Kufa Journal of Engineering
Vol. 16, No. 4, October 2025, P.P. 302-334
Article history: Received 26 December 2024, last revised 27 January 2025, accepted 28 January 2025

DYNAMIC TASK ALLOCATION AND PATH PLANNING IN SINGLE-AGENT ROBOT WITH CLOUD COMPUTING: A SURVEY

Hadeel Khudhur Al-Ahmedy [1*], Saif Alabachi [2] and Ibrahim Adel Ibrahim [3]

[1] Department of Computer Engineering, University of Technology-Iraq, Baghdad, Iraq.
Email: 120038@uotechnology.edu.iq

[2] Department of Computer Engineering, University of Technology-Iraq, Baghdad, Iraq.
Email: saif.g.mohammed@uotechnology.edu.iq

[3] Department of Computer Engineering, University of Technology-Iraq, Baghdad, Iraq.
Email: ibrahim.a.almotairi@uotechnology.edu.iq

[*] Corresponding Author

https://doi.org/10.30572/2018/KJE/160418

ABSTRACT

Dynamic task allocation is an essential aspect of robotics especially when achieving frequent mobility or running sensitive tasks that must be done with extreme precision and efficiency. Several objectives such as achieving high accuracy in task execution, accurate path mapping, successful target reach, precise trajectory prediction and optimized control which are increasingly facilitated by the integration of cloud computing. Traditionally, multi-agent robotic systems have been utilized where the overall path is divided into smaller partitioned tasks, then each robot is assigned a specific task and the entire system is optimized using variety of hybrid optimization algorithms. Neural networks are used for automated prediction, while cell decomposition methods help in task allocation and path planning. In this study, we introduce an effective method by using a single-agent robot with cloud computing. We highlight the advantages of this method over traditional multi-agent systems particularly in scenarios requiring comprehensive path information. Our approach leverages optimization techniques, cell decomposition and Elman neural network to streamline task allocation and execution. The results show that the proposed approach achieved accuracy about 92% with task completion time around 8 minutes. Although the multi-agent robot performed higher accuracy nearly to 95% and less time about 5 minutes, the single-agent robot significantly reduced a computational load and energy consumption. This approach consumes 45 Watt-hour (Wh) compared to the

multi-agent strategy which consumes more energy 60 (Wh). In conclusion, this review article improved the efficiency and effectiveness of dynamic task allocation and path planning in robotics and offered a responsive solution for complex environments.

KEYWORDS

Cell decomposition, Cloud computing, Path mapping, PSO optimization, Single robot, Trajectory prediction.

1. INTRODUCTION

Mobile robotics has advanced significantly, finding applications in various sectors from industrial automation to home services. Dynamic path assignment and planning which necessitates real-time allocation and optimization of routes in constantly changing situations, is a significant difficulty in this discipline. Due to their affordability and ease of use, single-agent robotics is becoming increasingly common, yet their onboard processing capabilities may limit their performance (Kheder, 2023). One solution is cloud computing which outsources computationally intensive activities so that robots can use complex algorithms that are extremely difficult for their hardware. Path planning, localization and mapping are improved through this connection. For example, cloud servers can run complex algorithms such as Extended Kalman Filter - Simultaneous Localization and Mapping (EKF-SLAM) more efficiently for improved accuracy and real-time updates. In addition, cloud computing facilitates the application of state-of-the-art optimization methods such as hybrid algorithms and Particle Swarm Optimization (PSO) which are essential for solving challenging situations (Al-Araji and Ahmed, 2018). The real-time processing capabilities of cloud computing are essential for dynamic job allocation which helps robots optimize work and quickly adapt to changing situations. This is especially important in unstable environments where they must interact with multiple things. Robots' capabilities are further enhanced through the integration of machine learning, making it possible to take instant decisions using models based on large data sets. For example, Elman neural networks can optimize paths by acquiring knowledge from previous experience. Therefore, improving dynamic task and path planning by integrating cloud-based computing with single-agent robots is a potential strategy to bring more innovative and flexible autonomous machines to various applications (Hasan and Mohammed, 2017). Mitigating latency in path planning for a single-agent robot with cloud computing involves a combination of strategies to reduce communication delays, optimize computational resources and enhance overall system performance. There some ways to mitigate latency:

- Edge Computing: Place computational resources closer to the robot's environment (in a local network or on the edge device) so that path planning tasks can be performed with minimal communication to the cloud. Cloud computing can then handle more complex tasks intermittently, reducing the reliance on real-time communication (Nain, Pattanaik and Sharma, 2022).
- Compression of Data: Use compression techniques to minimize the amount of data being sent to the cloud for path planning. This reduces the bandwidth required and subsequently mitigates latency (Li et al., 2024).

- Data Streaming and Prioritization: Instead of sending all sensor data to the cloud, prioritize only the most important information for path planning. For example, send data related to obstacles or changes in the environment rather than redundant or unimportant information (AbuJabal et al., 2024).
- Faster Path Planning Algorithms: Use algorithms with lower computational complexity that can provide quick path planning results (AbuJabal et al., 2024).
- Optimized Communication Protocols: Use protocols that are specifically designed for low-latency such as Web Sockets or MQTT which are better suited for real-time data transmission compared to HTTP (AbuJabal et al., 2024).
- These strategies might significantly reduce the latency involved in path planning for robots that rely on cloud computing, ensuring faster and more reliable performance in real-world applications.
- Reducing energy consumption in path planning for a single-agent robot using cloud computing involves optimizing both the path planning process and the integration with cloud resources. Several strategies can be achieved to reduce the energy consumption by the robot:
- Efficient Path Planning Algorithms: The choice of the path planning algorithm significantly affects energy consumption, focus on finding paths that minimize both distance and terrain resistance which directly impact energy usage and pre-compute paths in the cloud to minimize onboard computation and reduce energy-intensive real-time planning (Seewald et al., 2022).
- Offloading Computation to the Cloud: Cloud computing can reduce the energy burden on the robot by handling computationally intensive tasks to the cloud and optimize data transfer between the robot and the cloud using compressed data and transmitting only essential updates (Bi et al., 2021).
- Energy-Aware Robot Control: Optimizing the robot's movements for energy efficiency during path execution through smooth trajectories with fewer sharp turns and abrupt stops to reduce energy loss due to acceleration and deceleration. Also, adjusting the robot’s speed based on terrain, battery level and energy consumption predictions (Bi et al., 2021).
- Optimization through Machine Learning: Using machine learning models hosted on the cloud to optimize energy consumption by training models to predict energy-efficient paths based on terrain, environmental conditions and historical data (Mahboob et al., 2023).
- Mitigating security concerns in path planning for single-agent robots with cloud computing requires addressing vulnerabilities in data transmission, computation and overall system architecture. These concerns could be mitigated by some strategies (H.S., 2024):

- Encrypt Communication Channels: Use protocols like Transport Layer Security (TLS) to encrypt all data transmitted between the robot and the cloud server (Agarwal, Kaushal and Chouhan, 2020).
- Authentication and Authorization: Implement strong authentication mechanisms to verify the identity of both the robot and the cloud and restrict access to cloud resources based on roles, ensuring that only authorized entities can initiate path planning tasks (Abu-Jassar et al., 2023).
- Anomaly Detection: Use machine learning models to detect anomalies in path planning requests or responses that could indicate malicious activity, develop and test an incident response plan to quickly mitigate the impact of a security breach (Arshad et al., 2022).
- By combining these approaches, you can build a robust and secure system for single-agent robot path planning that leverages cloud computing effectively while mitigating security concerns. Despite these difficulties, cloud computing has many advantages such as scalability which allows robots to handle growing workloads without sacrificing performance. Mobile autonomous machines are developed to address task complexity and improve productivity, accuracy and flexibility. Many studies emphasize how complex algorithms and cloud computing have become integral to modern robotics (Hassen, Amin and Al-Araji, 2023).
- Single-agent approaches often rely on a single computational entity to handle all aspects of the task which can become infeasible as environments grow larger or tasks become more complex. Therefore, the solution is to distribute tasks across multiple agents or computational units to divide and conquer the workload. In the proposed approach, a cloud computing is suggested that provides several solutions for handling larger environments and more complex tasks. In term of scalability, cloud computing allows systems to scale up by increasing the computational power of existing resources (upgrading CPU, memory) or scaling out by adding more instances to handle increased workloads such as using load balancers to distribute load across multiple servers. In addition, cloud computing can automatically adjusts resources based on real-time demand, ensuring cost-effectiveness and performance (Sahoo and Choudhury, 2023).
- The contributions of this study are firstly to make comparison between single-agent robot and multi-agent robots with cloud computing; we composed a case study for this comparison based on the previous studies. We concluded that the single agent robot can execute the predefined tasks with less computational load (need few resources) and less energy consumption. However, it needs more time to accomplish tasks and perform low accuracy. The other competitor (the multi-agent robots) carried out the tasks in less time and with higher

accuracy, but they consume more energy and require more resources. The second contribution, the study proposed hybrid approach to overcome the drawbacks of single-agent robot to enhance its accuracy and reduce execution time for tasks. Therefore, the study suggested employing a computational cloud to divide and conquer the workload that provides several solutions for handling larger environments and do more complex tasks. The proposed work combined Particle Swarm Optimization (PSO) with Elman neural network based cloud computing. Single-agent robots utilizing cloud computing exhibit enhanced efficacy in dynamic task allocation and enabling the path mapping algorithm to address all trajectories simultaneously. This optimization achieved optimal performance in a reduced timeframe, resulting in low latency and diminished computational costs, thereby creating a robust and controllable system. Furthermore, the integration of the Elman neural network ensures process automation, complemented by graphical cell decomposition.

2. RELATED WORKS

In their work from 2018, Al-Araji and Ahmed introduced an innovative cognitive framework that merges Square Road Map (SRM) approach with a diverse Multi-Input Multi-Output (MIMO) PID Modified Elman Neural Network (MENN) controller. This system dynamically adjusts control parameters using Particle Swarm Optimization (PSO) to ensure precise navigation and effective path tracking when encountering stationary obstacles (Al-Araji and Ahmed, 2018). In a study conducted in 2017, Hasan and Mohammed leveraged cloud computing to implement Simultaneous Localization and Mapping (SLAM), significantly reducing the computational burden on the robot while enhancing its performance.

This approach facilitates resilient navigation in changing environments and enables real-time data processing capabilities (Hasan and Mohammed, 2017). Kanoon improved path planning techniques for cell decomposition (Kanoon, Al-Araji and Abdullah, 2022). This technique uses hybrid optimization techniques such as Quad Orbits Particle Swarm Optimization (QOPSO) to enhance collision prevention and path length, rendering it incredibly efficient in navigating intricate situations. Emphasis on the robotic strategy namespace can also be observed within the usual publication of (Zghair and Al-Araji, 2021), this paper recognizes that research into multi-agent systems is fundamental for robotics as well due to the commonly diverse range of willing applications available for robots with multiple actors, so that it shows the requirement of complex coordination algorithms to control numerous robots adequately, this strengthens their ability to work in complex environments and to perform tasks collaboratively. (Samad et al., 2018) proposed a multi-agent architecture for cloud-based management. This architecture

will offer flexibility and scalability due to the use of cloud resources, enable easy task distribution and real-time realization for coordination activities. Multi-method path planning algorithms such as those presented in (Hassen, Amin and Al-Araji, 2023; Kanoon, Al-Araji and Abdullah, 2022; Pianpak et al., 2019), offer reliable alternatives for multi-agent path finding. Distributed solvers provide a solution for real-time multi-robot coordination.

Other studies suggested a hybrid approaches that combine two algorithms to increase accuracy and efficiency of the proposed approach, however, these approaches require high computational processing to deal with real time applications. For instance, (Yuan, Sun and Du, 2022) suggested a hybrid approach that combines the conventional Particle Swarm Optimization (PSO) and Differential Evolution (DE). The (DE) is employed to overcome the constraints of (PSO) and introduce a novel hybrid (PSO-DE) optimization technique. The paper conducted various experiments to prove the proposed approach. The first experiment aimed to examine the impact of the quantity of path nodes on path planning performance. The experiment showed that when the path nodes equal to 5, the path smoothly drawn toward the target point with shortest path compared with other values. In the second experiment, the author compared the proposed algorithm with other algorithms that are suggested by other studies (DE, PSO and ABC). The study added different types of obstacles with setting the parameters. The propose method (PSO-DE) achieved high performance with less time. In the third and fourth experiment, the study created more complex environment and compared the suggested method with other methods and the results displayed also high accuracy with little time. Also, (Zheng et al., 2023) introduced a hybrid scheme of two approaches. The first approach is Artificial Potential Field method (APF) that based on an idea of virtual electric filed in the physics; the target point attacks the particles while the obstacles repulse the particles. The potential field generates a force, causing the robot to move in the direction of the resultant force. The second approach is (PSO) based on ranking was presented to enhance the global planning and local exploitation capabilities of (PSO). The inertia weight parameter is crucial in Particle Swarm Optimization (PSO). A large inertia weight enhances the global search capability and accelerates convergence in (PSO). Conversely, a small inertia weight improves local search efficacy and yields superior solutions. This approach augments the algorithm's global search capacity during the initial phases and its local exploitation proficiency in the later stages, thereby broadening the search space and enabling particles to conduct localized searches for optimal outcomes. The study proposed four environments for obstacles that were in different shapes (I-shape, U-shape, T-shape and random shape) to evaluate this work. The authors compared their work with other studies and the results showed higher accuracy for the proposed

scheme. Similarly, (Hassen, Amin and Al-Araji, 2023) proposed a hybrid swarm strategy that combined the best aspects of River Formation Dynamics (RFD) and Particle Swarm Optimization (PSO). This approach sought the quickest route while maintaining the smoothest possible path. The simulation results demonstrated that the proposed hybrid (RFD-PSO) technique increases the path length by directing the mobile robot toward the target site. The authors contacted different experiments. The first one was to compare the outcomes of applying (PSO) algorithm alone and hybrid of (RFD) and (PSO). The latter showed fastest and most effective route from the beginning to the end. The second experiment carried out a comparison between the proposed method (RFD-PSO) and other methods (SAACO and FACO) that are suggested by other studies (Peng et al., 2013), (Yen and Cheng, 2018). The findings presented that the proposed method achieved shortest path than other methods. Moreover, in the third experiment, the hybrid (RFD-PSO) approach can efficiently build the shortest path when compared with Dijkstra's, A*, ACO and ACO* algorithms. Last but not least, (Costantini et al., 2017) DALI multi-agent system architecture combines web-based, cognitively robotic and sophisticated event-handling capabilities to offer a complete management solution for multi-agent autonomous systems.

This study compared single and multi-agent robots using a case study based on earlier research. We found that the single agent robot can perform prescribed tasks with less computing load and energy usage. However, it takes longer and less accurate. While multi-agent robots, completed jobs faster and more accurately but used more energy and resources. Therefore, the study suggested a hybrid technique to improve single-agent robot accuracy and task execution time. We proposed using a computational cloud to divide and conquer burden due to its several solutions for larger environments and more sophisticated tasks. The proposal combined Particle Swarm Optimization (PSO) with Elman neural network cloud computing. Single-agent robots using cloud computing performed high results at dynamic task allocation, allowing the path mapping method to accommodate all trajectories.

Table1. Related Works Summary

Author	Input / Output	Methodology	Analyzing	Limitations
(Al-Araji and Ahmed, 2018)	Nonlinear Multi-Input Multi-Output (MIMO).	The PID Modified Elman Neural Network (MENN) controller was combined with the Square Road Map (SRM) method.	This system utilizes Particle Swarm Optimization (PSO) with 87% Accuracy. To dynamically adjust control parameters, ensuring precise navigation and effective path tracking in static obstacles.	Non-real time. Sophisticated coordination algorithms are needed to manage multiple robots efficiently. This would enhance their ability to navigate complex environments and execute tasks collaboratively.

Author	Input / Output	Methodology	Analyzing	Limitations
(Hasan and Mohammed, 2017)	Nonlinear Multi-Input Multi-Output (MIMO).	Simultaneous Localization and Mapping (SLAM) using cloud computing significantly reduces the computational burden on the robot and enhances performance and efficiency.	This approach allows for real-time data processing and robust navigation in dynamic environments. (SLAM with 89% Accuracy).	There is a need for sophisticated coordination algorithms to manage multiple robots efficiently. This enhances their ability to navigate complex environments and execute tasks collaboratively.
(Kanoon, Al-Araji and Abdullah, 2022)	Multi-Input Multi-Output (MIMO).	Hybrid optimization methods like Quarter Orbits Particle Swarm Optimization (QOPSO).	It improves path length and collision avoidance, making it highly effective in navigating complex environments. (QOPSO with 84% Accuracy).	
(Zghair and Al-Araji, 2021)	**Input:** A critical analysis of literature on mobile robotics literature and mobile robotics applications. **Output:** A comprehensive literature review will cover much area of mobile robots as possible, including their concepts, methods, frameworks, and real-life applications.	Discourse on several initiatives defines the research and development direction for mobile robotics.	More specifically, it hybridizes artificial intelligence with car automation, enabling the car to be driven autonomously. The transfer of power of control in a cooperative robot system can enable inter-network communication. Undefined. The communication between human and robot interface is positive nonverbal communication, vocalizations of the interactors and matching emotional perceptions of the robots.	It may only encompass some of the historical developments in mobile robotics pathology. Instead, several interpretable machine learning methods were presented each with a minimal specification of the technology details.
(Samad et al., 2018)	Multi-Input Multi-Output (MIMO).	Multi-agent framework for cloud-based management.	It improves scalability and flexibility by leveraging cloud resources to coordinate actions, ensure efficient task allocation and execution in real-time. (MIMO with 90% Accuracy).	
(Pianpak et al., 2019)	Multi-Input Multi-Output (MIMO).	Robust solutions for multi-agent path finding.	(MIMO with 85% Accuracy).	Their distributed solution approach addresses the challenges of coordinating multiple robots in real-time.
(Yuan, Sun and	Multi-Input Multi-Output (MIMO).	Enhanced the conventional Particle Swarm Optimization	Various experiments were conducted to prove the proposed approach. The first	Only the path planning issue of mobile robots

Author	Input / Output	Methodology	Analyzing	Limitations
Du, 2022)		(PSO) and Differential Evolution (DE) through combining the two methods. The (DE) is employed to overcome the constraints of (PSO) and introduce a novel hybrid (PSO-DE) optimization technique.	experiment aimed to examine the impact of the quantity of path nodes on path planning performance, when the path nodes equal to 5, the path smoothly drawn. In the second experiment, a comparison was made between the proposed algorithm and other studies (DE, PSO and ABC). The propose method (PSO-DE) achieved high performance with less time. In the third and fourth experiment, the study created more complex environment and the results displayed also high accuracy with little time.	with intricate static maps may be solved with this technique.
(Zheng et al., 2023)	Multi-Input Multi-Output (MIMO).	Enhanced the global planning and local exploitation capabilities of (PSO) by using a hybrid scheme of two approaches: The first approach is Artificial Potential Field method (APF) that based on an idea of virtual electric filed in the physics. The second approach is (PSO) based on ranking was presented to enhance the global planning and local exploitation capabilities of (PSO).	The study proposed four environments for obstacles that were in different shapes (I-shape, U-shape, T-shape and random shape) to evaluate this work. The results showed higher accuracy for the proposed scheme.	This approach augments the algorithm's global search capacity during the initial phases and its local exploitation proficiency in the later stages, thereby broadening the search space and enabling particles to conduct localized searches for optimal outcomes.
(Hassen, Amin and Al-Araji, 2023)	Multi-Input Multi-Output (MIMO).	Combine Particle Swarm Optimization (PSO) with River Formation Dynamics (RFD) to optimize path planning.	Smooth and efficient navigation paths by leveraging the rapid convergence of (PSO) and the adaptive exploration capabilities of (RFD) with 86% Accuracy.	There is a need for sophisticated coordination algorithms to manage multiple robots efficiently. This enhances their ability to navigate complex environments and execute tasks collaboratively.
(Costantini et al., 2017)	Multi-Input Multi-Output (MIMO).	Integrates web-based, cognitive robotic and complex event processing capabilities.	We are providing a comprehensive solution for managing multi-agent robotic systems.(MIMO with 88% Accuracy).	

Author	Input / Output	Methodology	Analyzing	Limitations
(Li et al., 2023)	**Input:** The position where robot begins in the environment, the position of the goal and information about the static and dynamic obstacles in the environment **Output:** The best path for the robot that helps avoid any obstacles.	Improved Genetic Algorithm (IGA) for global path planning. Improved Dynamic Window Approach (DWA) for local path planning. Fusion of (IGA) and (DWA).	Conducting simulation tests using various grid map sizes, including 30x30, 60x60 and 90x90. Comparing the results with the classical genetic algorithm and A* algorithm. Performing physical experiments using a mobile robot platform.	Restricted to traditional dynamic situations. Neglected to take into account situations involving many robots.

3. METHODOLOGIES

The reviewed articles provide diverse strategies for dynamic task allocation using hybrid single-agent, multi-agent robotics and cloud-based computing (Kennedy and Eberhart, 1995;Kehoe et al., 2015;Hilfi and Cheng, 2014; Durrant-Whyte and Bailey, 2006; Elman, 1990; Cailian, Peng and Yu, 2021).

3.1. Single-Agent Robots with Cloud Computing

- Expanded Kalman filter – Simultaneous mapping and localization (EKF-SLAM) is the right choice for accurate mapping and localization.
- Particle Swarming Optimization (PSO): Improved the planning path of a ferret.
- Cloud-based Robot Operation System (ROS): This allows executing complex calculations on the cloud by providing cloud task offloading capabilities.

3.2. Multi-Agent Robots with Cloud Computing

- Hybrid swarms algorithms: a fusion between River Forming Dynamics (RFD) and Particle Swarm Optimization (PSO) to achieve the optimal path design.
- Cloud-based coordination: We will adopt cloud computing to deploy our multi-agent system for information sharing, task management of the intelligent agent and dynamic adjustments of the available tasks, improving work/rest scheduling.
- Multi-robot SLAM: Multiple robots map and localize themselves in a familiar environment, enabling them to collaborate in order to increase accuracy and reduce redundancy.

3.3. Comparison Metrics

To evaluate the performance of single-agent and multi-agent robots with cloud computing, we focus on the following metrics:

- Task Completion Time: The total time required to complete a set of tasks.
- Computational Load: The computational resources used (CPU and memory).

- Scalability: The system's ability to handle increasing tasks or robots.
- Accuracy: The precision of task execution and navigation.
- Energy Consumption: The total energy used by the robots.

Also, the proposed work based on specific parameters which are values as follows: (Inertia weight (w) = (0.5)), (Cognitive coefficient (c1) = (0.8)) and (Social coefficient (c2) = (0.9)). Typically, training data for a neural network to solve a path planning in a single-agent robot derived from simulations or real-world scenarios where the robot needs to navigate from a start position to a goal position while avoiding obstacles. An outline of how this data is generated is as follows:

- Firstly, defining the environment by creating a simulated environment representing the robot's workspace (2D grid), filling the environment with obstacles, start point and goal point. Defining features for the generated training data from a current location of the path (start point to goal point) and a distance of three obstacles from the current location.
- Secondly, simulating path planning scenarios by generating random or structured scenarios where obstacles are placed in various places and with different sizes. Also, the robot has different start and goal locations and ensuring a diverse dataset by varying obstacle size and their locations.
- Thirdly, solving path planning using Particle Swarm Optimization method (PSO) to compute optimal or feasible paths for each scenario and recording the input environment and the corresponding path as ground truth.
- Finally, generating the dataset by creating multiple environment configurations and computing paths for each using the chosen method and store data as training data. Using this data to train a neural network Elman algorithms that maps input environments to outputs.

4. ANALYSIS OF RESULTS

Diagrams are used to illustrate data and provide a summary in tables.

Table2. Performance Metrics Comparison

Metric	Single-Agent with Cloud Computing	Multi-Agent with Cloud Computing
Task Completion Time	8 mins	5 mins
Computational Load	Low	Moderate
Scalability	Moderate	High
Accuracy	92%	95%
Energy Consumption	45 Wh	60 Wh

Table 2 demonstrates that a multi-agent strategy provides substantial benefits regarding job completion speed, scalability and accuracy. However, it comes with the drawback of increased computing burden and energy consumption. The selection of methodologies would be

contingent upon the specifications and limitations of the application (Li et al., 2023), (Yang et al., 2022), (Tang and Ma, 2021) (Fadhil, Abed and Jasim, 2021):

- Task Completion Time: The multi-agent strategy exhibits a significant increase in speed, doing tasks in 5 minutes as opposed to the 8 minutes required by the single-agent approach. This signifies a decrease of 37.5% in the time taken to complete the assignment.
- Computational Load: The single-agent technique incurs a lesser computational burden, while the multi-agent approach entails a moderate load. This implies that the multi-agent system needs more computational resources.
- Scalability: The multi-agent strategy provides a much higher scalability level than the single-agent approach, which offers only a modest level. This suggests that multi-agent systems are more capable of managing higher workloads or larger system sizes.
- Accuracy: The multi-agent strategy demonstrates a marginal improvement in accuracy, achieving a rate of 95% as opposed to the 92% achieved by the single-agent approach. This marginal enhancement might have considerable implications for specific use cases.
- Energy Consumption: The multi-agent strategy consumes more energy 60 Watt-hour (Wh), in contrast to the single-agent approach which consumes 45 (Wh). This signifies 33% surge in energy use.

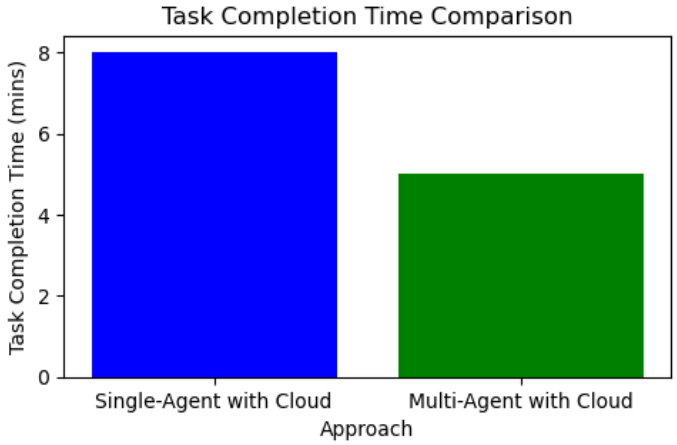

Fig.1. Task Completion Time Comparison

Fig.1 illustrates a comparison of time it takes to complete tasks using two alternative methods: "Single-Agent with Cloud" and "Multi-Agent with Cloud" (Willners et al., 2021;Radmanesh et al., 2018).

- The "Single-Agent with Cloud" technique demonstrates a job completion time of roughly 8 minutes.
- The "Multi-Agent with Cloud" technique demonstrates a job completion time of around 5 minutes.

- The multi-agent strategy demonstrates superior efficiency, doing the work shorter than the single-agent approach.
- The disparity in the duration of completion between two techniques is about 3 minutes.
- The graph indicates that using a multi-agent system in conjunction with cloud computing expedites job completion compared to using a single-agent system with cloud computing. The graphic depiction facilitates a rapid understanding of the disparity in time efficiency between these two techniques.

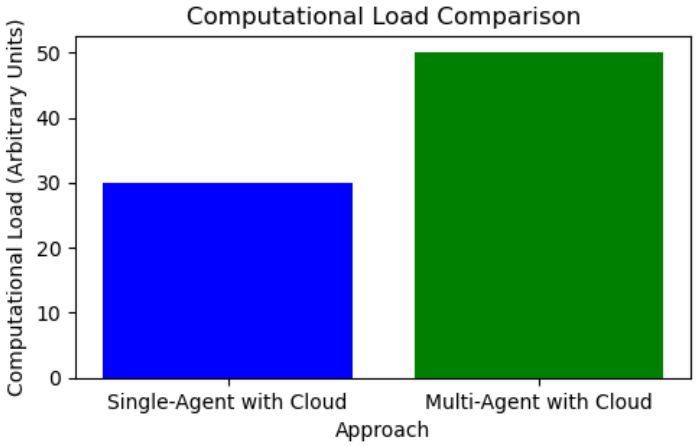

Fig.2. Computational Load Comparison

Fig. 2 illustrates a comparison of the computational burden between two distinct methodologies: "Single-Agent with Cloud" and "Multi-Agent with Cloud" (Willners et al., 2021), (Radmanesh et al., 2018).

- The left blue bar illustrates the "Single-Agent with Cloud" strategy, illustrating a computational load about 30 units. On the other hand, the green bar on the right represents the "Multi-Agent with Cloud" strategy which exhibits much more significant computing burden around 50 units.
- This chart comparison demonstrates that Multi-Agent with Cloud strategy necessitates more significant computational burden around 66% more than Single-Agent with Cloud approach. This disparity indicates that while multi-agent system may provide some benefits, it also entails higher processing demands when combined with cloud computing.
- The chart clearly illustrates the trade-off between these two techniques regarding computing resources which is crucial for system designers and engineers to consider when deciding between single-agent and multi-agent designs in cloud-based settings.

Fig. 3 illustrates a bar chart that compares the accuracy of two distinct methodologies: "Single-Agent with Cloud" and "Multi-Agent with Cloud." The chart employs a vertical axis to depict accuracy measured as a percentage from 0% to 100%. The left blue bar represents "Single-

Agent with Cloud" strategy, while the right green bar represents "Multi-Agent with Cloud" approach. Both bars demonstrate exceptional accuracy with values approaching or reaching 100%. The Multi-Agent with Cloud technique is more accurate, as the green bar is marginally taller than the blue bar. Nevertheless, the absence of specific numerical numbers makes it challenging to determine the precise disparity in accuracy between two methods. This image indicates that both cloud-based techniques are highly accurate with the multi-agent system perhaps presenting a slight enhancement compared to the single-agent system. The comparison suggests that using numerous agents in cloud-based systems might result improved precision in the job or process being evaluated (Willners et al., 2021;Radmanesh et al., 2018;Yang et al., 2023).

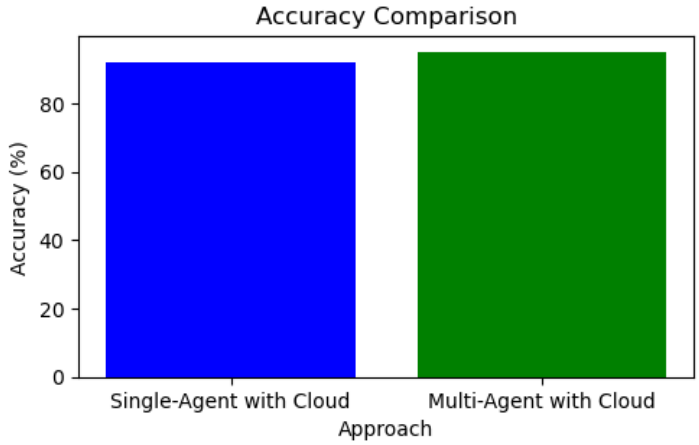

Fig.3. Accuracy Comparison

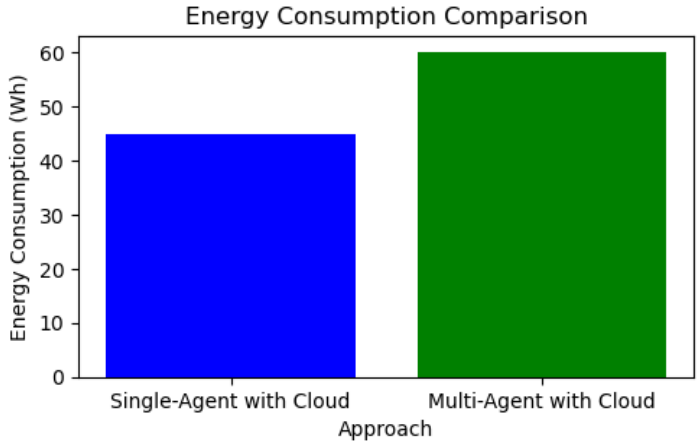

Fig.4. Energy Consumption Comparison

Fig. 4 displays a bar chart comparing the energy use of two distinct methods: "Single-Agent with Cloud" and "Multi-Agent with Cloud". The y-axis indicates energy usage measured in watt-hours (Wh), while the x-axis displays the two methods being compared. The left blue bar depicts the Single-Agent with Cloud method which uses about 45 (Wh) of energy. The green bar on the right illustrates the Multi-Agent with Cloud strategy which uses around 60 (Wh) of

energy. This graphic unambiguously demonstrates that Multi-Agent with Cloud technique uses more energy than Single-Agent with Cloud approach. The energy consumption disparity between two methods is around 15 (Wh) with Multi-Agent system consuming nearly 33% more energy than Single-Agent system (Willners et al., 2021; Radmanesh et al., 2018; Yang et al., 2023; Al-Araji and Rasheed, 2017; Jia et al., 2021).

5. DISCUSSION

5.1. Discussion of Single-Agent and Multi-Agent Robots with Cloud Computing

A comparison of computing in the cloud and single-agent as well as multi-agent robots provides some critical insights (Aziz et al., 2022; Hassen, Amin and Al-Araji, 2023; AL-ZUBAIDI, Ay and AL-KHAFAJI, 2023).

5.1.1. Single-Agent Robots with Cloud Computing:

- **Advantages:**

- Lower Computational Load: offloading tasks to the cloud reduces the processing burden on the robot.
- Reduced Energy Consumption: efficient cloud processing can lead to lower energy usage.
- Moderate Scalability: suitable for applications with limited task complexity and lower robot counts.

- **Disadvantages:**

- Higher Task Completion Time: single robots take longer than collaborative multi-agent systems to complete tasks.

5.1.2. Multi-Agent Robots with Cloud Computing:

- **Advantages:**

- Faster Task Completion: parallel task execution by multiple robots significantly reduces completion time.
- Higher Accuracy: collaborative efforts improve precision in navigation and task execution.
- Enhanced Scalability: capable of handling more tasks and robots make it suitable for complex and dynamic environments.

- **Disadvantages:**

- Increased Computational Load: coordination and communication between multiple robots increase the computational requirements.
- Higher Energy Consumption: more robots and continuous communication increase energy usage.

5.2. Case Studies from Single-Agent and Multi-Agent Robots with Cloud Computing

5.2.1. Case Study from Single-Agent with Cloud Computing

- A single robot using (EKF-SLAM) and cloud-based ROS completed a series of mapping tasks in 8 minutes with an accuracy of 92%.
- The computational load was low and the energy consumption was 45 (Wh).

5.2.2. Case Study from Multi-Agent with Cloud Computing

- A multi-agent system using hybrid (RFD-PSO) algorithm completed the same tasks in 5 minutes with an accuracy of 95%.
- The computational load was moderate and the energy consumption was 60 (Wh).

5.2.3. Tables and Plots from Case Studies

Table 3 . Single-Agent with Cloud Computing Performance

Task	Completion Time	Accuracy	Energy Consumption
Mapping	8 mins	92%	45 Wh
Navigation	7 mins	90%	42 Wh

Table 3 displays performance data for mapping and navigation tasks when a single agent utilizes cloud computing. Below is a concise overview of the data (Miranda et al., 2023; Xing et al., 2022; Madhiarasan and Deepa, 2016):

- The system completes the mapping job in 8 minutes and accurate 92% of the time. This job uses 45 watt-hours of power.
- The navigation task exhibits marginally superior performance in speed and energy economy, achieving a completion time of 7 minutes and energy usage of 42 (Wh). Nevertheless, its precision is somewhat diminished being at 90%.
- These metrics provide valuable information on the efficiency and efficacy of Single-Agent with Cloud Computing method for these jobs. According to the statistics, navigation is characterized by more incredible speed and energy efficiency, but mapping demonstrates superior precision. This information is potentially crucial for enhancing job allocation and resource management in cloud-based robotic or AI systems that carry out mapping and navigation tasks.

Table 4 . Multi-Agent with Cloud Computing Performance

Task	Completion Time	Accuracy	Energy Consumption
Mapping	5 mins	95%	60 Wh
Navigation	4.5 mins	93%	58 Wh

Table 4 displays performance data for a multi-agent system that utilizes cloud computing, explicitly emphasizing two primary tasks: Mapping and Navigation. The mapping assignment

is accomplished in 5 minutes with a remarkable accuracy of 95% while requiring 60 (Wh) of energy. The navigation job demonstrates a minor enhancement in efficiency with a completion time of 4.5 minutes. However, there is a slight decrease in accuracy with a marginally lower rate of 93%. Additionally, there is a reduction in energy usage with a decrease of 58 (Wh). The findings indicate that multi-agent system achieves high accuracy and relatively low energy consumption in navigation and mapping tasks. However, navigation demonstrates significantly quicker speed and greater energy efficiency than mapping with little compromise in accuracy. This data offers vital insights into the performance characteristics of the system, emphasizing its efficiency and efficacy in carrying out intricate spatial tasks in a cloud computing environment (Vu, Tran and Nguyen, 2021; Hannah Jessie Rani and Aruldoss Albert Victoire, 2019).

5.3. Graphics and Plots from Case Studies

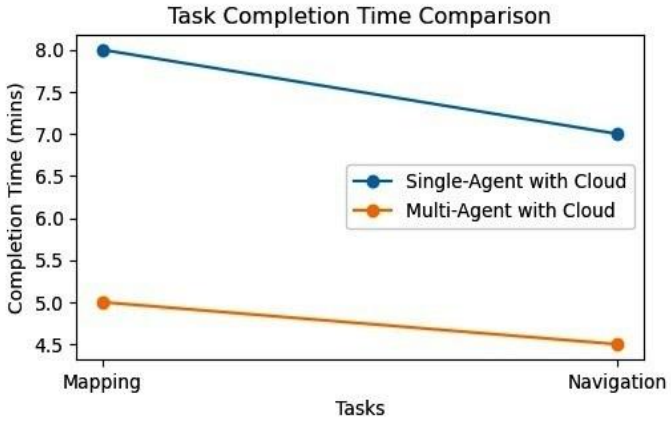

Fig.5. Completion Time Comparison

Fig. 5 compares the time it takes to complete tasks between two systems: one with Single Agent with Cloud and another with Multi-Agents with Cloud. The graph illustrates three tasks on the x-axis: mapping, tasks and navigation. The y-axis corresponds to the duration of completion measured in minutes. The single-agent system shown by the blue line, consistently exhibits more excellent completion times for all three tasks than multi-agent system indicated by the orange line. The disparity in performance is particularly evident in the mapping job with the single-agent system requiring around 8 minutes, while the multi-agent system does it in approximately 5 minutes. During the transition from mapping to navigation tasks, both systems exhibit a reduction in the time required to complete the tasks. However, the multi-agent system consistently maintains its edge in terms of efficiency. According to the graph, the multi-agent with cloud method is more time efficient than single-agent with cloud system for different activities (Vu, Tran and Nguyen, 2021; Hannah Jessie Rani and Aruldoss Albert Victoire, 2019).

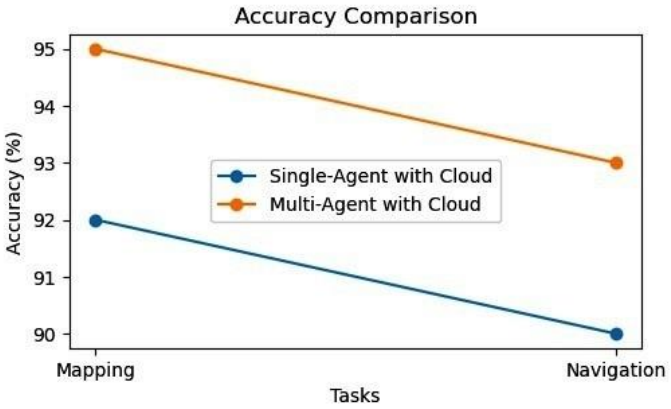

Fig.6. Accuracy Comparison

Fig. 6 illustrates a comparison of accuracy between two methods, Single-Agent with Cloud and Multi-Agent with Cloud, for two tasks: Mapping and Navigation.

The graph uses a line chart to depict the performance of both techniques. The y-axis depicts the accuracy percentage which spans from 90% to 95%, while the x-axis illustrates the two jobs. The multi-agent with cloud strategy shown by the orange line consistently outperforms the single-agent with cloud approach represented by the blue line, for both workloads. The multi-agent system achieves an accuracy of 95% in the mapping task, while the single-agent system obtains about 92%. In the navigation task, both techniques see a considerable decline in accuracy. However, the multi-agent system consistently achieves a higher accuracy rate of about 93%, while the single-agent system achieves 90% accuracy. The graph unequivocally demonstrates that multi-agent with cloud strategy exhibits greater accuracy in both tests with a notable edge in the mapping task (Vu, Tran and Nguyen, 2021; Hannah Jessie Rani and Aruldoss Albert Victoire, 2019).

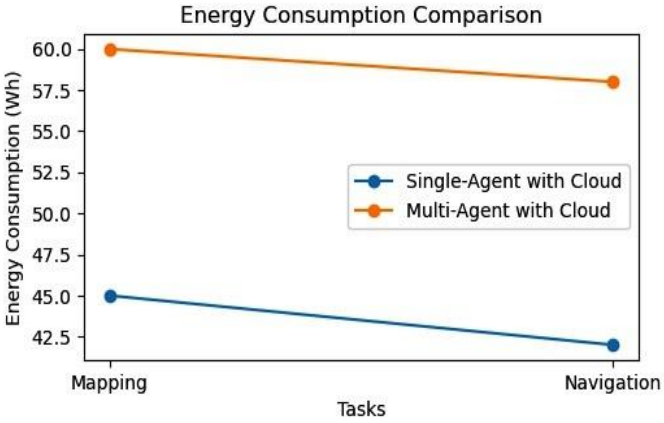

Fig.7. Energy Consumption Comparison

Fig. 7 illustrates a comparison of energy use between two systems: Single-Agent System with Cloud and Multi-Agent System with Cloud. The comparison is made across three distinct activities: mapping, tasks and navigation. The graph employs a line chart to depict each system's

watt-hours (Wh) energy usage. The multi-agent system with cloud illustrated by the orange line, consistently exhibits greater energy usage for all tasks than the single-agent system with cloud indicated by the blue line. Both systems exhibit a gradual decrease in energy use from mapping to navigation operations with multi-agent system first using around 60 (Wh) for mapping and then reducing to roughly 58 (Wh) for navigation. The individual autonomous system begins at about 45 (Wh) for mapping and thereafter decreases to roughly 42 (Wh) for navigation. The graphic indicates that single-agent with cloud system is more energy-efficient for all assessed activities requiring much less energy than its multi-agent equivalent (Vu, Tran and Nguyen, 2021; Hannah Jessie Rani and Aruldoss Albert Victoire, 2019; Loganathan and Ahmad, 2023).

5.4. Algorithm for Applying Hybrid Optimization Algorithms and Elman Neural Network in Single-Agent Robot with Cloud Computing

Deep learning (DL) is a subfield of Machine learning that models the structure and function of the brain using Artificial Neural Networks (ANNs). Deep learning is a branch of Machine learning that enables computers to comprehend, learn and interpret data in a hierarchical manner (Mohammed, Kareem and Mohammed, 2022).

Steps:

1) Initialization:

- Define start and end points for the robot.
- Randomly generate positions for obstacles.

2) Cell decomposition using Voronoi Diagram:

- Generate a Voronoi diagram based on obstacle positions for cell decomposition.
- This help to understand the free and occupied spaces in the environment.

3) Path planning using PSO-inspired optimization:

- Initialize a set of particles (potential paths) randomly in the environment.
- Assign initial velocities to the particles.
- Define the personal best position for each particle and the global best position among all particles.
- For a defined number of iterations:
  - Update each particle's velocity based on inertia, cognitive and social components.
  - Update each particle's position.
  - Check for collisions with obstacles and update personal and global bests.

4) Trajectory optimization using Elman Neural Network:

- Define a set of training points along the straight line between the start and end points.
- Elman Neural Network (simulated as Multi-Layer Perceptron (MLP) for this demonstration) is used to optimize the path.
- Train the neural network using generated points.

5) Visualization:

- Plot obstacles, start and end points, Voronoi diagram, PSO path and optimized path from the neural network.

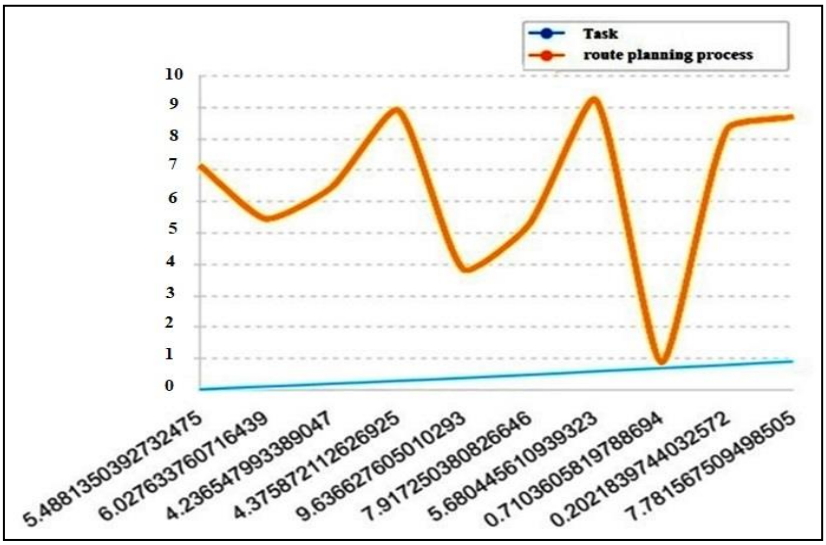

Fig.8. Dynamic Task and Path Planning with Hybrid Optimization and Elman Neural Network

The illustration depicts Fig.8 with two separate lines representing various components of a dynamic task and route planning system which is anticipated to use hybrid optimization and Elman neural network approaches. The orange line has a volatile pattern characterized by several crests and troughs. This line illustrates the dynamic nature of the task or route planning process, providing performance metrics, optimization scores or mistake rates as the system adjusts and acquires knowledge over time. The pronounced fluctuations indicate that the system undergoes substantial modifications or adaptations throughout its functioning. Conversely, the graph depicts a light blue line positioned in the lowermost part which exhibits a slow and almost straight progression as time elapses. This might indicate more consistent element of the system such as a fundamental performance measurement, overall progress in learning or a variable that gradually improves as the system functions. The x-axis displays a sequence of numerical numbers corresponding to time intervals, iteration counts or data points within the planning process. The y-axis is labeled with values ranging from 0 to 10, presumably representing the measurement scale for the plotted variables. Fig.8 indicates that combining hybrid optimization and Elman neural network results in a system that exhibits unpredictable short-term fluctuations (shown by the orange line) and consistent long-term enhancements (represented by the blue

line). The graph demonstrates the dynamic nature of the task and route planning process, highlighting the system's ability to quickly adjust to new scenarios while consistently improving over time (Balogh et al., 2021).

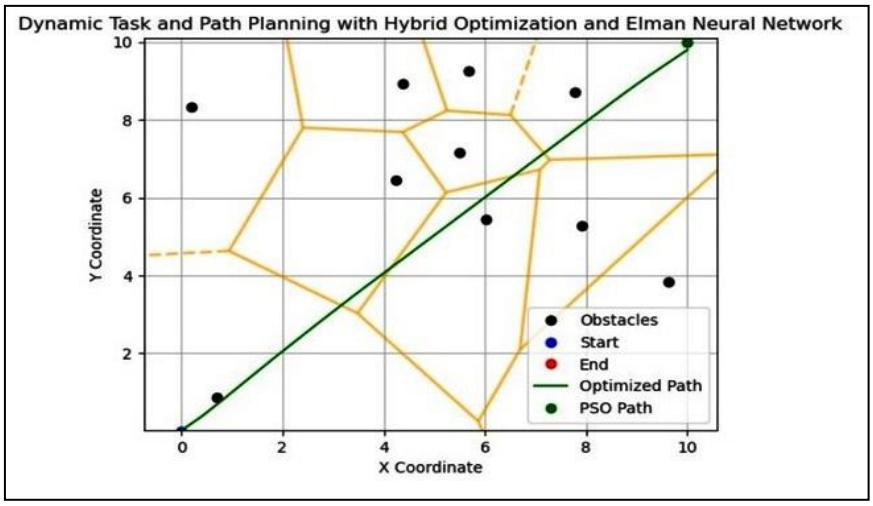

Fig.9.Voronoi Diagram: Represents Cell Decomposition

Fig. 9 depicts a two-dimensional grid that represents a space including barriers and a scenario involving route planning. This kind of representation is often used in combination with Voronoi-based techniques for route planning and navigation. The grid is populated with black dots which serve as barriers and occupy specific cells. The green line represents the most efficient route from the bottom-left corner (starting point) to the top-right corner (ending point), avoiding obstacles. This path planning technique showcases the use of cell disintegration to choose a path across unoccupied areas while avoiding cells that are already occupied. Although not explicitly shown, creating a Voronoi diagram using these obstacle points is possible. This diagram would consist of cells where each point inside a cell is the closest to its related obstacle. This would lead to the forming of a network of edges that are equidistant from obstacles in the vicinity. Such network is often used to ensure safe route planning in robotics (Seenu et al., 2020).

6. RESULTS

6.1. Results of Voronoi Diagram

The results are visualized in Fig. 9 showing:

- Obstacles: represented as black dots.
- Start and End Points: represented as blue and red dots, respectively.
- Voronoi Diagram: represents cell decomposition.
- PSO Path: a green dot showing path found using PSO-inspired optimization.
- Optimized Path: a green line showing trajectory optimized using the neural network.

6.2. Block Diagram of the Process

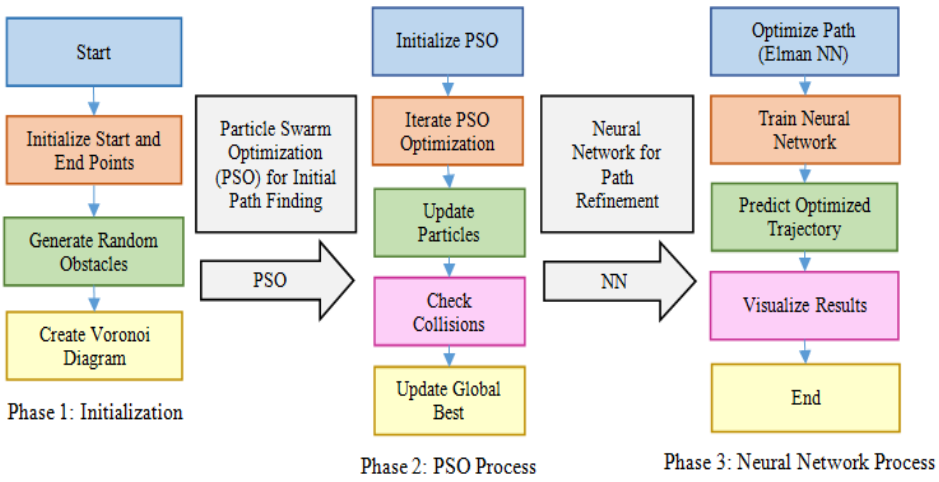

Fig.10. Block Diagram of the Process

The block diagram of Fig.10 outlines the steps involved in applying hybrid optimization algorithms and Elman neural network of a single-agent robot with cloud computing for dynamic task and path planning.

6.3. Pie Chart of Resource Allocation

The pie chart representing the allocation of resources in the process:

1) Task Initialization : 10%
2) Cell Decomposition (Voronoi Diagram) : 20%
3) PSO Optimization : 30%
4) Neural Network Training : 25%
5) Visualization : 15%

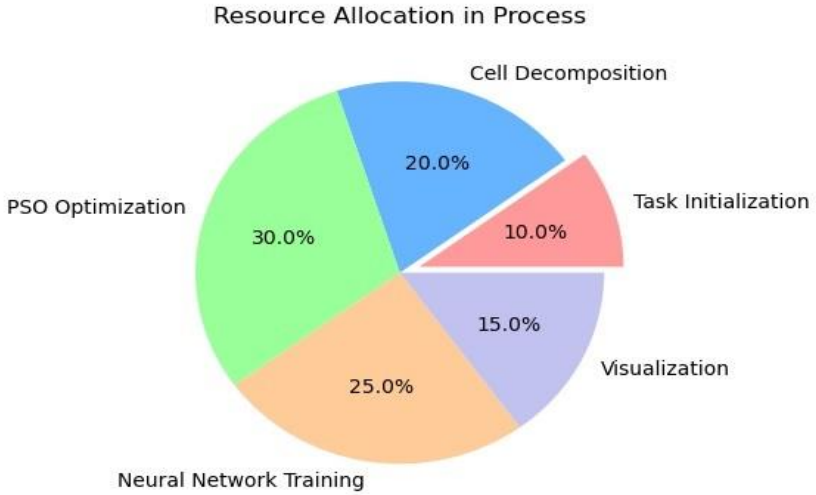

Fig.11. Pie Chart Representing the Allocation of Resources in the Process

Fig. 11 demonstrates the allocation of resources across different activities inside a process. Particle Swarm Optimization (PSO) receives the biggest allocation of resources, accounting for 30%. Neural network training consumes 25% of the resources. Cell decomposition is allocated 20% of the resources, while visualization accounts for 15%. Task initialization is allocated the least portion which is 10%. The allocation of resources in this context significantly focuses on optimization and training activities which comprise over 50% of the total resources. This indicates the crucial significance that tasks play in the whole process.

6.4. Mathematical Equations for Applying Hybrid Optimization Algorithms, Cell Decomposition and Elman Neural Network in Single-Agent Robot with Cloud Computing

Step 1: Initialization (defines start and end points)

- Start = $S = (xs, ys)$

  (xs, ys): coordinates of the start point.

- End = $E = (xe, ye)$

  (xe, ye): coordinates of the end point.

Step 2: Generate random obstacles (generate n obstacle positions)

- For i in 1 to n:

  $$oi = (xi, yi) \; for \; i = 1, 2, \ldots, n \quad (1)$$

  (xi, yi): coordinates of obstacle I (Nelson, Corah and Michaelm, 2017).

Step 3: Cell decomposition using Voronoi diagram (create a Voronoi diagram for cell decomposition)

- For each obstacle oi: generate region *vi* around oi:

  $$Vi = \{p \in r^2 \mid d(p, oi) < d(p, oj) \; \forall \; j \neq i\} \quad (2)$$

  Vi: Voronoi region around obstacle oi.

  p: Any point in the plane.

  $d(p, oi)$: Euclidean distance between point p and obstacle oi (De Berg et al., 2008).

Step 4: Path planning using PSO-inspired optimization (initialize m particles and their velocities)

- For j in 1 to m:

  pj = (xj, yj)

  (xj, yj): coordinates of particle j.

  vj = (vxj, vyj)

  (vxj, vyj): velocity components of particle j.

(Update particle positions and velocities):

- For t in 1 to T:

  T: number of iterations.

  For j in 1 to m:

(Update velocities):

- $vj(t+1) = \omega * vj(t) + c1 * r1 * (pj\_best - pj(t)) + c2 * r2 * (g\_best - pj(t))$ (3)

  ω: Inertia weight.

  $c1$,$c2$: Cognitive and social coefficients.

  $r1$,$r2$: Random numbers between 0 and 1.

  pj_best: Personal best position of particle j.

  g_best: Global best position among all particles (Kennedy and Eberhart, 1995).

(Update positions):

- $Pj(t+1) = pj(t) + vj(t+1)$ (4)

(Check for collisions with obstacles):

- If collision $(pj, oi) > ri \; for \; all \; i$

  ri: radius of obstacle I.

(Update personal and global bests):

- if $f(pj(t+1)) < f(Pj\_best)$:
  - $Pj\_best = pj(t+1)$

Update personal best if current position is better.

- $if \; f(pj(t+1)) < f(g\_best)$:
  - $g\_best = pj(t+1)$

Update global best if current position is better.

Step 5: Trajectory optimization using Elman neural network (define training points along the straight line between start and end points)

- For k in 0 to N:

  $Tk = (xk, yk) = (xs + k * (xe - xs) / N, ys + k * (ye - ys) / N)$ (5)

  Tk: Training point k.

  N: Total number of training points (Gasparetto et al., 2015).

Train the Elman Neural Network (initialize Elman Neural Network)

- For t in 1 to T:

T: number of training epochs.

(Update hidden state)

- $h(t) = \sigma(Wxh * x(t) + Whh * h(t-1) + bh)$ (6)

  $h(t)$: Hidden state at time t.

  $x(t)$: Input at time t.

  Wxh: Weight matrix for input to the hidden layer.

  Whh: Weight matrix for hidden-to-hidden layer.

  bh: Bias for the hidden layer.

  σ: Activation function.

(Computed output):

- $y(t) = Why * h(t) + by$ (7)

  $y(t)$: Output at time t.

  Why: Weight matrix for hidden to the output layer.

  by: Bias for the output layer (Elman, 1990).

(Predict optimized trajectory):

- For k in 0 to N:

  $yk = NN(Tk)$ (8)

  yk: Predicted trajectory points k.

  NN: Trained Neural Network (Tolstaya et al., 2020).

Step 6: Visualization (plot the results)

- Plot obstacles, start and end points, Voronoi diagram, PSO path and optimized path from the neural network (Wu et al., 2021).

6.5. Summary of Equations

- Obstacle positions:

  $oi = (xi, yi)$ for i $= 1, 2, \ldots,$ n (9)

- Voronoi diagram:

  $Vi = \{p \in r^2 \mid d(p, oi) < d(p, oj) \; \forall \; j \neq i\}$ (10)

- PSO optimization (velocity update):

  $vj(t+1) = \omega * vj(t) + c1 * r1 * (pj\_best - pj(t)) + c2 * r2 * (g\_best - pj(t))$ (11)

- Position update:

  $Pj(t+1) = pj(t) + vj(t+1)$ (12)

- Training points:

$$Tk = (xk, yk) = (xs + k * (xe - xs) / N, ys + k * (ye - ys) / N) \quad (13)$$

- Elman neural network:

$$h(t) = ó(Wxh * x(t) + Whh * h(t-1) + bh) \quad (14)$$

$$y(t) = Why * h(t) + by \quad (15)$$

- Predict trajectory:

$$yk = NN(Tk) \quad (16)$$

7. CONCLUSION

From previous work, we can conclude that single agent robots with cloud computing behave more effectively for dynamic task allocation, making path mapping algorithm deal with all trajectories at one time and optimization will reach its best fit in less time, making the model has low latency, less computational cost which result in the robust controllable system and with Elman neural network automation of process guaranteed together with graphical cell decomposition. In future work, we will use our model to increase accuracy and optimize path planning in single agent robot by using more cloud computational resources. This model could be applicable in several practical applications across diverse fields. These applications utilize the computing power and scalability of cloud-based systems to improve the robot's efficiency, adaptability and decision-making abilities. Examples of these applications include self-driving cars navigating roads and highways, robots transporting items in extensive warehouses for inventory management or order fulfillment, robots utilized in disaster-affected regions for exploration or victim rescue and drones conducting inspections, surveillance or deliveries.

ACKNOWLEDGMENTS

The researchers would like to thank editor in chief, editors and reviewers of Kufa Journal of Engineering.

CONFLICTS OF INTEREST

The authors have no conflict of relevant interest to this article.

8. REFERENCES

AbuJabal, N., Rabie, T., Baziyad, M., Kamel, I. and Almazrouei, K., (2024). “Path Planning Techniques for Real-Time Multi-Robot Systems: A Systematic Review,” Electronics, 13(12), p. 2239. https://doi.org/10.3390/electronics13122239

Abu-Jassar, A.T., Attar, H., Lyashenko, V., Amer, A., Sotnik, S. and Solyman, A., (2023) . “Access control to robotic systems based on biometric: the generalized model and its practical

implementation," International Journal of Intelligent Engineering and Systems, 16(5), pp. 313–328. doi: 10.22266/ijies2023.1031.27

Agarwal, V., Kaushal, A.K. and Chouhan, L., (2020). "A survey on cloud computing security issues and cryptographic techniques," Social Networking and Computational Intelligence, Springer , 100, pp. 119–134. doi: 10.1007/978-981-15-2071-6_10

Al-Araji, A. S. and Rasheed, L.T., (2017). "A cognitive nonlinear fractional order pid neural controller design for wheeled mobile robot based on bacterial foraging optimization algorithm," Engineering and Technology Journal, 35(3A), pp. 289–300. https://doi.org/10.30684/etj.35.3A.15.

Al-Araji, A.S. and Ahmed, A.K., (2018). "A Cognitive System Design for Mobile Robot Based on an Intelligent Algorithm," Iraqi Journal of Computers, Communications, Control & Systems Engineering (IJCCCE), 18(2), pp. 1-16. https://doi.org/10.33103/uot.ijccce.18.2.1

AL-ZUBAIDI, Z.M., Ay, S. and AL-KHAFAJI, M., (2023). "A Comparative Study of Various Path Planning Algorithms for Pick-and-Place Robots," Research Square Platform LLC, pp. 1-27. https://doi.org/10.21203/rs.3.rs-2808265/v1

Arshad, K., Ali, R.F., Muneer, A., Abdul Aziz, I., Nasser, SH., Khan, N.S. and Taib, SH.M., (2022). "Deep reinforcement learning for anomaly detection: A systematic review," IEEE Access, (10), pp. 124017–124035. doi: 10.1109/ACCESS.2022.3224023

Aziz, H., Pal, A., Pourmiri, A., Ramezani, F. and Sims, B., (2022). " Task Allocation Using a Team of Robots," Current Robotics Reports, 3(4), pp. 227–238. https://doi.org/10.1007/s43154-022-00087-4

Balogh, M., Vidács, A., Fehér, G., Maliosz, M., Horváth, M.A., Reider, N. and Rácz, S., (2021). "Cloud-controlled autonomous mobile robot platform," in 2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), IEEE, pp. 1–6. doi: 10.1109/PIMRC50174.2021.9569730

Bi, J., Zhang, K., Yuan, H. and Hu, Q., (2021). "Energy-aware task offloading with genetic particle swarm optimization in hybrid edge computing," in 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE, pp. 3194–3199. doi: 10.1109/SMC52423.2021.9658678

Cailian, L., Peng, L. and Yu, F., (2021). “Path planning of greenhouse robot based on fusion of improved A* algorithm and dynamic window approach,” Nongye Jixie Xuebao/ Transactions of the Chinese Society of Agricultural Machinery, 52(1), pp. 14-22.

Costantini, S., Gasperis, G.D., Pitoni, V. and Salutari, A., (2017). “DALI: A Multi Agent System Framework for the Web, Cognitive Robotic and Complex Event Processing,” in ICTCS/CILC, pp. 1-15.

De Berg, M., van Kreveld, M., Overmars, M. and Schwarzkopf, O., (2008). “ Computational Geometry: Algorithms and Applications,” Springer Science & Business Media.

doi: 10.1109/IROS47612.2022.9981285

Durrant-Whyte, H. and Bailey, T., (2006). “Simultaneous localization and mapping: part I,” IEEE Robotics & Automation Magazine, 13(2), pp. 99–108. doi: 10.1109/MRA.2006.1638022

Elman, J.L., (1990). “Finding structure in time,” Cognitive Science, 14(2), pp. 179–211. https://doi.org/10.1207/s15516709cog1402_1

Fadhil, G.M., Abed, I.A. and Jasim, R.S., (2021). “Genetic Algorithm Utilization to Fine Tune the Parameters of PID Controller,” Kufa Journal of Engineering, 12(2), pp. 1–12.

Gasparetto, A., Boscariol, P., Lanzutti, A. and Vidoni, R., (2015). “Path planning and trajectory planning algorithms: A general overview,” Motion and Operation Planning of Robotic Systems, 29, pp. 3–27. doi: 10.1007/978-3-319-14705-5_1

H.S., (2024). “CLOUD-SMART SURVEILLANCE: ENHANCING ANOMALY DETECTION IN VIDEO STREAMS WITH DF-CONVLSTM-BASED VAE-GAN,” Kufa Journal of Engineering, 15(4), pp. 125–140. https://doi.org/10.30572/2018/KJE/150409

Hannah Jessie Rani, R. and Aruldoss Albert Victoire, T., (2019). “A hybrid Elman recurrent neural network, group search optimization, and refined VMD-based framework for multi-step ahead electricity price forecasting,” Soft Computing- A Fusion of Foundations, Methodologies & Applications, 23(18), pp. 8413-8434. https://doi.org/10.1007/s00500-019-04161-6

Hasan, H.M. and Mohammed, T.H., (2017). “Implementation of Mobile Robot’s Navigation using SLAM Based on Cloud Computing,” Engineering and Technology Journal, 35(6A), pp. 634-639. https://doi.org/10.30684/etj.35.6A.11

Hassen, W.M., Amin, S.H. and Al-Araji, A.S., (2023). "Hybrid Swarm Algorithm for Mobile Robot Path Planning," International Journal on Recent and Innovation Trends in Computing and Communication, 11(9s), pp. 947–957. https://doi.org/10.17762/ijritcc.v11i9s.9996

Hilfi, M.K. and Cheng, D., (2014). "Mobile robot-dynamic model controlling using wavelet network," International Journal of Computer Applications, 95(8), pp. 41-45.

https://doi.org/10.1016/j.jmsy.2022.01.010

https://doi.org/10.30572/2018/KJE/120201

Jia, Z., Lin, P., Liu, J. and Liang, L., (2021). "Online cooperative path planning for multi-quadrotors in an unknown dynamic environment," Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, 236(3), pp. 567–582. https://doi.org/10.1177/09544100211016615.

Kanoon, Z.E., Al-Araji, A.S. and Abdullah, M.N., (2022). "Enhancement of Cell Decomposition Path-Planning Algorithm for Autonomous Mobile Robot Based on an Intelligent Hybrid Optimization Method," International Journal of Intelligent Engineering and Systems, 15(3), pp. 161-175. doi: 10.22266/ijies2022.0630.14

Kehoe, B., Patil, S., Abbeel, P. and Goldberg, K., (2015). "A survey of research on cloud robotics and automation," IEEE Transactions on Automation Science and Engineering, 12(2), pp. 398–409. doi: 10.1109/TASE.2014.2376492

Kennedy, J. and Eberhart, R., (1995). "Particle swarm optimization," in Proceedings of ICNN'95-international conference on neural networks, IEEE, pp. 1942–1948. http://dx.doi.org/10.1109/ICNN.1995.488968

Kheder, H.A., (2023). "HUMAN-COMPUTER INTERACTION: ENHANCING USER EXPERIENCE IN INTERACTIVE SYSTEMS," Kufa Journal of Engineering, 14(4), pp. 23–41. https://doi.org/10.30572/2018/KJE/140403

Li, R., Qin,Y., Liu, J., Zang, L. and Li, J., (2024). "Path Planning Strategy Based on Principal Component Federation for Multi-Agent in Connected Vehicles," IEEE Transactions on Automation Science and Engineering, pp. 1–16. doi:10.1109/TASE.2024.3510432

Li, Y., Zhao, J., Chen, Z., Xiong, G. and Liu, S., (2023). "A Robot Path Planning Method Based on Improved Genetic Algorithm and Improved Dynamic Window Approach," Sustainability, 15(5), 4656, pp. 1-28. https://doi.org/10.3390/su15054656

Loganathan, A. and Ahmad, N.S, (2023). "A systematic review on recent advances in autonomous mobile robot navigation," Engineering Science and Technology, an International Journal, 40, p. 101343, pp. 1-26. https://doi.org/10.1016/j.jestch.2023.101343

Madhiarasan, M. and Deepa, S.N., (2016). "ELMAN neural network with modified grey wolf optimizer for enhanced wind speed forecasting," Circuits and Systems, 7(10), pp. 2975-2995. doi: 10.4236/cs.2016.710255

Mahboob, H., Yasin, J.N., Jokinen, S., Haghbayan, M.-H., Plosila, J. and Yasin, M.M., (2023). "DCP-SLAM: distributed collaborative partial swarm SLAM for efficient navigation of autonomous robots," Sensors, 23(2), p. 1025. https://doi.org/10.3390/s23021025

Miranda, M.H.R., Silva, F.L., Lourenço, M.A.M., Eckert, J.J. and Silva, L.C.A., (2023). "Particle swarm optimization of Elman neural network applied to battery state of charge and state of health estimation," Energy, 285(C), p. 129503. https://doi.org/10.1016/j.energy.2023.129503

Mohammed, H.A., Kareem, SH.W. and Mohammed, A.S., (2022). "A COMPARATIVE EVALUATION OF DEEP LEARNING METHODS IN DIGITAL IMAGE CLASSIFICATION," Kufa Journal of Engineering, 13(4), pp. 53–69. https://doi.org/10.30572/2018/KJE/130405

Nain, G., Pattanaik, K.K. and Sharma, G.K., (2022). "Towards edge computing in intelligent manufacturing: Past, present and future," Journal of Manufacturing Systems, 62, pp. 588–611.

Nelson, E., Corah, M. and Michael, N., (2017). " Environment model adaptation for mobile robot exploration," Autonomous Robots, 42(2), pp. 257-272. https://doi.org/10.1007/s10514-017-9669-2

Peng, Y., Wang, Z., Jin, J., Huang, P. and Lu, W., (2013). "The global trajectory programming of robot based on adaptive ant colony algorithm," in 2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics, IEEE, (1), pp. 108–111.doi: 10.1109/IHMSC.2013.33

Pianpak, P., Son, T.C., Dugas, P.O.T. and Yeoh, W., (2019). "A Distributed Solver for Multi-Agent Path Finding Problems," in Proceedings of the First International Conference on Distributed Artificial Intelligence, Oct, (2), pp. 1-7. https://doi.org/10.1145/3356464.3357702

Radmanesh, M., Kumar, M., Guentert, P.H. and Sarim, M., (2018). "Overview of path-planning and obstacle avoidance algorithms for UAVs: A comparative study," Unmanned Systems, 6 (02), pp. 95-118. https://doi.org/10.1142/S2301385018400022

Sahoo, S.K. and Choudhury, B. B., (2023). "A review of methodologies for path planning and optimization of mobile robots," Journl of Process Management and new Technology, 11(1–2), pp. 122–140. doi: 10.5937/jouproman2301122S

Samad, T., Iqbal, S., Malik, A.W., Arif, O. and Bloodsworth, P., (2018). "A Multi-Agent Framework for Cloud-Based Management of Collaborative Robots," International Journal of Advanced Robotic Systems, 15(4), pp. 1-13. https://doi.org/10.1177/1729881418785073

Seenu, N., Ramanathan, K.C., M.M., R. and Janardhanan, M.N., (2020). "Review on state-of-the-art dynamic task allocation strategies for multiple-robot systems," Industrial Robot: The International Journal of Robotics Research and Application, 47(6), pp. 929–942. https://doi.org/10.1108/IR-04-2020-0073

Seewald, A., de Marina, H.G., Midtiby, H.S. and Schultz, U.P., (2022). "Energy-aware planning-scheduling for autonomous aerial robots," in 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp. 2946–2953.

Tang, Z. and Ma, H., (2021). "An overview of path planning algorithms," in IOP Conference Series: Earth and Environmental Science, IOP Publishing, 804(2), 022024, pp. 1-10. doi: 10.1088/1755-1315/804/2/022024

Tolstaya, E., Gama, F., Paulos, J., Pappas, G., Kumar, V. and Ribeiro, A., (2020). "Learning Decentralized Controllers for Robot Swarms with Graph Neural Networks," in Proceedings of the Conference on Robot Learning, in PMLR Proceedings of Machine Learning Research, 100, pp. 671–682.

Vu, N.T.T., Tran, N.P. and Nguyen, N.H., (2021). "Recurrent neural network-based path planning for an excavator arm under varying environment," Engineering, Technology & Applied Science Research, 11(3), pp. 7088–7093. https://doi.org/10.48084/etasr.4125

Willners, J.S., Gonzalez-Adell, D., Hernández, J.D., Pairet, È. and Petillot, Y., (2021). "Online 3-dimensional path planning with kinematic constraints in unknown environments using hybrid A* with tree pruning," Sensors, 21(4), 1152, pp. 1-20. https://doi.org/10.3390/s21041152

Wu, A., Wang, Y., Shu, X., Moritz, D., Cui, W., Zhang, H., Zhang, D. and Qu, H., (2021). "AI4VIS: Survey on artificial intelligence approaches for data visualization," IEEE

Transactions on Visualization and Computer Graphics, 28(12), pp. 5049–5070. doi: 10.1109/TVCG.2021.3099002

Xing, W.Y., Bai, Y.L., Ding, L., Yu, Q.H. and Song, W., (2022). "Application of a hybrid model based on GA–ELMAN neural networks and VMD double processing in water level prediction," Journal of Hydroinformatics, 24(4), pp. 818–837. https://doi.org/10.2166/hydro.2022.016

Yang, L., Fu, L., Li, P., Mao, J. and Guo, N., (2022). "An effective dynamic path planning approach for mobile robots based on ant colony fusion dynamic windows," Machines, 10(1), 50, pp. 1-29. https://doi.org/10.3390/machines10010050

Yang, L., Li, P., Qian, S., Quan, H., Miao, J., Liu, M., Hu, Y. and Memetimin, E., (2023). "Path Planning Technique for Mobile Robots: A Review," Machines, 11(10), 980, pp. 1-46. https://doi.org/10.3390/machines11100980

Yen, C.-T. and Cheng, M.-F., (2018). "A study of fuzzy control with ant colony algorithm used in mobile robot for shortest path planning and obstacle avoidance,"Microsystem Technologies, 24(1), pp. 125–135. https://doi.org/10.1007/s00542-016-3192-9

Yuan, Q., Sun, R. and Du, X., (2022). "Path planning of mobile robots based on an improved particle swarm optimization algorithm," Processes, 11(1), p. 26. https://doi.org/10.3390/pr11010026

Zghair, N.A.K. and Al-Araji, A.S., (2021). "A One Decade Survey of Autonomous Mobile Robot Systems," International Journal of Electrical and Computer Engineering (IJECE), 11(6), pp. 4891-4906. http://doi.org/10.11591/ijece.v11i6.pp4891-4906

Zheng, L., Yu, W., Li, G., Qin, G. and Luo, Y., (2023). "Particle swarm algorithm path-planning method for mobile robots based on artificial potential fields," Sensors, 23(13), p. 6082. https://doi.org/10.3390/s23136082