacle avoidance* dan navigasi robot dilakukan menggunakan simulasi di lingkungan Gazebo. Dalam penelitian tersebut robot mampu melakukan proses *mapping* dan navigasi walaupun masih terdapat beberapa kekurangan.

Sedangkan pada penelitian [15] membahas perancangan sistem *mapping* dan navigasi untuk robot banyak (3 robot) berbasis ROS dan SLAM yang juga dilakukan menggunakan simulasi Gazebo. Selanjutnya penelitian [12] dan [13] membahas tentang navigasi otonom robot menggunakan ROS, algoritma SLAM dan algoritma *path planning*. Algoritma SLAM diimplementasikan menggunakan *GMapping Tool*. Sedangkan robot yang digunakan adalah *turtlebot* yang disimulasikan di lingkungan Gazebo. Dalam penelitian tersebut robot dapat mencapai navigasi yang presisi melalui pengembangan software dan beberapa analisa.

Terdapat pula penelitian yang dilakukan secara riil seperti pada penelitian [14] yang membahas robot yang dapat bergerak secara otonom menggunakan ROS dan algoritma SLAM. Robot yang digunakan adalah *tracked mobile robot*. Pengujian robot dilakukan pada ruang koridor berbentuk huruf L. Hasil pengujian menunjukkan bahwa ROS *platform* sangat mempermudah pengembangan robot dan SLAM dapat dengan mudah direalisasikan pada ROS untuk menghasilkan pergerakan robot secara otonom. Kemudian pada penelitian [16] melakukan perancangan sistem *mapping* dan lokalisasi menggunakan metode SLAM. Robot yang digunakan adalah jenis *omnidirectional* yang dilengkapi dengan sensor LiDAR. Pengujian dilakukan di dalam gedung dan hasil *mapping* dan *localization* sudah optimal walaupun terdapat sedikit kendala karena kondisi lingkungan terdapat kaca yang membiaskan cahaya laser dari sensor LiDAR sehingga mempengaruhi saat pembuatan peta global. Penelitian lainnya [17] juga membahas tentang pengujian metoda SLAM dan menerapkan algoritma *Gmapping* pada robot 2 roda

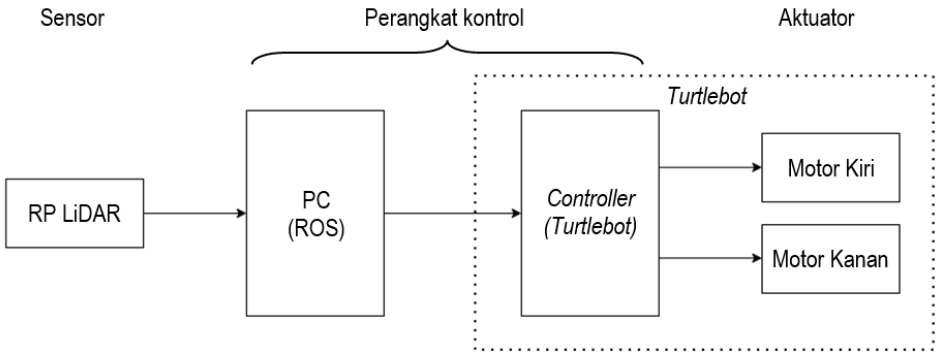
differential drive untuk mendapatkan informasi peta halangan sekitar robot menggunakan simulasi 3D secara *real time* di simulator Gazebo yang langsung terhubung dengan robot.

3. Metode Penelitian

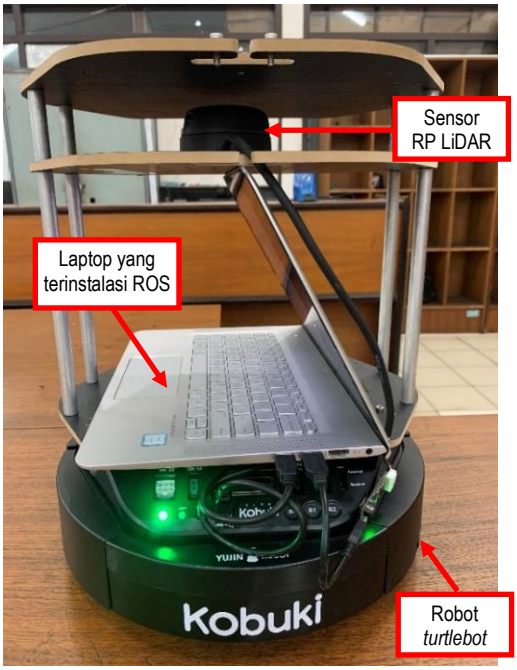
Sistem robot pengantar makanan berbasis ROS dirancang dengan menggunakan sensor RP LiDAR dan robot *turtlebot2* berbasis *Yujin Kobuki*. Pengontrolan sistem dilakukan dengan menggunakan ROS sebagai *framework* komunikasi antar proses. Sistem dirancang untuk melakukan *mapping* ruangan. Dengan menggunakan hasil *mapping*, dilakukan *localization* dan *path planning* untuk menghasilkan jalur yang dapat digunakan pada navigasi dari dapur menuju ke meja tujuan di restoran. Selama proses navigasi, robot dapat menghindari halangan-halangan dengan menggunakan algoritma *obstacle avoidance* [5].

3.1. Diagram Blok Sistem

Gambar 1 memperlihatkan diagram blok dari sistem robot pengantar makanan. Sistem robot pengantar makanan berbasis ROS terdiri dari sensor, perangkat kontrol, dan aktuator. Sensor yang digunakan pada sistem adalah sensor RP LiDAR yang merupakan sensor 2D yang dapat melakukan *scanning* 360° dengan menggunakan laser inframerah. Sensor ini mengadopsi prinsip *laser triangulation* dengan menggunakan kamera yang menangkap sinar inframerah yang dipantulkan ke objek di depan sensor, dan waktu pantul sinar digunakan untuk menghitung jarak antara sensor dengan objek atau halangan [7]. Perangkat kontrol pada sistem terbagi menjadi dua, yaitu: laptop yang terinstalasi ROS dan *controller* pada basis *turtlebot*. ROS adalah *framework* robotik yang bersifat gratis dan dapat digunakan untuk komunikasi antar proses pada sebuah sistem. Sistem yang memproses data dari sensor dan mengontrol aktuator memerlukan beberapa proses sehingga digunakan ROS [8]. *Controller* pada robot digunakan untuk mengaktifkan dan mengontrol aktuator yaitu motor kiri dan motor kanan robot *turtlebot* [9]. Gambar 2 memperlihatkan hasil akhir robot, terdiri atas *turtlebot* yang terpasang *mounting kit* untuk meletakkan sensor RP LiDAR dan laptop yang telah terinstalasi sistem ROS sebagai pengendali proses. Pada tingkat pertama, yaitu di atas robot, terdapat laptop yang digunakan untuk memberikan masukan nomor meja tujuan dan juga tombol yang dapat digunakan untuk memerintahan robot kembali ke dapur. Pada tingkat kedua diletakkan sensor RP LiDAR. Tingkat kedua memiliki ketinggian yang tidak terlalu besar untuk menghindari adanya sentuhan pada sensor yang sedang berputar. Tingkat paling atas menjadi tempat untuk meletakkan makanan yang akan diantar.



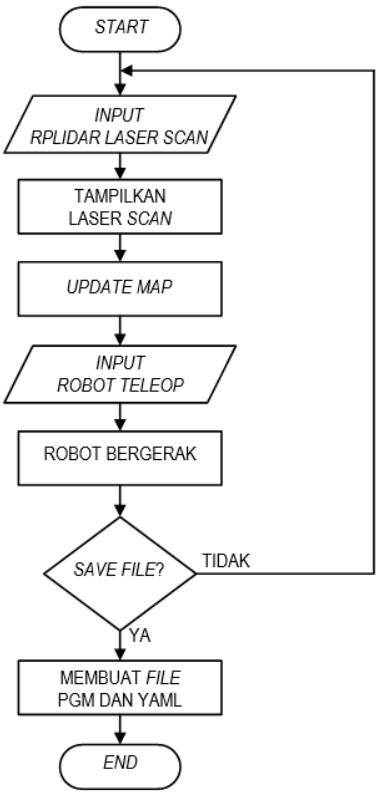
Gambar 1. Blok diagram robot pengantar makanan



Gambar 2. Hasil akhir robot

3.2. Diagram Alir Sistem

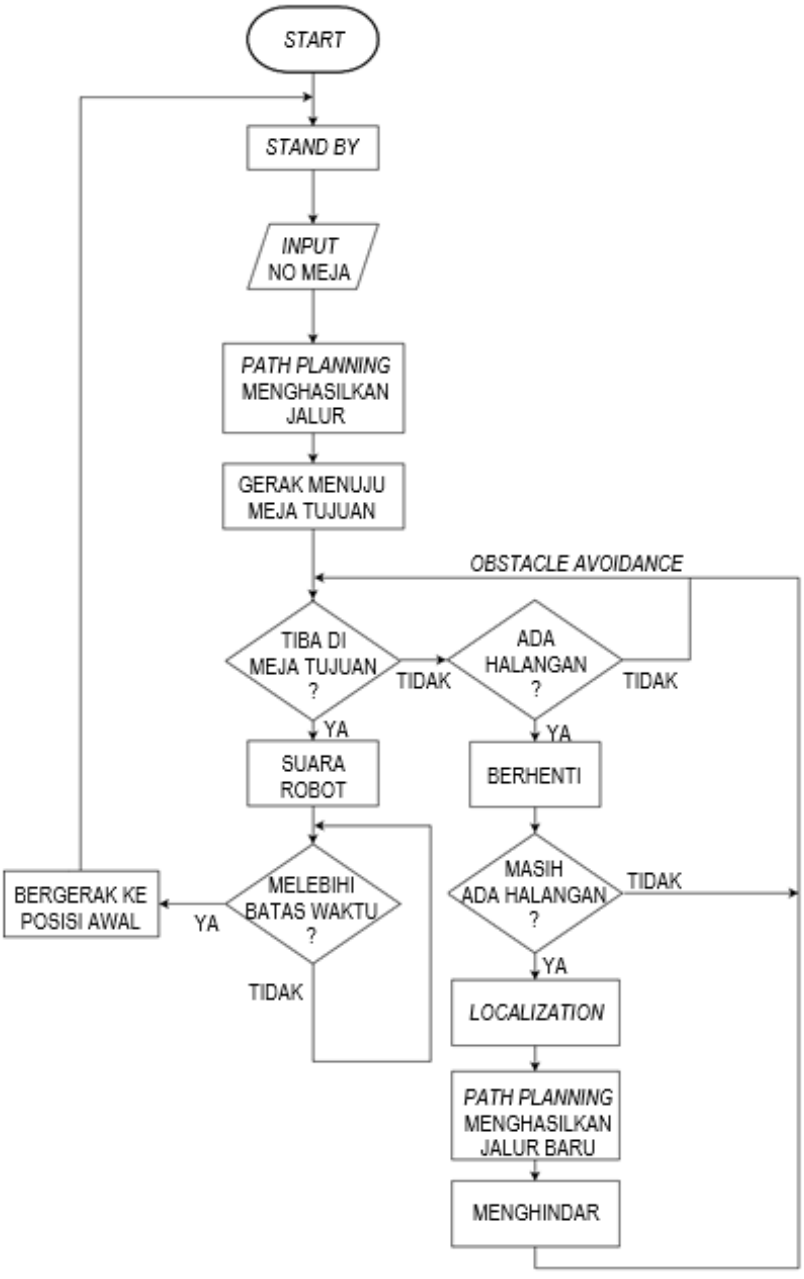
Diagram alir pada sistem robot pengantar makanan dapat dibagi menjadi dua, yaitu: diagram alir pada proses *mapping* dan diagram alir pada proses navigasi atau proses pengantaran makanan.



Gambar 3. Diagram alir proses *mapping*

Diagram alir proses *mapping* ditampilkan pada Gambar 3. Pada diagram alir proses *mapping*, terdapat dua masukan yaitu *laser scan* dari sensor RP LiDAR dan robot *teleop* yang digunakan untuk mengontrol pergerakan robot.

Masukan *laser scan* RP LiDAR akan ditampilkan dan peta yang dihasilkan akan terus diperbarui dengan hasil *laser scan* yang baru. Deteksi dari sensor akan berbeda dikarenakan perubahan lingkungan akibat pergerakan dari masukan robot *teleop*. Proses akan terus berulang hingga terdapat masukan untuk melakukan *save file*. *File* yang dihasilkan akan berupa .pgm dan .yaml. Tipe file .pgm menyimpan peta dalam bentuk gambar sedangkan tipe file .yaml merupakan program yang digunakan untuk memanggil data peta pada aplikasi visualisasi untuk proses *localization* dan navigasi.



Gambar 4. Diagram alir proses pengantaran makanan

Gambar 4 menampilkan diagram alir pada proses navigasi atau pengantaran makanan. Sistem akan dimulai dengan robot yang berada di posisi awal yang telah dimasukkan terlebih dulu ke sistem, yaitu dapur. Sistem akan menunggu masukan yang berupa posisi akhir sesuai dengan nomor meja yang akan dituju. Hasil mapping dan hasil pembacaan sensor RP Lidar menghasilkan sebuah area dengan nilai path weight kecil, sedang dan besar. Setelah menerima masukan, algoritma path planning akan menghasilkan jalur terpendek dari dapur menuju meja tujuan dengan nilai path weight terkecil, sesuai dengan prinsip kerja algoritma Djikstra, dan robot mulai bergerak. Seiring dengan berjalannya robot dan sensor RP Lidar yang terus mengirimkan hasil pembacaan baru, sistem path planning akan menghasilkan jalur baru dengan path weight terkecil dan jarak terpendek. Proses ini akan terjadi secara berulang hingga robot mencapai tujuan. Selama robot bergerak menuju ke meja tujuan, algoritma obstacle avoidance akan menyala dan melakukan deteksi di depan robot untuk memastikan bahwa tidak ada halangan.

Jika terdapat halangan, maka robot akan berhenti dan jika halangan di depan robot masih ada, maka robot akan melakukan *localization* dan algoritma *path planning* akan membentuk jalur yang baru dan robot akan menghindari halangan dengan mengikuti jalur yang baru.

Setelah tiba di tujuan, robot akan mengeluarkan suara yang berupa konfirmasi untuk pelanggan yaitu nomor meja tujuan. Robot akan menunggu di samping meja sesuai dengan waktu yang telah diatur pada bagian pemrograman dan kemudian akan bergerak kembali ke dapur.

3.3. Penggunaan *Library* dan *Package*

Sistem berbasis ROS menggunakan *package* dan *library* untuk melakukan pengontrolan sistem. Pada *library* dan *package* terdapat *launch file* yang berisi program untuk mengontrol sistem. Beberapa *launch file* yang digunakan adalah: *turbot_bringup minimal.launch, turbot_teleop keyboard.launch, turbot_slam laser_gmapping_demo.launch, turbot_rviz nav.launch,* dan *turbot_slam laser_amcl_demo.launch*.

3.4. *Tuning* Parameter

Tuning parameter dilakukan untuk mencapai navigasi yang optimal dan aman. Untuk melakukan *tuning* parameter, *file* harus dibuka sebagai *superuser* menggunakan *command sudo* (*super user do*). Proses *tuning* parameter dilakukan berdasarkan referensi "*ROS Navigation Tuning Guide*" oleh Kaiyu Zheng (2016) [10]. Beberapa *tuning* parameter yang dilakukan antara lain: Kecepatan dan akselerasi (Tabel 1), *path planner* (Tabel 2 dan Tabel 3), *costmap* (Tabel 4), dan amcl (Tabel 5).

Tabel 1. *Tuning* parameter kecepatan dan akselerasi

Parameter	*Default*	Nilai Baru
speed_lim_v	0.8	0.2
speed_lim_w	2.4	1.0
accel_lim_v	1.0	0.3
accel_lim_w	2.0	0.7

Parameter *speed_lim* adalah parameter kecepatan dan parameter *accel_lim* adalah parameter akselerasi. Variabel v menunjukkan pergerakan linier dan variabel w menunjukkan pergerakan rotasi. Kecepatan diturunkan untuk menghindari makanan atau minuman yang tertumpah dan memberi waktu agar sistem dapat memproses

lingkungan sekitarnya. Akselerasi diturunkan untuk menghasilkan pergerakan yang lebih mulus.

Parameter berikutnya adalah parameter *path planner* yang terbagi menjadi *global* dan *local path planner*. *Global path planner* menghasilkan jalur menggunakan data dari *mapping* sedangkan *local path planner* menghasilkan jalur menggunakan data dari sensor. Pada parameter *global path planner*, parameter *allow_unknown, use_djikstra, use_quadratic, use_grid_path,* dan *old_navfn_behavior* digunakan sesuai dengan pengaturan awal yaitu: *true, true, true, false,* dan *false.* Parameter *allow_unknown* akan memungkinkan sistem untuk menghasilkan jalur pada bagian peta yang belum terdeteksi atau dalam kondisi *unknown*. Parameter *use_djikstra, use_quadratic, use_grid_path,* dan *old_navfn_behavior* akan menentukan algoritma *path planning* yang diaktifkan.

Parameter lainnya adalah *lethal_cost, neutral_cost,* dan *cost_factor.* Nilai *cost_factor* menentukan kualitas pergerakan robot saat melewati belokan. Nilai yang terlalu rendah mengakibatkan robot untuk berbelok tajam, sedangkan nilai terlalu tinggi menghasilkan terlalu banyak gerakan yang tidak diperlukan. Pengaruh nilai *neutral*_cost dapat dilihat secara jelas saat robot harus melewati dua belokan untuk mencapai tujuan. Nilai yang terlalu rendah mengakibatkan robot yang berbelok tajam sedangkan saat diberi nilai yang tinggi robot mencoba untuk mengambil jalur yang mendekati garis lurus. Nilai *lethal_cost* yang terlalu rendah mengakibatkan *path* yang tidak dapat dibentuk.

Tabel 2. *Tuning* parameter *global planner*

Parameter	*Default*	Nilai Baru
lethal_cost	253	253
neutral_cost	50	66
cost_factor	3.0	0.55

Parameter *local planner* dapat dibagi menjadi dua, yaitu parameter untuk proses simulasi navigasi dan parameter untuk menghasilkan jalur. Parameter yang berhubungan dengan simulasi navigasi adalah: *sim_time* (*simulation time*) dan *v_sample*. Nilai *sim_time* menunjukkan waktu yang diberikan agar dapat menyimulasikan sebuah jalur. Nilai yang terlalu besar dapat membebani proses komputasi dan menghasilkan jalur yang terlalu panjang sedangkan nilai terlalu rendah menghasilkan performa terbatas dikarenakan waktu yang sedikit. Parameter *v_sample* terbagi menjadi dua, yaitu *vx_sample* dan *vth_sample*. vx menunjukkan kecepatan translasi pada sumbu x dan vth adalah kecepatan rotasi. Pada umumnya, nilai *sampling* pergerakan rotasi akan lebih tinggi dikarenakan pergerakan rotasi lebih kompleks dibandingkan pergerakan translasi.

Tabel 3. *Tuning* parameter *local planner*

Parameter	*Default*	Nilai Baru
vx_sample (*samples*)	3	20
nth_sample (*samples*)	20	40

Parameter yang digunakan untuk menghasilkan jalur adalah: *path_distance_bias*, *goal_distance_bias*, dan *occdist_scale.* Parameter *path_distance_bias* adalah nilai kemiripan *local path* dengan *global path*. Nilai yang terlalu rendah akan menghasilkan jalur yang jauh berbeda dari *global path* tetapi nilai terlalu tinggi dapat menurunkan kualitas dari *obstacle avoidance*. Parameter berikutnya, *goal_distance_bias*, menunjukkan usaha robot untuk mencapai tujuan akhir. Nilai yang terlalu tinggi akan menghasilkan jalur yang sangat berbeda dari *global path* karena tujuan dari robot hanya mencapai *goal*, dan nilai yang

terlalu rendah menghasilkan robot yang tidak membentuk jalur baru untuk mencapai tujuan. Parameter *occdist_scale* menunjukkan usaha robot untuk menghindari halangan. Peningkatan nilai ini akan menghasilkan robot yang sering terjebak dan menyerah untuk mencari jalur lain. Ketiga parameter ini digunakan sesuai dengan nilai *default*.

Tuning parameter *costmap* ditujukan untuk menghasilkan robot yang bergerak tepat di antara dua halangan (tidak terlalu dekat ke satu halangan). Terdapat dua parameter, yaitu: *inflation_radius* yang mengatur radius inflasi halangan dan *cost_scaling_factor* yang menunjukkan *cost* dari setiap *cell*. Nilai dari parameter ini dapat diubah sesuai dengan jarak antar halangan pada peta yang paling sering ditemukan. Nilai *inflation_radius* yang ditambah akan menutupi lebih banyak daerah antara kedua halangan dan *cost_scaling_factor* akan diturunkan untuk menghasilkan *cost* yang rendah pada titik tengah dari kedua halangan.

Tabel 4. *Tuning* parameter *costmap*

Parameter	*Default*	Nilai Baru
inflation_radius (m)	0.55	1.25
cost_scaling_factor	5.0	2.8

Tuning parameter terakhir berhubungan dengan algoritma *localization*, yaitu amcl dengan tujuan untuk mengurangi *offset* yang dihasilkan oleh hasil *scanning* (*real*) dan pada peta (simulasi). *Tuning* parameter ini dilakukan sesuai dengan sensor yang digunakan, yaitu sensor RP LiDAR.

Tabel 5. *Tuning* parameter amcl

Parameter	*Default*	Nilai Baru
laser_z_hit	0.5	0.9
laser_sigma_hit	0.2	0.1
laser_z_rand	0.5	0.5
laser_likelihood_max_dist	2.0	2.0
kld_err	0.01	0.1
kld_z	0.99	0.5
odom_alpha (1,2,3,4)	0.005	0.04

4. Hasil dan Analisis

4.1. Pengaruh Beban Terhadap Kecepatan Robot

Pengujian pengaruh beban terhadap kecepatan robot, dilakukan dengan mengukur waktu yang dibutuhkan robot untuk bergerak sejauh 1 meter dengan beban yang berbeda. Beban yang diukur tidak termasuk robot, laptop dan sensor. Digunakan botol air mineral 600 ml sebagai beban. Percobaan dilakukan mulai dari robot yang tidak diberikan beban (0 g) hingga diberi beban sampai dengan 8 botol air mineral (4800 g) dengan lima kali pengambilan data untuk setiap nilai beban. Didapatkan data seperti pada Tabel 6.

Dari data yang didapatkan, waktu yang dibutuhkan robot untuk bergerak sejauh 1 m relatif sama meskipun diberikan beban yang berbeda. Dari data tersebut dapat disimpulkan bahwa beban dari robot (sampai dengan 4800 g) tidak mempengaruhi kecepatan dari pergerakan robot.

Tabel 6. Perbandingan beban - kecepatan

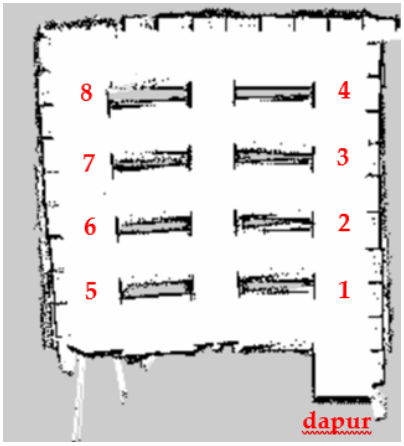
Beban (g)	Waktu 1	Waktu 2	Waktu 3	Waktu 4	Waktu 5	Rata-rata
0	1,59 s	1,45 s	1,49 s	1,56 s	1,56 s	1,53 s
600	1.46 s	1,57 s	1,45 s	1,56 s	1,55 s	1,52 s
1200	1,47 s	1,55 s	1,51 s	1,49 s	1,56 s	1,52 s
1800	1,50 s	1,47 s	1,60 s	1,52 s	1,54 s	1,53 s
2400	1,66 s	1,53 s	1,44 s	1,51 s	1,48 s	1,52 s
3000	1,51 s	1,50 s	1,54 s	1,59 s	1,64 s	1,56 s
3600	1,56 s	1,55 s	1,53 s	1,47 s	1,59 s	1.54 s
4200	1,55 s	1,55 s	1,52 s	1,49 s	1,50 s	1,52 s
4800	1,57 s	1,60 s	1,51 s	1,62 s	1,58 s	1,58 s

4.2. Pengujian *Mapping* dan Navigasi

Pengujian *mapping* dan navigasi dilakukan di ruangan yang berisi delapan meja seperti terlihat pada Gambar 5.

Gambar 5. Ruangan dengan 8 meja untuk pengujian *mapping* dan navigasi

Hasil *mapping* dan penomoran meja untuk ruangan tersebut terlihat pada Gambar 6. Pengujian navigasi dilakukan sebanyak lima kali ke setiap meja. Waktu yang dibutuhkan robot untuk bernavigasi dari dapur ke meja dan dari meja kembali ke dapur diukur, serta indikasi suara yang menunjukkan keberhasilan navigasi. Hasil pengujian navigasi ke meja 1 hingga meja 8 ditampilkan pada Tabel 7 hingga Tabel 14.



Gambar 6. Hasil *mapping* dan penomoran meja

Tabel 7. Pengujian Navigasi (Meja 1)

Percobaan	Navigasi dari dapur ke meja	Suara	Navigasi dari meja ke dapur
1	Berhasil (45 detik)	Berhasil	Berhasil (27 detik)
2	Berhasil (17 detik)	Berhasil	Berhasil (44 detik)
3	Berhasil (20 detik)	Berhasil	Berhasil (26 detik)
4	Berhasil (17 detik)	Berhasil	Berhasil (18 detik)
5	Berhasil (21 detik)	Berhasil	Berhasil (20 detik)
Waktu rata-rata	24 detik	-	27 detik
Keberhasilan	100%	100%	100%

Tabel 8. Pengujian Navigasi (Meja 2)

Percobaan	Navigasi dari dapur ke meja	Suara	Navigasi dari meja ke dapur
1	Berhasil (28 detik)	Berhasil	Berhasil (33 detik)
2	Berhasil (27 detik)	Berhasil	Berhasil (34 detik)
3	Berhasil (46 detik)	Berhasil	Berhasil (45 detik)
4	Berhasil (29 detik)	Berhasil	Gagal (*error when rotating*)
5	Berhasil (26 detik)	Berhasil	Berhasil (32 detik)
Waktu rata-rata	31 detik	-	36 detik
Keberhasilan	100%	100%	80%

Tabel 9. Pengujian Navigasi (Meja 3)

Percobaan	Navigasi dari dapur ke meja	Suara	Navigasi dari meja ke dapur
1	Berhasil (40 detik)	Berhasil	Berhasil (44 detik)
2	Berhasil (115 detik)	Gagal	Berhasil (49 detik)
3	Berhasil (39 detik)	Berhasil	Berhasil (43 detik)
4	Berhasil (115 detik)	Gagal	Berhasil (37 detik)
5	Berhasil (37 detik)	Berhasil	Berhasil (42 detik)
Waktu rata-rata	69 detik	-	43 detik
Keberhasilan	100%	60%	100%

Tabel 10. Pengujian Navigasi (Meja 4)

Percobaan	Navigasi dari dapur ke meja	Suara	Navigasi dari meja ke dapur
1	Berhasil (46 detik)	Berhasil	Berhasil (55 detik)
2	Berhasil (48 detik)	Berhasil	Berhasil (53 detik)
3	Berhasil (45 detik)	Berhasil	Berhasil (50 detik)
4	Berhasil (47 detik)	Berhasil	Berhasil (53 detik)
5	Berhasil (46 detik)	Berhasil	Berhasil (56 detik)
Waktu rata-rata	46 detik	-	53 detik
Keberhasilan	100%	100%	100%

Tabel 11. Pengujian Navigasi (Meja 5)

Percobaan	Navigasi dari dapur ke meja	Suara	Navigasi dari meja ke dapur
1	Berhasil (30 detik)	Berhasil	Berhasil (37 detik)
2	Berhasil (60 detik)	Berhasil	Berhasil (38 detik)
3	Berhasil (40 detik)	Berhasil	Gagal (tidak dapat berotasi)
4	Berhasil (30 detik)	Berhasil	Berhasil (45 detik)
5	Berhasil (40 detik)	Berhasil	Berhasil (40 detik)
Waktu rata-rata	40 detik	-	40 detik
Keberhasilan	100%	100%	80%

Tabel 12. Pengujian Navigasi (Meja 6)

Percobaan	Navigasi dari dapur ke meja	Suara	Navigasi dari meja ke dapur
1	Berhasil (110 detik)	Gagal	Berhasil (47 detik)
2	Berhasil (40 detik)	Berhasil	Berhasil (43 detik)
3	Berhasil (41 detik)	Berhasil	Berhasil (45 detik)
4	Gagal (menabrak meja)	Gagal	Gagal (tidak dapat berotasi)
5	Berhasil (45 detik)	Berhasil	Berhasil (44 detik)
Waktu rata-rata	59 detik	-	45 detik
Keberhasilan	80%	60%	80%

Tabel 13. Pengujian Navigasi (Meja 7)

Percobaan	Navigasi dari dapur ke meja	Suara	Navigasi dari meja ke dapur
1	Berhasil (48 detik)	Berhasil	Berhasil (60 detik)
2	Berhasil (49 detik)	Berhasil	Berhasil (58 detik)
3	Berhasil (49 detik)	Berhasil	Berhasil (57 detik)
4	Berhasil (50 detik)	Berhasil	Berhasil (55 detik)
5	Berhasil (50 detik)	Berhasil	Berhasil (61 detik)
Waktu rata-rata	49 detik	-	58 detik
Keberhasilan	100%	100%	100%

Tabel 14. Pengujian Navigasi (Meja 8)

Percobaan	Navigasi dari dapur ke meja	Suara	Navigasi dari meja ke dapur
1	Berhasil (58 detik)	Berhasil	Berhasil (70 detik)
2	Berhasil (60 detik)	Berhasil	Berhasil (64 detik)
3	Berhasil (58 detik)	Berhasil	Berhasil (66 detik)
4	Berhasil (59 detik)	Berhasil	Berhasil (63 detik)
5	Berhasil (60 detik)	Berhasil	Berhasil (64 detik)
Waktu rata-rata	59 detik	-	65 detik
Keberhasilan	100%	100%	100%

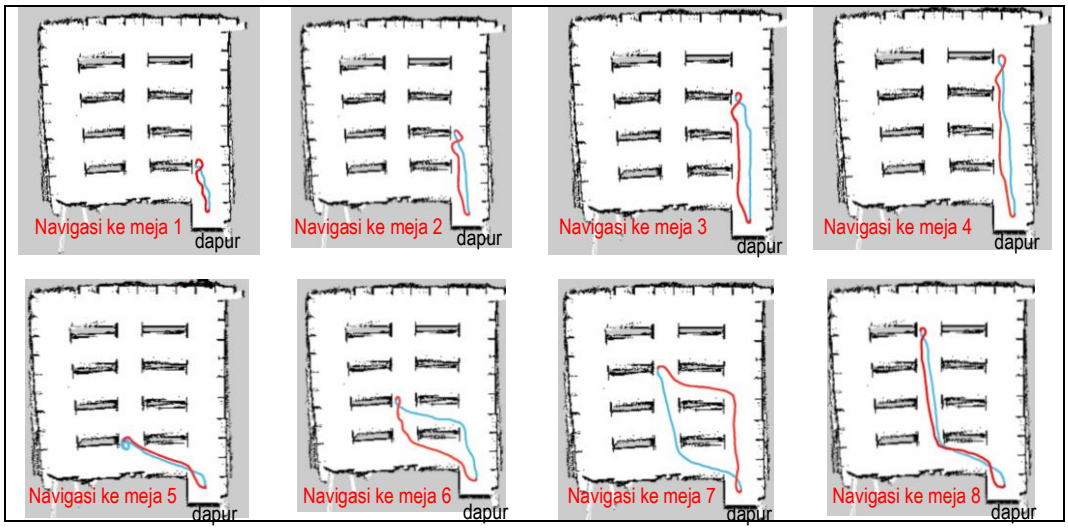
Dari pengujian yang dilakukan, terjadi beberapa kegagalan. Kegagalan-kegagalan yang terjadi dapat disebabkan oleh beberapa faktor seperti lantai yang tidak rata, beberapa benda yang tidak terdeteksi, robot terlalu dekat dengan halangan, dan pemilihan koordinat terlalu dekat dengan meja sehingga pada beberapa percobaan robot gagal untuk melakukan rotasi. Selain kegagalan yang terjadi, beberapa percobaan navigasi menggunakan waktu yang lebih lama untuk mencapai tujuan. Ini disebabkan dikarenakan sistem yang terkadang melakukan lokalisasi secara acak untuk memastikan ketepatan posisi dan orientasi robot. Rangkuman hasil pengujian navigasi seluruh meja dapat dilihat pada Tabel 15. Waktu rata-rata robot bernavigasi dari dapur ke meja tujuan yang tercepat adalah ke meja 1 yaitu 24 detik dan yang terlama adalah ke meja 3 yaitu 69 detik. Sedangkan waktu rata-rata robot bernavigasi dari meja tujuan kembali ke dapur yang tercepat adalah dari meja 1 yaitu 27 detik dan yang terlama adalah dari meja 8 yaitu 65 detik. Jika dihitung dari waktu *round trip* maka yang tercepat adalah ke meja 1 yaitu 51 detik dan yang terlama adalah ke meja 8 yaitu 124 detik. Hal ini wajar karena meja 1 memiliki jarak yang terdekat dengan dapur dan meja 8 adalah yang terjauh. Tingkat keberhasilan robot bernavigasi berkisar antara 60% sampai dengan 100%.

Contoh hasil pengujian navigasi yang berhasil untuk meja 1 sampai 8 ditunjukkan oleh Gambar 7. Garis biru menunjukkan jalur pergerakan robot dari dapur ke meja dan garis merah menunjukkan pergerakan robot dari meja ke dapur.

Gambar 8 memperlihatkan saat robot sedang melakukan navigasi atau mengantar makanan ke meja 6.

Tabel 15. Rangkuman Pengujian Navigasi

Meja Tujuan	Navigasi dari dapur ke meja		Keberhasilan suara	Navigasi dari meja ke dapur		*Round trip*
	Waktu rata-rata	Keberhasilan		Waktu rata-rata	Keberhasilan	
Meja 1	24 detik	100%	100%	27 detik	100%	51 detik
Meja 2	31 detik	100%	100%	36 detik	80%	67 detik
Meja 3	69 detik	100%	60%	43 detik	100%	112 detik
Meja 4	46 detik	100%	100%	53 detik	100%	99 detik
Meja 5	40 detik	100%	100%	40 detik	80%	80 detik
Meja 6	59 detik	80%	60%	45 detik	80%	104 detik
Meja 7	49 detik	100%	100%	58 detik	100%	107 detik
Meja 8	59 detik	100%	100%	65 detik	100%	124 detik



Gambar 7. Contoh hasil pengujian navigasi yang berhasil untuk meja 1 sampai 8

Gambar 8. Robot sedang melakukan navigasi ke meja 6

4.3. Pengujian Pengantaran Makanan dan *Obstacle Avoidance*

Pada pengujian ini dilakukan simulasi pengantaran makanan dengan halangan seperti di restoran. Pengantaran makanan akan dilakukan ke meja 1, 3, 6, dan 8 sebanyak lima kali dengan halangan bergerak dan lima kali dengan halangan yang diam.

Tabel 16 hingga Tabel 23 menampilkan waktu yang diperlukan robot untuk mencapai tujuan dan upaya yang dilakukan untuk menghindari halangan. Hasil pengujian pengantaran makanan dengan halangan bergerak dan halangan yang diam untuk meja 1, 3, 6 dan 8 jika dibandingkan dengan pengujian tanpa halangan dapat

dilihat pada Tabel 24. Dari pengujian ini terlihat bahwa tidak ada perbedaan waktu yang berarti saat robot diberikan halangan bergerak atau halangan diam. Tingkat keberhasilan robot dalam mengantar makanan baik dengan halangan maupun tanpa halangan adalah 60% sampai 100%

Tabel 16. Pengantaran makanan ke meja 1 (Halangan bergerak)

Percobaan	Pengantaran Makanan	Navigasi kembali ke dapur	*Obstacle Avoidance*
1	Berhasil (30 detik)	Berhasil (23 detik)	*Localization* ulang
2	Berhasil (26 detik)	Berhasil (22 detik)	Robot melambat
3	Berhasil (48 detik)	Berhasil (24 detik)	Berhenti sebentar
4	Berhasil (22 detik)	Berhasil (22 detik)	Melambat dan berhenti
5	Berhasil (20 detik)	Berhasil (20 detik)	Berhenti sebentar
Waktu rata-rata	29 detik	22 detik	
Keberhasilan	100%	100%	

Tabel 17. Pengantaran makanan ke meja 1 (Halangan diam)

Percobaan	Pengantaran Makanan	Navigasi kembali ke dapur	*Obstacle Avoidance*
1	Berhasil (48 detik)	Berhasil (23 detik)	Berhenti dan menghindar
2	Berhasil (33 detik)	Berhasil (23 detik)	lokalisasi dan menghindar
3	Berhasil (54 detik)	Berhasil (24 detik)	menghindar
4	Berhasil (34 detik)	Berhasil (23 detik)	menghindar
5	Berhasil (20 detik)	Gagal (menabrak halangan)	menghindar
Waktu rata-rata	38 detik	23 detik	
Keberhasilan	100%	80%	

Tabel 18. Pengantaran makanan ke meja 3 (Halangan bergerak)

Percobaan	Pengantaran Makanan	Navigasi kembali ke dapur	Obstacle Avoidance
1	Berhasil (125 detik)	Berhasil (43 detik)	Localization ulang
2	Berhasil (31 detik)	Berhasil (45 detik)	berhenti sebentar
3	Berhasil (53 detik)	Gagal (tidak dapat berotasi)	Localization ulang
4	Berhasil (37 detik)	Berhasil (40 detik)	melambat dan berhenti
5	Berhasil (54 detik)	Gagal (tidak dapat berotasi)	Robot melambat
Waktu rata-rata	60 detik	43 detik	
Keberhasilan	100%	60%	

Tabel 19. Pengantaran makanan ke meja 3 (Halangan diam)

Percobaan	Pengantaran Makanan	Navigasi kembali ke dapur	*Obstacle Avoidance*
1	Berhasil (122 detik)	Berhasil (39 detik)	menghindar
2	Berhasil (57 detik)	Berhasil (43 detik)	lokalisasi dan menghindar
3	Berhasil (30 detik)	Berhasil (43 detik)	menghindar
4	Berhasil (40 detik)	Berhasil (43 detik)	menghindar
5	Gagal (menabrak)	Gagal (tidak dapat berotasi)	menghindar
Waktu rata-rata	62 detik	42 detik	
Keberhasilan	80%	80%	

Tabel 20. Pengantaran makanan ke meja 6 (Halangan bergerak)

Percobaan	Pengantaran Makanan	Navigasi kembali ke dapur	*Obstacle Avoidance*
1	Berhasil (40 detik)	Berhasil (43 detik)	robot melambat
2	Berhasil (43 detik)	Berhasil (43 detik)	berhenti sebentar
3	Berhasil (40 detik)	Berhasil (52 detik)	berhenti sebentar
4	Berhasil (45 detik)	Berhasil (38 detik)	menghindar
5	Berhasil (40 detik)	Berhasil (42 detik)	robot melambat
Waktu rata-rata	42 detik	44 detik	
Keberhasilan	100%	100%	

Tabel 21. Pengantaran makanan ke meja 6 (Halangan diam)

Percobaan	Pengantaran Makanan	Navigasi kembali ke dapur	*Obstacle Avoidance*
1	Berhasil (84 detik)	Berhasil (54 detik)	berhenti dan menunggu
2	Berhasil (54 detik)	Berhasil (40 detik)	menghindar
3	Berhasil (70 detik)	Berhasil (42 detik)	tertabrak tetapi berhasil sampai di meja tujuan
4	Berhasil (58 detik)	Gagal (menabrak halangan)	menghindar
5	Berhasil (70 detik)	Berhasil (42 detik)	tertabrak tetapi berhasil sampai di meja tujuan
Waktu rata-rata	67 detik	45 detik	
Keberhasilan	100%	80%	

Tabel 22. Pengantaran makanan ke meja 8 (Halangan bergerak)

Percobaan	Pengantaran Makanan	Navigasi kembali ke dapur	*Obstacle Avoidance*
1	Berhasil (84 detik)	Berhasil (70 detik)	robot melambat
2	Berhasil (74 detik)	Berhasil (66 detik)	berbelok (menghindar)
3	Berhasil (89 detik)	Berhasil (64 detik)	robot melambat
4	Berhasil (92 detik)	Berhasil (64 detik)	berbelok (menghindar)
5	Berhasil (86 detik)	Gagal (gagal berotasi)	berhenti dan lokalisasi
Waktu rata-rata	85 detik	66 detik	
Keberhasilan	100%	80%	

Tabel 23. Pengantaran makanan ke meja 8 (Halangan diam)

Percobaan	Pengantaran Makanan	Navigasi kembali ke dapur	*Obstacle Avoidance*
1	Berhasil (104 detik)	Berhasil (64 detik)	berhenti dan menghindar
2	Berhasil (72 detik)	Berhasil (68 detik)	melewati halangan tepat diantara keduanya
3	Berhasil (78 detik)	Berhasil (73 detik)	menghindar
4	Berhasil (72 detik)	Berhasil (68 detik)	melewati halangan tepat diantara keduanya
5	Berhasil (78 detik)	Berhasil (73 detik)	menghindar
Waktu rata-rata	81 detik	69 detik	
Keberhasilan	100%	100%	

Tabel 24. Perbandingan hasil pengantaran makanan dengan dan tanpa halangan untuk meja 1, 3, 6 dan 8

Meja Tujuan	Navigasi dari dapur ke meja		Navigasi dari meja ke dapur		*Round trip*
	Waktu rata-rata	Keberhasilan	Waktu rata-rata	Keberhasilan	
Meja 1 (tanpa halangan)	24 detik	100%	27 detik	100%	51 detik
Meja 1 (halangan bergerak)	29 detik	100%	22 detik	100%	51 detik
Meja 1 (halangan diam)	38 detik	100%	23 detik	80%	61 detik
Meja 3 (tanpa halangan)	69 detik	100%	43 detik	100%	112 detik
Meja 3 (halangan bergerak)	60 detik	100%	43 detik	60%	103 detik
Meja 3 (halangan diam)	62 detik	80%	42 detik	80%	104 detik
Meja 6 (tanpa halangan)	59 detik	80%	45 detik	80%	104 detik
Meja 6 (halangan bergerak)	42 detik	100%	44 detik	100%	86 detik
Meja 6 (halangan diam)	67 detik	100%	45 detik	80%	112 detik
Meja 8 (tanpa halangan)	59 detik	100%	65 detik	100%	124 detik
Meja 8 (halangan bergerak)	85 detik	100%	66 detik	80%	151 detik
Meja 8 (halangan diam)	81 detik	100%	69 detik	100%	150 detik

Gambar 9 memperlihatkan saat dilakukan pengujian pengantaran makanan dengan halangan yang bergerak yaitu orang yang berjalan di depan robot dan Gambar 10 memperlihatkan pengujian dengan halangan yang diam berupa benda kotak yang diletakkan di dekat lintasan robot.

Gambar 9. Pengujian pengantaran makanan dengan halangan yang bergerak

Gambar 10. Pengujian pengantaran makanan dengan halangan yang diam

Berdasarkan percobaan-percobaan yang dilakukan, dapat disimpulkan bahwa beberapa upaya yang dilakukan robot untuk menghindari halangan adalah: menghindar, melambat, berhenti, atau melakukan lokalisasi ulang.

5. Kesimpulan

Dari pengujian yang dilakukan dapat disimpulkan bahwa robot *turtlebot* dengan sensor RP LiDAR yang dikontrol menggunakan ROS dapat melakukan *mapping* dan navigasi untuk pengantaran makanan. Proses *mapping* dipengaruhi oleh kemampuan komputasi dan ukuran ruangan. Ruangan yang lebih kecil memberikan beban komputasi lebih kecil sehingga hasil *mapping* lebih jelas. Jalur yang dihasilkan dapat berbeda meskipun titik awal, titik akhir, dan kondisi ruangan sama. Pada pengujian dengan *obstacle avoidance* atau halangan, robot dapat bernavigasi menuju ke meja tujuan saat diberikan halangan yang bergerak maupun diam. Aksi yang dilakukan robot untuk menghindari halangan, antara lain: menghindar, melambat, berhenti, dan melakukan lokalisasi ulang. Faktor yang dapat menghambat navigasi robot adalah lantai atau keramik yang tidak rata sehingga menghambat perputaran motor pada kecepatan rendah, beberapa bagian dari benda (meja/kursi) yang tidak terdeteksi, sistem yang melakukan belokan terlalu dekat dengan halangan, dan pemilihan koordinat yang terlalu dekat dengan meja sehingga robot gagal untuk melakukan rotasi saat akan bernavigasi kembali ke dapur. Waktu rata-rata yang dibutuhkan robot untuk mengantar makanan ke meja tujuan dan kembali lagi ke dapur (*round trip*) untuk meja terdekat yaitu meja 1 adalah 51

detik dan untuk meja yang terjauh yaitu meja 8 adalah 124 detik. Sedangkan pada pengujian dengan *obstacle avoidance,* robot dapat melewati halangan dan tidak terdapat perbedaan waktu yang berarti. Tingkat keberhasilan robot dalam mengantar makanan ke meja tujuan yaitu antara 60% sampai dengan 100%.

Daftar Pustaka

[1] M. Paisal, "Pelaksanaan Pelayanan Karyawan PT Antar Lintas Sumatera Pekanbaru Ditinjau Menurut Ekonomi Islam," Universitas Islam Negeri Sultan Syarif Kasim Riau, Riau, 2015.

[2] F. E. AS dan T. M. M. Aktalina, "Pentingnya Peranan Skill dan Menu Knowledge Waiter/Waiters Terhadap Kepuasan Pelanggan di Food and Beverage Service Departemen," *Pesona,* vol. 18, no. 1, 2016.

[3] V. N. Lu, J. Wirtz, W. H. Kunz, S. Paluch, T. Gruber, A. Martins dan P. G. Patterson, "Service robots, customers and service employees: what can we learn from the academic literature and where are the gaps?," *Journal of Service Theory and Practice,* 2020.

[4] A. Rahman, "Penerapan SLAM Gmapping dengan Robot Operating System Menggunakan Laser Scanner pada Turtlebot," *Jurnal Rekayasa Elektrika,* vol. 16, no. 2, pp. 103-109, 2020.

[5] S. Pietrzik dan B. Chandrasekaran, "Testing Autonomous Path Planning Algorithms and Setup for Robotic Vehicle Navigation," *IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON),* vol. 9, 2018.

[6] D. Ester, "Path Planning and Optimization on SLAM-Based Maps," University of Stuttgart, Stuttgart, 2016.

[7] SLAMTEC, "RPLIDAR A2 Introduction and Datasheet".

[8] L. Joseph, Robot Operating System for Absolute Beginners - Robotics Programming Made Easy, Apress, 2018.

[9] Turtlebot, "What is a Turtlebot?," Open Source Robotics Foundation, Inc.

[10] K. Zheng, "ROS Navigation Tuning Guide," 2016.

[11] X. Yang, "Slam and navigation of indoor robot based on ROS and lidar," *J. Phys. Conf. Ser.*, vol. 1748, no. 2, 2021, doi: 10.1088/1742-6596/1748/2/022038.

[12] S. Pramod Thale, M. Mangesh Prabhu, P. Vinod Thakur, and P. Kadam, "ROS based SLAM implementation for Autonomous navigation using Turtlebot," *ITM Web Conf.*, vol. 32, p. 01011, 2020, doi: 10.1051/itmconf/20203201011.

[13] A. Pajaziti and P. Avdullahu, "SLAM – Map Building and Navigation via ROS," *Int. J. Intell. Syst. Appl. Eng.*, vol. 2, no. 4, pp. 71–75, 2014, doi: 10.1039/b000000x.

[14] Z. An, L. Hao, Y. Liu, and L. Dai, "Development of Mobile Robot SLAM Based on ROS," *Int. J. Mech. Eng. Robot. Res.*, vol. 5, no. 1, pp. 47–51, 2016, doi: 10.18178/ijmerr.5.1.47-51.

[15] Z. Zheng, H. Gong, R. Duan, X. Qiu, and J. Zou, "Design of multi-robot collaborative navigation and control system based on ROS and laser SLAM," *J. Phys. Conf. Ser.*, vol. 2284, no. 1, 2022, doi: 10.1088/1742-6596/2284/1/012008.

[16]A. A. Fikri and L. Anifah, "Mapping And Localization System Pada Mobile Robot Menggunakan Metode SLAM Berbasis LiDAR," *J. Inf. Eng. Educ. Technol.*, vol. 5, no. 1, pp. 27–33, 2021, doi: 10.26740/jieet.v5n1.p27-33.

[17]F. Rohman and W. Harsanti, "Analisis Kinerja Teknologi Slam - Gmapping Untuk Robot Beroda Menggunakan Sensor Hokuyo Laser Scanner," *PENA Tek. J. Ilm. Ilmu-Ilmu Tek.*, vol. 3, no. 1, pp. 67–72, 2018, doi: 10.51557/pt_jiit.v3i1.168.