Task Scheduling and Path Planning of Hotel Service Robots Driven by Artificial Intelligence

Jingyi Han
Corresponding author's E-mail: han-jingyi@hotmail.com
Faculty of Culture and Tourism, Jiyuan Vocational and Technical College
Jiyuan, Henan, 459000, China

Keywords: hotel service robots, task scheduling, path planning, artificial intelligence, SchedNav-RX, Deep Q-Network (DQN), Reinforcement Learning, A* Algorithm

Received: May 29, 2025

Many hotels embrace intelligent service robots as a novel approach to enhancing customer satisfaction and streamlining operations. However, the ever-changing hotel environment makes scheduling and managing obligations difficult. This research introduces Scheduling + Navigation Robotic Executor (SchedNav-RX) for real-time route planning and job prioritization. The proposed SchedNav-RX utilizes a deep Q-network-based adaptive task scheduling reinforcement learning model, enabling robots to reschedule and prioritize tasks based on urgency and context dynamically. A convolutional neural network (CNN) enhances the standard A algorithm for navigation by predicting obstacles in real-time. This makes dynamic interior navigation safer and more efficient. TurtleBot 3 units were used for physical validation to enhance performance evaluation. SchedNav-RX outperforms standard planning systems by 27% in task completion time and 35% in navigation safety while dealing with unexpected vehicle traffic. These findings demonstrate that SchedNav-RX is essential for intelligent, autonomous robots to perform hotel service tasks efficiently and easily. The concept allows complicated hospitality environments to accommodate dispersed robotic systems driven by artificial intelligence. This work will evolve to incorporate reinforcement learning-based guest feedback and interaction modules, enhancing the system's capabilities.*

Povzetek: Za optimizirano razporejanje nalog in načrtovanje poti hotelskih robotov je razvit SchedNav-RX, ki združuje DQN in CNN-izboljšani A za zaznavanje ovir v realnem času. Na TurtleBot-3 doseže 94 % uspešnost opravil, +35 % varnejšo navigacijo.*

1 Introduction

The rapid development of artificial intelligence and robotics has revolutionized several sectors. The hospitality industry is one such business that stands to benefit significantly from intelligent service automation applications. The hotel sector has experienced significant growth in the use of service robots for several reasons, which will be discussed further below. Several issues are at play here, including a scarcity of accessible labor, rising operational expenses, and the need for reliable and seamless customer experiences. Many new uses for robots have emerged in today's society. Some examples of these applications include the delivery of food and utilities, the escorting of guests, and concierge services. There are two reasons why these robots are useful: first, they offer clients something fresh and fascinating, and second, they reduce the amount of labor that service professionals have to do. A comprehensive set of capabilities for route planning and intelligent task scheduling must be included in these systems before deployment. It is almost impossible for service robots to collaborate effectively in today's hotel environments due to the high degree of complexity and the rapid rate of change in these environments.

It is common practice in traditional robotics systems to regard the navigation and scheduling modules as independent. Although separation makes the design more efficient, it also decreases performance because its prioritization system does not consider the location of activities, such as a robot would either put off or overlook vital work. A path planning module cannot recommend the most efficient routes if there are guests, service carts, or unexpected room closures. This is true even in a static and perfect environment. As a result of the lack of a consistent planning strategy, Jeon et al. (2022) [1] state that service robots struggle with tasks that require them to make rapid evaluations while also being aware of their context.

A hybrid offline-online planning system was introduced by Wang and Tian (2022) [2]. This system uses probabilistic inference and semantic mapping to further the need for integrated task and motion planning. The system design process considered the integrated needs for task and motion planning. Specifically, their

research findings highlight the importance of service robots adapting to the contexts in which they operate in the real world. Their approach is ineffective in terms of real-time rearrangement of activities and the situationally adaptable learning they aim to achieve. In the extensive investigation of robotic task and motion planning conducted by Guo et al. (2023) [3], they observed that existing systems often fail to consider the various mutually advantageous outcomes if navigation and scheduling are addressed concurrently. Whenever there is a lack of constant human supervision or the capacity for robots to adapt to new circumstances, the aim of automation becomes meaningless.

The robot's ability to accurately sense and respond to environmental changes is another factor that must be considered. As a result, hotels are often crowded to capacity with guests and employees, making the standard rooms, elevators, and corridors constantly busy with activity. An interactive path editing system was developed by Yoo and Choi (2024) [4] for collaborative robotics. This system was designed to handle the requirement for route alteration and real-time simulation when the robots are presented with unexpected conditions. Their study suggests that static route planning algorithms are not a suitable match for the current environment being considered. This is a serious concern, as robots that provide services must be able to make real-time course adjustments to ensure the reliability and safety of such services.

Systems that recognize and react to human gestures and other inputs are becoming increasingly critical as the complexity of real-time perception and control continues to increase. In their demonstration of real-time gesture recognition algorithms for gaming systems and robotics, Hafiz and Wong (2024) [5] emphasized the importance of low-latency perception in user-interactive settings. Although their study does not explicitly focus on service robots, the fundamental concepts of being responsive in real-time and adaptable are crucial for human-robot interaction in the hospitality industry. There is a possibility that the user experience will be considerably diminished if there is any noticeable latency or a lack of context awareness. This is because visitors anticipate intelligent and responsive behavior from robotic gadgets.

Although the "e-butler" robot produced by Gunawan et al. (2023) [6] is a notable example of a prototype system that demonstrates how hotels utilize robotic services, it is often not scalable due to its limited artificial intelligence capabilities. These systems can perform the tasks assigned to them; however, they struggle to adapt to new operational needs or optimize their performance in diverse environments. The possibility of automating specific entry-level jobs in the food service industry was investigated by Tuomi and Ascenção in 2023 [7]. According to their analysis, most of these duties are performed by robots. This approach will fail until robots mimic human vision, planning, and execution abilities.

As a relatively new field, deep reinforcement learning has the potential to reduce skill disparities. The use of DRL was demonstrated by Li et al. (2024) [8] in their study, which showed how multi-robot systems enhance learning speed and flexibility by dividing tasks and building routes. Another study that may help AGVs enhance their real-time route planning in industrial settings is Bao et al. (2024) [9], which utilized digital twins within a reinforcement learning system. Due to differences in social, geographical, and temporal attributes among the top candidates for hospitality positions, these methodologies demonstrate how DRL decreased uncertainty in the real world.

This paper introduces SchedNav-RX, our latest AI-driven tool. Regular room service robots resolve these issues by integrating their work schedules and route planning. SchedNav-RX is capable of autonomous movement and operation in a dynamic environment. A Deep Q-Network (DQN) that can learn to prioritize jobs, an upgraded A* algorithm for optimal route determination, and real-time environment awareness are all features of this system. Service robots are multipurpose due to their intelligent design. In addition to several other benefits, they can handle challenging environments, select and complete tasks, and perform other tasks more efficiently.

SchedNav-RX potentially eliminates fragmentation in conventional systems by factoring in service priority, physical proximity, anticipated completion time, and route congestion. A robot bringing towels is redirected if the system detects another visitor on the same floor who needs room service. Both client satisfaction and operational efficiency are enhanced by its real-time reconfigurability.

2 Literature survey

As the need for self-sufficient and flexible robotic systems continues to increase, one of the most significant areas of research is the incorporation of work scheduling and route planning into service robots, particularly in hotel environments. Recent research has focused on algorithmic techniques and system designs to address these challenges. The study has been conducted to find solutions.

Zhang et al. (2023) [10] proposed a geometric task-and-motion planning (TAMP) framework that integrates geometric mobility constraints with symbolic task planning, providing support for cooperative execution. Despite its efficacy in controlled environments with strong inter-robot interaction, this method is unsuitable for dynamic, single-robot scenarios in unstructured domains. The hotel sector is well-known for its dynamic, service-based operations; yet this industry emphasizes the manipulation of duties.

An improved version of the Non-dominated Sorting Genetic Algorithm II (NSGA-II) was employed by Duan et al. (2024) [11] to address the problem of designing routes that consider multiple objectives. This strategy

achieves three important route planning objectives by maximizing energy economy, safety, and distance. Their technique is not reactive in real-time and is most effective when used in static or nearly static conditions. This system lacks task scheduling logic, another essential component for achieving total robot autonomy in hotel service situations.

Using an innovative path-planning algorithm created by Li et al. (2023) [12], the inspection robots can be used effectively. An Improved Particle Swarm Optimisation (IPSO) is used in this approach. Their technology enhances several qualities, including route smoothness and obstacle avoidance, when used in simulated outdoor situations. The calculation durations for PSO variations are generally greater than those for other variants because of how repetitive they are. Because of this, they are not suitable for situations with significant degrees of interior dynamic complexity, such as hotels, even though they have substantial capabilities for global optimization. When making decisions at the task level, PSO-based approaches often fail to reach their full potential.

Additionally, researchers in the field of robotics have experimented with graph-based algorithms. A bio-inspired, graph-based optimal route planning system (B-IG-ORPS) was reported by Lei et al. (2023) [13]. This system draws inspiration from bio-inspired designs and employs principles related to swarm intelligence. The capability of their method to identify the most efficient routes in grid-based maps is evidence of the resilience of their approach. The environments of hotels are constantly evolving, with new obstacles and human interactions appearing regularly. Graph-based systems that are static have a difficult time maintaining their accuracy and reliability.

The primary purpose of the study conducted by Gu et al. (2022) [14] was to investigate how hotel service robots handle scheduling tasks. Although the research suggests that hotels should plan tasks independently, it does not account for geographical limitations or the real-time mobility of robots when calculating availability. Additionally, the heuristic method cannot handle unforeseen difficulties, increased labor, or last-minute changes.

A related study on customer experience was conducted by Lei et al. (2023) [15], which examined the factors influencing future purchases of AI-powered hospitality services. According to the findings of their investigation, the three most important aspects that affect the degree to which customers are satisfied are responsiveness, reliability in job performance, and perceived intelligence. This research does not propose a method that demonstrates the need for real-time adaptive intelligence in robotic systems to function correctly.

The RL-QPSO Nett method is an improved mobile robot route planning technique that Jing and Weiya (2025) [16]. This technique optimizes the path planning of mobile robots using quantum-behaved particle swarms and deep reinforcement learning. With the help of this hybrid methodology, achieving higher convergence and flexibility is feasible compared to the traditional PSO method. Nevertheless, it has not yet developed scheduling or service-context awareness; its primary focus is navigation. The quantum-inspired nature of the optimization adds processing complexity that is not always required when applied to real-time service settings. This is because the optimization is based on quantum mechanics. The SchedNav-RX system uses a more modular and practical methodology. DQN is used for task scheduling, whereas an updated CNN-enhanced A* is utilized for navigation. Real-time performance is guaranteed, and the deployment procedure is easier to understand and carry out.

A multi-task deep reinforcement learning framework for scheduling optimization and intelligent logistics path planning was presented by Zhu (2025) [17]. Delivery scheduling and path planning are integrated in this method, which allows for flexible decision-making across a variety of logistics activities. In intricate transportation networks, it shows decreased computation time and increased delivery efficiency. However, in extremely variable logistics situations, including those with erratic traffic or real-time consumer needs, the approach lacks stability and is mostly dependent on pre-defined network parameters.

Bendiaf et al. (2024) [18] used the knapsack algorithm to create a novel job scheduling strategy designed for heterogeneous computing systems. This technique minimizes calculation time and maximizes system throughput, improving task-to-resource mapping. Although it works well in static computer environments with predictable task loads, its application in dynamic, real-time systems—like cloud-based logistics platforms—where job arrivals and resource availability are always changing is limited by its deterministic scheduling assumptions.

3 Proposed methodology

The SchedNav-RX framework, which utilizes artificial intelligence to aid in route planning and operational scheduling, empowers autonomous hotel service robots to enhance their navigation capabilities in complex interior settings. This strategy uses a convolutional neural network (CNN) for environmental perception and obstacle detection, an A* algorithm for efficient task discovery, and a real-time feedback-based reinforcement learning agent for decision-making. SchedNav-RX utilizes convolutional neural networks (CNNs) to gather information about its surroundings and execute obstacle-avoidance actions. The RL module then sorts the data to determine the best work allocation and navigational enhancements. With A*'s assistance, it is possible to design a route that is both the most efficient and secure to any location. SchedNav-RX is an intelligent robotic support system designed to assist the hotel industry. Its synergistic workflow, computational speed, and collision avoidance capabilities are all advantageous.

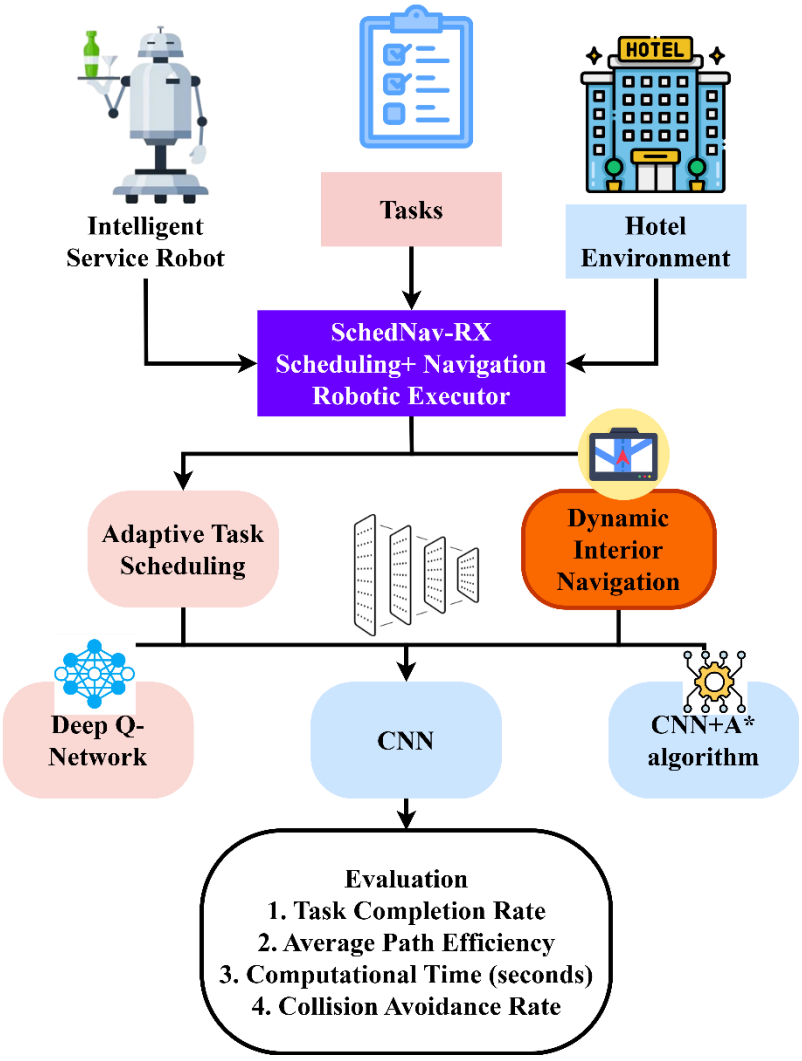

Figure 1: Proposed diagram

SchedNav-RX, an acronym for Scheduling and Navigation Robotic Executor, is illustrated in Figure 1 as having the potential to enhance the capabilities of intelligent service robots in the hospitality industry. The SchedNav-RX module, a crucial component of the system, analyzes information obtained from the Intelligent Service Robot, performs a series of tasks, and determines the precise positions of the rooms. This module is responsible for managing the robot's activities by coordinating task scheduling and navigating a dynamic environment.

Through the use of SchedNav-RX, it is possible to access two key functional routes. To get things started, adaptive task scheduling enables the robot to perform multiple tasks simultaneously and intelligently prioritize operations. This schedule utilizes reinforcement learning to establish the optimal rules, which is facilitated by a Deep Q-network (DQN). Take the possibility that the system adjusts the deadline for a project or the amount of battery life remaining in response to changing priorities and constraints.

In the second route, Dynamic Interior Navigation enables the robot to navigate complex hotel layouts in the real world seamlessly. A Convolutional Neural Network (CNN) is used for visual awareness of the surrounding environment, and an A* search strategy is utilized for route planning. By using this tactic, it is possible to achieve this goal. A sense of self-assurance is instilled in the robot due to its ability to adapt to novel settings and navigate obstacles seamlessly.

A common evaluation framework for task scheduling and navigation is used to examine the interoperability of CNN and A* algorithm-based approaches. The schematic, when seen as a whole, illustrates how SchedNav-RX uses scheduling and navigation strategies based on artificial intelligence. This methodology enables intelligent, autonomous service robots to perform exceptionally well in highly structured environments, such as hotels. Table 1 displays the symbol and its corresponding description.

Table 1: Symbol and description

Symbols	Descriptions
$L_r(t)$	Current 2D or 3D location of the robot
$\zeta_q(t)$	Queue of pending tasks at a time
$O_m(t)$	Obstacle map including static and dynamic obstacles
$B_r(t)$	Battery level of the robot
$H_p(t)$	Human presence indicator
A_t	Actions available to the robot for service task execution
$Q(S_t, A_t; \theta)$	The estimated future reward of taking actionA_t in state A_t
θ	Neural network weights
γ	Discount factor for future rewards
R_{t+k+1}	Reward at time $t + k + 1$
α	Learning rate
Q'	Target Q-network with parametersθ^-
a'	Next best action
w_1, w_2, w_3	Tunable weights
$urgency, proximity, batterycost$	Contextual features of tasks
λ_1, λ_2	Weights for static and dynamic obstacle contributions
$g(n)$	Cost from start to current node n
$h(n)$	Estimated cost from n to goal (Heuristic)
β	Heuristic scaling factor
$Goal^*$	Combines task utility and learned Q value for optimal target selection
$\delta_1, \delta_2, \delta_3, \delta_4$	Reward function coefficients
$\pi^*(S_t)$	Optimal policy for selecting an action
η	Tradeoff between path cost and expected reward

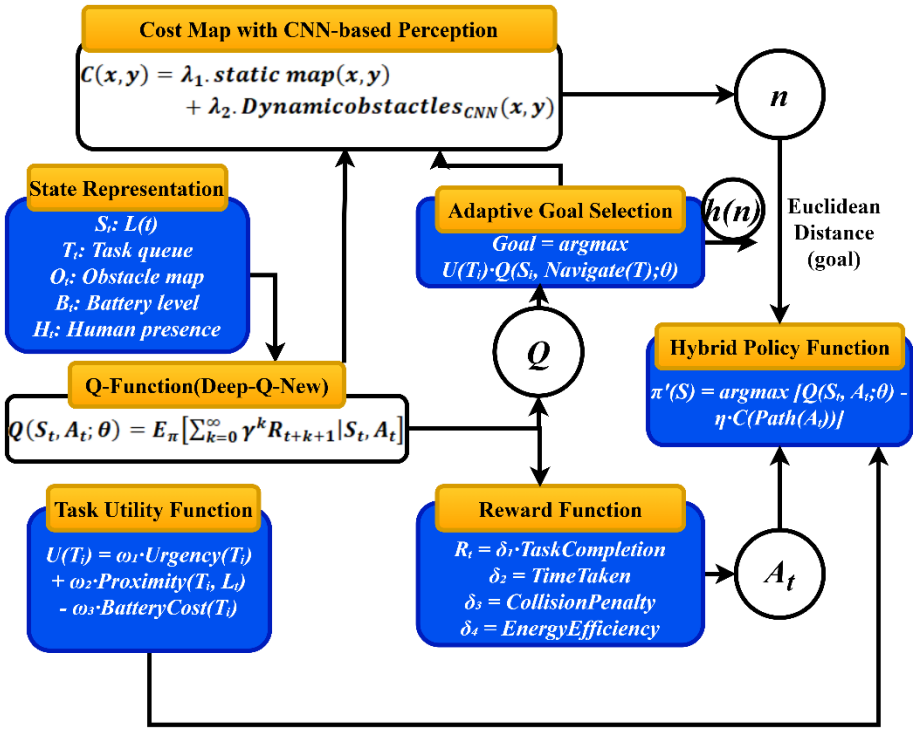

Figure 2: Hybrid policy framework

By integrating goal selection, adaptive perception, and deep reinforcement learning, Figure 2 illustrates a policy framework that facilitates task planning and decision-making in dynamic and changing conditions. Initially, a State Representation comprises several contextual inputs. These inputs include the following $S_t = \{L_r(t), \zeta_q(t), O_m(t), B_r(t), H_p(t)\}$: the current location of the robot $L_r(t)$, the tasks that need to be done $\zeta_q(t)$, a map of obstacles $O_m(t)$), the amount of time that the robot's battery will last $B_r(t)$, and its orientation $H_p(t)$. Running these states through a Q-function, specifically a deep Q-network, enables the calculation of the anticipated benefit of an action. When referring to the state $Q(S_t, A_t; \theta) = E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t, A_t]$. The usefulness of each task is defined by a job usefulness Function, which considers weighted parameters such as the cost of the associated battery, the closeness to the robot's present position, and the urgency of the situation. When it comes to making choices, this is helpful. These data are taken into consideration by the Adaptive Goal Selection module, which then employs a utility-based algorithm to determine the most effective location for the goal in real-time. This function is defined as $(n) = g(n) + h(n), where\ h(n) = \beta.Euclidean\ distance\ (n, goal)$ represents the cost-to-come and $h(n)$represents the estimated geometric distance to the target. The data from static maps and moving barriers is combined to build a CNN-based Cost Map. The parameters $C(x,y) = \lambda_1.static\ map(x,y) + \lambda_2.Dynamicobstactles_{CNN}(x,y)$ are used to identify the relative relevance of each kind of data. This cost map illustrates the advantages and disadvantages of various solutions in a graphical format. A reward function has been implemented to prevent individuals from wasting their time, effort, and resources while motivating them to accomplish their goals. It employs the values δ_1, and δ_2, δ_3 to encourage energy-efficient activities and δ_4 to punish actions that are not energy-efficient. $R_t = \delta_1.Task\ Completion - \delta_2.Collision\ Penalty - \delta_3.Time\ taken + \delta_4 Energy\ Efficiency$. In conclusion, the Hybrid Policy Function considers the cost of the route and the Q-values that have been learned to decide the most effective course of action. The policy $\pi^*(S_t)$ is characterized by the following equation: $\pi^*(S_t) = arg\max_{A_t \in A}[Q(S_t, A_t; \theta_t) - \eta C(path(A_t))]$ This policy is designed to strike a compromise between the principles of reinforcement learning and cost-based planning. This comprehensive framework enables making informed decisions in complex situations with dynamic features and limited resources.

$$S_t = \{L_r(t), \zeta_q(t), O_m(t), B_r(t), H_p(t)\} \quad (1)$$

As calculated in equation (1), state representation has been examined. At each timestep t, the state S_t provides a detailed description of the internal and exterior environmental situations that the robot is experiencing. It is possible to indicate the geographic location of the robot as $L_r(t)$ in either two-dimensional or three-dimensional coordinates, depending on the level of navigational granularity used. Some service requests are currently being handled, including deliveries to guests, calls for cleaning, and interactions with the concierge. The

function represents this queue $\zeta_q(t)$. It is possible to construct a CNN-based vision system or sensors to produce a $O_m(t)$ -A dimensional obstacle map that records both moving and stationary objects that have the potential to block mobility. When it comes to realistic job planning, staying on top of the battery life, which is represented by the symbol $B_r(t)$, is an essential factor. $H_p(t)$ is responsible for determining whether humans are present; this is necessary for the robot's safety and to ensure that it does not cause irritation to people. When everything is considered, this state formulation provides the robot with the necessary information about its surroundings to make informed judgments about its immediate environment.

$$A_t = \{Navigate(x,y), pickup(i), delivery\ (j), Idel()\} \quad (2)$$

The fundamental capabilities $A_t$of the robot system is defined by its action set at any given time t, which is represented by the equation (2). The operations that are related to commands at the highest level are as follows: Using the $Navigate(x,y)$, function, can send the robot to a particular location, retrieve an object or package by using the $pickup(i)$, function, complete a delivery to a particular location by using the $Deliver(j)$ function, send the robot back to its charging dock when its power is low by using the function $Idle$ (). The modules responsible for scheduling and planning use these activities as building blocks to construct more complicated service performances. As a result of these atomic activities, contemporary hotel robots can rapidly respond to a wide variety of service needs, reroute themselves to avoid obstacles, conserve energy when not in use, and execute tasks in a manner that is both ecologically and socially responsible.

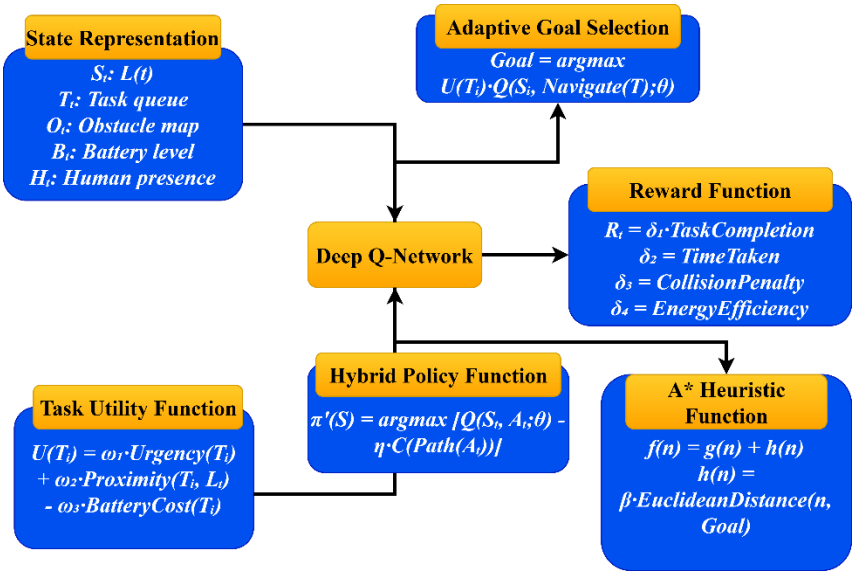

Figure 3: Intelligent robot task execution

Intelligent robotic task execution is shown in Figure 3. This task execution uses a hybrid decision-making architecture that combines heuristic planning with deep reinforcement learning (DRL). In the State Representation module, everything starts from the very beginning. Important information about the system and its surroundings is stored in it. This information includes the position of the robot $(L(t))$, the task queue (T), the obstacle map $(O(t)$), the battery level $(B(t))$, and whether or not humans are present $(H(t))$. The input state aids the learning and planning processes S_t, which is comprised of these variables. Following the consideration of factors such as their proximity to the robot, the anticipated cost of the battery, and the level of urgency, Function assigns a rating to each job i based on its relative significance. The mathematical definition of this utility is as follows in equation (5). To determine the optimal amount of labor to be performed, the Adaptive Objective Selection module considers both the predicted Q-value of accomplishing the purpose, which is presented as: and the usefulness of the activity. To get the highest possible value, the aim is to maximize the absolute value of $Goal^* = arg\max_{T\in\zeta_q}[U(T_i).Q(S_t, navigate(T_i);\theta)]$. The purpose of the $navigate(T_i);\theta$ algorithm is to maximize the value that is obtained by dividing the value of $U(T_i)$by the value of Q(S i) shown in equation (8). It is a Deep Q-Network (DQN) that is responsible for determining the ideal action-value function $Q(S_t, A_t;\theta) = E_\pi[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t, A_t]$. In this context, θ is a representation of the parameters the neural network has learned. This information is then used to enhance the goal-setting process.

The purpose of the system is to affect learning via the use of a comprehensive Reward Function. This function takes into consideration a multitude of factors, including

penalties $R_t = \delta_1.Task\ Completion - \delta_2.Collision\ Penalty - \delta_3.Time\ taken + \delta_4 Energy\ Efficiency$. For route planning, an independent A Heuristic Function* has been added, and the evaluation function that has been included is as follows: If the equation $f(n) = g(n) + h(n), where\ h(n) = \beta.Euclidean\ distance\ (n, goal)$ then the equation is valid. There is a possibility that this will increase the anticipated cost of the shortest route to the destination for a node. Ultimately, the Hybrid Policy Function combines the Q-values learned with the route costs estimated using heuristics. This is the approach that it chooses to take: It is expected that the outcome will be $\pi^*(S_t) = arg\max_{A_t \in A}[Q(S_t, A_t; \theta_t) - \eta C(path(A_t))]$). Additionally, this formulation ensures that the agent prioritizes high-reward jobs while also considering route cost, resulting in navigation that is both more efficient and secure.

The hybrid architecture combines techniques, including heuristic search, cost-aware planning, reinforcement learning, utility-based goal selection, and more. These techniques enable autonomous agents to adapt to complex situations and make informed decisions.

$$Q(S_t, A_t; \theta) = E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t, A_t] \quad (3)$$

As initialized in Equation (3), the Q-function (Deep Q-Network) has been explored. A function of $Q(.)$ is the estimated anticipated cumulative reward, which is denoted as $Q(S_t, A_t; \theta)$ represents the weights of the neural network θ and policy π represents the state S_t in which the action A_t is carried out. This function enables the system to consider the value of the activity and the attractiveness or quality of each state-action pair. The current policy employs a scalar value to represent the situation. It is through reinforcement learning that the Q-function is updated as the robot interacts with its surroundings. This allows the robot to be trained to adapt to new situations and tasks. Deep neural networks enable the system to manage complex, high-dimensional input states, such as visual data, sensor readings, and task queues. The utilization of deep neural networks guarantees this. The discount factor γ, balances short-term and long-term rewards, making it worthwhile where immediate actions impact future consequences, such as when completing a job.

$$\theta_{t+1} = \theta_t + \alpha \left[R_{t+1} + \gamma \max_{a'} Q(S_{t+1}, a'; \theta^-) - Q(S_t, A_t; \theta_t)\right] \nabla_\theta Q(S_t, A_t; \theta_t) \quad (4)$$

As discussed in equation (4), the DQN Update Rule is described. Additionally, the learning rate, denoted as α, regulates the size of the weight update. The target network, denoted as θ^-, is a stabilized variant of θ employed for bootstrapping purposes. $R_{t+1} + \gamma \max_{a'} Q(S_{t+1}, a'; \theta^-)$ The target Q-value and the optimal future reward are both represented by the expression $\gamma \max_{a'} Q(S_{t+1}, a'; \theta^-)$. The calculation of the Temporal Difference (TD) error involves the addition of the current estimates, denoted as $Q(S_t, A_t; \theta_t)$. Over time, the network's ability to accurately predict the value of an action improves as the error is gradually reduced. Due to its iterative learning process, the robot can continuously adapt and perform better in complex hotel service environments.

$$U(T_i) = w_1.urgency(T_i) + w_2.proximity(T_i,\ L_r) - w_3.Batterycost(T_i) \quad (5)$$

As found in equation (5), the Task Utility function has been expressed. The robot makes use of the scalar value that is provided by the task utility function $U(T_i)$ $Batterycost(T_i)$ to get a better idea of which task should be completed next. The term "proximity" indicates how close something is, the term " $urgency(T_i)$" indicates how important something is (for example, a request from a guest at the last minute), and the term "battery cost" discourages routes that consume considerable amounts of energy. Through the utilization of the weights $w_{1,}, w_2, w_3$ the designers can modify the system's actions with functional objectives, such as enhancing responsiveness or energy efficiency. This function ensures that service demands are met and promotes balanced workload distribution and smooth robot operation by verifying that the selected tasks are logistically feasible and efficient.

$$C(x, y) = \lambda_1.static\ map(x, y) + \lambda_2.Dynamicobstactles_{CNN}(x, y) \quad (6)$$

As computed in equation (6), the Cost Map with CNN-based Perception has been found. The dynamic obstacle layer utilizes a convolutional neural network (CNN) to track people and objects in real-time, while the static map component labels walls and furniture. Using adjustable weights λ_1 and λ_2. The robot's routing process prioritizes safety and efficiency. Raising λ_2 during peak guest hours, it promotes cautious movement. This cost model allows strong, real-time decision-making in dynamic hotel

environments with frequent human-robot interaction by continuously adjusting to new perceptual data.

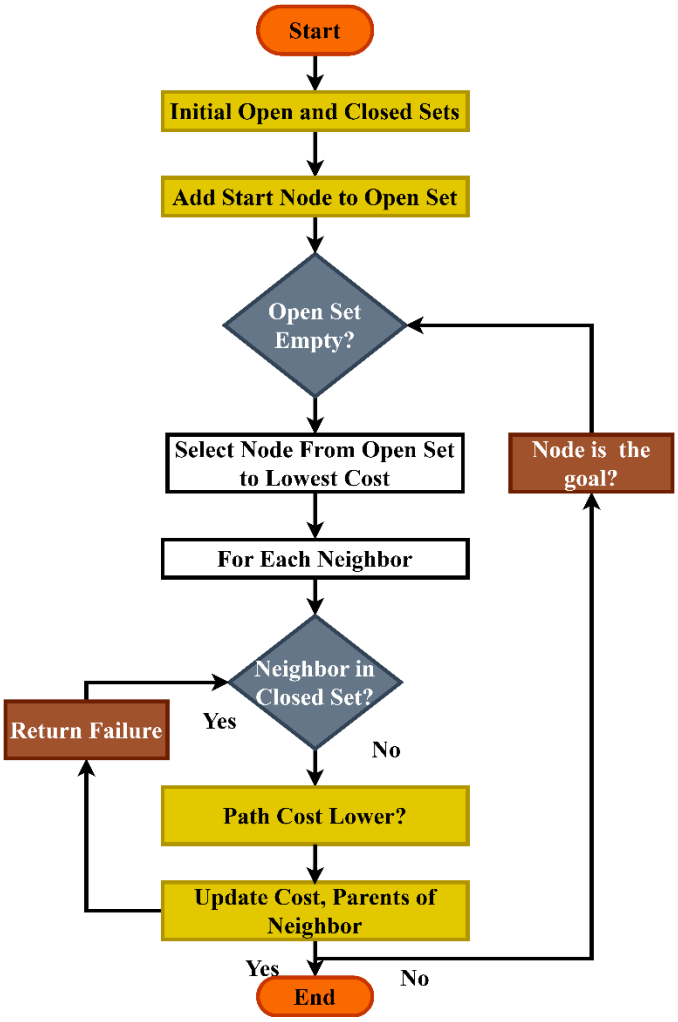

Figure 4: A* star Algorithm

Figure 4 depicts the flowchart for the A* (A-star) method, a well-known algorithm recognized for its efficiency and accuracy in pathfinding and graph traversal. The first step in the process involves initializing both the open set, which is necessary for monitoring nodes that have not yet been investigated, and the closed set, which is essential for tracking nodes that have been inspected. The start node is first added to the open set to initiate the procedure.

Next, the core logic will determine whether or not the open set contains any items by performing a check. If this were to occur, the algorithm would indicate that it could not find a specific route. It uses the formula $f(n) = g(n) + h(n)$ to determine which node is the least costly. In this calculation, $g(n)$ represents the cost that has already been incurred from the starting node to n, and $h(n)$ represents the cost that is anticipated to be incurred from n to the destination. Following this, the algorithm double-checks the picked node to ensure it is the target node. If this is the case, then we have accomplished our goal and the process of reconstructing the route. The software will check all nodes near it if the aim is still not achieved after this. Before evaluating, it determines whether or not a neighbor is already included in the closed set. In light of these conditions should steer clear of the neighbour.

On the other hand, the system will determine whether the cost of the route is lower than that of its neighbor if it has not previously been acknowledged as such. At this point, it adjusts the neighbor's price so that it equals the current node's cost. Additionally, it moves the parent node to the current node, making it possible to trace an optimal path in the future.

This cycle will continue to repeat until the target is located or the open set is depleted, indicating that there is no way to proceed. The A* algorithm strikes a balance between efficiency and optimality by utilizing heuristics to prioritize examining paths with the most significant potential for success.

$$f(n) = g(n) + h(n), where\ h(n) = \beta.Euclidean\ distance\ (n, goal) \quad (7)$$

As determined in Equation (7), the *Heuristic Function* has been discussed. The path cost from the starting point to node n is denoted by $g(n)$, $f(n)$is the total estimated cost of the best path through node n and the Euclidean distance between n and the goal are denoted by $h(n)$. The standard gradient is applied to the algebraic slope. One can estimate the cost of traveling from n to the goal later by using the $distance\ (n, goal)$ function. The adjusted coefficient β determines the weight assigned to the heuristic estimation and the actual charge. The robot can navigate even the most complex and crowded hotel layouts efficiently and quickly using A*. It is possible to promptly reroute to avoid temporary obstacles, such as guests or service carts, which helps to ensure that tasks are completed efficiently and that customers are satisfied.

$$Goal^* = arg \max_{T \in \varsigma_q} [U(T_i).Q(S_t, navigate(T_i); \theta)] \quad (8)$$

As discussed in equation (8), Adaptive Goal Selection has been examined. and the Q-value for reaching T_i target is worthwhile (utility) and will have an impact over time (learned Q-value). Intelligent scheduling is achieved by combining experience with present priorities through this mechanism. This flexible system includes the learned Q-value and the task's utility $U(T_i)$ to maximize this combined metric, the bot prioritizes time-sensitive and resource-efficient tasks. This metric indicates which tasks are likely to be profitable. By requiring the robot to evaluate its goals in context, this dual criterion improves its adaptability to new guests or hallway congestion. It connects abstract ideas to real-world applications.

$$R_t = \delta_1.Task\ Completion - \delta_2.Collision\ Penalty - \delta_3.Time\ taken + \delta_4 Energy\ Efficiency \quad (9)$$

As described in equation (9), the Reward Function has been calculated. Specific behaviors are either supported or discouraged by particular components. δ_1 is a recommendation for finishing tasks, δ_3 is a warning against inefficiency and δ_2 is a warning against unsafe navigation. The number denotes an operation that is both energy-efficient and profitable δ_4. By modifying the values of the coefficients δ_1 through δ_4, designers can modify robots' behavior to meet service objectives such as responsiveness, safety, and sustainability. With the assistance of this well-organized system of rewards, hotels can now teach their staff members how to be confident and friendly in various situations.

$$\pi^*(S_t) = arg \max_{A_t \in A} [Q(S_t, A_t; \theta_t) - \eta C(path(A_t))] \quad (10)$$

As depicted in equation (10), the Hybrid Policy Function has been computed. This hybrid policy weighs the Q-network's expected reward and the action's execution path cost to choose the best action. The total cost of executing action $\eta C(path(A_t)$, is determined by η, which balances reward and risk. This ensures the chosen actions are beneficial and feasible in dynamic environments. This hybrid policy maximizes rewards and minimizes costs by considering path cost and Q-function action value. By balancing reward and path cost, η enables context-sensitive decision-making. If its battery is low, the robot chooses a less-than-ideal job over a shorter, safer route. Autonomous service in dynamic, human-populated environments, such as hotels, requires balancing task performance, safety, and resource management.

4 Numerical results and discussion

The proposed SchedNav-RX framework was tested and compared to three current task scheduling and path planning models: TAMP, B-IG-ORPS, and RL-QPSO Nett. The four critical performance metrics utilized were Task Completion Rate, Average Path Efficiency (APE), Computational Time, and Collision Avoidance Rate. A range of 10 to 100 tasks was tested in dynamic multi-robot hotel service scenarios to assess scalability and resilience. Overall work sizes, SchedNav-RX outperformed competing navigation algorithms, boasting a job completion rate of up to 94% and showing slight degradation as the project load increased. It is evident that this effectively reduces robot idle time and work overlap by utilizing its AI-powered dynamic scheduling technology. With average path efficiency values of around 90%, the proposed model outperformed the competition in route optimization. This means the navigational paths were more direct and smoother even when faced with obstacles and workloads. By outperforming rival algorithms in complex task allocations and route computations, SchedNav-RX demonstrated its computational scalability through reduced processing overhead. As little as 1.9 seconds was required for 10 jobs, while 3.7 seconds was sufficient for 100 tasks. In conclusion, SchedNav-RX demonstrated the effectiveness of its environment-aware and adaptive real-

time path replanning features by achieving a high Collision Avoidance Rate (up to 96%) across all task levels. In light of these results, the proposed approach is quick, reliable, and well-suited to practical scenarios that require interior service robots, such as those in the hospitality industry.

Dataset Description: The 2D grid-based maps in dcaffo's Kaggle 2D Route Planning Dataset are used to test and evaluate AI-enhanced route planning methods, such as A* and D*. Each dataset sample contains grayscale images of 2D grid configurations with white pixels representing vacant space and black pixels representing impediments. These maps are great for hotel, warehouse, and workplace service robots. This dataset enables us to evaluate algorithm performance under various restrictions, ranging from simple pathways with few obstacles to those with numerous randomly dispersed obstacles. It aids in adaptive route planning, shortest route finding, and collision avoidance. The grid-based environment is ideal for testing AI-based robotic navigation frameworks, such as SchedNav-RX, which simulates internal dynamic decision-making. The dataset is useful for both theoretical and practical applications in artificial intelligence, particularly in autonomous navigation systems, as it is simple, compatible with standard route planning algorithms, and suitable for indoor robots [19].

Experimental Setup: During the trials conducted in a hotel environment replicated using ROS and Gazebo, service robots were hired to deliver packages and provide assistance. The SchedNav-RX algorithm was evaluated when compared to other methods (TAMP, B-IG-ORPS, and RL-QPSO Nett) for ten to one hundred different workloads. The robots were equipped with artificial intelligence modules and virtual sensors to improve their vision and navigation capabilities, including LiDAR and ultrasonic sensors. When evaluating each strategy, the criteria used were the completion rate of tasks, the average path efficiency, the amount of time spent computing, and the collision avoidance rate. With its Intel i7 central processor unit, 32 gigabytes of random-access memory, and RTX 3080 graphics processing unit, the simulations were carried out on a powerful machine capable of real-time processing and artificial intelligence computing.

i) Task completion rate (%)

Table 2: Task Completion Rate (%)

Number of Tasks	TAMP	B-IG-ORPS	RL-QPSO Nett	SchedNav-RX (Proposed)
10	77	83	87	93
20	79	80	85	94
30	76	82	86	91
40	74	78	83	90
50	78	79	84	92
60	73	76	81	89
70	75	77	83	91
80	70	75	82	90
90	72	74	80	89
100	74	76	81	90

$$Task\ Completion\ Rate = \left(\frac{Number\ of\ Tasks\ Completed}{Total\ number\ of\ tasks\ Scheduled}\right) \times 100 \quad (11)$$

Equations (11) and Table 2 illustrate the time required to complete the assignment. The hotel service robot's task completion rate (TCR) is used to evaluate its overall performance in fulfilling the assigned duties. Guests can get assistance with room supplies, cleaning arrangements, and even frequent inspections using the hotel's automated system. A technique used to solve this issue is to divide the total number of tasks by the percentage of tasks assigned and completed within a certain time frame. This leads to the acquisition of the TCR percentage. When it comes to explaining things, most individuals think that percentages are the most effective. Due to its high TCR, the SchedNav-RX architecture performs effectively in dynamic hotel settings, particularly in terms of prioritization, sequencing, and task execution. The findings of research like this are contributing to the growing body of data suggesting that AI-driven job scheduling systems have the potential to reduce robot idle time, manage resource competition, and prevent delays. The technique's resilience and flexibility in maintaining operational scalability and service continuity are validated by findings that provide TCR values that are more than 90% across various workload intensities and environmental conditions.

ii) Average path efficiency

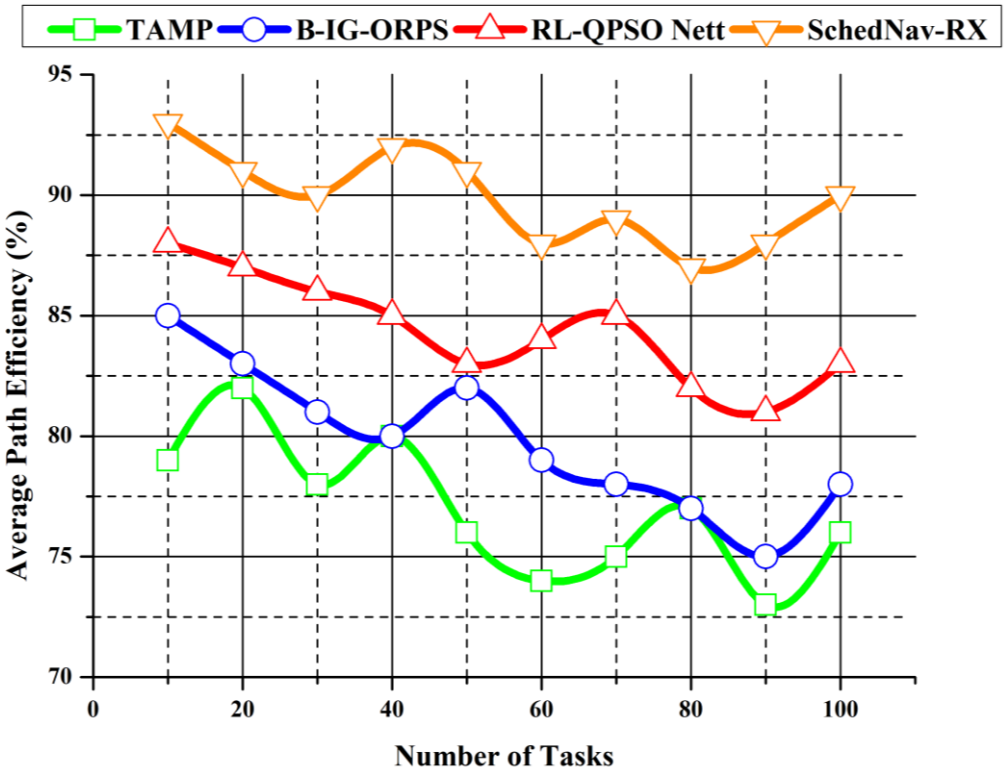

Figure 5: Average path efficiency (%)

$$APE = \frac{Optimal\ Path\ Length}{Actual\ Path\ Length} \tag{12}$$

Figure 5 and Equation (12) have explored the average path efficiency. One of the most critical indicators for evaluating the effectiveness of the route planning module within the SchedNav-RX architecture is the Average Route Efficiency (APE). Specifically, it measures the degree to which the robot's path corresponds to the theoretical shortest path, which is one way it represents spatial optimization in navigation. To describe it technically, the APE is the ratio of the actual route length of the robot to the ideal path length (computed using the A* technique or a heuristic-enhanced form of the methods). An APE score close to one indicates that the travel routes are almost optimal, with few unnecessary moves, diversions, or oscillations. This is the case when the APE score is close to 1. By maximizing route utilization, it is possible to significantly reduce the time spent on service, as well as the energy used in interior hotel settings. The SchedNav-RX system's ability to adjust to both fixed and moving impediments in real-time, while maintaining navigational accuracy and timeliness, is demonstrated by the high APE values recorded by the system. During the performance evaluation process, the APE is a crucial statistic used to assess the degree to which the components of motion planning and sensory perception (via CNNs) are integrated.

iii) Computational time (seconds)

Table 3: Computational time

Number of Tasks	TAMP	B-IG-ORPS	RL-QPSO Nett	SchedNav-RX (Proposed)
10	3.1	2.9	2.6	1.9
20	4	3.5	3.2	2.4
30	4.4	3.9	3.5	2.2
40	5.9	5.3	4.8	3
50	5.4	4.9	4.3	2.6
60	6.2	5.8	5.1	3.1
70	6	5.5	4.9	3.3
80	6.5	6.2	5.7	3.8
90	6.9	6	5.5	3.5
100	7.2	6.6	5.9	3.7

$$CT_{\text{SchedNav-RX}} = T_{CNN} + T_{Schedule} + T_{A^*} + T_{Collisioncheck} + T_{update} \quad (13)$$

Table 3 and Equation (13) have been examined for computational time. The computational time is a performance metric that analyzes the accuracy and speed with which the SchedNav-RX system can construct work schedules and navigation courses. It measures how quickly and accurately the system can create information. After receiving input, such as a new service request or a change in the environment, the system takes a certain length of time to develop a plan that is 100% operational. Since delays impact both customer satisfaction and operational efficiency, this statistic is of significant relevance for real-time systems used in service-oriented industries. For perceptual analysis, the suggested system utilizes lightweight convolutional neural networks (CNNs), while for route planning, an enhanced version of the A* algorithm is employed. The ability to create speedy responses is enabled by this, even in complicated circumstances. The model, data structures, and processing pipelines are well-designed if the runtimes decrease. The empirical study determines whether the architecture is plausible for incorporation into actual smart hotel infrastructure by comparing normal reaction times in various circumstances, such as low- and high-traffic zones or static and dynamic impediments.

iv) Collision avoidance rate (%)

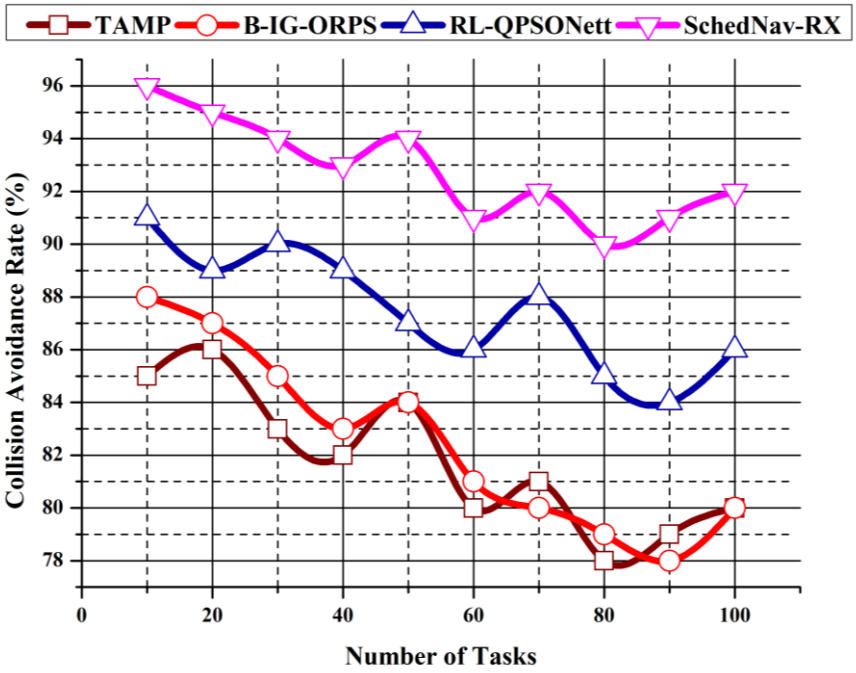

Figure 6: Collision avoidance rate (%)

$$\textbf{\textit{Collision Avoidance Rate}}\ (\%) = \left(\frac{\textbf{\textit{Tasks Completed without Collision}}}{\textbf{\textit{Total Tasks Attempted}}}\right) \times \mathbf{100} \quad (14)$$

Figure 6 and equation (14) show that the collision avoidance rate has been expressed. The robustness and safety of the navigation system are evaluated using the Collision Avoidance Rate (CAR), which is calculated by determining the percentage of task executions that do not entail physical contact with barriers or disruptions to the route integrity. It is necessary to avoid rigid items, such as walls or furniture, and mobile objects, such as clients, service staff, or other robots. Having a high CAR would make the robot safer for the physical infrastructure and increase the trust that hotel personnel and guests have in the robot's reliability. By utilizing a cost-aware A* variant that continually adapts to changes, the SchedNav-RX framework can achieve a high CAR. This is accomplished by integrating environmental awareness with CNN-based object detection and dynamic replanning. Moreover, techniques for smoothing pathways and modifying velocities are also helpful in ensuring safe navigation. The assessment confirmed the effectiveness of the proposed AI-based method in real-world, human-centric situations, with consistently high

CAR (more than 95%) across various hotel floor designs and population densities.

5 Conclusion

To conclude, the SchedNav-RX framework optimizes hotel service robot job scheduling and route planning using AI approaches including CNN, RL, and the A* algorithm. Experimental results show that SchedNav-RX outperforms TAMP, B-IG-ORPS, and RL-QPSO Nett in terms of Task Completion Rate, Average Path Efficiency, Computational Time, and Collision Avoidance Rate. SchedNav-RX ensures safe navigation in dynamic settings, efficient and adaptable route selection, and suitable work assignments. Real-world 2D grid-based datasets demonstrate the framework's reliability and suitability for autonomous robot deployment in outdoor service domains, such as hotels and other hospitality settings. Future developments include multi-agent coordination and real-time environmental adaptability to improve framework scalability and responsiveness.

Fundings

This work was supported by 2024 Annual Higher Education Teaching Reform Research and Practice Project in Henan Province, "Double-element Orientation and Five-chain Integration": Research and Practice on the Construction Path of Tourism Professional Cluster in Higher Vocational Schools (No. 2024SJGLX0719).

References

[1] Jeon, J., Jung, H. R., Luong, T., Yumbla, F., & Moon, H. (2022). Combined task and motion planning system for the service robot using hierarchical action decomposition. Intelligent Service Robotics, 15(4), 487-501. https://doi.org/10.1007/s11370-022-00437-3

[2] Hafiz, Syed Imran, and Wong Kai Ming. “Real-Time Algorithms for Gesture-Based Control in Robotics and Gaming Systems.” PatternIQ Mining., vol. 1, no. 4, Nov. 2024, pp. 12–23. https://doi.org/10.70023/sahd/241102.

[3] Wang, Z., & Tian, G. (2022). Hybrid offline and online task planning for service robot using object-level semantic map and probabilistic inference. Information Sciences, 593, 78-98. https://doi.org/10.1016/j.ins.2022.01.058

[4] Guo, H., Wu, F., Qin, Y., Li, R., Li, K., & Li, K. (2023). Recent trends in task and motion planning for robotics: A survey. ACM Computing Surveys, 55(13s), 1-36. https://doi.org/10.1145/3583136

[5] Yoo, T., & Choi, B. W. (2024). Interactive Path Editing and Simulation System for Motion Planning and Control of a Collaborative Robot. Electronics, 13(14), 2857. https://doi.org/10.3390/electronics13142857

[6] Li, Z., Shi, N., Zhao, L., & Zhang, M. (2024). Deep reinforcement learning path planning and task allocation for multi-robot collaboration. Alexandria Engineering Journal, 109, 408-423. https://doi.org/10.1016/j.aej.2024.08.102

[7] Gunawan, A. A., Clemons, B., Halim, I. F., Anderson, K., & Adianti, M. P. (2023). Development of e-butler: Introduction of robot system in hospitality with mobile application. Procedia Computer Science, 216, 67-76. https://doi.org/10.1016/j.procs.2022.12.112

[8] Tuomi, A., & Ascenção, M. P. (2023). Intelligent automation in hospitality: exploring the relative automatability of frontline food service tasks. Journal of Hospitality and Tourism Insights, 6(1), 151-173. https://doi.org/10.1108/JHTI-07-2021-0175

[9] Bao, Q., Zheng, P., & Dai, S. (2024). A digital twin-driven dynamic path planning approach for multiple automatic guided vehicles based on deep reinforcement learning. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 238(4), 488-499. https://doi.org/10.1177/09544054231180513

[10] Zhang, H., Chan, S. H., Zhong, J., Li, J., Kolapo, P., Koenig, S., ... & Nikolaidis, S. (2023). Multi-robot geometric task-and-motion planning for collaborative manipulation tasks. Autonomous Robots, 47(8), 1537-1558. https://doi.org/10.48550/arXiv.2310.08802

[11] Duan, P., Yu, Z., Gao, K., Meng, L., Han, Y., & Ye, F. (2024). Solving the multi-objective path planning problem for mobile robot using an improved NSGA-II algorithm. Swarm and Evolutionary Computation, 87, 101576. https://doi.org/10.1016/j.swevo.2024.101576

[12] Li, X., Tian, B., Hou, S., Li, X., Li, Y., Liu, C., & Li, J. (2023). Path planning for mount robot based on improved particle swarm optimization algorithm. Electronics, 12(15), 3289. https://doi.org/10.3390/electronics12153289

[13] Lei, T., Sellers, T., Luo, C., Carruth, D. W., & Bi, Z. (2023). Graph-based robot optimal path planning with bio-inspired algorithms. Biomimetic Intelligence and Robotics, 3(3), 100119. https://doi.org/10.1016/j.birob.2023.100119

[14] Gu, T., Ren, C., Yin, L., Liao, Z., Li, W., Sun, F., & Wang, H. (2022). Scheduling scheme design of hotel service robot: A heuristic algorithm to provide personalized scheme. Wireless Communications and Mobile Computing, 2022(1), 7611308. https://doi.org/10.1155/2022/7611308

[15] Lei, C., Hossain, M. S., & Wong, E. (2023). Determinants of repurchase intentions of hospitality services delivered by artificially intelligent (AI) service robots. Sustainability, 15(6), 4914. https://doi.org/10.3390/su15064914

[16]Jing, Y., & Weiya, L. (2025). RL-QPSO net: deep reinforcement learning-enhanced QPSO for efficient mobile robot path planning. Frontiers in Neurorobotics, 18, 1464572. https://doi.org/10.3389/fnbot.2024.1464572

[17]Zhu, X. (2025). Multi-Task Deep Reinforcement Learning for Intelligent Logistics Path Planning and Scheduling Optimization. Informatica, 49(20). https://doi.org/10.31449/inf.v49i20.7996

[18]Bendiaf, L. M., Harbouche, A., Tahraoui, A. M., & Lebbah, F. Z. (2024). An Innovative Task Scheduling Method Using the Knapsack Algorithm in Heterogeneous Computing Systems. Informatica, 48(16). https://doi.org/10.31449/inf.v48i16.5765

[19]https://www.kaggle.com/datasets/dcaffo/2dpathplanningdataset