Pengembangan Kontrol Robot Mobil dengan Kerangka Kerja Robot Operating System

Wea Aqil Dhimas S[a], Santoso[b], Kukuh Setyadjit[c], Ratna Hartayu[d]

[a,b,c,d]*Department of Electrical Engineering, Universitas 17 Agustus 1945 Surabaya, Indonesia*

ARTICLE INFO

Article history:
Received 2 April 2024
Received in revised form
10 May 2024
Accepted 2 June 2024
Available online 31 June 2024

Keywords:
Mobile
Robot
Simulation
Autonomous
Navigation

ABSTRACT

This study develops and implements ROS-based programming control to simulate a mobile robot. ROS, as a popular open-source framework in the robotics community, provides tools and libraries that enable the development of complex robotic applications with ease. The main objective of this research is to create an accurate and efficient simulation of a mobile robot capable of operating in indoor environments. The design process involves integrating various sensors and actuators, as well as implementing algorithms for autonomous navigation. Simulation results demonstrate that the developed system can navigate autonomously and respond to real-time environmental changes. This research makes a significant contribution to the field of robotics, particularly in the development of mobile robot systems applicable to various industrial and research applications.

1 Pendahuluan

Dalam beberapa tahun terakhir, bidang robotika telah mengalami kemajuan signifikan berkat pengembangan dan implementasi sistem kontrol dan kerangka kerja canggih seperti ROS. ROS, sebuah platform open-source yang banyak diadopsi dalam komunitas robotika, menyediakan beragam alat dan perpustakaan yang memfasilitasi pembuatan aplikasi robotika kompleks. Salah satu aplikasi utama dalam kerangka kerja ini adalah simulasi robot mobil, memainkan peran penting dalam pengujian dan penyempurnaan algoritma sebelum diterapkan di dunia nyata[1], [2], [3].

Berfokus pada pengembangan dan implementasi sistem kontrol berbasis ROS untuk mensimulasikan robot mobil yang mampu beroperasi di lingkungan dalam ruangan. Tujuan utamanya adalah menciptakan lingkungan simulasi akurat dan efisien mendekati kondisi dunia nyata secara dekat. Ini melibatkan integrasi berbagai sensor dan aktuator ke dalam platform robotik serta implementasi algoritma untuk navigasi otonom. Tujuan akhirnya adalah untuk menunjukkan bahwa sistem dikembangkan dapat melakukan navigasi secara mandiri dan responsif terhadap perubahan lingkungan secara real-time[4], [5], [6], [7].

Penelitian ini bertujuan memberikan kontribusi penting dalam memajukan teknologi melalui pengembangan sistem kontrol berbasis ROS. Dengan fokus pada simulasi robotika, penelitian ini mengurangi risiko dan biaya implementasi fisik, serta membuka peluang eksplorasi dan pengujian algoritma baru dengan lebih fleksibel dan efisien. Hasil penelitian diharapkan mengarah pada pengembangan sistem robotik yang lebih adaptif mampu menghadapi tantangan lingkungan kompleks dan dinamis. Selain itu, teknologi yang dikembangkan diharapkan memberikan solusi inovatif dan efektif untuk berbagai aplikasi industri dan penelitian di masa depan[8].

Penelitian ini untuk memvalidasi keakuratan sistem kontrol yang dikembangkan dalam simulasi. Validasi ini melibatkan pengujian sistem dalam berbagai skenario navigasi dan lingkungan simulasi yang berbeda-beda, untuk memastikan bahwa respons sistem terhadap perintah navigasi dan adaptasi terhadap perubahan lingkungan sesuai dengan harapan. Penelitian ini tidak hanya berfokus pada pengembangan teknologi baru, tetapi juga pada pengujian dan verifikasi kinerja teknologi tersebut. Hasil dari validasi ini diharapkan dapat memberikan kontribusi yang berarti dalam mengembangkan sistem robotik lebih handal[9].

2 Studi Literatur

ROS telah menjadi kerangka kerja sangat populer dalam pengembangan aplikasi robotika. ROS menyediakan alat dan pustaka memudahkan pengembangan perangkat lunak robotik dengan fitur seperti komunikasi antar proses, manajemen perangkat keras, dan dukungan untuk simulasi, memungkinkan para peneliti dan pengembang untuk berbagi kode dan hasil penelitian mereka dengan lebih mudah, sehingga mempercepat kemajuan di bidang ini[7].

Penelitian menggunakan ROS mengembangkan sistem navigasi otonom untuk robot mobil. Mereka mengintegrasikan sensor seperti LiDAR, kamera, dan IMU untuk memfasilitasi navigasi akurat dalam lingkungan tidak terstruktur. Hasil penelitian menunjukkan bahwa sistem berbasis ROS dapat beradaptasi dengan perubahan lingkungan secara real-time, meningkatkan kehandalan dan efisiensi navigasi robot[10].

Selain itu, penelitian lain memperkenalkan sistem navigasi berbasis ROS yang disebut Hector SLAM, yang khusus dirancang untuk operasi dalam lingkungan dalam ruangan. Sistem ini menggunakan sensor laser untuk membangun peta 2D dari lingkungan dan telah berhasil diimplementasikan dalam berbagai aplikasi, mulai dari robot penyelamat hingga robot kesetimbangan[11], [12], [13]. Penelitian lain yang mendukung adalah studi tentang algoritma kontrol dan perencanaan jalur. Penelitian oleh LaValle tentang "Planning Algorithms" menyediakan dasar teori yang kuat untuk pengembangan algoritma perencanaan jalur yang efisien untuk robot mobile. Salah satu algoritma yaitu PID, algoritma ini memungkinkan robot untuk menemukan jalur optimal di antara rintangan, merupakan aspek penting dalam navigasi otonom[14], [15], [16].

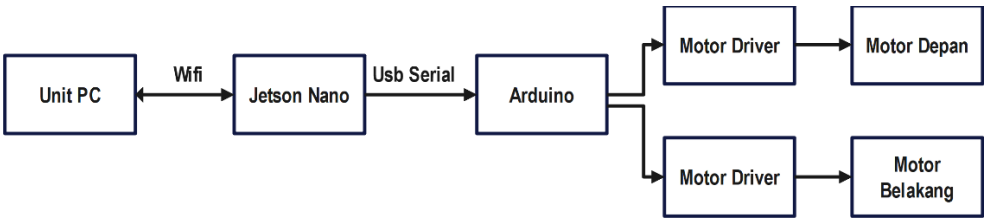
3 Metodologi

Penelitian ini bertujuan untuk mengembangkan dan mengimplementasikan kontrol pemrograman berbasis ROS untuk mensimulasikan robot mobil yang mampu beroperasi di lingkungan dalam ruangan.

Corresponding Author:
Email Addresses: weadhimas@gmail.com

Doi:

Elsains: Jurnal Elektro
p-ISSN: 2527-6336 | e-ISSN: 2656-7075



Gambar 1. Blok diagram sistem robot dengan ROS

Metodologi penelitian ini mencakup beberapa tahap utama, yaitu: perancangan sistem, integrasi perangkat keras dan perangkat lunak, pengembangan algoritma navigasi, simulasi, dan evaluasi kinerja.

3.1 Perancangan Sistem

Tahap awal penelitian ini mencakup perancangan sistem robot mobil yang akan disimulasikan secara mendetail. Desain sistem mencakup spesifikasi teknis robot, serta pemilihan sensor dan aktuator yang sesuai dengan kebutuhan. Komponen utama yang dirancang meliputi pengembangan model 3D dari robot mobil untuk simulasi yang akurat. Selain itu, pemilihan motor dan roda yang optimal untuk memastikan pergerakan robot juga menjadi fokus utama dalam tahap ini.

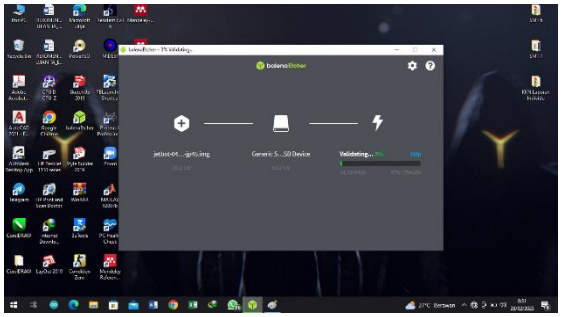
a. Jetson Nano

Untuk mengkondisikan Jetson Nano agar siap digunakan dengan Ubuntu dan ROS, sejumlah langkah terperinci perlu diikuti. Kebutuhan perangkat keras seperti Jetson Nano 4GB Developer Kit, kartu microSD minimal 32GB, power supply 5V 4A, monitor, keyboard, mouse, serta internet atau modul Wi-Fi harus disiapkan terlebih dahulu.

Selanjutnya, JetPack SDK yang mencakup sistem operasi Ubuntu yang dioptimalkan untuk Jetson Nano serta driver dan pustaka yang diperlukan diunduh dari situs web NVIDIA Developer. File image JetPack kemudian diunduh dan dibakar ke kartu microSD menggunakan alat seperti Etcher. Setelah itu, kartu microSD dimasukkan ke slot pada Jetson Nano, monitor, keyboard, dan mouse disambungkan, serta daya dinyalakan. Petunjuk di layar diikuti untuk menyelesaikan pengaturan awal Ubuntu[8].

b. Ubuntu untuk ROS

Ubuntu sebagai platform dasar untuk menginstal dan menjalankan ROS. Salah satu alasan utamanya adalah kompatibilitas dan dukungan. ROS dikembangkan dengan mempertimbangkan Ubuntu sebagai platform utama. Setiap versi ROS umumnya disertai dengan dukungan khusus untuk versi tertentu dari Ubuntu. Misalnya, ROS Noetic didukung di Ubuntu 20.04 LTS. Selain itu, Ubuntu memiliki versi LTS (Long-Term Support) yang didukung selama lima tahun, memberikan stabilitas dan pembaruan keamanan yang penting untuk aplikasi robotik.



Gambar 2. Pengaturan Jetson Nano

Instalasi Jetson Nano melibatkan memasang kartu microSD yang telah dipersiapkan dengan sistem operasi, menghubungkan monitor, keyboard, dan mouse, serta menyambungkan daya melalui adaptor DC. Setelah itu, perangkat dapat diaktifkan dan konfigurasi awal sistem dilakukan melalui antarmuka grafis atau SSH, termasuk pengaturan jaringan, instalasi perangkat lunak tambahan, dan penyesuaian pengaturan sesuai kebutuhan aplikasi yang diinginkan, sehingga Jetson Nano siap digunakan untuk pengembangan dan aplikasi AI atau pengolahan gambar.

Gambar 3. Pengaturan Ubuntu

c. Instalasi ROS Melodic

ROS Melodic adalah versi yang sesuai untuk Ubuntu 18.04 digunakan oleh Jetson Nano dengan JetPack. Untuk memulai, buka terminal dan tambahkan repositori ROS serta kunci GPG dengan perintah berikut:

```
sudo apt update
sudo apt install curl
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
sudo apt update
```

Kemudian, menginstal paket ROS Melodic yang lengkap:

```
sudo apt install ros-melodic-desktop-full
```

Inisialisasi `rosdep` diperlukan untuk menginstal dependensi sistem untuk paket ROS:

```
sudo rosdep init
rosdep update
```

Menambahkan ROS ke environment bash sehingga di-load setiap kali terminal dibuka:

```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Menginstal alat-alat yang diperlukan untuk membangun paket ROS:

Corresponding Author:
Email Addresses: weadhimas@gmail.com
Doi:

sudo apt install python-rosinstall python-rosinstall-generator python-wstool build-essential

Membuat direktori workspace dan inisialisasi dengan catkin:

mkdir -p ~/catkin_ws/src
cd ~/catkin_ws/
catkin_make

Menaambahkan workspace ke file bashrc agar environment workspace di-load setiap kali terminal dibuka:

echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
source ~/.bashrc

Untuk memastikan ROS berfungsi dengan benar, menjalankan `roscore`:

roscore

Membuat dan bangun paket sederhana untuk memastikan workspace bekerja:

cd ~/catkin_ws/src
catkin_create_pkg my_first_package std_msgs rospy roscpp
cd ~/catkin_ws
catkin_make

Mengubah mode daya untuk meningkatkan kinerja:

sudo nvpmodel -m 0
sudo jetson_clocks

Menginstal pustaka AI seperti TensorFlow untuk memanfaatkan GPU Jetson Nano:

sudo apt install python3-pip
pip3 install tensorflow

Jetson Nano siap digunakan dengan Ubuntu dan ROS, memungkinkan untuk mengembangkan aplikasi robotik dan AI yang kompleks.

3.2 Integrasi Perangkat Keras dan Perangkat Lunak

Setelah perancangan sistem, langkah berikutnya adalah integrasi perangkat keras dan perangkat lunak menggunakan ROS. Tahapan ini meliputi instalasi dan konfigurasi ROS pada sistem komputer yang digunakan untuk simulasi. Penggunaan pustaka ROS untuk mengintegrasikan sensor dan aktuator ke dalam sistem ROS. Pembuatan node-node ROS yang bertugas untuk mengontrol sensor, aktuator, dan komunikasi antar node.

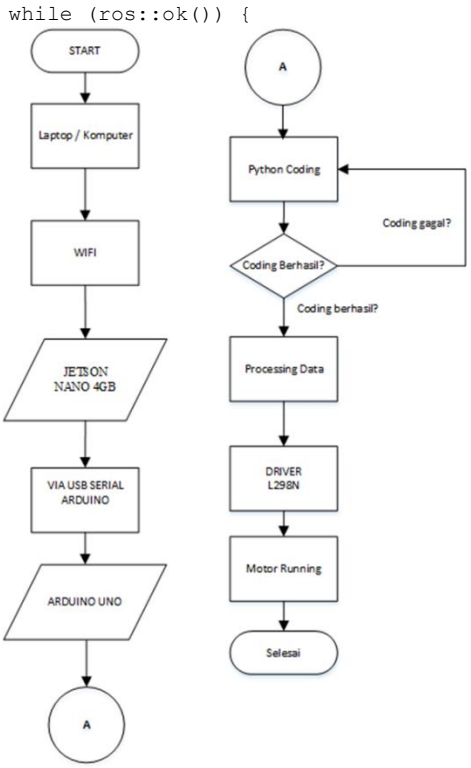
3.3 Pengembangan Algoritma Navigasi

Tahap ini melibatkan pengembangan dan implementasi algoritma navigasi otonom untuk robot mobil, termasuk perencanaan jalur untuk menentukan rute optimal di lingkungan dinamis [1]. Algoritma ini juga mengembangkan metode deteksi dan menghindari rintangan sepanjang jalur yang direncanakan dengan memanfaatkan data dari sensor kamera dan sensor lainnya..

```
include <ros/ros.h>
include <geometry_msgs/Twist.h>

int main(int argc, char argv) {
    ros::init(argc, argv,
"simple_mobile_robot");
    ros::NodeHandle nh;
    ros::Publisher vel_pub =
nh.advertise<geometry_msgs::Twist>("/cmd_ve
l", 10);

    ros::Rate rate(10);
```



Gambar 4. Diagram alir integrasi perangkat keras dan lunak

```
        geometry_msgs::Twist vel_msg;
        vel_msg.linear.x = 0.5;
        vel_msg.angular.z = 0.0;

        vel_pub.publish(      vel_msg);

        rate.sleep();
    }

    return 0;
}
```

Kode C++ tersebut adalah implementasi node ROS yang mengontrol simulasi robot mobil. Node ini menginisialisasi koneksi dengan sistem ROS dan menggunakan publisher untuk mengirimkan perintah kecepatan kepada robot melalui topik "/cmd_vel". Dalam loop utama, node secara terus-menerus mengirimkan pesan `geometry_msgs::Twist` yang memuat kecepatan linear sebesar 0,5 m/s pada sumbu x tanpa kecepatan angular, dengan menggunakan metode `publish` dari objek publisher yang telah dibuat. Untuk menjaga kestabilan frekuensi pengiriman pesan, digunakan objek `ros::Rate` dengan frekuensi 10 Hz. Kode ini dirancang untuk berjalan selama sistem ROS aktif (`ros::ok()` mengembalikan true), dan akan berakhir saat node dihentikan atau program keluar dari loop[9].

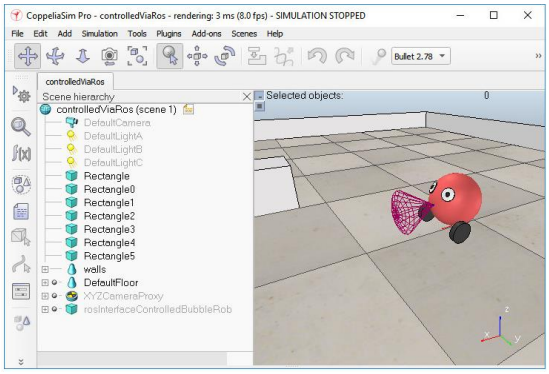
3.4 Simulasi

Simulasi dilakukan menggunakan simulator Coppelia Sim yang kompatibel dengan ROS. Tahapan ini meliputi pembuatan lingkungan dalam ruangan yang akan digunakan untuk menguji algoritma navigasi dan kontroler gerak di lingkungan simulasi. Pengumpulan data dari hasil simulasi untuk analisis lebih lanjut.

Corresponding Author:
Email Addresses: weadhimas@gmail.com

Doi:

Elsains: Jurnal Elektro
p-ISSN: 2527-6336 | e-ISSN: 2656-7075

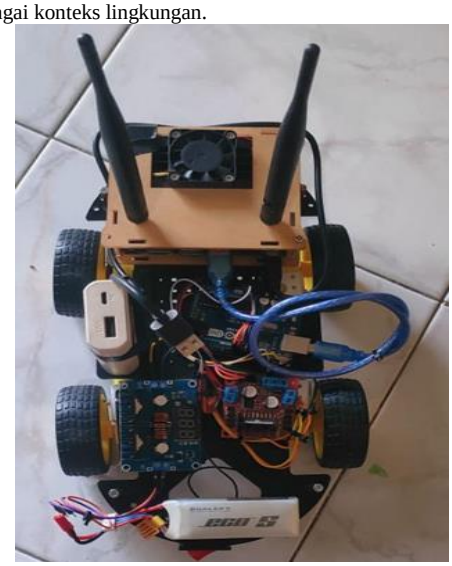


Gambar 5. Simulasi lingkungan robot dengan Coppelia Sim

ROS dan CopeliaSim terintegrasi untuk simulasi robot kompleks. CopeliaSim menyediakan lingkungan simulasi, sementara ROS mengelola kontrol dan data sensor. Integrasi ini memungkinkan pengujian algoritma robotika secara aman dan efisien sebelum diterapkan pada perangkat keras nyata. Proses ini difasilitasi oleh ROS Interface yang memungkinkan komunikasi mulus antara kedua platform, sehingga meningkatkan fleksibilitas, keandalan, dan efektivitas pengujian sistem robotik secara keseluruhan. Integrasi ini sangat bermanfaat bagi pengembangan teknologi robotika.

4 Pembahasan

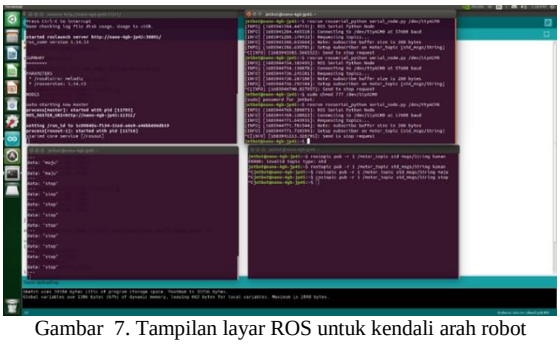
Sistem berhasil diimplementasikan pada robot mobil menggunakan Jetson Nano dan Arduino Uno sebagai kontroler utama. Komunikasi antara laptop sebagai monitoring dengan Jetson Nano melalui Wi-Fi, serta antara Jetson Nano dengan Arduino melalui USB serial, berjalan lancar. Sensor terintegrasi dengan baik, memberikan data yang akurat untuk navigasi otonom. Penggunaan motor DC dengan modul L298N berhasil diuji dalam berbagai kondisi simulasi. Analisis menyeluruh menunjukkan sistem ini dapat menangani tantangan navigasi dalam lingkungan dalam ruangan dengan baik, memvalidasi kehandalan dan kinerja sistem secara keseluruhan. Hasil kajian ini menegaskan bahwa kontrol pemrograman berbasis ROS mampu menggerakkan robot mobil secara otonom dan responsif, mendukung aplikasi yang memerlukan navigasi kompleks dalam berbagai konteks lingkungan.



Gambar 6. Robot mobil dengan ROS

Tabel 1. Langkah pergerakan robot pada keyboard

Arah	Tombol Keyboard	Kecepatan	Arah Roda	Berhasil / Tidak?
Maju	I	100	Semua Roda Bergerak Searah Jarum Jam	Berhasil
Mundur	M	100	Semua Roda Bergerak Berlawanan Jaruh Jam	Berhasil
Kanan	L	100	2 Roda Kiri Bergerak Searah Jarum Jam, 2 Roda Kanan Tidak Bergerak	Berhasil
Kiri	J	100	2 Roda Kanan Bergerak Searah Jarum Jam, 2 Roda Kiri Tidak Bergerak	Berhasil
Stop	K	100	4 Roda Semuanya Berhenti	Berhasil



Gambar 7. Tampilan layar ROS untuk kendali arah robot

Langkah pertama adalah membuka terminal baru dan menjalankan perintah-perintah tertentu yang memungkinkan pengguna mengendalikan robot langsung melalui tombol-tombol pada keyboard komputer. Ini melibatkan menjalankan program atau skrip yang telah diprogram untuk menerima input dari keyboard dan mengubahnya menjadi pergerakan atau tindakan yang diinginkan oleh robot, seperti maju, mundur, berbelok, atau berhenti.

5 Kesimpulan

Penelitian ini menunjukkan bahwa pengembangan sistem kontrol berbasis ROS untuk simulasi robot mobil mampu memberikan solusi yang efektif dalam lingkungan simulasi. Penelitian ini berhasil merancang dan mengimplementasikan algoritma navigasi otonom serta integrasi sensor yang tepat untuk mendukung operasi robot dalam lingkungan dinamis. Hasil simulasi menunjukkan bahwa sistem dapat menghasilkan

Corresponding Author:
Email Addresses: weadhimas@gmail.com

Doi:

navigasi yang responsif dan adaptif terhadap perubahan lingkungan secara real-time. Kontribusi utama dari penelitian ini terletak pada kemampuan untuk menguji dan mengoptimalkan algoritma navigasi tanpa memerlukan risiko dan biaya yang terkait dengan implementasi fisik. Dengan demikian, penelitian ini memberikan lsan yang kuat untuk pengembangan lebih lanjut dalam bidang robotika, khususnya dalam aplikasi industri dan penelitian yang memerlukan robot mobil yang mampu beroperasi secara otonom dalam lingkungan kompleks.

Kontribusi Penulis

Penelitian ini melibatkan kolaborasi dari beberapa penulis: Penulis 1, Penulis 2, Penulis 3, dan Penulis 4. Penulis 1 dan Penulis 2 mengembangkan konsep awal dan tujuan, merancang metodologi, serta mengelola aspek teknis dan perangkat lunak. Penulis 1 juga bertanggung jawab atas analisis data, eksperimen, pengelolaan sumber daya, dan kurasi data. Selain itu, Penulis 1 menulis draf awal, menyempurnakan penulisan, dan membuat visualisasi yang mendukung artikel. Penulis 1 juga memastikan pengawasan dan administrasi proyek berjalan lancar. Penulis 2 fokus pada perolehan dana untuk mendukung proyek ini. Penulis 3 dan Penulis 4 berkontribusi dalam pengumpulan data, analisis tambahan, serta penyempurnaan dan revisi naskah akhir.

Ucapan Terima Kasih

Penulis ingin mengucapkan terima kasih kepada semua pihak yang telah berperan dalam penelitian ini. Kami mengapresiasi kontribusi rekan penulis dalam merumuskan konsep, merancang metodologi, dan mengembangkan perangkat lunak. Terima kasih kepada tim validasi atas uji coba dan evaluasi sistem yang berharga. Kami juga mengucapkan terima kasih atas dukungan sumber daya, administrasi proyek, dan perolehan pendanaan. Kontribusi dan dukungan dari semua pihak sangat berarti bagi kesuksesan penelitian ini.

Referensi

[1] ‘SIMULASI AUTONOMOUS VEHICLE DI UNIVERSITAS KRISTEN SATYA WACANA SALATIGA | Techné : Jurnal Ilmiah Elektroteknika’. Accessed: Jun. 28, 2024. [Online]. Available: https://ojs.jurnaltechne.org/index.php/techne/article/view/93

[2] ‘Modeling and Simulating a Narrow Tilting Car Using Robotics Formalism | IEEE Journals & Magazine | IEEE Xplore’. Accessed: Jun. 28, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6689308?casa_token=JhOK-NbtMsgAAAAA:QkhdmVgyuN8jQFDyzaHabZAfOI_dUou7LlmH9HNzozUqnlRD0h1mLPy80T_VlSgViAxV23s7EKZTf9U

[3] C. Urs, ‘Simulation driven approach to study the feasibility of involving Collaborative Robot for windshield loading process in a car manufacturing plant’, in *2021 IEEE Transportation Electrification Conference (ITEC-India)*, Dec. 2021, pp. 1–4. doi: 10.1109/ITEC-India53713.2021.9932506.

[4] S. Parihar and S. Dutta, ‘Enhancing Line Following Robot Navigating with Smart Assistant Capabilities’, in *2023 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*, Dec. 2023, pp. 1–7. doi: 10.1109/ICSES60034.2023.10465582.

[5] B. Fares, H. Souifi, M. Ghribi, and Y. Bouslimani, ‘Omnidirectional Platform for Autonomous Mobile Industrial Robot’, in *2021 IEEE 3rd Eurasia Conference on IOT, Communication and Engineering (ECICE)*, Oct. 2021, pp. 633–638. doi: 10.1109/ECICE52819.2021.9645621.

[6] M. I. Zulfiqar, I. Abdullah, A. Akram, S. Mehmood, Fareeha, and A. Ali, ‘An Efficient Autonomous Maze Solving Robot based on Microcontroller by Avoiding Obstacles Using a Decision-Making Algorithm’, in *2023 International Conference on IT and Industrial Technologies (ICIT)*, Oct. 2023, pp. 1–6. doi: 10.1109/ICIT59216.2023.10335850.

[7] ‘Mastering ROS for Robotics Programming: Best practices and troubleshooting solutions when working with ROS | Packt Publishing books | IEEE Xplore’. Accessed: Jun. 28, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10162816

[8] Z. Ma, L. Zhu, P. Wang, and Y. Zhao, ‘ROS-Based Multi-Robot System Simulator’, in *2019 Chinese Automation Congress (CAC)*, Nov. 2019, pp. 4228–4232. doi: 10.1109/CAC48633.2019.8996843.

[9] Z. Cheng, B. Li, and B. Liu, ‘Research on Path Planning of Mobile Robot Based on Dynamic Environment’, in *2022 IEEE International Conference on Mechatronics and Automation (ICMA)*, Aug. 2022, pp. 140–145. doi: 10.1109/ICMA54519.2022.9856220.

[10] S. Gatesichapakorn, J. Takamatsu, and M. Ruchanurucks, ‘ROS based Autonomous Mobile Robot Navigation using 2D LiDAR and RGB-D Camera’, in *2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP)*, Jan. 2019, pp. 151–154. doi: 10.1109/ICA-SYMP.2019.8645984.

[11] H.-S. Juang and K.-Y. Lurrr, ‘Design and control of a two-wheel self-balancing robot using the arduino microcontroller board’, in *2013 10th IEEE International Conference on Control and Automation (ICCA)*, IEEE, 2013, pp. 634–639.

[12] R. Hartayu, S. Santoso, A. O. U. Kaleka, and Moh. K. Musakhol, ‘Desain Simulasi Robot Kesetimbangan Dua Roda Dengan Kecerdasan Buatan’, *J. Sains Dan Inform.*, vol. 6, no. 2, pp. 175–182, Dec. 2020, doi: 10.34128/jsi.v6i2.232.

[13] A. Chhotray, M. K. Pradhan, K. K. Pandey, and D. R. Parhi, ‘Kinematic Analysis of a Two-Wheeled Self-Balancing Mobile Robot’, p. 8.

[14] Santoso, I. A. Wardah, P. Slamet, A. H. Andriawan, and A. R. Algopiki, ‘Design and Build Two Wheel Balancing Robot Simulation with Fuzzy PID’, presented at the International Conference on Advanced Engineering and Technology, Jun. 2024, pp. 124–128. Accessed: Jun. 28, 2024. [Online]. Available: https://www.scitepress.org/PublicationsDetail.aspx?ID=KDP9hf247lo=&t=1

[15] S. Santoso and S. Yuliananda, ‘Design and Control Self Balancing Robot’, presented at the Proceedings of the 1st Asian Conference on Humanities, Industry, and Technology for Society, ACHITS 2019, 30-31 July 2019, Surabaya, Indonesia, Sep. 2019. Accessed: Jun. 28, 2024. [Online]. Available: https://eudl.eu/doi/10.4108/eai.30-7-2019.2287765

[16] S. Santoso and S. Mursyid, ‘KONTROL PROPORTIONAL INTEGRAL (PI) PADA ROBOT LINE FOLLOWER’, *J. Sains Dan Inform.*, vol. 1, no. 1, pp. 10–10, Sep. 2017.

Corresponding Author:
Email Addresses: weadhimas@gmail.com

Doi: