International Journal of Robotics and Control Systems
Vol. 6, No. 2, 2026, pp. 1550-1572
ISSN 2775-2658
http://pubs2.ascee.org/index.php/ijrcs

AI-Driven Mobile Robot Navigation with Multi-Objective Task Scheduling and Reinforcement Learning

Abdullah A. Algethami [a,1,*]

[a] Mechanical Engineering Department, Taif University, Taif, Saudi Arabia
[1] a_algethami@tu.edu.sa
* Corresponding Author

ARTICLE INFO

Article history
Received January 10, 2026
Revised March 11, 2026
Accepted May 12, 2026

Keywords
Mobile Robot Navigation;
Reinforcement Learning;
Multi-Objective Task Scheduling;
Hybrid Metaheuristic Optimization;
Dynamic Obstacle Avoidance

ABSTRACT

Mobile robots are increasingly used in dynamic and cluttered environments, where efficient navigation, task execution, and energy management are critical. This study presents a hybrid AI-driven mobile robot navigation framework that integrates multi-objective task scheduling, reinforcement learning–based path planning, and model predictive control for trajectory tracking. A hybrid Sparrow Search–Bat Optimization method is employed to generate energy-efficient task scheduling, while a Deep Q-Network is used for collision-free path planning in dynamic environments. Obstacle detection and avoidance are supported using transformer-based deep learning models for environment perception, and a recharging strategy is included to support continuous operation. The proposed approach is validated through simulation studies in dynamic navigation environments and comparative analysis with existing optimization techniques. energy-efficient task scheduling. The results indicate improvements in navigation efficiency, reduced energy consumption, and higher navigation success rates (up to 97%), indicating the effectiveness of the proposed framework for mobile robot navigation applications.

1. Introduction

Motion planning for mobile robots has attracted significant interest due to its wide range of applications in social and industrial environments [1]-[4]. Advances in digital electronics and computing have enabled increasingly autonomous robotic systems that integrate artificial intelligence techniques [5]. Finding an appropriate collision-free path from the start position to the target location is important for the mobile robot to navigate in situations when obstacles are present [6]. The navigation path should be optimal or nearly ideal in terms of time, distance, energy, and smoothness throughout the robot motion sequence, which among the most desirable criteria is the minimum length. Mobile robot movement planning has been a developing field for the past forty years [7], [8].

A fundamental method of autonomous mobile robots, path planning is currently a hot topic in the field of mobile robot navigation development. Path planning may be divided into two categories based on the objectives and environments: local path planning and global path planning [9]. Researchers and academics have investigated the Global Path Planning (GPP) for known environments in detail [10]-[12]. Consequently, one of the most popular methods in use today is the optimized algorithm, which is a very efficient way to find the shortest path over a grid map. However, when a robot tracks this

planned course, it is not favorable to reducing collision risk since the path prepared by the standard algorithm is next to obstacles [13], [14]. Complex applications with human interaction in working circumstances, moving robots, or unexpected barrier appearance are in greater demand these days [15]-[17]. Currently, the dynamic environment is made up of moveable impediments, such as humans or robots. Since then, there has been a lot of interest in the camera application-based solution to the "Navigation" problem since it can overcome the drawbacks of the conventional line-detecting technique and also offers the capability to optimize the path [18]. Furthermore, the robot's working environment is typically a warehouse or small space, thus it must have a flexible operation that allows it to follow a predefined path and avoid any moving impediments without deviating from the safe planning path [19].

As a result, many conventional studies and methodologies are currently in use, with a primary focus on autonomous path planning over mobile robots in situations where the environment is static, mapped, and barrier positions are taken for granted [20]. Due to these approaches' time-consuming creation and upkeep of the dynamic environment map, the prediction's accuracy is low [21]. The primary issues with mobile robot planning for movement are computing complexity, adaptability, and choosing the best navigation path, which prevents scenarios like dead ends and guarantees safe and successful travel [22]. The scientific community has always looked for a different, more efficient way to approach these issues. Path planning is amenable to optimization, namely in terms of shortest distance, within specific constraints, such as specific circumstances and collision-free conditions [23], [24]. Recent advances in mobile robot navigation have increasingly incorporated deep learning, reinforcement learning, and sensor fusion techniques to improve perception, decision-making, and robustness in complex environments [25]-[30].

Robot route planning is one area where Deep Reinforcement Learning (DRL), a significant machine learning technique, is seeing an increase in interest and applications [31]. By trial and error, the agent learns by exploring its surroundings and gaining knowledge. Path planning benefits greatly from the DRL technique, which also requires less environmental knowledge beforehand [32]-[36]. While conventional DRL approaches such as DDPG and TD3 learn reactive control policies, they often struggle to capture long-horizon dependencies in sequential navigation decisions; the Decision Transformer addresses this limitation by modeling navigation actions using trajectory-level context and sequence learning mechanisms [37]-[40].

This study suggests a mobile robot tracking system driven by AI that is intended for dynamic decluttering purposes. Reinforcement learning will be used for adaptive path planning in dynamic situations, Multi-Objective Optimization will be used for task planning, and Deep Learning (DL) will be used for environment perception. The goal of this work is to create a reliable, effective, and flexible method for automating decluttering chores.

The creation and assessment of a mobile robot navigation system driven by AI for dynamic decluttering applications is the main goal of this work. This comprises:

- Constructing a DL model to perceive the surroundings and recognize objects.
- Using Multi-Objective Optimization approaches to schedule tasks and arrange paths while taking energy consumption, path length, and decluttering efficiency into account.
- Educating an RL agent that plans collision-free routes while adjusting to unforeseen impediments in dynamic situations.
- combining path tracking with motion control techniques to enable precise and fluid robot motions.
- creating a reliable system that integrates robot recharging into the task schedule.

The paper is organized as follows: Section 2 reviews related works; Section 3 presents the problem statement; Section 4 describes the system model; Section 5 details the proposed methodology; Section 6 presents the results and discussion; and Section 7 concludes the paper.

The research contribution of this study is the development of an integrated framework that combines multi-objective task scheduling, reinforcement learning–based path planning, and energy-aware navigation strategies to improve the efficiency and robustness of mobile robot navigation in dynamic environments.

2. Related Works

Utilizing a three-wheeled mobile robot, Le Duc et al [41] suggested an efficient real-time adaptive obstacle avoidance system based on Q-learning in the actual environment. Initially, the Q-learning process is simulated, and then, using trained data, the Q table result is applied to the actual mobile robot to enable it to carry out the task in the actual scene. The outcomes are contrasted with the intelligent control approach for scenarios involving both static and dynamic impediments. By conducting trials, the findings demonstrate that the mobile robot performs better to achieve the target position.

In an environment with erratic obstacles, Hongxia et al [42] have presented a robot route planning strategy that employs an expanded dynamic window approach and the improved A* algorithm to address the issue of failing along the planned path. The search point selection technique and evaluation mechanism are optimized in the re-vised A* algorithm to increase its effectiveness. This paper proposes an online path planning technique that combines improved dynamic window technique and GPP information to enable local obstacle avoidance.

Fatma, et al [43] propose a novel robotic system designed for the use of emergency scenarios and hospital settings when saving human life and taking immediate action are critical. The framework is primarily concerned with the accurate and timely distribution of pharmaceuticals or medical supplies inside a designated region, all the while preventing robot collisions and other obstructions. The proposed reinforcement learning-based route planning system (RPA) collects and transmits data between medical robots and human physicians, thereby optimizing medical services.

Ravi, et al [44] gathering information for surveillance purposes is a multi-objective optimization problem with time and energy restrictions. Three primary goals are taken into account when collecting data for this study. This approach proposes an intelligent particle swarm optimization (PSO) strategy which establishes fitness value by reducing the optimization problem for reaching the shortest path for MR navigation. Comparing the simulation results with various benchmark functions produced for PSO, it is clear that this PSO-based technique can realize high accuracy and fast convergence.

Tianrui and colleagues [45] have investigated hybrid path planning schemes to address the shortcomings of a single robot's task execution capacity. To start, a better PSO with a GPP model is suggested as a remedy for the inadequate effectiveness of robot path planning. To enhance the conventional artificial potential field (APF) technique, a multi-robot formation local route planning model is built using a simulated annealing algorithm. The proposed method can improve the path exploring and handling performance of robot formation.

A comprehensive strategy of DRL for independent mobile robot navigation in an unfamiliar environment is put forth by Min-Fan et al [46]. The authors suggest two forms of deep Q-learning (DQN) agents, namely DQN and double DQN (DDQN) agents, to allow mobile robots to learn about collision avoidance and navigation. When navigating an autonomous mobile robot in an unfamiliar area, a deep neural network model is used to recognize the target object. The DQN and DDQN technique is then used to navigate to the target object. In the test training, the DDQN agent performed 5.06% better than the DQN agent in finding the target object location.

Shumin et al [47] suggested an obstacle avoidance technique called the Trans-formable and Self-reconfigurable Omni-Directional Robotic Modules (STORM), to operate safely and intelligently in an unfamiliar environment. This study also presents a well-thought-out hardware and software framework with features that allow for future extension and parallel development created for the ongoing STORM programs. In addition, detailed tests and experiments carried out in both simulated

and real-world environments are described, along with comparisons between several training runs of the neural networks with various parameters.

Xunyu, et al [48] introduces a novel hybrid path planning technique that combines the adaptive window technique and the A* algorithm to perform real-time tracking, GPP, and obstacle avoidance for mobile robots. First, the risk expense and distance cost can be more easily calculated with the help of a safe A* algorithm. Second, to minimize the number of grid nodes required for efficient path tracking, important path points are taken out of the planned path produced by the safe A*. Ultimately, real-time motion planning with an adaptive window technique is utilized to accomplish both key path point switching and instantaneous path tracking and obstacle escaping.

A navigation control methodology is proposed by Nayab, et al [49] which utilizes a hybrid concept of grey wolves' optimization (GWO) algorithm and an APF method for on-time path planning of mobile robots. The proposed methodology runs in two folds. The first fold defense is the focus region (FR), which shows the obstacle-free lo-cations of all possible robot movements. In the second fold step, the GWO algorithm searches the shortest path by minimizing the APF value of the location generated within FR.

3. Problem Statement

Among the shortcomings of earlier research is that it attempted to apply an objective function, such as the quickest and/or smoothest path in a dynamic or static environment, without considering robot size. Furthermore, grid-based environment modeling wastes space and is rigid in dynamic contexts [50]. Inspired by the advantages and disadvantages of the previously mentioned research, this paper offers a collision-free shortest-path planning method that takes into account the size of the obstacles in the algorithm for routing and fits in both static and dynamic environments. The environments under consideration are modeled using a free-space modeling approach. One of thean important study subjects is obstacle avoidance for mobile robots to move from a start position to the intended target. This challenge remains an open problem in mobile robot research. Numerous problems plagued the traditional obstacle avoidance method in the dynamic, complicated environment. The flexibility and resilience needed for dynamic cleaning tasks are frequently lacking from existing mobile robot navigation technologies [51]. Manual design of obstacle avoidance and path planning is time-consuming and inefficient to manually design obstacle avoidance and path planning, particularly in surroundings that change frequently. Furthermore, conventional path planning algorithms might not maximize certain goals, such as energy conservation and clutter reduction effectiveness. This study offers an innovative strategy that incorporates the following to address these limitations.

- Deep Learning for analyzing the surroundings and recognizing objects in real-time.
- Multi-objective optimization to plan and schedule tasks more effectively.
- Adaptive navigation in changing circumstances with unforeseen obstacles using reinforcement learning.

4. System Model

The structure of the mobile robot is a wheeled device with a small caster wheel at the bottom and two separate primary wheels locked on the same axis up front. A ceiling-mounted RGBD camera is mounted at a height of 0.85 m above ground level. An overview of the robotic system is shown in Fig. 1.

The intended shortest-length path planning's computed desired location and orientation posture are shown by $R_d = [X_d, Y_d, \phi_d]^h$. The coordinate $R = [X, Y, \phi]^h$ is the real position and orientation as determined by the sensors mounted on the robot. A control mechanism is put in place to track the path that the suggested path-planning strategy creates to verify the algorithm. The locations of barriers

and other details about the immediate surroundings are gathered using computer vision techniques. To create the map that the suggested path planning method will use, this information is necessary. After creating an obstacle-free path between the starting location and the goal point, a kinematics controller helps the robot maintain the path by producing the instantaneous linear & rotational velocity (V_1 and V_2). This is done using the most recent readings of the robot's instantaneous attitude about the origin frame.

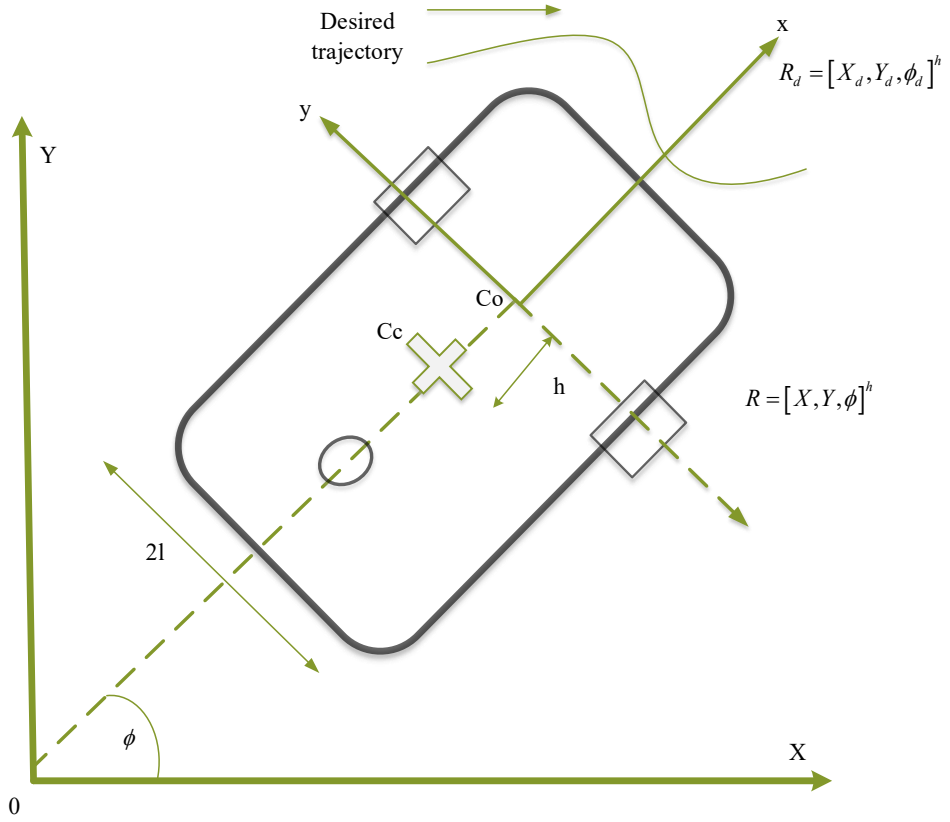

Fig. 1. Overview of the robotic system

5. Proposed Methodology

The overall hierarchical architecture of the proposed framework is illustrated in Fig. 2. The system is organized into distinct layers to ensure modularity and control clarity. At the mission level, the task scheduler performs multi-objective optimization using the Hybrid Sparrow Search and Bat (HSSB) algorithm to determine the optimal task sequence. The decision layer consists of the DQN, which generates discrete navigation actions toward the scheduled task goal. The Decision Transformer operates as an obstacle-aware refinement module, adjusting motion decisions under dynamic environmental conditions. These decisions are filtered through dynamic window–based feasibility constraints before being executed by the Improved Model Predictive Control (IMPC) module, which produces the final steering command. The recharging module supervises battery status and temporarily overrides mission execution when energy falls below a predefined threshold. This layered design ensures structured signal flow, stability, and physical feasibility.

Real-time feasibility: Inference is performed once per control step, and the IMPC executes at the control sampling rate. The overall latency depends on the selected onboard compute hardware and the chosen sampling time; hardware timing benchmarks are left for future work.

5.1. Task Scheduler

Create an effective work plan by using Multi-Objective Optimization strategies to minimize energy consumption, robot idle time, path length, job completion time, and overall task completion

time. Utilizing the hybrid optimization technique, the method of optimization looks for a set of results that reflect trade-offs between these competing aims.

5.1.1. Hybrid Sparrow Search and Bat (HSSB) Optimization

In the proposed framework, task scheduling is formulated as a multi-objective combinatorial optimization problem involving task order, path efficiency, and energy consumption. This structure resembles routing and scheduling problems such as Traveling Salesman–type formulations, which are generally NP-hard; therefore, the Hybrid Sparrow Search and Bat (HSSB) algorithm is employed to efficiently obtain near-optimal solutions in dynamic environments [52]-[54].

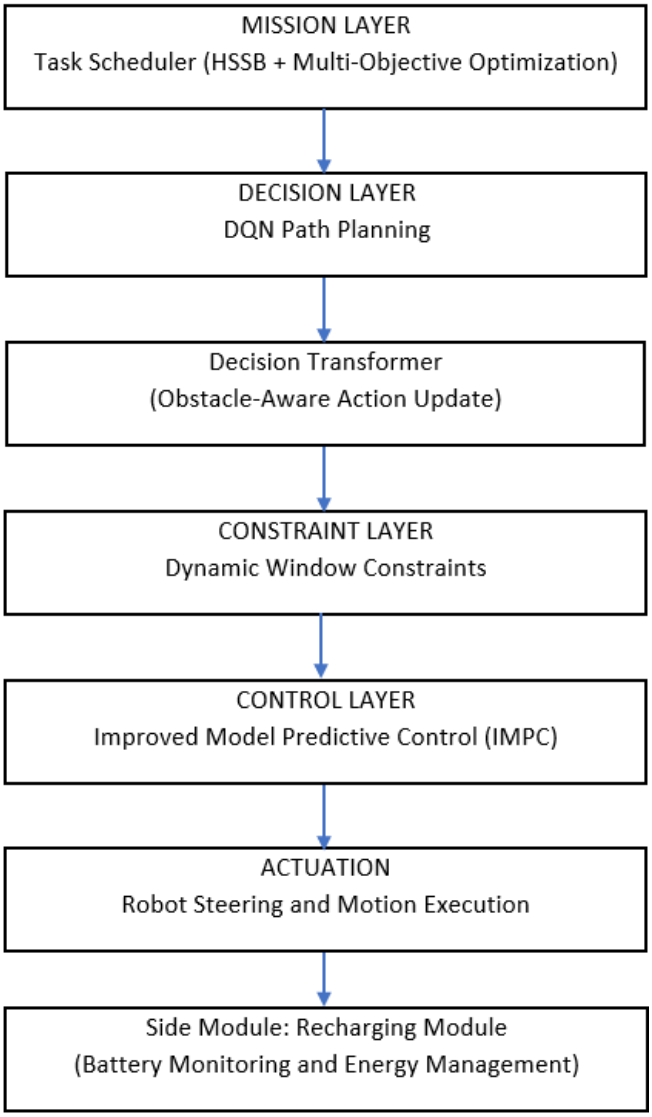

Fig. 2. Hierarchical architecture of the proposed robotic navigation framework

Develop a mathematical framework to build the algorithm for finding sparrows based on the prior sparrow description. The sparrows emit frightening chirps as soon as they identify the predator [55]. The producers must guide all scavengers to a safe location when the alarm value exceeds the safety threshold. Every sparrow has the potential to become a producer if it looks for additional food sources of information, but the share of producers and scavengers in the population remains constant. The producers would be the birds with the highest energy. To obtain additional energy, several famished scavengers are more likely to take to the skies in search of food. The scavengers hunt for food by trailing the producer who can offer the best food. The following Eq. (1) can be used to illustrate the sparrows' positions:

$$\ddot{X} = \begin{bmatrix} \ddot{X}_{1,1} & \ddot{X}_{1,2} & \dots & \ddot{X}_{1,d} \\ \ddot{X}_{2,1} & \ddot{X}_{2,2} & \dots & \ddot{X}_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ \ddot{X}_{k,1} & \ddot{X}_{k,2} & \dots & \ddot{X}_{k,d} \end{bmatrix} \quad (1)$$

Let, $k$is denoted as the number of sparrows and d displays the dimension of the factors that need to be adjusted. Then, the subsequent vector can be used to express the overall sparrows' fitness number in Eq. (2).

$$f\ddot{X} = \begin{bmatrix} f\ddot{X}_{1,1} & f\ddot{X}_{1,2} & \dots & f\ddot{X}_{1,d} \\ f\ddot{X}_{2,1} & f\ddot{X}_{2,2} & \dots & f\ddot{X}_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ f\ddot{X}_{k,1} & f\ddot{X}_{k,2} & \dots & f\ddot{X}_{k,d} \end{bmatrix} \quad (2)$$

Each row's value in $f\ddot{X}$ symbolizes the person's level of fitness. In the search procedure for food in the SSA, producers with higher fitness levels are given precedence. Furthermore, it is the producers' job to look for food and direct the population's movement. As a result, producers have access to a wider variety of locations for food hunting than do scavengers. The producer's position is updated as follows during each iteration by Eq. (3).

$$\ddot{X}_{i,j}^{t} = \begin{cases} \ddot{X}_{i,j}^{t}.exp\left(\frac{-i}{\delta.I_m}\right) & if(r_2 < s_t) \\ \ddot{X}_{i,j}^{t} + G.U & if(r_2 \geq s_t) \end{cases} \quad (3)$$

Let, t indicates the current iteration, $j = 1,2,3,\dots.d$. $\ddot{X}_{i,j}^{t}$ symbolizes the worth of the j^{th} size of the i^{th} sparrow during the repetition t. I_m is the constant that has undergone the most iterations. $\delta \in (0,1)$ is denoted as a random number. $r_2(r_2 \in [0,1])$ and $s_t(s_t \in [0.5,1.0])$ Indicate the safety threshold with the alarm value. G is an arbitrary number that follows the normal distribution. Udisplays a matrix of $1 \times d$ which each internal element is 1. When $r_2 < s_t$, indicating the absence of predators, the producer switches to wide searching mode. If $r_2 \geq s_t$, it indicates that a few sparrows have found the predator, and the rest of the sparrows must immediately fly to other secure places.

Regarding the scavengers, they must uphold regulations. As was already established, some scroungers keep a closer eye on the producers. They instantly abandon their current position to battle for food as soon as they learn that the producer has discovered nice food. If they triumph, they can immediately take the food from the producer; if not, they must continue following the guidelines. In this phase, update the velocities of bats $\vec{V}_i^t$ for enhancing the search space process [56]. The following is an explanation of the scrounger's position update using Eq. (4).

$$\ddot{X}_{i,j}^{t+1} = \begin{cases} G.exp\left(\frac{\ddot{X}_{worst}^{t} - \ddot{X}_{i,j}^{t}}{i^2}\right) + \vec{V}_t^i & if\left(i > \frac{k}{2}\right) \\ \ddot{X}_{op}^{t+1} + \left|\ddot{X}_{i,j}^{t} - \ddot{X}_{op}^{t+1}\right| + W^+.U + \vec{V}_t^i & otherwise \end{cases} \quad (4)$$

Let, $\ddot{X}_{op}$ is the ideal position that the producer holds. $\ddot{X}_{worst}$ indicates the worst place on Earth right now. W depicts a matrix of $1 \times d$ This assigns a random element within each 1 or−1, and $W^+ = W^t(WW^t)^{-1}$. When $i > \frac{k}{2}$, It implies that the i^{th}Scroungers with the lowest fitness rating are probably starving. The velocities of the bat are determined using Eq. (5).

$$\vec{V}_i^t = \vec{V}_i^{t+1} + (\vec{X}_i^t - \vec{X}_*)\vec{f}_i \quad (5)$$

Let, $\vec{X}_i^t$ is denoted as a new search space solution for the bats, $\vec{X}_*$ is considered the current global best solution, $\vec{f}_i$ is denoted as frequency. These birds' starting locations are chosen at random from the

population. This phase involves updating the new solution $\vec{\vec{X}}_{new}$ for each bat for the local search portion. A local random walk is used to produce the new solution for each bat after a solution is chosen from the existing pool of best solutions. The mathematical framework can be represented as follows in Eq. (6).

$$\ddot{X}_{i,j}^{t+1} = \begin{Bmatrix} \ddot{X}_{best}^{t} + \alpha.\left(\ddot{X}_{i,j}^{t} - \ddot{X}_{best}^{t}\right) + \vec{\vec{X}}_{new} & if\left(f^{i} > f^{g}\right) \\ \ddot{X}_{i,j}^{t} + H.\left(\frac{\left|\ddot{X}_{i,j}^{t} - \ddot{X}_{worst}^{t}\right|}{(f^{i} - f^{r}) + \varepsilon}\right) + \vec{\vec{X}}_{new} & if\left(f^{i} = f^{g}\right) \end{Bmatrix} \quad (6)$$

Let, $\ddot{X}_{best}$ is the stage size control variable, and the current global ideal location is an arbitrary number distribution having a mean of 0 and a variability of 1. $H \in [-1,1]$ is considered a random number. Here f^{i} is the current sparrow's fitness value. f^{g} and f^{r} are the finest and poorest fitness values, ε is the minimum constant necessary to prevent a zero-division error. For ease of understanding, when $f^{i} > f^{g}$shows that the small bird is at the group's boundary. $\ddot{X}_{best}$symbolizes the location of the population center and the area that is safe around it. $f^{i} = f^{g}$ demonstrates that the middle-sized group of sparrows is aware of the threat and needs to go closer to the others. H is both step by step size control coefficient and indicates the direction of the sparrow's movement.

5.1.2. Energy Consumption Model

Saving Energy in Mobile Robots: It is imperative for mobile robots to effectively manage their energy usage when operating alone and in teams. Due to unequal allocation of energy, a group of MRs working on a job may never be able to complete it or may grow less tolerant of unfavorable circumstances that arise while the task is being completed. Since the MR uses a variety of sensors for a range of actions, the energy usage differs based on the behavior. The energy used by MRs' sensors and motors has a major impact on their energy level. The total energy capacity of an MR, or the amount available once its battery is fully charged, is a fixed amount. The quantity of energy left (e_r) Following completion of MRs' surveillance assignments is computed using Eq. (7).

$$e_r = (e_i - e_u) \quad (7)$$

Let, e_i is what MR's starting energy with a fully charged battery, and e_u is the total amount of energy expended while monitoring.

5.1.3. Information Density Model

Information density: Users will probably be able to estimate the unknown quantity for further coordinate locations in the area of surveillance by utilizing the density measurement of discrete information. The fitting approach is appropriate when working with discrete data. The process of fitting involves creating a mathematical function that most closely matches a set of discrete data points. The link between is where the minimizing difficulty originates $f(a,b)$and σ in Eq. (8).

$$\min_{f \in \Pi_n^2} \sum_{q} f(a_q, b_q) - \sigma_q^2 \quad (8)$$

The function for data dispersion is denoted by f and acquired from Π_n^2. Let's take that into consideration n points reachable from specific places (a_q, b_q)in $\Re^2$, where $q \in \{1,2,3,\ldots,n\}$. The information density is σ_q for the region of i^{th}grid. The importance of $f(a,b)$is indicated by $f(a_q, b_q)$ within the role (a_q, b_q)It calculates the given scalar quantities σ_q.

The information distribution system for the surveillance region is defined as follows: $f(a,b)$ Eq. (9) provides the amount of information, and the total surveillance area is indicated by E_s.

$$E_s = \iint_{a_s} f(a,b) da db \quad (9)$$

Let, a_s is the region including the whole surveillance area. Eq. (10) Shortcuts the amount of data collected by the MR:

$$D_s = \sum_{i=1}^{n} f(a_i, b_i)\, Sr_i \tag{10}$$

Sr_i displays the monitoring area's search area r_i.

5.2. Path Planning and Motion Control

In a simulated setting, teach a Reinforcement Learning agent (Deep Q-Network (DQN)) [57] to plan courses that avoid collisions while taking into account dynamic barriers and optimizing for many objectives. Multi-objective optimization in reinforcement learning might entail learning rules that strike a balance between goals like optimizing navigation speed, lowering collision risk, and using the least amount of energy. Every iteration of an RL-based agent provides the action to be performed in the environment after receiving two inputs: the state and the reward. The following provides a detailed description of the state, action, and reward utilized by mobile robot path planning utilizing DRL in unknown situations. Reinforcement learning is employed because classical planners struggle in unknown and dynamic environments where obstacle configurations change over time. The DQN enables adaptive decision-making through interaction with the environment and reward shaping, allowing the robot to optimize speed, safety, and energy consumption simultaneously.

State: A state is an environmental observation that explains the circumstances of the moment. This is significant to the agent since it would compute and respond based on the condition. Of the 29 states, 24 have laser distance sensor (LDS) values. The remaining five are the following: the angle to the target object location, the distance to the closest obstacle, the assurance of the recognized target object, and the travel distance to the goal and closest obstacle. Eq. (11) expresses the state.

$$s(t) = l_1 + c_1 + d_1 + a_1 + d_2 + a_2 \tag{11}$$

Let, $s(t)$: state; l_1: LDS (24 values); c_1: confidence of the detected target object (1 value); d_1: distance to the goal (1 value); a_1: angle to the goal (1 value); d_2: distance to the nearest obstacle (1 value); and a_2: angle to the nearest obstacle (1 value).

Action: Action is what an agent can do in each state. The mobile robot has five actions that it can act on depending on the type of state. The mobile robot has a fixed linear velocity of 0.15 m/s and the angular velocity is determined by the action. Initially, the agent knows nothing about the environment, so the agent cannot distinguish between the actions in the starting state in the first episodes.

The choices the agent makes determine the information it gathers and, consequently, the information it can learn from. Our agent's policy can be described as a function that accepts states and returns actions, as shown in Eq. (12).

$$A(t) = f\big(s(t)\big) \tag{12}$$

where $A(t)$is denoted as action; $f$is denoted as function and $s(t)$is considered a state.

Reward: Rather than aiding in the agent's prediction-making process, reward serves as a means of reaching a decision. A reward is a function that, for an agent in a given state and action, returns a scalar number that represents a positive reward along with a negative reward (penalty). The incentive gauges how effective it is to carry out a particular action for a particular state. Based on the incentives obtained for different state–action combinations, an agent modifies its policy. In general, some acts are rewarded positively to promote them, and other actions are discouraged by penalties. A thoughtfully created reward serves as the agent's guide to optimize the cumulative reward.

The mobile robot and the target object position in the training simulation are initially set to (0, 0) and random locations, respectively, at the beginning of the first program in the training phase of the

simulation. When the mobile robot finds the goal object, it will begin exploring its surroundings in quest of it and will be rewarded with 0 before it does. The DQL agent gets paid 500 right away after identifying the target item with a confidence value higher than the minimum. Next, the reward function ($r(\theta)$) for an angle can be obtained using Eq. (13) from the angular direction.

$$r(d) = 2^{\frac{d_c}{d_g}\Lambda(\theta)} = \begin{Bmatrix} > 2, & d_c < d_g \\ [1,2] & otherwise \end{Bmatrix} \quad (13)$$

Λ: a function of the angle between the moving robot and the objective; d_c: current separation from the objective; d_g: total separation from the objective. The mobile robot's angular direction indicates whether or not it is facing the destination directly. For example, the angle is 0 if the robot turns exactly toward the goal ($\theta = 0$) and the activity is equivalent to $A(t) = 2$. A robot's action varies by the angle it rotates in, which varies from 0 to positive 180 when it rotates in a clockwise motion from the objective, and in the opposite direction, which varies from 0 to negative 180 when it rotates counterclockwise from the objective. Depending on the angle between the robot and the target, the action taken by a block of lines tells what needs to be done. The robot's angular direction towards the goal is explained by Eq. (14).

$$\theta = \frac{\pi}{2} + A(t) + \frac{\pi}{8} + \psi \quad (14)$$

Let, $A(t)$is denoted as action; θ is considered as the angle between the objective and the mobile robot; ψ: is the yaw of the mobile robot. If the angle falls within negative 90 and positive ninety degrees, the reward is higher or equal to 0, and if it falls outside of those ranges, it is less than 0. Additionally, the reward system ($r(d)$) for the distance can be obtained using Eq. (15) based on the linear direction.

$$r(\theta) = 5 \times 1^{-\Lambda(\theta)} = \begin{Bmatrix} \geq 0, & \frac{-1}{2}\pi < \theta < \frac{1}{2}\pi \\ < 0, & otherwise \end{Bmatrix} \quad (15)$$

The robot's progress to the destination is shown by the linear direction. Equation (12) states that the reward from the distance is over two if the current location of the goal is less compared to the absolute distance to the goal, and less than or equivalent to 1 if the current separation from the goal is higher or equal to the unconditional distance from the goal. The DQL agent's development is directed by the reward function in Eq. (16).

$$r(t) = r(d) + r(\theta) \quad (16)$$

Let, $r(\theta)$: angle reward; $r(d)$: distance reward; and $r(t)$: reward. The reward function for multi-objective reinforcement learning consists of the weighted average of the individual objectives. It establishes the reward function for the developed model as detailed in Eq. (17).

$$r(t) = W_1.r(speed) + W_2.r(collision) + W_3.r(energy) \quad (17)$$

Let, $r(speed)$ is denoted as the maximizing navigation speed reward, $r(collision)$ is the minimizing collision risk-reward, $r(energy)$ is the minimizing energy usage reward, and W_1, W_2, and W_3 are the weights attributed to every goal.

Ultimately, as the mobile robot approaches the (goal) target object position, the DQL agent is rewarded with 100, and it incurs a penalty of -100 if it goes further away from the desired object location or runs into obstacles. The DQN method is a method for approximating the value function via a neural network. The neural network's weight is changed to approximate the ideal value function. The parameters are modified by the update value function. The parameters are set once the neural network training is finished, and the associated function value will remain constant. After that, the training phase converges. The Eq. (18) for the location updates is:

$$Q(s(t),a(t)) = Q(s(t),a(t)) + \alpha[r(t) + \gamma\, max_{a'(t)}\, Q\,(s(t+1),a'(t)) - Q(s(t),a(t))] \quad (18)$$

α is denoted as the learning rate, γ is considered a discount factor, $r(t)$ is denoted as an immediate reward, and $s(t+1)$ is seen as the subsequent state of the following action $a(t)$. In practice, the sample velocity range must be constrained by the mobile robot's limitations as well as external constraints, even though there are an endless number of groups in the velocity space. The following Eq. (19) Expression for mobile robot speed constraints:

$$S_m = \{(s,e)|s \in [smax_{min}, e \in [emax_{min}]]|\} \quad (19)$$

The speed restriction of the mobile robot resulting from the motor's addition and subtraction limitations can be computed in the dynamic window's moving period using the following Eq. (20).

$$S_d = \{(s,e)|s \in [s_c - \hat{s}_b \Delta t, s_c + \hat{s}_a \Delta t], e \in [e_c - \hat{e}_b \Delta t, e_c + \hat{e}_a \Delta t]|\} \quad (20)$$

Let, s_c and e_c denote current speed, $\hat{s}_a$ and $\hat{e}_a$ indicate the mobile robot's maximum acceleration., $\hat{s}_b$ and $\hat{e}_b$ indicate the mobile robot's greatest deceleration. The mobile robot's safety must be guaranteed when avoiding impediments in its immediate surroundings. The speed can be lowered to 0 m/s before contact under the maximum deceleration constraint; this deceleration constraint is stated as Eq. (21).

$$S_d = \left\{(s,e)\,\middle|\, s \le (2d(s,e)\hat{s}_b)^{\frac{1}{2}}, e \le (2d(s,e)\hat{e}_b)^{\frac{1}{2}}\right\} \quad (21)$$

Let, $d(s,e)$ indicates the shortest path between the two trajectories (s,e) and the obstacle. When the device is in a condition $s(t)$, After determining the maximum value of the related actions in that state, the robot will take the appropriate action and modify the information in that cell $Q(s(t),a(t))$. Continue until the robot has mastered the rules (via the Q-table) to enable it to avoid obstacles in a variety of scenarios.

5.3. Path Tracking System

Improved Model Predictive Control (IMPC) is used in this phase to follow the course accurately even in the presence of small disruptions. A mobile robot has to make decisions at every sample time because the traffic and road conditions are always changing. In addition, the MB can only be used inside the legal limits of the road. As a result, route following controller design is based on the MPC approach. The sideslip angle, yaw rate, and yaw angle are among the pertinent states that are taken to be directly retrieved by an RT3002 onboard sensor unit in this publication. Global Positioning System (GPS) signals allow for immediate measurement of the vehicle's position. There is no consideration for the actuator's management, or the steering motor. It is vital to forecast the vehicle's potential output states to reduce the lateral mismatch between the current position and the intended lateral position. The current plant data is provided by the vehicle states and can be obtained through assessment. The IMPC framework is selected to ensure robust trajectory tracking under dynamic disturbances and model uncertainties. By incorporating prediction and control horizons, IMPC anticipates future system states and minimizes lateral deviation while respecting actuator constraints.

Assuming that the time t; $t > 0$; The current vehicle variables can be used to anticipate the future dynamics of the vehicle, and the prediction horizon is p_h as the optimization window's duration. The sample time and prediction horizon can be used to calculate the predictive steps. Assumed to be the control horizon, the predictive horizon benefits greatly from the control action c_h. The control horizon & sample duration can also be used to compute it. The correlation between the control horizon & the predictive horizon is $p_h \ge c_h$. Assume that the stage of prediction is P and that the stage of control is C. This indicates that following a time step, the control input stays constant. C, that is $u(t+C) = u(t+C+1) = \ldots = u(t+P-1)$Furthermore, presuming that throughout the prediction horizon the longitudinal velocity remains invariant. In this instance, given the mobile robot's present conditions, the \following Cfuture output states can be anticipated using Eq. (22).

$$x(t+1/t) \triangleq W_{xx}x(t) + W_{xu}u(t) + W_{xn}n(t)$$
$$y(t+1/t) \triangleq W_x x(t) + W_u u(t) + W_n n(t) \quad (22)$$

One of the control demands is to make the mobile robot journey as precisely as possible toward the road's centerline. From this, the formulas for the vectors and matrices in Eq. (22) can be inferred. Accordingly, the reference input sequence that defines the system's desired outcome is defined as follows in Eq. (23).

$$r(t) = \left[f\left(x_{road,t}\right), f\left(x_{road,t+1}\right), \ldots, f\left(x_{road,t+P-1}\right)\right]^k \quad (23)$$

Let, $f\left(x_{road,t+i-1}\right)$is derived from the centreline's discretization. Then, reduce the gap between the expected output and the road centerline to guarantee that the vehicle conforms to the road centerline while staying inside the designated viable region. Particularly, the mechanical method's restrictions limit the steering motor's conceivable operations. Furthermore, the lateral and longitudinal distances traveled in a single sampling interval can be written as Eq. (24).

$$\Delta x_n(t+i) = v(t).n_s, \Delta y_n(t+i) = y_o(t+i-1), i = 1, \ldots . P \quad (24)$$

The mobile robot may reduce its energy consumption if it can find the shortest path among every pair of points in each predicted interval. To minimize driving routes, weighing criteria are included. Consequently, the following multi-objective cost function for path tracking is obtained using Eq. (25).

$$Q = \left\|\Gamma_y\left(y(t+1/t) - r(t)\right)\right\|^2 + \|\Gamma_u u(t)\|^2 + \sum_{i=1}^{P} \Gamma_{n,i}(\|\Delta x_n(t+i)\|^2 + \|\Delta y_n(t+i)\|^2) \quad (25)$$

Let, Γ_y and Γ_u are the matrices used for weighting and $\Gamma_{n,i}$ are the variables used for weighting. Limiting the front end's lateral locations helps prevent collisions with the road's edges ($\tilde{f}$) and rear end ($\tilde{r}$) of the MB to stay inside the designated driving lanes in Eq. (26).

$$\tilde{f}'(t) \le y_i(t) \le \tilde{f}_l'(t), i = \tilde{f}, \tilde{r} \quad (26)$$

The connection involves the front end's $\tilde{f}$lateral position and the vehicle, rear-end $\tilde{r}$, and the following is how the expression can be explained in Eq. (27) and (28).

$$y_{\tilde{f}}(t) = y\ \ (t) + l_{\tilde{f}}\, sin(\psi + \alpha)\,(t) \quad (27)$$

$$y_{\tilde{r}}(t) = y\ \ (t) + l_{\tilde{r}}\, sin(\psi + \alpha)\,(t) \quad (28)$$

Let, y is considered as the vehicle's lateral position at the CoG; $y_{\tilde{f}}$ and $y_{\tilde{r}}$is denoted as the front-end and rear-end lateral position of the vehicle; ψ is considered as yaw angle; and α is denoted as sideslip angle. Next, the connections between y, $\tilde{f}$, $\tilde{r}$ can be made simpler using Eqs. (29) and (30).

$$y_{\tilde{f}}(t) = y\ \ (t) + l_{\tilde{f}}\, s(\psi + \alpha)\,(t) \quad (29)$$

$$y_{\tilde{r}}(t) = y\ \ (t) + l_{\tilde{r}}\, s(\psi + \alpha)\,(t) \quad (30)$$

The actuator's properties, namely the ranges within which the steering angle and steering rate are feasible, must be taken into account to produce a workable control input using Eqs. (31) and (32).

$$\delta\tilde{f}(t+1)_{\tilde{f}_{max}\tilde{f}_{min}} \quad (31)$$

$$\delta\tilde{f}(t+1)_{\tilde{f}_{max}\tilde{f}_{min}} \quad (32)$$

Let, $\Delta\delta_{\tilde{f}}(t+1)$is denoted as control increment; $\delta_{\tilde{f}_{max}}$the greatest steering angle of the front wheels; and $\delta_{\tilde{f}_{min}}$is the minimum front wheel steering angle. After resolving the aforementioned optimization issue, the initial component of the ideal control sequence $u(t)$ is used with the automotive system. Therefore, at that precise moment t, Thus, a state feedback control law is created using Eq. (33).

$$\hat{u}_o^* = [1,0,\ldots.,0]u(t) \tag{33}$$

Let, $\hat{u}_o^*$ is the best vector for the angle of the front wheel steering and acts as a control signal for the MB. The forecast horizon is advanced by one interval at the subsequent time instant, and the optimization issue is resolvable with the recently captured process measurements.

5.4. Obstacle Avoidance

Obstacle recognition is performed using real-time environmental observations obtained from the Laser Distance Sensor (LDS) and RGBD-based depth mapping. The LDS provides 24 directional distance measurements, while the RGBD sensor contributes spatial depth information to estimate obstacle proximity and free-space regions. These sensor readings are integrated into the state representation of the DQN and Decision Transformer modules to enable environment-aware decision-making. Conditional action generation, in which the output action is dependent on the intended future rewards denoted by the reward-to-go (RTG), is the fundamental concept underlying Decision Transformers (DT) [58] using Eq. (34).

$$\overrightarrow{r_t} = \sum_{t'=t}^{T} R_t^{'} \tag{34}$$

Let, t and T consist of the final phase of time and the present time step. A series of RTGs, findings, and actions are fed into the DT. This series is known as a trajectory ςand is expressed in Eq. (35).

$$\varsigma = (\overrightarrow{r_0}, a_0, o_0, \ldots.\overrightarrow{r_t}, o_t) \tag{35}$$

The output sequence's ground truth is a_t . Then choose cross-entropy ξ_{ce} because employ discrete activities for lost function training. The greatest collected reward for the task is estimated to resemble the true RTG, which is unknown at inference time. Then utilize the detection rating since the greatest reward for every single step, the initial RTG setting $\overrightarrow{r_0}$ to the duration of an episode T. Next, the true reward received at each stage is subtracted to update the RTG $\overrightarrow{r_{t+1}} = \overrightarrow{r_t} - R_t$.

Batch sampling via the data buffer is necessary for DT training since it adheres to the same methodology as supervised training. The duration of the trajectory determines the likelihood of sampling a single trajectory. To promote exploration and give the model access to a wide range of samples, sample the data buffer proportionately to the variation of the rewards in the trajectory. Utilize the variability of the rewards as a tractable proxy because the variance of the states is hard to compute. In particular, the probability for the i-the trajectory is detailed in Eq. (36).

$$Pr_i = \frac{V(\varsigma_i)}{\sum_i^n V(\varsigma_i)} \tag{36}$$

Let, n is the total amount of paths used during training. Then include an entropy keyword ξ_{en} to further improve exploration of the predicted action, leading to the ultimate loss function using Eq. (37).

$$\xi = \xi_{ce} - \lambda\xi_{en} \tag{37}$$

Let, λ is experimentally set as 0.1 to equalize the two training losses. The architecture of the decision transformer is shown in Fig. 3.

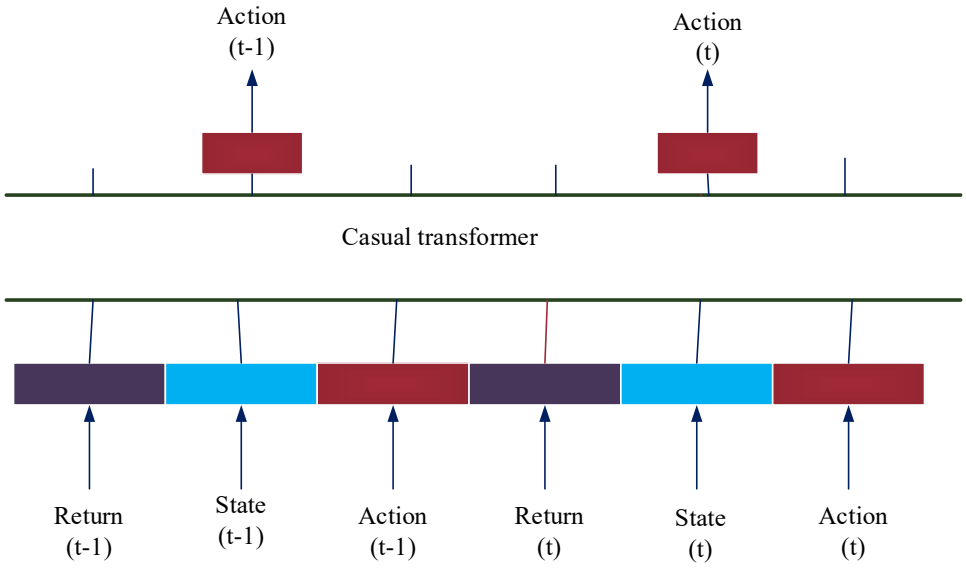

Fig. 3. Design of DT

5.5. Recharging Module

Design a charge strategy in the work plan that will direct the robot to a charging station when its battery runs low. Initially, incorporating a recharging technique into an autonomous robot's task schedule is to continuously monitor the robot's battery level. The operational structure of the robot establishes a threshold battery level that, when achieved, initiates the recharging procedure. When the robot does its assigned duties, it determines if there is enough battery life left to finish the mission and, in case it is needed, to find the closest charging station. If the battery level drops below the predetermined threshold while the task is being executed, the robot will instantly halt and use path planning to find the shortest route to the nearest charging station. When the robot reaches the charging station, it begins to recharge itself until its battery is fully restored. The robot resumes its paused duty after recharging, thus there is no significant disturbance to its scheduled operations. This tactic makes sure the robot stays operating at peak efficiency and prevents unplanned shutdowns brought on by low battery levels, which improves the robot's dependability and productivity.

6. Results and Discussion

The developed model technique is implemented in the Python tool and the performance of the designed technique is validated with existing optimization techniques such as performance score, energy consumption, navigation success rate, and total path length. The process of developing a model to reach the destination is shown in Fig. 4. For fairness, GA, PSO, and ACO were implemented as global path planners without adaptive local replanning mechanisms. All methods were evaluated under identical dynamic simulation conditions across 100 independent trials.

6.1. Performance Metrics

The proposed system will be evaluated in a simulated environment with varying degrees of clutter and dynamic obstacles. Metrics for evaluation will include path length, energy consumption, navigation success rate, performance score. The developed model is validated with existing optimization techniques such as Genetic Algorithm (GA) [59], Particle Swarm Optimization (PSO) [60], and Ant Colony Optimization (ACO) [61]. Path length is evaluated in terms of accumulated grid-based distance (number of discrete cells traversed), rather than physical distance in meters.

Fig. 5 compares the performance of different methods in terms of path length, measured as the total number of grid steps. PSO achieved a path length of 136.48, GA reached 170.71, and ACO obtained 61.87. Under the same simulation conditions, the proposed method resulted in the shortest path, indicating a significant reduction in travel distance. Path length values were computed over

successful trials only. Failed trials (due to collision or timeout) were excluded from the path length averaging.

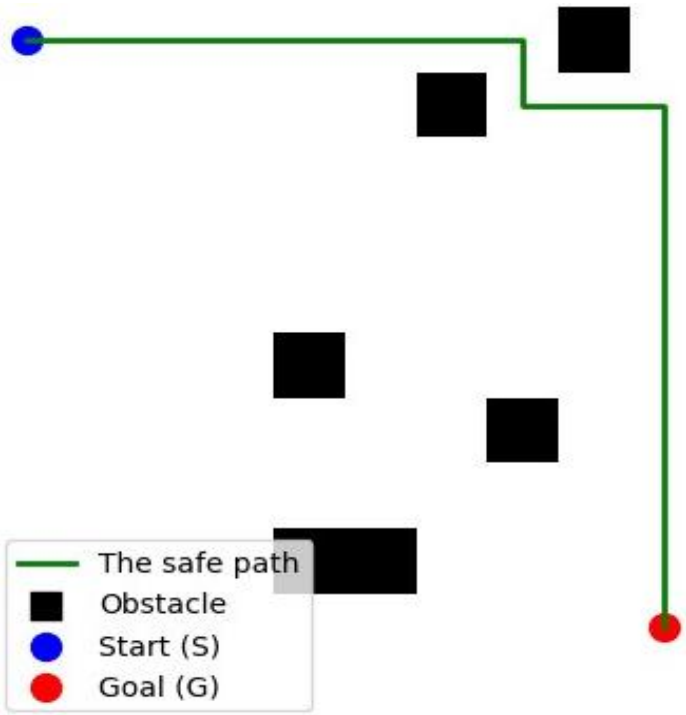

Fig. 4. Execution diagram of the developed model

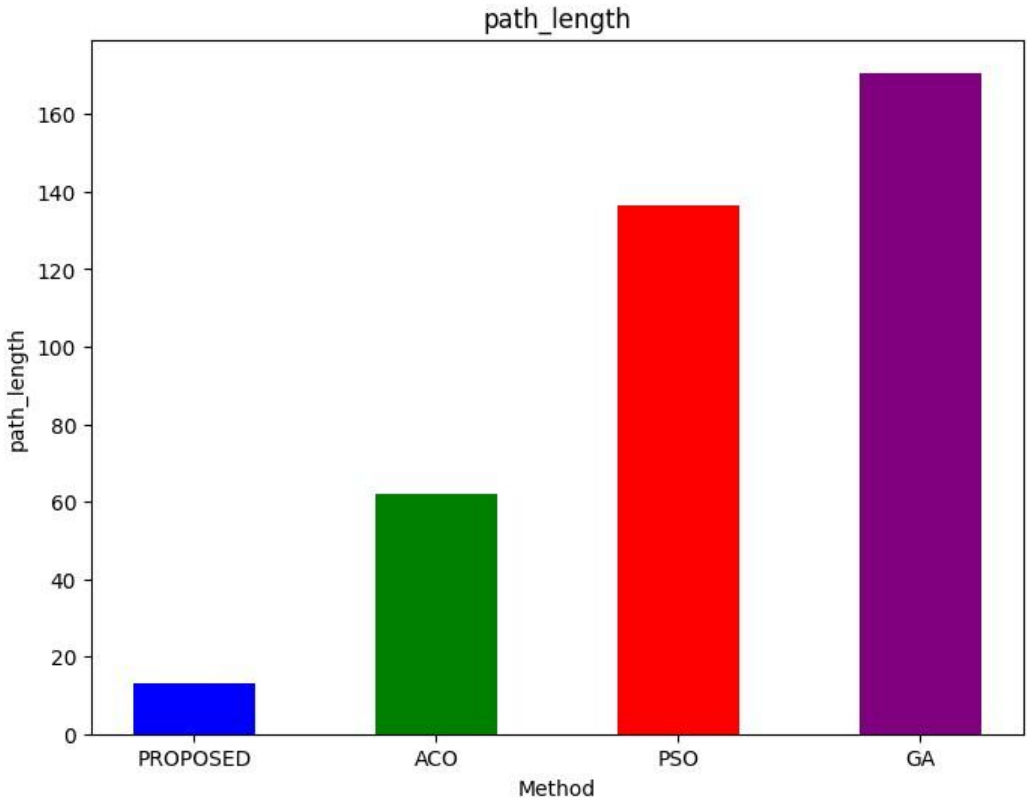

Fig. 5. Comparison of total grid-based path length achieved by each method in the simulated environment

Energy Consumption: Battery usage during the task.

Fig. 6 presents a comparative analysis of several procedures concerning their energy use. PSO uses somewhat less energy than GA, with a value of 8 units, compared to GA's 9 units. ACO displays a 12-unit higher energy consumption. On the other hand, the Proposed method's energy consumption of 8 units matches PSO, demonstrating its efficiency in energy use.

Navigation Success Rate: Ability to reach target locations and avoid collisions

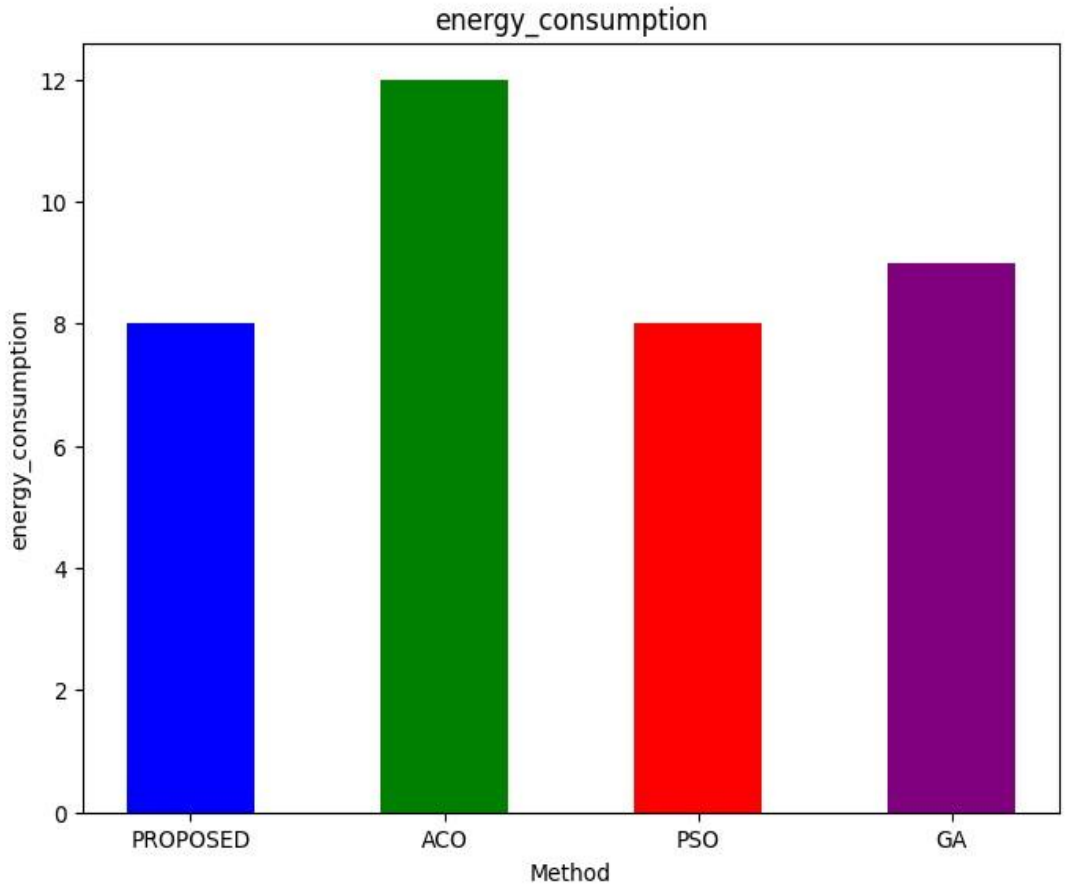

Fig. 6. Comparison of total energy consumption (battery units) for each method during task execution

The low success rates of GA (5%), PSO (19%), and ACO (6%) are attributed to their non-adaptive global optimization structure shown in Fig. 7. In dynamic environments with time-varying obstacles, these methods lack real-time trajectory correction, leading to frequent mission failures. In contrast, the proposed framework integrates learning-based policy refinement and dynamic feasibility filtering, enabling robust navigation under changing conditions.

Performance score: A mobile robot's total performance score is a comprehensive metric that assesses the robot's dependability, efficacy, and efficiency in carrying out its missions or duties. The performance score is calculated as:

$$PS = 100(w_1 L + w_2 E + w_3 S), \quad w_1 + w_2 + w_3 = 1 \tag{38}$$

Where L, E and S denote the normalized path length, energy consumption, and success rate respectively.

Fig. 8 compares the evaluated strategies based on the normalized and scaled performance score (scaled to 0–100). GA achieved a score of 95.24, PSO obtained 87.01, and ACO reached 94.36, indicating comparable performance among these methods. Under the same evaluation criteria, the proposed method achieved the highest performance score of 99, reflecting improved overall performance relative to the compared approaches.

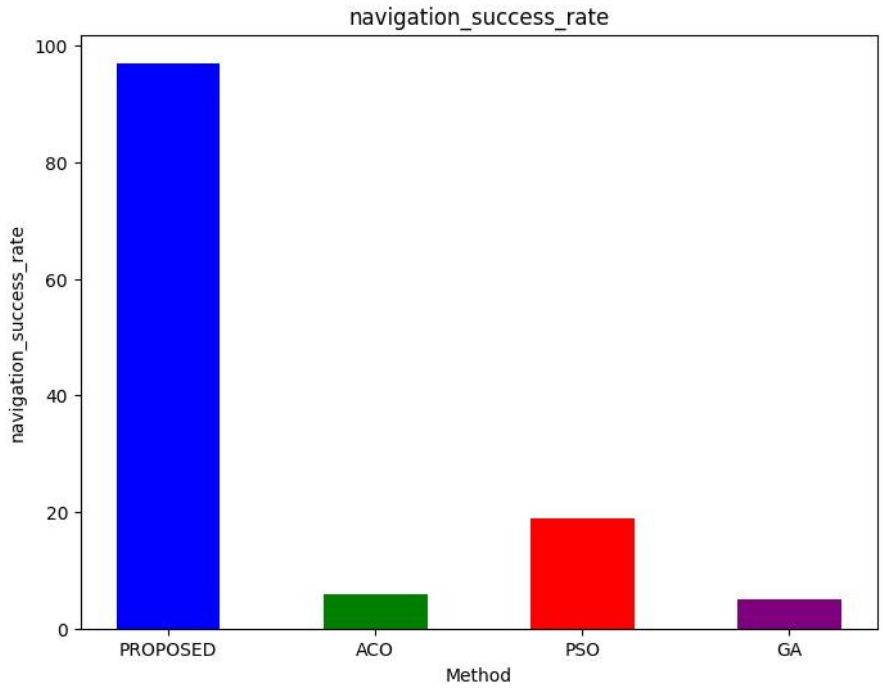

Fig. 7. Comparison of navigation success rates (%) among GA, PSO, ACO, and the proposed method under identical simulation conditions

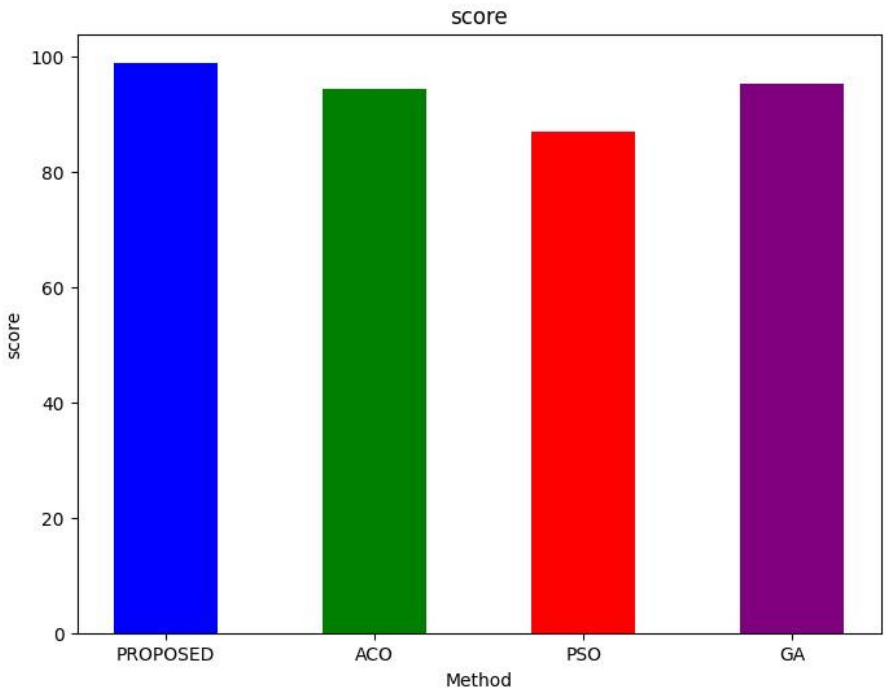

Fig. 8. Comparison of normalized and scaled performance scores (0–100) across the evaluated methods under the same experimental setup

6.2. Discussion

A triple line graph in Fig. 9 illustrates the effectiveness of an unidentified agent across a hundred incidents. In the first graph, "Scores over Episodes," a line begins at 0 and rises gradually to approximately 80 by the 100th episode. This implies that as the agent accumulates experience across the episodes, its score rises.

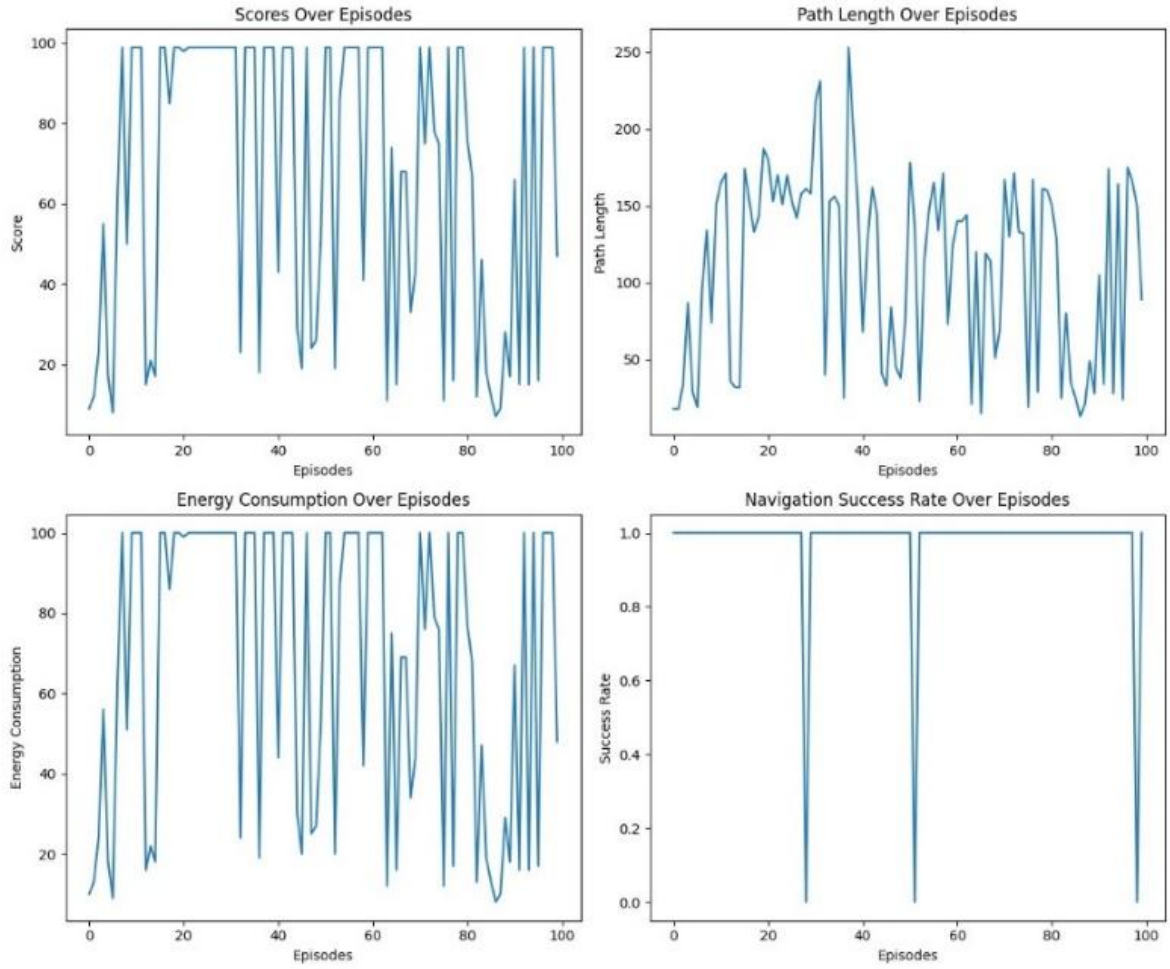

Fig. 9. Overall performance of the developed model with episodes

The line in the second graph, "Path Length over Episodes," starts at 200 and gradually drops to roughly 50 by the 100th episode. This implies that as the agent accumulates expertise across the episodes, it is choosing quicker routes to do its assignment. The final graph, "Navigation Success Rate over Episodes," displays a line that begins at 0 and rises gradually to approximately 1.0 by the 100th episode. This implies that as the agent accumulates experience across the episodes, it is navigating its surroundings more and more successfully. Overall, the results indicate that the agent is getting better throughout the 100 episodes in all three parameters.

Fig. 10 displays performance parameters, such as path length, energy consumption, navigation success rate, and performance score, for the proposed technique across several criteria. The Proposed technique exhibits effective navigation strategy with a path length of 13. Additionally, it exhibits low energy utilization with an energy consumption score of 8, suggesting an energy-efficient navigation strategy. Moreover, the Proposed technique's improved 97% navigation success rate highlights its efficacy in achieving planned destinations. With an outstanding performance score of 99, the proposed technique demonstrates its great performance in all assessed criteria. Overall; the outcomes demonstrate how well the suggested technique performs when it comes to navigation task optimization. It can be used to create short, energy-efficient paths with a high success rate, which makes it a viable method for navigation applications.

From a practical perspective, a navigation success rate of 97% implies that the robot can reliably complete assigned missions in cluttered environments with minimal collision risk. In real-world applications such as warehouse logistics or indoor service robotics, this translates to fewer task interruptions, reduced operational downtime, and improved energy efficiency.

Despite the promising results, the current study is limited to simulation-based evaluation and predefined environmental settings. Real-world deployment may introduce sensor noise, communication delays, and unforeseen dynamic behaviours that were not fully modeled. Future work will focus on real-world validation, scalability to larger environments, and computational efficiency optimization for embedded robotic platforms.

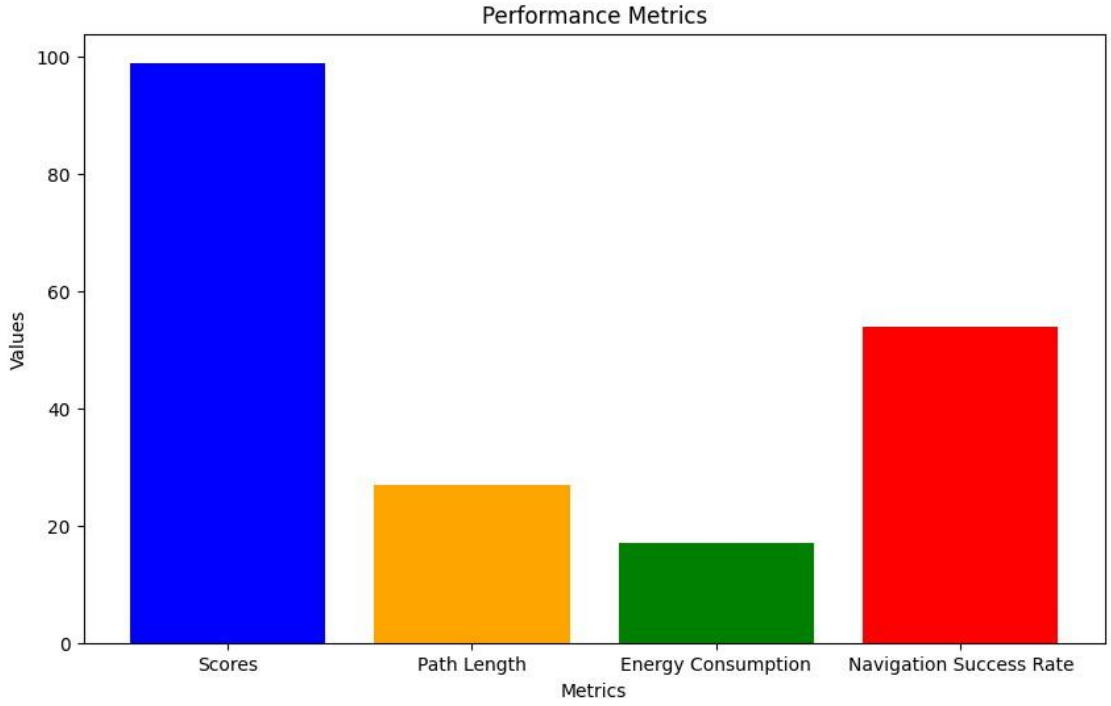

Fig. 10. Overall performance of the developed model

7. Conclusion

This study examines mobile robot tracking, global path planning, and obstacle avoidance in large-scale dynamic situations. A hybrid planning method is presented to address the real-time navigation requirement, using motion planning for obstacle avoidance and path tracking and the HSSB for global path planning. Real-time motion planning can satisfy mobile robots' utility needs in expansive, dynamic situations. Reinforcement Learning is used to create a mobile robot navigation system that is both reliable and effective, with increased flexibility to unforeseen obstacles and dynamic situations. The proposed model's results are verified using current techniques, yielding a reduced path length of 13, reduced energy consumption of 8, a high-performance score of 99, and an improved navigation success rate of 97. Overall, the outcomes demonstrate how well the suggested technique performs when it comes to navigation task optimization. It can provide short, energy-efficient paths with a high success rate, which demonstrates its suitability for mobile robot navigation applications. These results indicate the potential applicability of the proposed framework in real-world robotic navigation tasks such as warehouse automation, indoor service robots, and autonomous inspection systems. Despite the promising simulation results, the current study is limited to controlled simulation environments. Future work will focus on real-world implementation, scalability analysis, and computational optimization for embedded robotic platforms.

Author Contribution: All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] P. Gia Luan and N. T. Thinh, "Real-time hybrid navigation system-based path planning and obstacle avoidance for mobile robots," *Applied Sciences*, vol. 10, no. 10, p. 3355, 2020, https://doi.org/10.3390/app10103355.

[2] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 674-691, 2021, https://doi.org/10.26599/TST.2021.9010012.

[3] Y. F. Chen, M. Liu, M. Everett and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 285-292, 2017, https://doi.org/10.1109/ICRA.2017.7989037.

[4] A. D. J. Plasencia-Salgueiro, "Deep reinforcement learning for autonomous mobile robot navigation," in *Artificial Intelligence for Robotics and Autonomous Systems Applications*, pp. 195–237, 2023, https://doi.org/10.1007/978-3-031-28715-2_7.

[5] M. R. Islam, P. Protik, S. Das, and P. K. Boni, "Mobile robot path planning with obstacle avoidance using chemical reaction optimization," *Soft Computing*, vol. 25, no. 8, pp. 6283–6310, 2021, https://doi.org/10.1007/s00500-021-05615-6.

[6] X. Deng, R. Li, L. Zhao, K. Wang, and X. Gui, "Multi-obstacle path planning and optimization for mobile robot," *Expert Systems with Applications*, vol. 183, p. 115445, 2021, https://doi.org/10.1016/j.eswa.2021.115445.

[7] S. Alshammrei, S. Boubaker, and L. Kolsi, "Improved Dijkstra algorithm for mobile robot path planning and obstacle avoidance," *Computers, Materials & Continua*, vol. 72, no. 3, pp. 5939–5954, 2022, https://doi.org/10.32604/cmc.2022.028165.

[8] F. H. Ajeil, I. K. Ibraheem, A. T. Azar, and A. J. Humaidi, "Autonomous navigation and obstacle avoidance of an omnidirectional mobile robot using swarm optimization and sensors deployment," *International Journal of Advanced Robotic Systems*, vol. 17, no. 3, 2020, https://doi.org/10.1177/1729881420929498.

[9] H. Wang, X. Ma, and L. Zhu, "Obstacle avoidance path planning of mobile robot based on improved DWA," *Journal of Physics: Conference Series*, vol. 2383, no. 1, p. 012098, 2022, https://doi.org/10.1088/1742-6596/2383/1/012098.

[10] H. S. Hewawasam, M. Y. Ibrahim and G. K. Appuhamillage, "Past, Present and Future of Path-Planning Algorithms for Mobile Robot Navigation in Dynamic Environments," *IEEE Open Journal of the Industrial Electronics Society*, vol. 3, pp. 353-365, 2022, https://doi.org/10.1109/OJIES.2022.3179617.

[11] T. Feng *et al*., "The Optimal Global Path Planning of Mobile Robot Based on Improved Hybrid Adaptive Genetic Algorithm in Different Tasks and Complex Road Environments," *IEEE Access*, vol. 12, pp. 18400-18415, 2024, https://doi.org/10.1109/ACCESS.2024.3357990.

[12] Z. Garip, D. Karayel, and M. E. Çimen, "A study on path planning optimization of mobile robots based on hybrid algorithm," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 5, p. e6721, 2022, https://doi.org/10.1002/cpe.6721.

[13] Q. Wang, J. Li, L. Yang, Z. Yang, P. Li, and G. Xia, "Distributed multi-mobile robot path planning and obstacle avoidance based on ACO–DWA in unknown complex terrain," *Electronics*, vol. 11, no. 14, p. 2144, 2022, https://doi.org/10.3390/electronics11142144.

[14] K. Almazrouei, I. Kamel, and T. Rabie, "Dynamic obstacle avoidance and path planning through reinforcement learning," *Applied Sciences*, vol. 13, no. 14, p. 8174, 2023, https://doi.org/10.3390/app13148174.

[15] S. Das and S. K. Mishra, "A machine learning approach for collision avoidance and path planning of mobile robot under dense and cluttered environments," *Computers and Electrical Engineering*, vol. 103, p. 108376, 2022, https://doi.org/10.1016/j.compeleceng.2022.108376.

[16] P. Long, T. Fan, X. Liao, W. Liu, H. Zhang and J. Pan, "Towards Optimally Decentralized Multi-Robot Collision Avoidance via Deep Reinforcement Learning," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6252-6259, 2018, https://doi.org/10.1109/ICRA.2018.8461113.

[17] T. Fan, P. Long, W. Liu, and J. Pan, "Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios," *The International Journal of Robotics Research*, vol. 39, no. 7, pp. 856-892, 2020, https://doi.org/10.1177/0278364920916531.

[18] L. Zhang, Y. Zhang, and Y. Li, "Path planning for indoor mobile robot based on deep learning," *Optik*, vol. 219, p. 165096, 2020, https://doi.org/10.1016/j.ijleo.2020.165096.

[19] C. Ntakolia, S. Moustakidis, and A. Siouras, "Autonomous path planning with obstacle avoidance for smart assistive systems," *Expert Systems with Applications*, vol. 213, p. 119049, 2023, https://doi.org/10.1016/j.eswa.2022.119049.

[20] M. N. A. Wahab, C. M. Lee, M. F. Akbar and F. H. Hassan, "Path Planning for Mobile Robot Navigation in Unknown Indoor Environments Using Hybrid PSOFS Algorithm," *IEEE Access*, vol. 8, pp. 161805-161815, 2020, https://doi.org/10.1109/ACCESS.2020.3021605.

[21] W. Rahmaniar and A. E. Rakhmania, "Mobile robot path planning in a trajectory with multiple obstacles using genetic algorithms," *Journal of Robotics and Control (JRC)*, vol. 3, no. 1, pp. 1–7, 2022, https://doi.org/10.18196/jrc.v3i1.11024.

[22] T. V. Dang and N. T. Bui, "Obstacle avoidance strategy for mobile robot based on monocular camera," *Electronics*, vol. 12, no. 14, p. 1932, 2023, https://doi.org/10.3390/electronics12081932.

[23] K. P. Jayalakshmi, V. G. Nair and D. Sathish, "A Comprehensive Survey on Coverage Path Planning for Mobile Robots in Dynamic Environments," *IEEE Access*, vol. 13, pp. 60158-60185, 2025, https://doi.org/10.1109/ACCESS.2025.3556446.

[24] D. Agarwal and P. S. Bharti, "Implementing modified swarm intelligence algorithm based on Slime moulds for path planning and obstacle avoidance problem in mobile robots," *Applied Soft Computing*, vol. 107, p. 107372, 2021, https://doi.org/10.1016/j.asoc.2021.107372.

[25] Y. Zhu, W. Z. W. Hasan, H. R. H. Ramli, N. M. H. Norsahperi, M. S. M. Kassim, and Y. Yao, "Deep reinforcement learning of mobile robot navigation in dynamic environment: A review," *Sensors*, vol. 25, no. 11, p. 3394, 2025, https://doi.org/10.3390/s25113394.

[26] B. R. Kiran *et al*., "Deep Reinforcement Learning for Autonomous Driving: A Survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909-4926, 2022, https://doi.org/10.1109/TITS.2021.3054625.

[27] X. Wang *et al*., "Deep Reinforcement Learning: A Survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 4, pp. 5064-5078, 2024, https://doi.org/10.1109/TNNLS.2022.3207346.

[28] C. Chen, Y. Liu, S. Kreiss and A. Alahi, "Crowd-Robot Interaction: Crowd-Aware Robot Navigation With Attention-Based Deep Reinforcement Learning," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6015-6022, 2019, https://doi.org/10.1109/ICRA.2019.8794134.

[29] L. Tai, G. Paolo and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 31-36, 2017, https://doi.org/10.1109/IROS.2017.8202134.

[30] M. Hussain, M. O'Nils, J. Lundgren and S. J. Mousavirad, "A Comprehensive Review on Deep Learning-Based Data Fusion," *IEEE Access*, vol. 12, pp. 180093-180124, 2024, https://doi.org/10.1109/ACCESS.2024.3508271.

[31] H. -Y. Lin and X. -Z. Peng, "Autonomous Quadrotor Navigation With Vision Based Obstacle Avoidance and Path Planning," *IEEE Access*, vol. 9, pp. 102450-102459, 2021, https://doi.org/10.1109/ACCESS.2021.3097945.

[32] J. Gao, W. Ye, J. Guo, and Z. Li, "Deep reinforcement learning for indoor mobile robot path planning," *Sensors*, vol. 20, no. 19, p. 5493, 2020, https://doi.org/10.3390/s20195493.

[33] J. Li, J. Sun, L. Liu, and J. Xu, "Model predictive control for the tracking of autonomous mobile robot combined with a local path planning," *Measurement and Control*, vol. 54, no. 9-10, pp. 1319-1325, 2021, https://doi.org/10.1177/00202940211043070.

[34] R. Zhao, Y. Li, Y. Fan, F. Gao, M. Tsukada and Z. Gao, "A Survey on Recent Advancements in Autonomous Driving Using Deep Reinforcement Learning: Applications, Challenges, and Solutions,"

IEEE Transactions on Intelligent Transportation Systems, vol. 25, no. 12, pp. 19365-19398, 2024, https://doi.org/10.1109/TITS.2024.3452480.

[35] V. Ušinskis, M. Nowicki, A. Dzedzickis, and V. Bučinskas, "Sensor-fusion based navigation for autonomous mobile robot," *Sensors*, vol. 25, no. 4, p. 1248, 2025, https://doi.org/10.3390/s25041248.

[36] B. Tao and J. H. Kim, "Deep reinforcement learning-based local path planning in dynamic environments for mobile robot," *Journal of King Saud University - Computer and Information Sciences*, vol. 36, no. 10, p. 102254, 2024, https://doi.org/10.1016/j.jksuci.2024.102254.

[37] D. Wang, N. Gao, D. Liu, J. Li and F. L. Lewis, "Recent Progress in Reinforcement Learning and Adaptive Dynamic Programming for Advanced Control Applications," *IEEE/CAA Journal of Automatica Sinica*, vol. 11, no. 1, pp. 18-36, 2024, https://doi.org/10.1109/JAS.2023.123843.

[38] H. Kumar, A. Koppel, and A. Ribeiro, "On the sample complexity of actor-critic method for reinforcement learning with function approximation," *Machine Learning*, vol. 112, no. 7, pp. 2433-2467, 2023, https://doi.org/10.1007/s10994-023-06303-2.

[39] L. Chen *et al.*, "Decision transformer: Reinforcement learning via sequence modeling," *Advances in Neural Information Processing Systems*, vol. 34, pp. 15084–15097, 2021, https://proceedings.neurips.cc/paper/2021/hash/7f489f642a0ddb10272b5c31057f0663-Abstract.html.

[40] Z. Zhu and H. Zhao, "A Survey of Deep RL and IL for Autonomous Driving Policy Learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14043-14065, 2022, https://doi.org/10.1109/TITS.2021.3134702.

[41] L. D. Hanh, V. D. Cong, "Path following and avoiding obstacle for mobile robot under dynamic environments using reinforcement learning," *Journal of Robotics and Control (JRC)*, vol. 4, no. 2, pp. 157-164, 2023, https://doi.org/10.18196/jrc.v4i2.17368.

[42] H. Yang and X. Teng, "Mobile robot path planning based on enhanced dynamic window approach and improved A∗ algorithm," *Journal of Robotics*, vol. 2022, no. 1, p. 2183229, 2022, https://doi.org/10.1155/2022/2183229.

[43] F. M. Talaat *et al.*, "Route planning for autonomous mobile robots using a reinforcement learning algorithm," *Actuators*, vol. 12, no. 1, p. 12, 2022, https://doi.org/10.3390/act12010012.

[44] R. Raj and A. Kos, "An optimized energy and time constraints-based path planning for the navigation of mobile robots using an intelligent particle swarm optimization technique," *Applied Sciences*, vol. 13, no. 17, p. 9667, 2023, https://doi.org/10.3390/app13179667.

[45] T. Zhang, J. Xu and B. Wu, "Hybrid Path Planning Model for Multiple Robots Considering Obstacle Avoidance," *IEEE Access*, vol. 10, pp. 71914-71935, 2022, https://doi.org/10.1109/ACCESS.2022.3188784.

[46] M. F. R. Lee and S. H. Yusuf, "Mobile robot navigation using deep reinforcement learning," *Processes*, vol. 10, no. 12, p. 2748, 2022, https://doi.org/10.3390/pr10122748.

[47] S. Feng, B. Sebastian, and P. Ben-Tzvi, "A collision avoidance method based on deep reinforcement learning," *Robotics*, vol. 10, no. 2, p. 73, 2021, https://doi.org/10.3390/robotics10020073.

[48] X. Zhong, J. Tian, H. Hu, and X. Peng, "Hybrid path planning based on safe A* algorithm and adaptive window approach for mobile robot in large-scale dynamic environment," *Journal of Intelligent & Robotic Systems*, vol. 99, no. 1, pp. 65–77, 2020, https://doi.org/10.1007/s10846-019-01112-z.

[49] M. N. Zafar, J. C. Mohanta, and A. Keshari, "GWO-potential field method for mobile robot path planning and navigation control," *Arabian Journal for Science and Engineering*, vol. 46, no. 8, pp. 8087–8104, 2021, https://doi.org/10.1007/s13369-021-05487-w.

[50] M. R. Garey, D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," *W. H. Freeman: San Francisco*, 1979, https://books.google.co.id/books/about/Computers_and_Intractability.html?id=fjxGAQAAIAAJ&redir_esc=y.

[51] S. M. Tabatabaei, M. Asadian-Pakfar, and B. Sedaee, "Well placement optimization with a novel swarm intelligence optimization algorithm: Sparrow Search Algorithm," *Geoenergy Science and Engineering*, vol. 231, p. 212291, 2023, https://doi.org/10.1016/j.geoen.2023.212291.

[52] X. S. Yang and A. Hossein Gandomi, "Bat algorithm: a novel approach for global engineering optimization," *Engineering Computations*, vol. 29, no. 5, pp. 464-483, 2012, https://doi.org/10.1108/02644401211235834.

[53] T. F. Abaas and A. H. Shabeeb, "Autonomous mobile robot navigation based on PSO algorithm with inertia weight variants for optimal path planning," *IOP Conference Series: Materials Science and Engineering*, vol. 928, no. 2, p. 022005, 2020, https://doi.org/10.1088/1757-899X/928/2/022005.

[54] M. R. Azizi, A. Rastegarpanah, and R. Stolkin, "Motion planning and control of an omnidirectional mobile robot in dynamic environments," *Robotics*, vol. 10, no. 1, p. 48, 2021, https://doi.org/10.3390/robotics10010048.

[55] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm," *Systems Science & Control Engineering*, vol. 8, no. 1, pp. 22–34, 2020, https://doi.org/10.1080/21642583.2019.1708830.

[56] S. U. Umar, T. A. Rashid, A. M. Ahmed, B. A. Hassan, and M. R. Baker, "Modified Bat Algorithm: a newly proposed approach for solving complex and real-world problems," *Soft Computing*, vol. 28, no. 13, pp. 7983–7998, 2024, https://doi.org/10.1007/s00500-024-09761-5.

[57] X. Zhu *et al.*, "Path planning of multi-UAVs based on deep Q-network for energy-efficient data collection in UAVs-assisted IoT," *Vehicular Communications*, vol. 36, p. 100491, 2022, https://doi.org/10.1016/j.vehcom.2022.100491.

[58] W. Ding *et al.*, "Learning to View: Decision Transformers for Active Object Detection," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7140-7146, 2023, https://doi.org/10.1109/ICRA48891.2023.10160946.

[59] Y. V. Pehlivanoglu and P. Pehlivanoglu, "An enhanced genetic algorithm for path planning of autonomous UAV in target coverage problems," *Applied Soft Computing*, vol. 112, p. 107796, 2021, https://doi.org/10.1016/j.asoc.2021.107796.

[60] M. D. Phung and Q. P. Ha, "Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization," *Applied Soft Computing*, vol. 107, p. 107376, 2021, https://doi.org/10.1016/j.asoc.2021.107376.

[61] C. Miao, G. Chen, C. Yan, and Y. Wu, "Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm," *Computers & Industrial Engineering*, vol. 156, p. 107230, 2021, https://doi.org/10.1016/j.cie.2021.107230.