Purwarupa *Autonomous Mobile Robot* Dengan *Hoverboard* dan Sensor RPLIDAR Menggunakan Algoritme Hector SLAM Dan Navfn

Bambang Gunawan Tanjung[1], Rizal Maulana[2], Rakhmadhany Primananda[3]

Program Studi Teknik Komputer, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]bambanggunawan@student.ub.ac.id, [2]rizalmaulana@ub.ac.id, [3]rakhmadhany@ub.ac.id

Abstrak

Jumlah permintaan pengiriman barang jasa ekspedisi setiap tahunnya mengalami peningkatan. Kenaikan permintaan ini dipicu oleh eskalasi tren berbelanja *online*. Akibatnya banyak paket yang perlu dikelola di dalam gudang. Atas hal ini, diperlukan sistem distribusi otonom di dalam gudang agar terjadi efisiensi pengelolaan gudang terutama dalam waktu dan tenaga kerja. Sistem terdiri dari sensor *encoder*, IMU GY-521, RPLIDAR A1, dan *Robot Operating System* (ROS). Sistem dengan ciri-ciri robot seperti ini dapat digolongkan sebagai jenis *autonomous mobile robot* (AMR). Keistimewaan penelitian terdapat pada penerapan *hoverboard* sebagai sistem penggerak robot. Proses pembuatan peta ruangan lokal dilakukan oleh algoritme Hector SLAM. Perencanaan jalur global dan lokal dibuat menggunakan algoritme Navfn dan *Dynamic Window Approach* (DWA). Untuk lokalisasi dan pencarian lokasi terkini robot digunakan algoritme *Extended Kalman Filter* (EKF) dan *Adaptive Monte Carlo Localization* (AMCL). Hasil uji menunjukkan bahwa *hoverboard* dapat dipakai dalam sistem sebagai sistem penggerak dengan kemampuan angkut dengan maksimal berat 40Kg. Kinerja sensor *odometry*, IMU, dan RPLIDAR mendapatkan hitungan ketelitian sensor sebesar 99,61%, 81,93%, dan 99,84%. Untuk pembuatan peta, didapatkan hasil akurasi keberhasilan 80% menggunakan algoritme Hector SLAM. Pengujian navigasi global tanpa halangan mencapai akurasi 80%. Sedangkan navigasi lokal dengan halangan bergerak mencapai akurasi 80%.

Kata kunci: *ekspedisi, gudang, pemetaan, navigasi, autonomous mobile robot, ROS*

Abstract

The number of requests for freight forwarding services has increased every year. This increase in demand was triggered by the escalation of online shopping trends. As a result, many packages need to be managed in the warehouse. For this, an autonomous distribution system is needed in the warehouse so that warehouse management efficiency occurs, especially in time and labor. The system consists of a sensor encoder, IMU GY-521, RPLIDAR A1, and Robot Operating System (ROS). Systems with robotic characteristics like this can be classified as a type of autonomous mobile robot (AMR). The specialty of the research is the application of the hoverboard as a robot propulsion system. The local spatial map creation process is carried out by the Hector SLAM algorithm. Global and local path planning was created using the Navfn algorithm and the Dynamic Window Approach (DWA). For localization and finding the robot's current location, the Extended Kalman Filter (EKF) and Adaptive Monte Carlo Localization (AMCL) algorithms are used. The test results show that the hoverboard can be used in the system as a propulsion system with the ability to carry a maximum weight of 40Kg. The performance of the odometry, IMU, and RPLIDAR sensors get a sensor accuracy count of 99.61%, 81.93%, and 99.84%. For map making, the results obtained accuracy of 80% success using the Hector SLAM algorithm. Unhindered global navigation test achieves 80% accuracy. While local navigation with moving obstacles reaches 80% accuracy.

Keywords: *expedition, warehouse, mapping, navigation, autonomous mobile robot, ROS*

1. PENDAHULUAN

Setiap tahunnya, jumlah permintaan pengiriman barang jasa ekspedisi mengalami peningkatan (BPS 2017). Salah satu penyebabnya adalah meningkatnya frekuensi belanja *online* di platform *e-commerce* sejak tahun 2017 (Databooks 2019) yang akan terus

meningkat selama beberapa tahun ke depan (Statista 2019). Jika peningkatan ini tidak dibarengi dengan sistem distribusi yang lebih modern, maka akan menjadi masalah di tahun-tahun mendatang. Waktu pengiriman di Indonesia memiliki rata-rata sekitar 3,8 hari. Data mengenai hal ini didapat dari survei yang dinyatakan oleh (Parcel Perform 2019). Survei ini juga menunjukkan bahwa 36% pelanggan tidak puas dengan layanan kurir. Dari 36% pelanggan yang melaporkan ketidakpuasan, 90% tidak puas dengan keterlambatan pengiriman, waktu pengiriman, dan kurang informasi mengenai status pengiriman.

Dapat disimpulkan bahwa peningkatan jumlah permintaan pengiriman barang setiap tahunnya berdasarkan data tersebut. Namun seiring berjalannya waktu sistem distribusi di gudang menjadi bermasalah. Proses pengiriman yang cepat ini berdampak pada waktu, tenaga, dan biaya yang terkait. Karena banyaknya masalah jasa ekspedisi yang ada, penulis hanya membahas sebagian kecil dari masalah pembagian *camp*. Solusi sistem robot otomatis diyakini dapat mengatasi masalah distribusi produk di gudang. Hal ini bertujuan untuk mempercepat proses penyortiran dan pemindahan barang di gudang. Pada saat yang sama, kebutuhan transportasi penumpang berkurang, yang secara signifikan dapat mengurangi biaya operasional.

Dari permasalahan ini, diperlukan sistem yang mampu mengangkat dan sortir paket secara mandiri sehingga prosesnya dapat dilakukan secara otomatis. Sistem ini menggunakan data *odometry*, akselerasi, laser *point cloud*, dan sistem operasi robot sebagai piranti sistem. Sistem dapat menavigasi secara mandiri dan memahami kondisi lingkungan berdasarkan karakteristik ini. Robot bergerak otonom (AMR) adalah jenis robot terbaik untuk sistem ini. AMR memiliki tugas mengelola barang dari proses distribusi, penyortiran, dan pengangkutan.

Dalam implementasi tersebut digunakan sensor *encoder* motor BLDC, *Inertia Measurement Unit* (IMU), dan RPLIDAR. Untuk algoritme yang digunakan dalam perencanaan jalur global digunakan Navfn, sedangkan untuk perencanaan lokal digunakan DWA. Hector SLAM bertindak sebagai algoritma untuk proses pemetaan karena robot membutuhkan peta lokal sebelum melakukan navigasi. Lokalisasi dibutuhkan karena selama proses pemetaan dan navigasi, sistem perlu mengetahui lokasinya secara simultan, digunakan algoritme EKF dan AMCL.

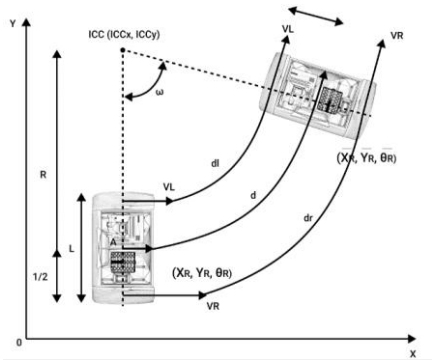
Implementasi dari sistem terotomatisasi dengan sistem *self-driving* ke dalam sebuah robot menjadi tujuan hasil akhir dari penelitian. Pemanfaatan dari sistem terotomatisasi untuk distribusi gudang dan pemajuan industri jasa ekspedisi di Indonesia menjadi harapan dari penelitian ini.

2. DASAR TEORI

2.1 Hoverboard

Hoverboard menggunakan dua motor BLDC yang berputar pada kecepatan tinggi dan menghasilkan torsi yang besar. *Hoverboard* memiliki *encoder* internal yang digunakan untuk memperoleh data *odometry*, sehingga dapat digunakan sebagai penggerak motor untuk robot beroda. Dalam penelitian ini, *hoverboard* digunakan sebagai sistem penggerak menggunakan motor BLDC dan *motherboard*.

2.2 Differential Drive



Gambar 1. Gerak Kinematika Differential Drive

Dalam jurnalnya (Dudek dan Jenkin 2001) menjelaskan bahwa gerak *differential* adalah sistem penggerak dua roda dengan aktuator independen untuk setiap roda Agar robot dapat berputar, ia harus berputar di sekitar titik di sepanjang sumbu kiri dan kanan. Pada *Gambar 1* menunjukkan diagram gerak penggerak diferensial yang memberikan rumus:

$$R = \left(\frac{L}{2}\frac{vr + vl}{vr - vl}\right) \quad (1)$$

$$\omega = \left(\frac{vr - vl}{l}\right) \quad (2)$$

Dari persamaan di atas, dapat diturunkan beberapa hal:

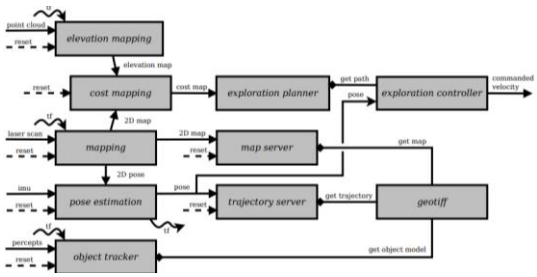
1. Ketika $vl = vr$, maka didapatkan gerak maju linier..
2. Ketika $vl = -vr$, maka didapatkan rotasi atau berputar di tempat (putaran 360^o).
3. Ketika $vl = 0$, maka didapatkan gerak poros ke kiri dikarenakan $R = \frac{l}{2}$, kebalikannya berlaku untuk roda sebelah kanan ketika vr = 0.

vl dan vr adalah nilai kecepatan pada roda yang dapat dikontrol untuk mendapatkan gerak rotasi yang dibutuhkan. l adalah jarak pusat antara kedua roda. Di sisi lain, R adalah nilai $l/2$ yang merupakan jarak antara pusat.

2.2.1 *Simultaneous Localization and Mapping* (SLAM)

SLAM adalah cara terbaru dalam memetakan lingkungan ke data di peta. SLAM melakukan eksplorasi dan pemetaan lingkungan yang tidak diketahui dan menggunakan sensor yang terpasang pada robot untuk memperkirakan sikap robot itu sendiri (Pyo, dkk. 2017). Untuk melakukan pemetaan, posisi robot terbaru harus diperbarui secara bersamaan. Sementara itu, Anda akan membutuhkan data peta untuk hasil pemetaan untuk memperbarui lokasi. Istilah SLAM digunakan untuk masalah pelokalan saat membuat peta.

2.2.1.1 Hector SLAM



Gambar 2. Skema Sistem Hector SLAM
Sumber: (Kohlbrecher, dkk. 2014)

Proses pembuatan peta lokal dengan menghubungkan peta dengan perkiraan posisi lingkungan sekitar yang tidak diketahui merupakan bagian dari satu set paket ROS Hector SLAM (Kohlbrecher, dkk. 2014). Dalam penelitian ini algoritme Hector SLAM menjadi proses pembuatan peta gudang lokal. Pemetaan menggunakan laser lidar frekuensi tinggi tanpa informasi *odometry*. Pemetaan Hector menggunakan algoritma pencocokan pemindaian yang dikembangkan dengan menyelaraskan data dari sensor untuk membuat peta. Pada Gambar 2 menunjukkan proses pemetaan Hector, Geotiff Hector, dan bentuk lain dari skema SLAM Hector dan alur proses terkaitnya.

2.2.1.2 Hector *Mapping*

Pemetaan Hector adalah algoritma SLAM yang membuat peta *grid* yang ditempati tanpa perlu menggunakan *odometry* dalam metode pencocokan pemindaian. Metode pencocokan pemindai untuk pemetaan Hector didasarkan pada pendekatan *Gauss-Newton* (Kohlbrecher, dkk. 2014). Untuk menstabilkan pemindaian laser dapat digunakan penggabungan sistem optimasi pose menggunakan unit *pitch* atau *roll*. Proses stabilisasi sistem memungkinkan untuk memetakan lingkungan bahkan di *surface* yang tidak beraturan. Terdapat beberapa persamaan yang menjadi rumus utama Hector *Mapping*. Persamaan 3 digunakan untuk melakukan pergeseran posisi robot dan Persamaan 4 merupakan pencarian nilai H yang dibutuhkan pada Persamaan 3.

$$\Delta\xi = H^{-1} \sum_{i=1}^{r} \left[\nabla M(S_i(\xi)) \frac{\partial S_i(\xi)}{\partial \xi}\right]^T [1 - M(S_i(\xi))] \quad (3)$$

$$H = \sum_{i=1}^{r} \left[\nabla M(S_i(\xi)) \frac{\partial S_i(\xi)}{\partial \xi}\right]^T \left[\nabla M(S_i(\xi)) \frac{\partial S_i(\xi)}{\partial \xi}\right] \quad (4)$$

$\Delta\xi$ merupakan nilai perpindahan posisi robot dalam koordinat *cartesian* dan $S_i(\xi)$ merupakan koordinat hasil pemindaian lidar. Nilai M merupakan probabilitas adanya halangan pada koordinat hasil pemindaian.

2.2.2 Perencanaan Jalur Global

Menurut (Pyo, dkk. 2017) dalam bukunya Navfn merupakan algoritme pengoptimalan algoritma *Dijkstra*. Fitur navigasi untuk merencanakan basis gerak atau perencanaan jalur disediakan pada Navfn *package* ROS *navigation.* Algoritma optimasi *Dijkstra*, Navfn, memiliki kesamaan dalam hal pencarian tetangga. Namun Navfn memiliki optimalisasi dalam pencarian tetangga dengan mengunjungi tetangga yang memiliki garis tujuan terdekat dengan *goal*, sedangkan tetangga yang sangat jauh dari garis tujuan akan di abaikan. Navfn juga menggunakan sistem pengukuran *grid* berdasarkan warna sehingga dapat memudahkan *backtrack* dari *goal* menuju titik tujuan.

2.2.3 Perencanaan Jalur Lokal

Di dalam pemrograman ROS terdapat paket *library* perencanaan lokal yang sangat efektif yaitu perencanaan lokal *Dynamic* Window *Approach* (DWA). Perencanaan jalur ini memiliki kinerja yang sangat baik karena mampu menginstruksi perencanaan global secara simultan untuk terus membuat jalur baru ketika jalur global utama tertutup halangan (Pyo, dkk. 2017). DWA menggunakan perhitungan percepatan translasi dan rotasi. Perhitungan ini digunakan untuk mengestimasi kemungkinan tubrukan saat navigasi dengan memaksimalkan fungsi objektif kecepatan dan pertimbangan arah gerak robot.

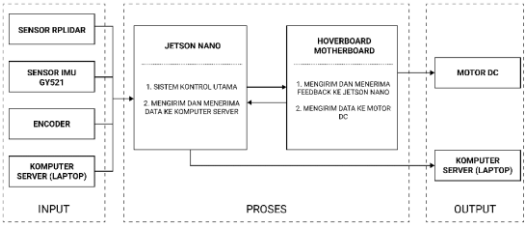
2.3 *Robot Operating System* (ROS)

ROS mengembangkan, mengelola, dan mengirimkan aplikasi paket untuk berbagai tujuan. Tujuan utamanya adalah untuk melakukan layanan abstraksi perangkat keras, manajemen paket, *sharing* pesan topik, dan kontrol perangkat *low level*. ROS dapat menjalankan beberapa paket dan sensor secara bersamaan, sehingga sistem terlihat paralel. ROS mengkhususkan diri dalam mendukung perpustakaan, aplikasi pihak ketiga, dan komunitas yang sangat aktif. Inilah alasan utama mengapa penelitian menggunakan ROS sebagai kerangka sistem utama.

3. PERANCANGAN DAN IMPLEMENTASI

3.1 Gambaran Umum Sistem

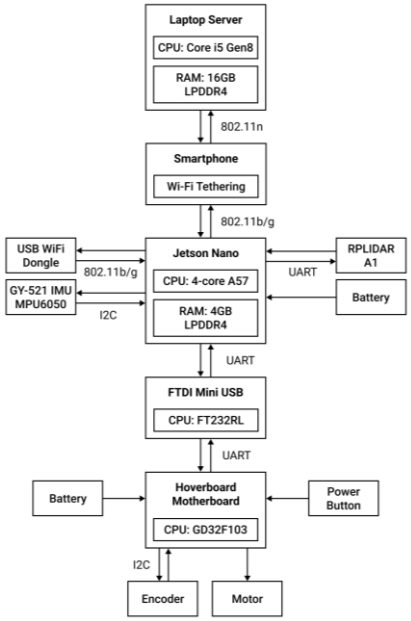
Fungsi pengangkutan barang dari titik awal menuju titik tujuan, bergerak dengan otonom, dan mampu melakukan pengindraan lingkungan menjadi tujuan sistem dibuat.



Gambar 3. Blok Diagram Sistem Autonomous Mobile Robot

Dari Gambar 3 menunjukkan diagram blok sistem yang terdiri dari tiga bagian: *input*, proses, dan *output*. *Section input* terdiri dari sensor *encoder*, IMU, dan RPLIDAR. Ketiga sensor ini digunakan sebagai alat pengindraan dari sistem. Hasil sensor RPLIDAR digunakan untuk lokalisasi menggunakan metode AMCL, yang menggabungkan data laser dan peta untuk membuat estimasi posisi.

3.2 Perancangan Perangkat Keras

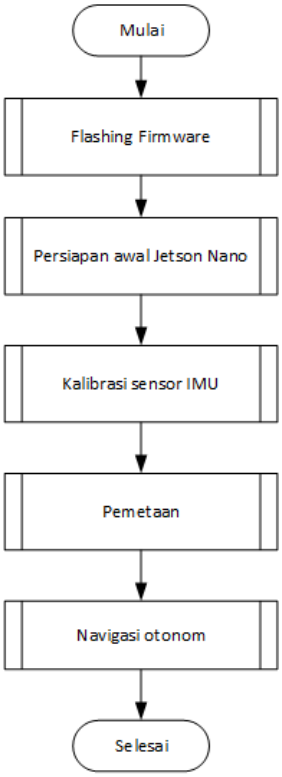


Gambar 4. Rancangan Umum Perangkat Keras

Diagram di atas pada Gambar 4 merupakan tata letak khas perangkat keras sistem. Protokol komunikasi I2C dan UART adalah protokol utama pada rancangan sistem. Pemroses utama yaitu Jetson Nano terhubung langsung *motherboard hoverboard* melalui FTDI melalui protokol UART. *Motherboard hoverboard* bertindak sebagai *driver* motor BLDC.

3.3 Perancangan Perangkat Lunak

Pada Gambar 5 menunjukkan diagram alir proses dari program utama. Seluruh rangkaian program ini sudah berisi persyaratan dan kriteria untuk membuat robot dalam sistem robot bergerak otonom. Hal pertama yang dilakukan pada penelitian ini adalah *flashing firmware*. Proses kedua adalah konfigurasi *Jetson* Nano. Proses ketiga adalah kalibrasi sensor IMU. Proses keempat adalah Pemetaan dan yang terakhir adalah Navigasi otonom. Alur ini tersusun menjadi program utama sistem yang setiap prosesnya memiliki sub bab proses tersendiri.



Gambar 5. Diagram Alir Program Utama

3.4 Implementasi Keseluruhan Sistem

Pada Gambar 6 menunjukkan hasil implementasi alat dan sistem. Sensor RPLIDAR terletak di depan robot, sehingga memudahkan untuk membaca jika ada objek di tengah robot. Robot tidak dapat melihat bagian depan robot ketika RPLIDAR diposisikan berada di atas *chassis* motor. Dengan posisi seperti Gambar 6 setidaknya membuat RPLIDAR dapat membaca lingkungan sebelum gerakan maju robot dan posisi rintangan di belakangnya, sehingga perencanaan rute tidak terhambat.

Gambar 6. Implementasi Keseluruhan Sistem

4. PENGUJIAN DAN ANALISIS

4.1 Pengujian Kinerja Sensor *Encoder*

Pengujian kinerja *odometry encoder* bertujuan untuk mengetahui unjuk kerja motor BLDC pada jumlah putaran yang ditentukan.

Tabel 1 Hasil Pembacaan Sensor Encoder

No	Masukan	*Odometry*		Seharusnya
	m/s	Rad/s	RPM	RPM
1	0,2	2,30384	22	22
2	0,3	3,76992	36	34
3	0,4	4,81712	46	45
4	0,5	6,07376	58	57
5	0,6	7,12896	68	68
6	0,7	8,48232	81	80
7	0,8	9,63424	92	93
8	0,9	10,89888	104	103
9	1	12,0428	115	115
10	1,1	13,19472	126	126
11	1,2	14,6688	140	138
12	1,3	15,788	151	149
13	1,4	16,7552	160	162
14	1,5	18,01184	172	174
15	1,6	19,47792	186	185

Tabel 2 Hasil Akurasi Pembacaan Odometry

Pengukuran Kesalahan	Total	
	Odometry	Seharusnya
	1557	1551
WAPE	0,39%	
Nilai akurasi	99,61%	

Didapatkan hasil akurasi sebesar 99,61% dari hasil pengujian yang dilakukan pada *Tabel 1*. Sensor *encoder* dari motor BLDC dinilai akurat berdasarkan performa skor akurasi pada *Tabel 2*.

4.2 Pengujian Kinerja Sensor IMU

Untuk mengetahui keakuratan pembacaan perubahan percepatan dilakukan pengujian sensor GY-521 untuk mendapatkan data IMU tepatnya pada nilai akselerasi saat robot bergerak. Hasil pengujian kemampuan alat GY-521 dijalankan sebanyak 20 kali pada percepatan statis.

Tabel 3 Hasil Pembacaan Sensor GY-521

No	*m/s*	*Sensor*			*Smartphone*		
		Xi	Yi	Zi	Xj	Yj	Zi
1	0,0	0,097	0,056	0,980	0,2942	0,3334	9,669
2	0,0	0,108	0,029	9,790	0,2942	0,3236	9,660
3	0,0	0,103	0,032	9,803	0,2942	0,3040	9,650
4	0,0	0,053	0,014	9,804	0,3432	0,2746	9,679
5	0,0	0,038	0,022	9,782	0,3432	0,6865	9,532
6	0,3	0,337	0,165	9,612	0,5001	0,1177	9,444
7	0,3	0,046	0,178	9,677	0,4119	1,0493	9,522
8	0,3	0,087	0,007	9,739	0,3334	0,1961	9,601
9	0,3	0,188	0,114	9,909	0,4805	0,1373	10,140
10	0,3	0,296	0,018	9,987	0,2059	0,0883	12,454
11	0,5	0,030	0,108	9,895	0,2550	0,3432	11,474
12	0,5	0,089	0,007	10,167	1,0199	1,3435	10,336
13	0,5	0,048	0,048	10,435	0,9120	0,4021	11,032
14	0,5	0,341	0,077	9,659	1,1866	2,0888	9,003
15	0,5	0,078	0,007	10,002	0,1863	0,2550	9,522

16	2,0	0,244	1,099	11,410	5,7957	2,3830	14,622
17	2,0	0,171	1,418	10,629	0,5394	1,8731	10,424
18	2,0	0,284	1,500	10,822	1,3533	2,0202	13,092
19	2,0	0,672	1,617	10,309	1,2160	1,4122	10,287
20	2,0	0,120	1,492	8,809	5,0014	3,3539	8,247

Hasil pengujian pada *Tabel 3* dilakukan untuk mengetahui penentuan pembacaan accelerometer oleh sensor GY-521 pada percepatan aktual robot untuk mencapai akurasi positioning. Berdasarkan *Tabel 4* didapatkan akurasi sebesar 81,93%.

Tabel 4 Hasil Akurasi Pembacaan Sensor IMU

Pengukuran Kesalahan	Total	
	Sensor	*Smartphone*
	202,658	247,343
WAPE	18,07%	
Nilai akurasi	81,93%	

4.3 Pengujian Kinerja Sensor RPLIDAR

Untuk keakuratan pembacaan jarak dengan data laser *point cloud* dilakukan pengujian pengukuran dengan jarak sebenarnya. Pengujian ini dilakukan untuk mengetahui kinerja dan akurasi perangkat RPLIDAR. Data laser *point cloud* menjadi nilai yang paling penting karena digunakan sebagai *sensing* pemetaan dan pendeteksian area lingkungan berdasarkan jarak.

Tabel 5 Hasil Pembacaan Sensor RPLIDAR

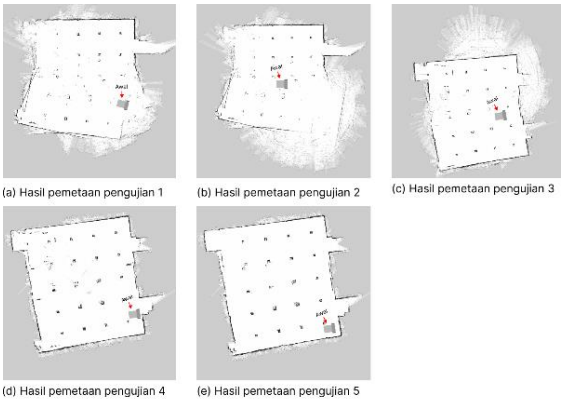
No.	Halangan	Pengukuran		RPLIDAR	
		Sudut	Jarak	Sudut	Jarak
1	Terdekat	180	9,7	180	1
2	Terjauh	0	6000	180	5988
3	Tembok	0	76	0	77
4	Tembok	90	65	90	66,3
5	Tembok	180	75	180	76,4
6	Tembok	270	65	270	66,5
7	Objek 1	180	43	180	43,7
8	Objek 2	0	60	0	61,3
9	Objek 3	135	50	141	51,5
10	Objek 4	315	42	330	43,5

Tabel 6 Hasil Akurasi Pembacaan Sensor RPLIDAR

Pengukuran Kesalahan	Total	
	Total jarak Observasi	Total jarak RPLIDAR
	6485,7	6475,2
WAPE	0,16%	
Nilai akurasi	99,84%	

Hasil pengujian *Tabel 5* dilakukan untuk mengetahui jarak pembacaan dengan jarak sebenarnya. Berdasarkan *Tabel 6* didapatkan akurasi sebesar 99,84%. Dengan hasil akurasi yang diperoleh maka dapat disimpulkan bahwa kinerja dan konfigurasi RPLIDAR yang digunakan pada sistem tergolong akurat.

4.4 Pengujian Pemetaan



Gambar 7 Hasil Pemetaan

Tujuan pengujian pemetaan menggunakan Hector SLAM. Hasil pengujian pemetaan SLAM Hector dijalankan sebanyak 5 kali pada posisi pemetaan awal yang berbeda. Berdasarkan bentuk denah basemen dapat disimpulkan bahwa 4 dari 5 pengujian yang dilakukan sesuai dengan bentuk peta denah *existing*. Didapatkan hasilnya akurasi 80% terdapat pada Persamaan 5.

$$Akurasi = 12/15 \times 100 = 80\% \quad (5)$$

Tabel 7 Hasil Pengukuran Peta Terhadap Denah

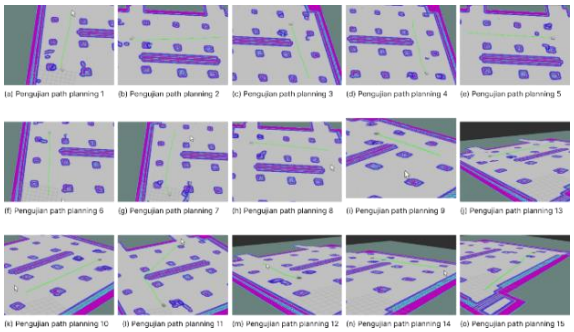
No.	Ukuran Peta		Ukuran Denah Sebenarnya	
	panjang	lebar	panjang	lebar
1	33.519	37.916	31.32	37.1
2	36.277	36.929	31.32	37.1
3	34.842	37.971	31.32	37.1
4	33.191	38.078	31.32	37.1
5	33.697	37.602	31.32	37.1

Hasil pengujian *Tabel 7* dilakukan untuk mengetahui keakuratan pembuatan peta terhadap ukuran peta sebenarnya. Berdasarkan *Tabel 8* didapatkan akurasi sebesar 94,43%. Dari lima peta yang berhasil dibuat, peta kedua gagal karena ukuran tidak sesuai dengan pengukuran denah dan hasil *occupancy grid* menempati posisi yang salah.

Tabel 8 Hasil Akurasi Pengukuran Peta Terhadap Ukuran Denah Basemen Gedung G

	Total	
	panjang	lebar
Total data	360,0252	342,1
WAPE	5,24%	
Nilai akurasi	94,76%	

4.5 Pengujian Navigasi Pada Halangan Diam



Gambar 8 Hasil Pengujian Perencanaan Jalur Global

Pengujian global dijalankan 15 kali di berbagai titik sasaran. Keberhasilan pengujian diwakili oleh kecepatan waktu komputasi algoritma Navfn, waktu tempuh manuver dan akurasi operasi robot saat berjalan seperti yang terlihat pada Gambar 8. Mengetahui keakuratan manuver pergerakan robot dan waktu komputasi menjadi tujuan dari pengujian ini. Pengujian ini juga sebagai penentu apakah sistem atau robot AMR dapat digunakan dan bekerja dengan stabil pada area gudang ekspedisi. Hasil pengujian pada Gambar 8 menunjukkan bahwa secara perencanaan global, robot dapat bergerak dengan baik dan mampu menyelesaikan tugas, yaitu menuju titik tujuan. Namun terdapat 3 kesalahan yang membuat sistem masih belum sempurna dalam bermanuver. Akurasi untuk pengujian perencanaan global mendapatkan nilai 80% yang dapat dilihat pada Persamaan 6.

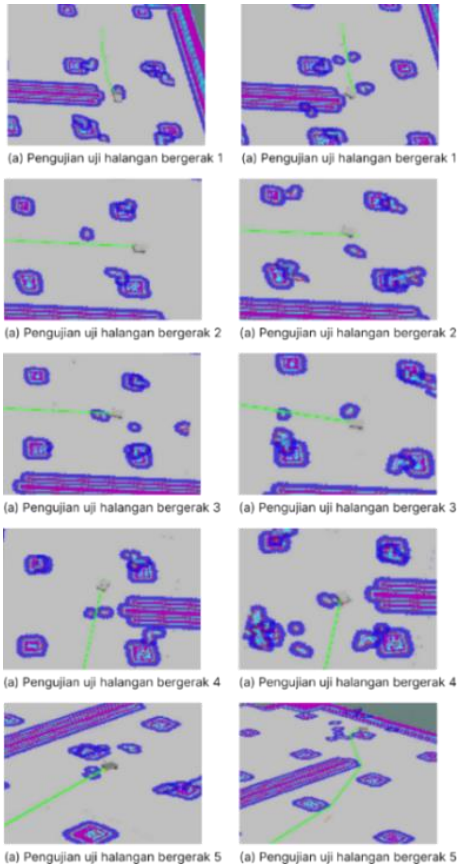
$$Akurasi = 12/15 \times 100 = 80\% \quad (6)$$

Tabel 9 Hasil Uji Waktu Komputasi Navigasi Dengan Algoritme Navfn dan DWA

No	Waktu Komputasi *Path Planner* (detik)	Waktu Tempuh Manuver (detik)	Total Waktu Navigasi (detik)
1	0.37	41.53	41.9
2	0.99	56.11	57.1
3	0.84	40.29	41.13
4	0.41	59.24	59.65
5	0.88	52.08	52.96
6	1.53	52.43	53.96
7	0.63	67.06	67.69
8	1.23	46.61	47.84
9	1.32	37.53	38.85
10	0.92		
11	0.98		
12	0.67	42.35	43.02
13	1.31	54.38	55.69
14	0.45	40.72	41.17
15	1.4		
Rata-rata waktu tempuh (detik)			50.08

Saat menguji perhitungan navigasi dan waktu tempuh yang tercantum dalam *Tabel 9* waktu tempuh rata-rata adalah 50,08 detik. Analisis ini dilakukan dengan menghitung waktu mulai dan titik akhir robot. Anda dapat menghitung selisih selama proses navigasi dengan menggunakan fungsi waktu dari aplikasi *Rviz,* atau lebih tepatnya jam dinding. Dari rata-rata waktu perhitungan ini, kita dapat melihat bahwa algoritma Navfn dan DWA memberikan waktu perhitungan yang cepat. Dengan waktu 1 detik, jalur global sudah dapat dibuat. Cepatnya pencarian jalur global menjadikan Navfn sebagai perencana jalur terbaik untuk lingkungan kerja di gudang atau ruangan.

4.6 Pengujian Navigasi Pada Halangan Bergerak



Gambar 9 Hasil Pengujian Menghindari Halangan Bergerak Dengan Perencanaan Jalur Lokal

Pengujian navigasi lokal dijalankan sebanyak 5 kali di berbagai titik sasaran. Keberhasilan pengujian diwakili oleh kecepatan waktu komputasi, waktu manuver, dan akurasi robot saat berjalan. Gambar 9 menunjukkan bahwa secara perencanaan lokal robot dapat bergerak dengan baik dan mampu menyelesaikan tugas. untuk menentukan jalur

global. Perhitungan sebesar 80% didapatkan pada Persamaan 7.

$$Akurasi = 12/15 \times 100 = 80\% \quad (7)$$

Kesimpulannya, sistem ini bagus dalam mendeteksi rintangan dan cenderung berhenti ketika rintangan di depan bergerak. Namun, jika rintangannya terlalu dekat atau terlalu cepat, Anda mungkin mendapatkan kesalahan perencanaan saat melacak ulang. Berkat *cloud* titik data RPLIDAR, sistem bahkan dapat menghindari pergerakan rintangan di depannya.

5. KESIMPULAN

Pada pengujian sensor *encoder*, IMU, dan RPLIDAR didapatkan masing-masing nilai akurat sebesar 99,61%, 81,93%, dan 99,84%. Keberhasilan ketiga sensor dan perangkat lunak atau *software* yang dirancang dapat dibuktikan dengan ketiga hasil akurasi tersebut. Pada pengujian pemetaan menggunakan algoritme Hector SLAM menghasilkan nilai akurasi sebesar 80%. Didapatkan hasil peta yang cenderung buruk dan kekacauan pada sinkronisasi posisi TF robot ketika sensor RPLIDAR tidak dapat mendeteksi tembok yang lebar di sekitarnya. Lalu didapatkan nilai akurasi sebesar 94,76% dari hasil pengujian pengukuran hasil peta terhadap ukuran denah sebenarnya.

Untuk navigasi global didapatkan hasil akurasi sebenar 80% setelah dilakukan pengujian tanpa halangan menggunakan algoritme Navfn dan DWA. Hasil ini didapat dari pengujian sebanyak 15 kali, didapatkan 3 kegagalan navigasi dari 12 keberhasilan pada pengujian. Dari hasil analisis didapatkan kesimpulan bahwa kegagalan navigasi akibat sensor RPLIDAR tidak mendeteksi adanya tembok yang lebar di sekitarnya. Untuk navigasi lokal didapatkan hasil akurasi sebesar 80% setelah dilakukan pengujian dengan halangan bergerak menggunakan algoritme Navfn dan DWA.

Dari keseluruhan pengujian dan implementasi, terdapat tiga hal yang dapat diperbaiki dan menjadi masukan untuk penelitian selanjutnya. Saran pertama menggunakan sistem kemudi *steering Ackerman* agar kemudi mudah dikendalikan. Kedua menggunakan variasi lebih banyak pengujian pemetaan dan navigasi. Ketiga menambahkan fitur bermanuver ke segala arah menggunakan roda *omni wheel* dengan kombinasi 4WD motor BLDC agar robot dapat bermanuver dengan bebas di dalam gudang yang sempit.

6. DAFTAR PUSTAKA

BPS. 2017. *Laporan Hasil Survei Triwulanan Kegiatan Usaha Terintegrasi 2017*. Jakarta: Badan Pusat Statistik.

Databooks. 2019. “Tren Pengguna E-Commerce Terus Tumbuh.” *Dkatadata.co.id*.

Dudek, Gregory, dan Michael Jenkin. 2001. “Computational Principles of Mobile Robotics.” in *Computational Principles of Mobile Robotics*.

Kohlbrecher, Stefan, Johannes Meyer, Thorsten Graber, Karen Petersen, Uwe Klingauf, dan Oskar Von Stryk. 2014. “Hector open source modules for autonomous mapping and navigation with rescue robots.” *Lecture Notes in Computer Science*.

Parcel Perform. 2019. “Consumers are still not happy with their e-commerce delivery experience, a new survey by Parcel Perform and iPrice Group reveals | Mini Me Insights.” *Minime Insights*. Diambil 12 Maret 2022.

Pyo, YoonSeok, HanCheol Cho, RyuWoon Jung, dan TaeHoon Lim. 2017. *Robot Programming From The Basic Concept To Practical Programming and Robot Application*. First Edit. Seoul, Republic of Korea: ROBOTIS Co., Ltd.

Statista. 2019. *E-commerce in Indonesia - statistics & Facts*.

Utomo, Eko Budi. 2015. “Autonomous Mobile Robot Berbasis Landmark Menggunakan Particle Filter Dan Occupancy Grid Maps Untuk Navigasi, Lokalisasi dan Mapping.”