PENGENDALIAN *DIFFERENTIAL WHEELED MOBILE ROBOT* UNTUK *HUMAN FOLLOWING*

Alfretz Nehemia Mangapul[1], Wahyudi[2*)] dan Hadha Afrisal[3]

[123]Departemen Teknik Elektro, Fakultas Teknik, Universitas Diponegoro, Semarang, Indonesia

*) *E-mail: wahyuditinom@elektro.undip.ac.id*

Abstrak

Mobile robot merupakan salah satu jenis implementasi dari teknologi robotika yang sering dikembangkan karena memungkinkan bergerak secara mudah, salah satunya pada industri perhotelan. Implementasi robot dalam pekerjaan-pekerjaan di hotel memerlukan suatu strategi sehingga penggunaan dan cara kerja robot dalam melakukan pelayanan di hotel dapat optimal. Salah satu strategi lainnnya dalam mengimplementasikan pergerakan robot adalah penggunaan manusia sebagai objek yang diikuti robot. Dengan strategi ini, robot dapat bergerak secara dinamis mengikuti manusia serta menjadi asisten manusia. Penelitian ini membahas mengenai pengendalian *differential wheeled mobile robot* untuk *human following*. Robot menggunakan pengendali PI untuk mengontrol kecepatan motor BLDC pada roda, menggunakan kontrol P untuk mengontrol pergerakan robot, mengimplementasikan pendeteksian manusia dengan metode YOLOv8 untuk membantu gerak robot dalam mengikuti manusia. Hasil pengujian menunjukkan bahwa kecepatan motor BLDC untuk berjalan maju, mundur, berputar kiri dan berputar kanan dapat dikendalikan dengan nilai persentase *error steady state* di bawah 3,41% dan tidak memiliki *overshoot*. Robot masih dapat melakukan pengendalian posisi dengan optimal sampai jarak 2,5 m. Pada skema perjalanan berbentuk persegi 4x4 meter, kecepatan linear rata-rata yang ditempuh robot dalam mengikuti perjalanan maju manusia adalah 0,33 m/s dan kecepatan angular rata-rata yang ditempuh robot dalam mengikuti pergerakan berputar manusia adalah 0,15 rad/s.

Kata kunci: inverse velocity kinematics, motor BLDC, kamera RGBD, kontrol PID

Abstract

Mobile robots are often developed because they can move easily, one of which is in the hotel industry. Implementing robots in hotel jobs requires a strategy so the usage and operation of robots in providing services in hotels can be optimal. Another strategy in implementing robot movement is the use of humans as objects followed by robots. Robots can move dynamically following humans and become human assistants. This research discusses the control of differential wheeled mobile robots for human following. The robot uses a PI controller to control the BLDC motor speed on the wheels, uses P control to control robot movement, implements human detection using the YOLOv8 method to assist robot movement in following humans. The results show that BLDC motor speed for moving forward, backward, turning left and turning right can be controlled with a steady state error percentage value below 3.41% and has no overshoot. Robot can still optimally control position up to a distance of 2.5 m. In a 4x4 meter square pattern traveling scheme, the average linear and angular velocity traveled by the robot in following the human's forward and turning movement are 0.33 m/s and 0.15 rad/s.

Keywords: inverse velocity kinematics, BLDC motor, RGBD camera, PID controller

1. Pendahuluan

Kemajuan teknologi robotika telah merubah lanskap industri, ilmu pengetahuan, dan kehidupan sehari-hari. Teknologi robotika dimanfaatkan untuk mempermudah pekerjaan manusia khususnya jenis pekerjaan yang bersifat high redundance, pola berulang, melelahkan, dan beresiko. Salah satu jenis teknologi robotika tersebut adalah mobile robot. Dalam konteks pengembangan mobile robot, beberapa jenis mobile robot telah muncul untuk membantu pekerjaan manusia. Salah satu pemanfaatan mobile robot yang paling menarik adalah fungsi robot tersebut sebagai asisten dalam industri perhotelan.

Penggunaan mobile robot dalam industri perhotelan telah menjadi tren yang semakin populer. Hotel-hotel modern mengintegrasikan mobile robot ini dalam upaya untuk meningkatkan efisiensi operasional dan memberikan pengalaman yang lebih baik kepada tamu. Kegunaan

mobile robot dalam pelayanan hotel tidak hanya meningkatkan efisiensi, tetapi juga memperkaya pengalaman tamu dengan teknologi yang menarik. Dalam realitanya, mobile robot dapat digunakan dalam bagian-bagian tertentu dari operasional hotel, contohnya, sebagai front desk, housekeeping, dan pramusaji restoran[1]. Robot didesain untuk melakukan pekerjaan tertentu yang bersifat perulangan seperti memberikan informasi, bergerak ke suatu titik, ataupun membawa barang bawaan.

Robot pembawa barang atau logistik sudah banyak diimplementasikan pada industri industri manufaktur ataupun warehouse. Banyak dari robot ini didesain dengan menggunakan sensor sensor dan aktuator yang mampu membaca lingkungan sekitar serta mampu melakukan estimasi posisi robot di dalam lingkungan[2]. Pada bidang hotel, robot pembawa barang sangat mungkin juga diimplementasikan. Ketika tamu atau pengunjung hotel banyak dan jumlah karyawan khususnya bellboy sedikit maka penggunaan robot ini dapat dijadikan salah satu alternatif dalam membantu pelayanan hotel. Untuk membuat robot bergerak secara dinamis, penggunaan suatu objek untuk diikuti pergerakannya bisa menjadi strategi yang relevan. Robot dapat bergerak dengan leluasa mengikuti pergerakan suatu objek di sekitar robot.

Berdasarkan uraian masalah tersebut, maka diajukan pengembangan dan pengimplementasian sistem human following robot. Pada penelitian ini dirancang sistem pengendalian robot untuk melakukan human following menggunakan kinematika differential drive. Robot ini menggunakan sensor Hall effect dan RGBD kamera serta menerapkan metode pendeteksian manusia dengan YOLOv8 sehingga robot dapat mengetahui keberadaan manusia. Sistem pengendalian robot menggunakan kontrol PID untuk menghasilkan pergerakan robot yang stabil. Selain itu, untuk mengangkut barang dengan beban yang berat, aktuator penggerak robot dirancang menggunakan motor BLDC dengan torsi yang cukup besar.

2. Metode

2.1. *Inverse Kinematics Differential Drive*

Differential Wheeled Robot adalah jenis robot bergerak yang gerakannya didasarkan pada dua roda yang bergerak secara terpisah. Gerakan maju dicapai saat kedua roda dikendalikan terpisah dengan kecepatan yang sama, belok kanan dicapai dengan menggerakkan roda kiri dengan kecepatan lebih tinggi dari roda kanan dan sebaliknya untuk belok kiri[3]. *Mobile robot* jenis ini dapat berputar di tempat dengan menggerakkan satu roda ke depan dan roda kedua berlawanan arah dengan kecepatan yang sama. Roda ketiga atau keempat adalah roda kastor yang dibutuhkan untuk stabilitas mobile robot. Persamaan untuk mencari kecepatan sudut masing-masing roda didefinisikan sebagai *inverse velocity kinematics* untuk *differential drive robot*. Hubungan antara kecepatan linear dan kecepatan angular dalam perhitungan *inverse velocity kinematics* dijabarkan pada persamaan 1 dan 2.

$$\dot{\phi}_L = \frac{v_x}{r} - \frac{L}{2}\frac{\omega}{r} \quad (1)$$

$$\dot{\phi}_R = \frac{v_x}{r} + \frac{L}{2}\frac{\omega}{r} \quad (2)$$

dimana

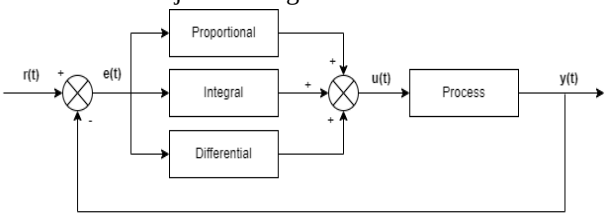
ω : Kecepatan angular robot (rad/s)
v_x : Kecepatan linear robot (m/s)
L : Jarak antar roda robot (m)
r : jari jari roda (m)
$\dot{\phi}_L$: kecepatan angular roda kiri
$\dot{\phi}_R$: kecepatan angular roda kiri

2.2. Kontrol PID Digital

PID digital pada dasarnya mengacu pada jenis perangkat keras digital dimana sistem kontrol PID tersebut ditanamkan. Ekspresi matematis PID digital direpresentasikan pada persamaan 3.

$$u(n) = K_p\left\{e(n) + \frac{1}{T_i}\left(e_{int_{prev}} + \left(\frac{(e_n + e_{n-1})T_s}{2}\right)\right) + \frac{T_d}{T_s}(e_n - e_{n-1})\right\} \quad (3)$$

Gambar 1 menunjukkan diagram blok kontroler PID.



Gambar 1. Diagram blok kontrol PID

Secara umum, terdapat 3 bagian utama dari PID. Proporsional (P) adalah operasi perkalian eror input dengan koefisien gain. Integral (I) merupakan operasi integral dari eror input. Derivatif (D) adalah operasi turunan dari eror input.

2.2.1. Penalaan PID dengan Ziegler-Nichols II

Pada metode Ziegler-Nichols II, penalaan dilakukan dalam kontrol tertutup dimana masukan referensi yang digunakan adalah fungsi tangga (step). Pengendali pada proses penalaan metode ini hanya pengendali proporsional. Nilai kp dinaikkan dari 0 hingga nilai kritis Kp, sehingga diperoleh keluaran yang terus-menerus berosilasi dengan amplitude yang sama. Nilai kritis Kp ini disebut sebagai ultimate gain.

Nilai ultimate period, Tu, diperoleh setelah keluaran sistem mencapai kondisi yang terus-menerus berosilasi. Nilai perioda dasar, Tu, dan penguatan dasar, Ku, digunakan

untuk menentukan konstanta-konstanta pengendali sesuai dengan tetapan empiris Ziegler-Nichols pada Tabel 1.

Tabel 1. Penalaan Ziegler-Nichols II

Kontrol	Kp	Ti	Td
P	$0{,}5 * K_u$	~	0
PI	$0{,}45 * K_u$	$0{,}8 * T_u$	0
PID	$0{,}59 * K_u$	$\frac{T_u}{2}$	$3 * \frac{T_u}{25}$

Nilai parameter Ti adalah konstanta integral dan nilai parameter Td adalah konstanta derivatif.

2.2.2. Penalaan PID dengan *Trial and Error*

Metode *trial and error* merupakan salah satu cara yang digunakan dalam menentukan parameter kontrol PID. Dalam metode *trial and error*, parameter P, I dan D diatur secara berurutan atau bersamaan. Setiap respon sistem terhadap perubahan input atau gangguan harus diamati. Berikut adalah prosedur dalam penalaan metode *trial and error*:

1. Masukan nilai parameter P kecil, kemudian Analisa respon sistem. Jika sistem lambat dan belum bisa mendekati nilai setpoint, naikkan nilai P secara bertahap.
2. Setelah ditemukan nilai P yang membuat sistem merespons dengan baik namun masih ada error steady-state, coba untuk menambahkan komponen I. Mulai dengan nilai I yang rendah dan amati apakah *error steady-state* berkurang. Komponen I membantu dalam mengatasi eror statis sistem.
3. Komponen D membantu mengurangi overshoot (kelebihan respons) dan mempercepat waktu tanggap sistem. Jika dianggap perlu, tambahkan komponen D dengan nilai rendah. Analisa respons sistem terhadap perubahan set point atau gangguan.

2.3. Object Detection dengan YOLOv8

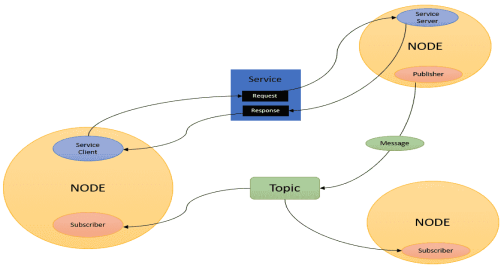
YOLOv8 adalah versi terbaru dari YOLO yang dikembangkan oleh Ultralytics. Ultralytics YOLOv8 mendukung beberapa mode yang dapat digunakan untuk melakukan tugas berbeda. Mode-mode ini antara lain, train, validate, predict, export, track, dan benchmark. YOLOv8 juga merupakan Kerangka kerja AI yang mendukung banyak tugas computer vision. Kerangka kerja ini dapat digunakan untuk melakukan segmentasi, klasifikasi, estimasi pose, dan deteksi[4]. Masing-masing tugas tersebut mempunyai tujuan dan kasus penggunaan yang berbeda. Segmentasi adalah tugas yang melibatkan segmentasi gambar ke dalam wilayah berbeda berdasarkan konten gambar. Setiap wilayah diberi label berdasarkan isinya. Klasifikasi adalah tugas yang melibatkan pengklasifikasian gambar ke dalam kategori yang berbeda. Deteksi pose/titik kunci adalah tugas yang melibatkan pendeteksian titik tertentu dalam bingkai gambar atau video. Titik-titik ini disebut sebagai titik kunci dan digunakan untuk melacak pergerakan atau memperkirakan pose. Gambar 2 menunjukkan implementasi deteksi objek dengan YOLOv8.

Gambar 2. Deteksi objek dengan YOLOv8

Deteksi adalah pendeteksian objek dalam bingkai gambar atau video dan pemberian kotak pembatas atau bounding box di sekelilingnya. Objek yang terdeteksi diklasifikasikan ke dalam kategori berbeda berdasarkan fiturnya.

2.4. Robot Operating System 2

Robot Operating System 2 (ROS2) adalah sumber terbuka untuk sistem *middleware* robotika yang menyediakan kumpulan libraries, tools, dan konvensi sebagai bantuan untuk membangun sistem robotika kompleks. ROS (pendahulu ROS2) telah menjadi platform terkenal untuk penelitian robotika dan juga telah membuktikan landasan yang fleksibel untuk membangun kontrol robot melalui perencanaan tugas[5]. ROS2 bukanlah sebuah operating system, tapi menyediakan layanan yang dirancang untuk menulis aplikasi robotika. ROS2 menyediakan abstraksi perangkat keras, driver perangkat, perpustakaan, visualisator, pengiriman pesan, manajemen paket, dan banyak lagi[6]. ROS2 memiliki ekosistem packages dan stacks sumber terbuka untuk aplikasi robotika seperti pemetaan, navigasi, manipulasi, persepsi, dan simulasi. Framework dan packages yang digunakan adalah ros_control dan realsense2_camera. Framework ros_control memberikan kemampuan untuk mengimplementasikan dan mengelola pengontrol robot dengan fokus pada dua hal yaitu real-time performance dan sharing of controllers dalam hal agnostisme robot[7]. Packages realsense2_camera adalah ROS wrapper yang disediakan oleh intel untuk menggunakan Intel RealSense Depth Camera dengan ROS dan ROS2[8]. Terdapat juga tools canggih untuk visualisasi dan simulasi seperti Rviz dan Gazebo. Gambar 3 menunjukkan konsep komunikasi antar node terjadi[9].



Gambar 3. Pertukaran *message* antar *node* pada ROS

Konsep utama ROS2 meliputi *node*, *message*, *topic*, *publisher*, *subscriber*, *service*, *act*, *bag*. *Node* adalah sebuah program yang dapat dieksekusi serta menjalankan proses untuk melakukan komputasi. Setiap *node* di ROS2 harus bertanggung jawab untuk satu tujuan modular, misalnya mengendalikan motor roda atau mempublikasikan data sensor dari *laser range-finder*[9]. Setiap *node* akan berkomunikasi satu sama lain untuk mengirim pesan atau *messages*. Suatu node berkomunikasi melalui *topic*, *service*, atau *act*.

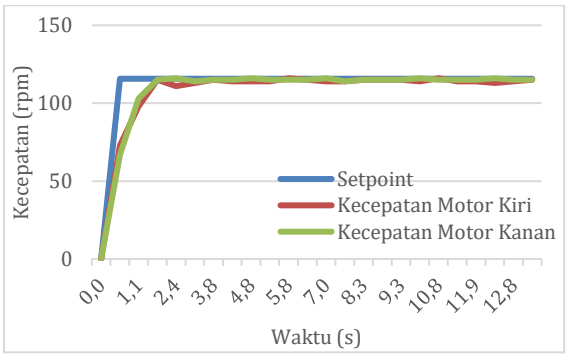
3. Hasil dan Pembahasan

3.1. Pengendalian Kecepatan Robot

Pengujian ini dilakukan untuk mengetahui bahwa kecepatan roda kiri dan kanan dapat beroperasi dengan baik sesuai nilai masukan hasil perhitungan *inverse velocity kinematics*. Pengujian ini juga dilakukan untuk mengetahui kemampuan kontrol PI dalam mencapai setpoint kecepatan angular roda.

3.1.1. Pengujian Berjalan Maju

Pengujian pergerakan robot dalam berjalan maju dilakukan dengan memberikan perintah kecepatan linear sebesar 1 m/s. Nilai kecepatan 1 m/s menghasilkan nilai kecepatan angular roda kiri dan kanan masing masing 115,7 rpm. Nilai kecepatan angular roda 115,7 rpm akan menjadi setpoint untuk tiap roda robot. Gambar 4 menunjukkan respon sistem saat berjalan maju.

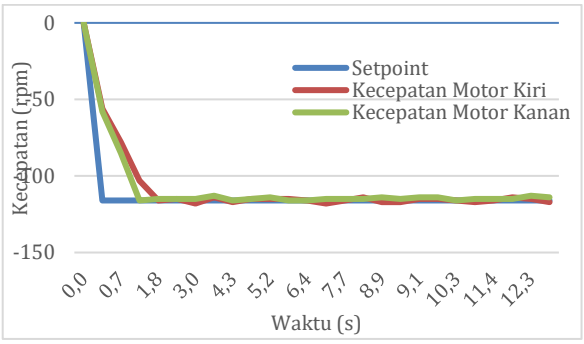


Gambar 4. Grafik respon sistem roda kiri dan kanan saat berjalan maju 1 m/s

Nilai *rise time* pada roda kiri dan kanan adalah 1,7 detik. Nilai puncak maksimum yang dihasilkan adalah 0%. Persentase *error steady state* pada roda kiri dan kanan masing masing sebesar 1,22% dan 0,48%.

3.1.2. Pengujian Berjalan Mundur

Pengujian pergerakan robot dalam berjalan mundur dilakukan dengan memberikan perintah kecepatan linear sebesar -1 m/s. Nilai kecepatan -1 m/s menghasilkan nilai kecepatan angular roda kiri dan kanan masing masing -115,7 rpm. Nilai kecepatan angular roda -115,7 rpm akan menjadi setpoint untuk tiap roda robot. Gambar 5 menunjukkan respon sistem saat berjalan mundur.



Gambar 5. Grafik respon sistem roda kiri dan kanan saat berjalan mundur -1 m/s

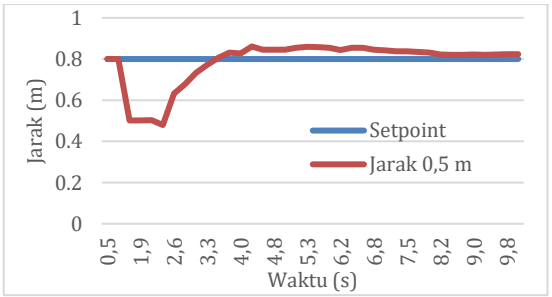
Nilai *rise time* pada roda kiri dan kanan adalah 1,8 detik dan 1,3 detik. Nilai puncak maksimum yang dihasilkan adalah 0%. Persentase error steady state pada roda kiri dan kanan masing masing sebesar 1,06% dan 1,01%.

3.2 Pengujian Konsistensi Pendeteksian Terhadap Manusia

Pengujian ini dilakukan untuk mengetahui bahwa robot dapat terus melakukan pergerakan dan terus mendeteksi target yang dituju. Pengujian ini akan mengambil variasi jarak linear manusia terhadap robot.

3.2.1. Pengujian dengan Jarak 0,5 meter

Pengujian ini memposisikan jarak manusia dengan robot sebesar 0,5 meter. Keberadaan manusia masih bisa terdeteksi oleh robot dan robot merespon pendeteksian dengan bergerak mundur ke belakang. Respon pergerakan robot terhadap manusia ditampilkan pada Gambar 6.



Gambar 6. Hasil pengujian pergerakan robot dengan jarak 0,5 meter.

Berdasarkan data yang dihasilkan, pendeteksian objek pada jarak 0,5 meter konsisten menampilkan bounding box dan robot berhasil bergerak mundur untuk menjauhi manusia. Respon sistem menghasilkan rise time 3,5 detik dengan nilai steady state error sebesar 4,8%. Pada variasi jarak ini, robot dapat melakukan pergerakan dengan stabil serta melakukan pengendalian untuk mengurangi eror.

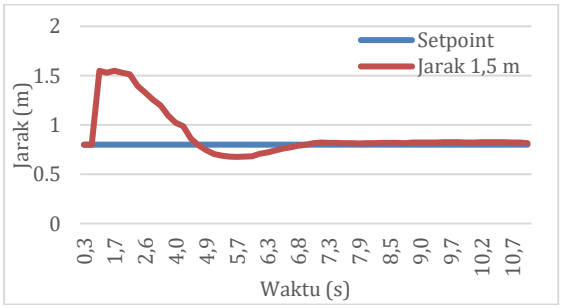
Gambar 7 menunjukkan hasil pendeteksian manusia pada jarak 0,5 meter.

Gambar 7. Hasil pendeteksian manusia pada jarak 0,5 meter.

Lokalisasi manusia dilakukan dengan *bounding box* yang melekat sesuai posisi manusia.

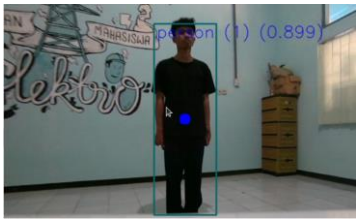
3.2.2 Pengujian dengan Jarak 1,5 meter

Pengujian ini memvariasikan jarak manusia dengan robot sebesar 1,5 meter. Keberadaan manusia masih bisa terdeteksi oleh robot dan robot merespon pendeteksian dengan bergerak maju ke depan mendekati objek. Respon pergerakan robot terhadap manusia ditampilkan pada Gambar 8.



Gambar 8. Hasil pengujian pergerakan robot dengan jarak 1,5 meter.

Berdasarkan data yang dihasilkan, pendeteksian objek pada jarak 1,5 meter konsisten menampilkan bounding box dan robot berhasil bergerak maju untuk mendekati manusia. Hal ini dikarenakan setpoint jarak linear robot dengan manusia adalah 0,8 meter. Respon sistem menghasilkan rise time 4,7 detik dengan steady state error sebesar 2,44%. Pada variasi jarak ini, robot dapat melakukan pergerakan dengan stabil serta melakukan pengendalian untuk mengurangi eror. Gambar 9 menunjukkan hasil pendeteksian manusia pada jarak 1,5 meter.

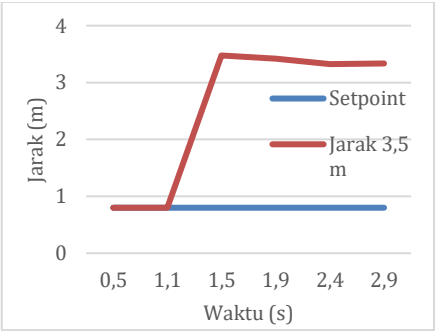


Gambar 9. Hasil pendeteksian manusia pada jarak 1,5 meter.

Lokalisasi manusia dilakukan dengan *bounding box* yang melekat sesuai posisi manusia.

3.2.3 Pengujian dengan Jarak 3,5 meter

Pengujian ini memvariasikan jarak manusia dengan robot sebesar 3,5 meter. Respon pergerakan robot terhadap manusia ditampilkan pada Gambar 10.



Gambar 10. Hasil pengujian pergerakan robot dengan jarak 3,5 meter.

Berdasarkan data yang dihasilkan, pendeteksian objek pada jarak 3,5 meter tidak konsisten menampilkan bounding box dan robot tidak berhasil bergerak maju untuk mendekati manusia. Hal ini dikarenakan penomoran target awal berubah menjadi nomor 4 pada sistem. Respon sistem tidak memiliki rise time dan steady state error sistem tidak diketahui. Pada jarak variasi ini, robot tidak dapat melakukan pergerakan dengan stabil serta tidak melakukan pengendalian untuk mengurangi eror. Gambar 11 menunjukkan hasil pendeteksian manusia pada jarak 3,5 meter.

Gambar 11. Hasil pendeteksian manusia pada jarak 3,5 meter.

4. Kesimpulan

Hasil pengujian kecepatan masing-masing roda pada robot dengan metode *inverse velocity kinematics* menunjukkan bahwa pergerakan robot maju, mundur, berputar ke kiri dan ke kanan memiliki persentase error steady state dibawah 3,41%. Metode inverse velocity kinematics telah berhasil diimplementasikan dan sistem pengendali PI berhasil untuk mengendalikan kecepatan robot. Hasil pengujian konsistensi pendeteksian terhadap manusia menunjukkan bahwa variasi jarak yang bervariasi dapat tetap membuat robot konsisten melakukan pergerakan ke arah manusia, dan variasi 3,5 meter tidak berhasil memiliki pengendalian karena penomoran objek berubah dari 1 menjadi 2. Sistem pendeteksian dan penjajakan manusia hanya dapat bekerja secara optimal pada jarak 0,5 meter, sampai 2,5 meter. Perancangan pengendalian *human following* telah berhasil diimplementasikan dan sistem pengendali P telah berhasil untuk mengendalikan posisi robot.

Referensi

[1] F. Binesh and S. Baloglu, "Are we ready for hotel robots after the pandemic? A profile analysis," *Comput. Human Behav.*, vol. 147, no. June, p. 107854, 2023, doi: 10.1016/j.chb.2023.107854.

[2] F. McLeay, V. S. Osburg, V. Yoganathan, and A. Patterson, "Replaced by a Robot: Service Implications in the Age of the Machine," *J. Serv. Res.*, vol. 24, no. 1, pp. 104–121, 2021, doi: 10.1177/1094670520933354.

[3] B. Sandeep Kumar Malu, J. Majumdar, S. Kumar Malu α, and J. Majumdar σ, "Kinematics, Localization and Control of Differential Drive Mobile Robot Global Journal of Researches in Engineering: H Kinematics, Localization and Control of Differential Drive Mobile Robot," *Type Double Blind Peer Rev. Int. Res. J. Publ. Glob. Journals Inc*, vol. 14, no. 1, 2014.

[4] A. glenn-jocher, sergiuwaxmann, "Ultralytics YOLOv8 Docs: Tasks." https://docs.ultralytics.com/tasks/.

[5] S. Bernardini, M. Fox, and D. Long, "Planning the behaviour of low-cost quadcopters for surveillance missions," *Proc. Int. Conf. Autom. Plan. Sched. ICAPS*, vol. 2014-Janua, no. January, pp. 445–453, 2014, doi: 10.1609/icaps.v24i1.13670.

[6] H. Lee, J. Yoon, M. S. Jang, and K. J. Park, "A robot operating system framework for secure uav communications," *Sensors (Switzerland)*, vol. 21, no. 4, pp. 1–20, 2021, doi: 10.3390/s21041369.

[7] S. Chitta *et al.*, "ros_control: A generic and simple control framework for ROS," *J. Open Source Softw.*, vol. 2, no. 20, p. 456, 2017, doi: 10.21105/joss.00456.

[8] IntelRealsense, "ROS & ROS2," [Online]. Available: https://dev.intelrealsense.com/docs/ros-wrapper.

[9] Open Robotics, "ROS2 Foxy Tutorials: Understanding nodes." https://docs.ros.org/en/foxy/Tutorials/Beginner-CLI-Tools/Understanding-ROS2-Nodes/Understanding-ROS2-Nodes.html.