International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 04 | April - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Multipurpose Autonomous Navigation Robot (M.A.N-R)

Nirmal Raju Gharpure
Department of Computer Engineering
PVPIT
nirmalgharpure@gmail.com

Sanket Liladhar Bhole
Department of Computer Engineering
PVPIT
sanketbhole127@gmail.com

Tanishq Hemant Shinde
Department of Computer Engineering
PVPIT
tanishq747s@gmail.com

Vrushali Rajesh Mohite
Department of Computer Engineering
PVPIT
vrushalimohite428@gmail.com

Under the guidance of

Prof. G. S. Wayal
Department of Computer Engineering
PVPIT
ganeshw2006@gmail.com

Abstract - The "Multipurpose Navigation Robot" project aims to develop an adaptive robot that can smoothly navigate various real-world environments. Past navigation robot techniques have often been limited to specific environments or tasks, lacking the flexibility needed for diverse operations. While earlier efforts mainly emphasized path planning, our system provides a comprehensive solution by incorporating both path scheduling and task management features.

Key Words*: *Arduino, IR sensors, IOT, Raspberry pi, PWM, PID controller, A-star algorithm.

1. INTRODUCTION

While using robotics in industrial and commercial use the ability to navigate seamlessly and precisely through real-world environments is important. While previous efforts have been made addressing different challenges, there remains a gap in the literature regarding systems capable of navigating through multiple and diverse environments. Many of the previous approaches focus on a particular environment when it comes to service robots like warehouses, hospitals, or restaurants. A robot system that can adapt to multiple environments and can be used easily can increase the involvement of robotics in small and medium businesses. This survey paper presents a comprehensive review of the "Multipurpose Autonomous Navigation Robot". Traditionally, robotics research is mainly focused on optimizing path-planning algorithms or training robots to excel in specific environments or tasks. However, these narrowed approaches often lack the adaptability required for real-world applications where environments are dynamic and tasks are diverse. This project stands apart by providing both path planning and task scheduling functionalities. This survey paper aims to provide insights into the current approaches and technologies related to robotic navigation and opportunities for future research and development.

2. Literature Survey

1. "Intelligent Warehouse Management System", Mohamed Dhouioui; Tarek Frikha

   This paper presents the design and implementation of an intelligent warehouse management system. The main emphasis is on three software components: An Xamarian IOT application that acts as the exchange interface with end users, a web application that communicates with a database and ensures the persistence and communication between the different components of the system, and a robot that moves products in the warehouse according to storage and shipping requests. A frontend application developed with Xamarian IOT and installed in a Raspberry card with Windows IOT operating system, this card is equipped with a touch screen and RFID module. This application is cross-platform and can be installed on Android IOS and Windows 10 tablets. The robot follows the instructions received from the user interface via the MQTT protocol. It locates packets by using RFID tags and moves them according to the instructions. This paper implements a line-following robot. The robot has a Raspberry Pi camera module integrated with it which allows it to track and traverse on the line. Raspberry Pi board plays the role of the robot's command system. Through the data provided by the Pi camera and the RFID modules, the developed robot can operate autonomously. A web application manages communication between the database and the robot. This web application is hosted on

the Azure web jobs. Robot has the task of transporting packages from the cross-docking area to the storage area. This mission is launched by the reception of an MQTT package from the web jobs after the validation of the locations by the cross-docking manager. This package contains a message that tells the robot the location of the package to be transported and the content of its tag. The robot arrives at the packet location in the cross-docking area following a line and is located by the RFID tag placed on the floor of the warehouse. It checks the availability of the package by its tag. The robot notifies the web jobs application and it in turn replies with information about the location of the package in the storage area.

2. "Path Planning of Restaurant Service Robot Based on A-star Algorithms with Updated Weights", Ruijun Yang; Liang Cheng

This paper introduces an improved A-star algorithm for optimal path selection for autonomous navigation robots in restaurants. This uses a real-time gridded restaurant congestion map, aiming at the shortest weighted path, and based on the degree of channel congestion to change the weight of the restaurant channel in real-time. A star algorithm with constant path weight and an improved A star algorithm can effectively avoid crowded channels and improve the mobile efficiency of service robots. While normal traditional navigation efforts use a gridded map this paper presents an improved way to find the more optimal path by considering live congestion in the restaurant channels. This uses a gridded map of the restaurant and implements a gray-scale simulation of channel congestion on the gridded map of the restaurant. The gray-scale value represents the congestion in the channel in the restaurant. The gray level generally ranges from 0 to 255, the white is represented by Lmax=255, and the black is represented by Lmin=0[11-12], the value of the gray scale is inversely proportional to the number of people represented on the map. Hence more crowned areas will be represented as darker and vice versa. This might need input data from sensors to calculate the value of the gray-scale but this is out of the scope of this paper. This paper only focuses on the navigation and calculation part. The A star algorithm is modified to calculate the optimal path based on the gray-scale value. This finds the fastest and most optimal path after considering the crowd in the restaurant.

3. "Improving Autonomous Robotic Navigation Using Imitation Learning", Brian Cesar-Tondreau; Garrette Warnell; Ethan Stump; Kevin Kochersberger; Nicholas R. Waytowich

This paper presents a new model for autonomous navigation in an improved way on the traditional models by integrating a machine learning module in a traditional layered navigation stack of global and local path planning modules. This model can be trained with human demonstrations of the preferred navigation behavior using a training procedure based on (BC)Behavior Cloning. This paper proposes a new framework for combining autonomous navigation and LfD(Learning from Demonstration). A Machine Learning (ML) module is inserted between traditional global and local path-planning modules to enhance autonomous navigation. This ML module takes sensor data and global planner features as input, utilizing lidar data, visual cues from images, goal bearing, and distance to the goal to determine local navigation goals. It predicts waypoints along a semicircle in front of the robot, which is then passed to the local planner. Training involves a two-phase process: first, pre-training the initial navigation policy using behavioral cloning on robot-generated trajectories, and then refining it with human demonstrations to incorporate specific navigation preferences or rules. Trajectories for training are collected either by observing the robot's state and selecting waypoints from the global plan or through human teleoperation, recording actions adhering to predefined rules. This approach iteratively improves the navigation policy, integrating ML with traditional methods to optimize navigation based on both environmental constraints and human preferences.

4. "Research and Implementation of Autonomous Navigation for Mobile Robots Based on SLAM Algorithm under ROS ", Jianwei Zhao; Shengyi Liu; Jinyu Li

This paper aims at the problem of low mapping accuracy, slow path planning efficiency, and high radar frequency requirements in the process of mobile robot mapping and navigation in an indoor environment. This paper proposes a four-wheel drive adaptive robot positioning and navigation system based on ROS. By comparing and analyzing the mapping effect of various 2D-SLAM algorithms like Gmapping, Karto SLAM, and Hector SLAM. The Karto SLAM algorithm is used for map building. By comparing the Dijkstra algorithm with the A* algorithm, the A* algorithm is used for heuristic searches, which improves the efficiency of path planning. The DWA algorithm is used for local path planning, and real-time path planning is carried out by combining sensor data, which has a good obstacle avoidance performance. The mathematical model of four-wheel adaptive robot sliding steering was established, and the URDF model of the mobile robot was established under a ROS system. The map environment was built in Gazebo, and the simulation experiment was carried out by integrating lidar and odometer data, so as to realize the functions of mobile robot scanning mapping and autonomous obstacle avoidance navigation.

5. "Development of a wireless communication platform for multiple-mobile robots using ROS ", Pipit Anggraeni; Mariem Mrabet; Michael Defoort; Mohamed Djemai

The paper discusses the development of a wireless communication platform for coordinating multiple mobile robots using the Robot Operating System (ROS). The authors present the MiniLab Enova mobile robot as the focal point and detail the implementation of a multi-master system using ROS to manage a wireless communication network among multiple robots. They emphasize the

importance of decentralized control architectures in modern manufacturing facilities and propose a solution using ROS's multi-master system to enable communication between robots while avoiding namespace conflicts. The paper outlines the hardware components of the MiniLab Enova robot, the principles of ROS, and the technical steps for configuring the decentralized communication architecture. Experimental results demonstrate the efficacy of the proposed framework for coordinating multiple robots in achieving consensus. Overall, the paper offers insights into leveraging ROS for efficient wireless communication and coordination in multi-robot systems.

6. "ZigBee based Small-World Home Area Networking for Decentralized Monitoring and Control of Smart Appliances", Rakesh Das; Jitendra Nath Bera

The paper presents a novel approach to decentralized monitoring and control of smart electric appliances within homes, utilizing ZigBee mesh topology for home area networking (HAN). All communication nodes within the household are interconnected via ZigBee, with the operational status of individual smart appliances displayed on an in-home display (IHD) or smartphone via Wi-Fi internet connectivity. To minimize ZigBee nodes, groups of smart appliances are linked to a single ZigBee module through a smart controller board housed inside the local switchboard. This design reduces the number of controller units required, equal to the maximum number of switchboards in the house. A Small World Mesh topology for ZigBee-based HAN ensures communication redundancy, allowing uninterrupted operation even if a ZigBee node fails. The proposed ZigBee Controller boards, IHD, and ZigBee to Wi-Fi gateway establish a robust network with reduced transmission delay. The ZigBee network operates based on the principles of small-world network theory, emphasizing efficient communication paths and decentralized control. Through mesh networking, decentralized monitoring and control are facilitated, ensuring reliable operation and quick response times. ZigBee's low power consumption, high noise immunity, and reliability make it a preferred choice for HAN communication. This approach contributes to the realization of energy-efficient, green homes by enabling responsive and intelligent management of electrical appliances.

3. Comparison Study

1. Navigation

a. Line Follower

In this navigation technique, the robot follows a black path instructed by the user. Conventionally, while implementing line-follower robots IR(Infrared) sensors are used for detecting the path. IR sensors are good for the detection of lines from a minimum of 100cm to a maximum of 500cm, low power, and fit in small spaces[1]. IR sensors are used to detect the properties of surroundings. It detects by either transmitting or detecting infrared radiation. As the light is incident on a dark surface less light is reflected than in the case of a white surface. This principle is used for detecting black-path which is followed by the robot. When light is reflected from the black path, IR sensors detect the intensity difference between the black path and surroundings. This is used to send a signal to the processing unit which then controls robot movement accordingly[1]. Line follower robots are simple to design and build but they are restrictive in terms of freedom of navigation. The robot can only follow the predefined black path. If there is an obstacle on the path then the robot can not avoid it hence line-follower robots do not have obstacle avoidance.

b. Visual SLAM

Visual SLAM is a technique for building maps of the environment and localizing a device within them. Visual SLAM relies on visual information to build a map of the environment. Visual SLAM works on the principle of feature extraction[8]. Typically cameras are used to gather visual data from the environment. Visual SLAM provides a 3D map of the environment costing less resources compared to other sensors like Lidar. Using the camera for capturing visual data consumes less power, is less costly, and can be easily integrated with a range of devices. While Visual SLAM is a cost-effective technique it fails to provide accuracy in low-light conditions. Also, Visual SLAM may prove to be less accurate while providing depth measurements. Hence Visual SLAM provides a good choice for cost-effective implementations and is suitable for certain environments[8].

c. LiDAR SLAM

LIDAR is an acronym for Light Detection and Ranging. It is a virtual perception device utilized to check the earth's exterior surface. It comes under the category of Time of Flight (ToF) sensors. LiDAR measures the object's distance by emitting a laser onto the object and capturing the traveling time[2]. LiDAR SLAM, known as Simultaneous Localization and Mapping. It uses laser-based measurements which give accurate depth-sensing readings[4]. It is simple and it is straightforward compared to some other algorithms like using feature extraction. This allows it to map and navigate in dynamic environments, ensuring stable performance even amidst fluctuations. The use of a laser allows it to work properly in low-light conditions as well. LiDAR SLAM provides high accuracy and accurate 3D measurements of the environment allowing precise mapping and localization. Lidar sensors can quickly generate point cloud data allowing real-time mapping and localization. LiDAR SLAM generates detailed 3D maps of the environment, which makes it a good choice for autonomous navigation[8]. Despite these benefits, Lidar sensors can be expensive compared to the other sensors which can increase the overall cost

of implementing LiDAR SLAM system. Lidar sensors also can consume more power compared to other sensors which needs to be considered while implementing battery-powered devices.

Metrics	Line-following	Visual SLAM	LiDAR SLAM
Accuracy	Depends on the precision of sensors.	Depends on different factors.	Provides high accuracy.
Robustness	May struggle in dynamic environments.	Relatively robust to lighting conditions.	Sensitive to changes in lighting.
Complexity	Simple to implement.	More complex	Complex
Cost	Cost-effective	More expensive.	Depends on hardware quality.
Environment Adaptability	Less adaptable.	More adaptable	More adaptable.
Power Consumption	Low	High	Moderate
Dynamic Environment Handling	Can not handle dynamic environment	Can adapt to dynamic conditions.	Can adapt to dynamic conditions.

For auto-navigating through closed and indoor environments like warehouses, hospitals, and restaurants LiDAR SLAM seems to be the preferred choice to build a 3D map of the environment with more accuracy and precision. While other methods like Visual SLAM prevail in cost and power effectiveness, LiDAR provides more accurate and precise measurements. LiDAR SLAM proves to be more accurate in low-light conditions and also provides more accurate measurements when it comes to depth calculating. For indoor environments where light conditions might be irregular and uncertain, LiDAR SLAM proves to be a better choice than other methods. Additionally, in congested and packed environments like warehouses more accurate depth measurements become important to avoid any accidents. Hence LiDAR SLAM seems to be a better choice in the field of indoor autonomous navigation.

2. Shortest Path

a. Dijkstra Algorithm

Dijkstra's algorithm is one of the most used algorithms for finding optimal solutions. It is often used for path-finding problems. It can be used to find the shortest path between nodes in a graph. Dijkstra's algorithms guarantee finding the shortest path in a weighted graph provided that all the edges are non-negative[5]. Dijkstra's algorithm is unable to deal with negative weight values. Dijkstra's algorithm is straightforward to understand and implement making it easily accessible to solve path-planning problems. It can be applied to a wide range of graph-based problems beyond just finding the shortest path. Dijkstra's algorithm explores all the nodes in the graph or network and evaluates the optimal path[6]. This might require huge memory and computing resources for complex and large maps due to the need to store information about visited nodes and their distances. Dijkstra's algorithm can be insufficient in dense graphs due to its need to explore all nodes.

b. Greedy Algorithm

Greedy algorithm is an algorithm used to solve optimization problems. Many optimization problems can be solved by greedy algorithms. Greedy algorithm solves the problem step by step. At each step, it considers the best option available at that time. This assumes that the local optimal option is part of the global optimal solution[5]. Greedy algorithm is simple and easy to implement. Greedy algorithm involves the search for the best candidate solution from a sample space of all available solutions at the time. Repeating this process again and again until the goal state is achieved or all sample space is exhausted. While Greedy algorithm is simple to implement, it does not provide a guaranteed global optimal solution as it does not consider future values or future options[6]. Hence even though they offer simplicity and efficiency, they may provide suboptimal solutions and have limited applicability in certain problem domains.

c. A* Algorithm

A* algorithm is one of the best-known path-planning algorithms. This algorithm uses heuristic search and search based on the shortest path. A* proves to be a good choice in the range of certain environments. A* uses the evaluation function: $f(n) = g(n) + h(n)$[5]. Where $g(n)$ is the cost so far to reach the nth node and $h(n)$ is the heuristic value of cost required to reach the goal node from the nth node. The heuristic value is an estimated value of the future path to reach the end node. A* algorithm prevails over other path-planning algorithms by considering future cost requirements and trying to find optimal global solutions[6]. Hence A* is optimal and efficient when the heuristic function is available and accurately estimates the cost from the given node to the goal node.

Metrics	A*	Dijkstra's	Greedy
Optimality	Always Optimal	Always Optimal	Not always Optimal
Completeness	Does not guarantees solution.	Guarantees solution.	Does not guarantees solution.
Time Complexity	Takes less time.	Takes more time.	Takes less time.

Space Complexity	Less space complexity.	More Space complexity.	Less space complexity.
Heuristic Function	Considers future values.	Does not consider future values.	Does not consider future values.
Implementation Complexity	More Complex	Less Complex	Lesser Complex.

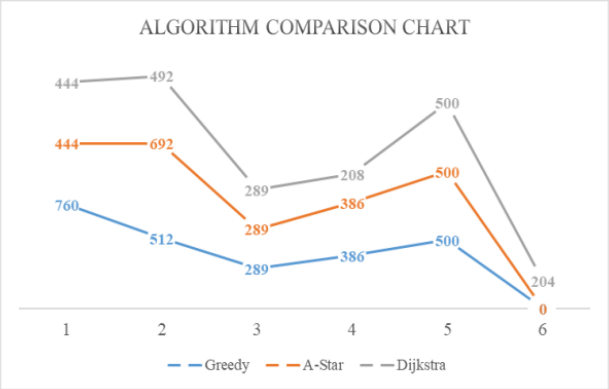

Fig 1 shows the study comparison of Greedy algorithm, A* algorithm, and Dijktra's algorithm on a graph[6]. All of these algorithms are used to find the shortest path on the same number of nodes. This image shows the comparison of the cost taken by 3 algorithms to reach the goal node from all other nodes in the graph. From the comparison, we can conclude that A* proves to be a more suitable choice for finding the optimal solution in a graph-based and pre-mapped environment. While Dijkstra's algorithm guarantees the shortest path and finding the solution while A* and Greedy algorithms may fail in some cases to find any solution, the insufficiency of Dijkstra's algorithm in dealing with negative values and more time and memory consumption due to exploring all available nodes makes it the secondary choice for selecting the shortest path[5]. Greedy algorithms prove to be more time-efficient than Dijkstra's algorithm but still fail to guarantee optimal solutions[5][6]. Hence time efficiency of the A* algorithm over Dijkstra's algorithm and finding the optimal solution over Greedy algorithm makes A* the preferable choice for finding the shortest path in the environment.

3. Communication

a. PySerial Communication

PySerial is a Python library used for serial communication with devices like Arduinos. It enables communication over a serial port (UART) between a computer and an Arduino board. To establish communication, you need to specify which serial port your Arduino is connected to and at what baud rate (communication speed) it should operate. Baud rate is a critical parameter and must match the baud rate set in your Arduino sketch (typically done using Serial.begin()).Once the serial port is open, you can read data from the Arduino using ser.readline() or ser.read(). This depends on how data is being sent from the Arduino side (e.g., line by line or byte by byte).To send data from Python to Arduino, use ser.write(). Data must be encoded into bytes before sending. Although pySerial is widely used and effective library for serial communication and disadvantages but it is specifically designed for serial communication over UART (Universal Asynchronous Receiver-Transmitter) ports. It cannot be used for other types of communication protocols like SPI (Serial Peripheral Interface) or I2C (Inter-Integrated Circuit), which are commonly used in embedded systems. PySerial does not provide extensive built-in error handling mechanisms. If communication errors occur (e.g., timeouts, buffer overflows), the user must implement custom error handling to manage and recover from these situations.

b. ROS based Distributed Communication

ROS communication is based on the publish-subscribe pattern, where nodes can publish data to topics, and subscribe to topics to receive data. Topics are named channels that carry messages, which are data structures that define the content of the communication. There are two main methods for nodes to communicate in ROS: topics and services. Topics are named buses that nodes can publish or subscribe to. When a node publishes a message to a topic, it sends it to the ROS master, which is a central node that manages the network. The ROS master then delivers the message to all the nodes that have subscribed to that topic. Topics are suitable for streaming data, such as sensor readings, commands, or feedback. Services are another way for nodes to communicate. A service is a pair of messages: one for the request and one for the response. When a node calls a service, it sends a request message to the node that provides the service, and waits for a response message. Services are useful for synchronous and one-time communication, such as querying a parameter, performing a computation, or triggering an action. For example, a node can publish a sensor_msgs/Image message to the /camera/image_raw topic, and another node can subscribe to that topic to receive the image data. ROS communication is asynchronous, meaning that the publisher and subscriber do not need to be synchronized or aware of each other.

c. Zigbee Communication

Zigbee is an open standard for a low-cost, low-power, wireless mesh network targeted at the wide development of devices for wireless control and monitoring applications. By building on top of the physical layer and media access control defined in the IEEE standard 802.15.4 it natively supports mesh networking as well as offering secure communications by default. Devices in a Zigbee network can operate

in three roles: coordinator, router, or end device. The coordinator initiates and manages the network, while routers and end devices relay messages within the network. Zigbee operates in the 2.4 GHz ISM (Industrial, Scientific, and Medical) band and divides this band into multiple channels. Devices select a specific channel for communication based on network configuration to avoid interference. Each Zigbee device has a unique 64-bit extended address and a 16-bit short address within the network. Devices use these addresses to send and receive messages. In a Zigbee mesh network, routers dynamically route messages to their destination based on network topology and signal strength. Zigbee defines a Media Access Control (MAC) layer protocol that manages channel access, acknowledgment of transmitted frames, and collision avoidance. This protocol optimizes energy consumption and ensures reliable communication in low-power wireless networks. Unfortunately they aren't particularly suitable for situations where there are many, many devices that need to communicate quickly but with much less data.

ROS facilitates a decentralized communication model where nodes (software modules) communicate with each other in a peer-to-peer fashion. This architecture allows for flexible and scalable systems where nodes can be added or removed dynamically without central coordination. ROS utilizes a publish-subscribe messaging paradigm, enabling efficient data distribution across multiple nodes. Nodes can publish messages to specific topics, and other nodes interested in these topics can subscribe to receive the messages. This decoupling of communication logic makes ROS well-suited for complex and distributed robotic systems, unlike PySerial and Zigbee[10], which often require direct point-to-point communication. ROS is deeply integrated with robotics frameworks and libraries, providing comprehensive support for robot control, perception, navigation, simulation, and more. This ecosystem enables developers to build complex robotic systems efficiently by leveraging existing components and tools. PySerial and Zigbee, while suitable for certain tasks like low-level device communication, lack the higher-level abstractions and functionalities required for advanced robotics applications. These factors make ROS[11] an ideal choice for building sophisticated and scalable robotic systems that require distributed communication and collaboration among heterogeneous components.

3. CONCLUSIONS

In conclusion, implementing a multipurpose autonomous navigation robot for commercial indoor and closed environments like warehouses, hospitals, and restaurants can increase robotics involvement in the commercial industry. This will require safe and secure as well as cost-effective and optimal methods for navigating. A durable and robust system that can adapt and navigate in multiple environments. Multipurpose autonomous navigation robot uses advanced techniques like LiDAR SLAM for navigation and localization and A* for path planning. This can provide robots the ability to navigate in multiple environments with accuracy and precision.

4. Future Direction

There are many studies and research being conducted and efforts being made in autonomous navigation. Whether it may be big car companies trying to provide completely autonomous driverless cars or e-commerce companies trying to deliver products with autonomous robots. As the demand for low-cost and more productive labor increases in various industries, many companies are turning towards AI and robotics to automate businesses. As more and more commercial efforts are being made to develop more cost-friendly equipment and ways to automate industrial processes, multipurpose robots can serve many needs. Multipurpose navigation robots can be used in many industries. Being highly modular any kind of module can be developed according to the needs of the industry and be attached to a multipurpose navigation robot. This approach can provide a solution for navigation in multiple and different environments and with modular chassis can be very useful.

REFERENCES

[1] Jagruti Chaudhari, Asmita Desai, and S. Gavarskar, "Line Following Robot Using Arduino for Hospitals", 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT), September 2019, pp. 330-332.

[2] Ifte Khairul Alam Bhuiyan, "LiDAR Sensor for Autonomous Vehicle ", Tampere University, September 2017.

[3] Brian Cèsar-Tondreau, Garrett Warnell, Ethan Stump, Kevin Kochersberger, and Nicholas R. Waytowich, "Improving autonomous robotic navigation using imitation learning", Frontiers in Robotics and AI, June 2021, pp. 1-10.

[4] Alif Ridzuan Khairuddin, Mohamad Shukor Talib, and Habibollah Haron, "Review on Simultaneous Localization and Mapping (SLAM) ", IEEE International Conference on Control System, Computing and Engineering, November 2015, pp. 85-90.

[5] Faiza Gul, Wan Rahiman, and Syed Sahal Nazli Alhady, "A comprehensive study for robot navigation techniques ", Cogent Engineering, 2019.

[6] M. Rhifky Wayahdi, Subhan Hafiz Nanda Ginting, and Dinur Syahputra, "Greedy, A-Star, and Dijkstra's Algorithms in Finding Shortest Path", International Journal of Advances in Data and Information Systems, February 2021, pp. 45-52.

[7] Kenta Takaya, Toshinori Asai, Valeri Kroumov, and Florentin Smarandache, "Simulation Environment for Mobile Robots Testing Using ROS and Gazebo ", 20th International Conference on System Theory, Control and Computing (ICSTCC), October 2016.

[8] Jianwei Zhao, Shengyi Liu, and Jinyu Li, "Research and Implementation of Autonomous Navigation for Mobile Robots Based on SLAM Algorithm under ROS ", Sensors 2022, May 2022.

[9] Mohamed Dhouioui, and Tarek Frikha, "Intelligent Warehouse Management System", 2020 IEEE International Conference on Design & Test of Integrated Micro & Nano-Systems (DTS), June 2020.

[10] Rakesh Das, and Jitendra Nath Bera, "ZigBee based Small-World Home Area Networking for Decentralized Monitoring and Control of Smart Appliances", 2021 5th International Conference on Smart Grid and Smart Cities (ICSGSC), August 2021.

[11] Pipit Anggraeni; Mariem Mrabet; Michael Defoort; Mohamed Djemai, "Development of a wireless communication platform for multiple-mobile robots using ROS", 2018 6th International Conference on Control Engineering & Information Technology (CEIT), July 2019.