Q-RCR: A Modular Framework for Collision-Free Multi-Package Transfer on Four-Wheeled Omnidirectional Conveyor Systems

Syamsiar Kautsar [1], Aulia Siti Aisjah [2*], Syamsul Arifin [3], Mat Syai'in [4]
[1, 2, 3] Engineering Physics Departement, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
[1] Engineering Departement, Politeknik Negeri Jember, Jember, Indonesia
[4] Marine Electrical Engineering Departement, Politeknik Perkapalan Negeri Surabaya, Indonesia
Email: [1] 7009232002@student.its.ac.id, [2] auliasa@ep.its.ac.id, [3] syamsul@ep.its.ac.id, [4] matt.syaiin@ppns.ac.id
*Corresponding Author

***Abstract*—Modern logistics systems increasingly require high flexibility in handling simultaneous package transfers in compact, dynamic environments without collisions. Improper handling of multi-package transfers in omnidirectional conveyor systems can lead to deadlocks, congestion, or delivery delays, particularly in grid-based environments where routing complexity increases with package variability and layout density. This research addresses these challenges by introducing Q-RCR, a modular Q-Learning-based framework with Rule-Based Conflict Resolution (RCR) for intelligent path planning and collision handling in Four-Wheeled Omnidirectional Cellular Conveyor (FOCC) systems. The research contribution is decoupling path learning and collision handling, enabling independent agent training while minimizing computational burden and improving convergence in multi-agent scenarios. The proposed Q-RCR framework integrates Q-Learning for route optimization with a rule-based conflict resolution module, applying four adaptive strategies: Sequential Transfer, Insert Path, Reroute, and Hybrid. The method is implemented in a grid-based FOCC environment, supporting eight-directional movement and handling various package sizes. Experiments were conducted in four scenarios with grid dimensions ranging from 8×11 to 12×12 and involving up to four simultaneous packages. Results show that Q-RCR consistently outperforms Double Q-Learning, RRT, and A* regarding delivery time, path smoothness, and the number of activated cells. The hybrid mode demonstrated the most effectiveness in handling frequent collisions and maintaining operational flow continuity. The proposed framework demonstrates strong adaptability, scalability, and responsiveness, offering a practical and intelligent solution for real-time multi-package coordination in flexible manufacturing and warehouse automation environments.**

Keywords*—Q-Learning; Adaptive Path Planning; Collision Avoidance; Omnidirectional Cellular Conveyor; Simultaneous Transfer; Logistics Optimization; Flexible Manufacturing.

I. INTRODUCTION

The Omnidirectional Cellular Conveyor (OCC) is an innovative conveyor system that utilizes omnidirectional wheels to facilitate the flexible movement of packages in various directions [1], [2], [3]. Each cell comprises several drive motors with omni wheels [4], [5], [6], [7]. These modules operate integrated, utilizing centralized or decentralized control to strategize routes, manage velocities, and prevent collisions among various packages. Omnidirectional conveyors are designed to overcome the constraints of traditional static conveyor systems[8], [9], [10] by providing enhanced maneuverability, route flexibility, and energy economy. The system supports simultaneous movement and sorting of multiple packages with high precision, making it highly suitable for modern logistics applications that demand adaptability to dynamic requirements [11], [12], [13].

An intelligent and adaptive path planning mechanism is essential to fully realize the potential of OCC systems, particularly under conditions involving multi-package coordination, dynamic routing, and limited space, Conventional rule-based routing or static scheduling techniques often lack the flexibility to respond to real-time layout changes, congestion, or the presence of varied package sizes. In this context, Reinforcement Learning (RL) [14], [15], [16], [17] presents a compelling solution, as it allows autonomous agents to learn navigation strategies directly from interaction with the environment, without requiring predefined models or explicit programming.

Among various Reinforcement Learning (RL) techniques, Q-Learning (QL) stands out due to its simplicity and proven effectiveness in discrete environments. Q-Learning enables agents to iteratively improve their decision-making policies based on reward feedback, making it especially suitable for systems with non-fixed or dynamic configurations such as Omnidirectional Cellular Conveyors (OCC) [18], [19], [20], [21], [22], [23]. Previous research, including the development of the Action Restrictions Q-Learning (ARQL) algorithm [24], has applied RL for route optimization in Four-Wheeled Omnidirectional Conveyor (FOCC) systems. However, this approach mainly focuses on single packet transfer with a uniform package size. Consequently, it fails to address the complex challenges of simultaneous multi-package transfer, variable object dimensions, and real-time conflict resolution in dense conveyor layouts.

To bridge this gap, this study introduces a hybrid framework, Q-RCR, combining the path planning and management conflict resolution in the FOCC system. Path planning is carried out through independent single-agent Q-Learning, where each package learns its optimal route individually. This decoupling of agents during training reduces computational overhead and mitigates convergence issues commonly associated with multi-agent RL setups. Meanwhile, collision handling is managed through a modular, deterministic Rule-Based Conflict Resolution (RCR) layer operating outside the learning process. This rule-based mechanism detects and resolves potential collisions in real time using predefined strategies, including Sequential Transfer, Insert Path, Rerouting, and hybrid combinations. Such separation streamlines the training phase and enhances the system's flexibility and adaptability to varying operational scenarios.

The Q-RCR framework is applied to a four-wheeled omnidirectional conveyor platform that provides eight directional movement possibilities, improving the six-directional configurations utilized in [25]. Moreover, the system is designed to handle packages of different sizes, thus accommodating more realistic and practical logistics scenarios. This aspect addresses a significant limitation in prior work [26], which often assumes static trajectories and uniform object dimensions. The key contributions of this research lie in creating a scalable and efficient framework for time-efficient multi-packet transfer on FOCC systems. By adopting a Q-Learning-based path planning approach with independent agent training, Q-RCR achieves reduced computational complexity and faster convergence. The deterministic rule-based conflict resolution ensures smooth and collision-free operation during runtime without impacting the learning process. Comparative experiments demonstrate that Q-RCR outperforms other baseline methods, including Tabular Q-Learning (TQL) [25], Double Q-Learning (DoQL), Rapidly-exploring Random Trees (RRT), and A*, regarding computational efficiency and success rate in complex, high-density environments. Overall, Q-RCR offers a flexible and practical solution for intelligent multi-package transfer in dynamic industrial conveyor systems.

II. RELATED WORKS

Automatic sorting systems (ASS) have evolved through various technological advancements to meet the increasing demands of the logistics sector[27], [28]. In ASS, path planning plays an important role in the smooth running of the package distribution process[29]. Complex material handling systems often encounter deadlock, livelock, or starvation problems, which can disrupt the operating system[30]. According to [31], bottlenecks in the distribution chain can be cut down by using the right scheduling algorithms, picking the correct sorting system layout, and innovative control methods. The GridSorter system with FlexConveyor was proposed by [32] as a solution to overcome the deadlock problem. A distributed control strategy utilizing negotiation across modules can prevent deadlocks using the concept of logical time. The GridSorter system is asserted to be more secure than the Automatic Guide Vehicle (AGV) system [33], [34], [35], [36], which is prone to deadlock issues, particularly when routes overlap and are inadequately managed. The combination of a manipulator in a conveyor-based sorting system was proposed by [37]. The operational process of the 5-DOF robotic arm manipulator in conjunction with the conveyor in this material sorting system, as described in this paper, operates automatically based on sensor detection and microcontroller-based control. The robotic arm features five degrees of freedom (DOF), allowing for flexibility in grasping and placing objects with high precision. However, the addition of a manipulator may increase production costs.

Furthermore, the development of mechanical wheel types, including omni wheels and mechanical wheels, led to the creation of an omnidirectional conveyor system. The three-wheeled omnidirectional cellular conveyor (TOCC) is a conveyor system consisting of hexagonal cells equipped with omnidirectional wheels [1]. Each cell consists of three omniwheels, forming a 120-degree angle. The OCC system can be arranged with various cell size configurations according to user needs. By utilizing the holonomic properties of the omnidirectional wheel drive [38], [39], [40], [41], package distribution can be carried out in multiple directions like a swiveling roller conveyor but with fewer actuators. In addition, using OCC allows rotational movements with a minimum radius compared to swivel conveyors.

The OCC system is designed to accommodate dynamic variations in entry (inbound) and exit (outbound) gate positions, reflecting real-world logistics operations where goods flow between warehouses and loading areas through interconnected conveyor belts. To address the path planning needs within this flexible architecture, [25] investigated the use of RL in a TOCC. Their research evaluated various reinforcement learning algorithms, including TQL, DoQL, Deep Q-Learning, and Double Deep Q-Learning, in the context of multi-package sorting scenarios. The results demonstrated that TQL and DQL achieved faster convergence and more stable collision management compared to their deep-learning counterparts. However, these approaches still rely on multi-agent training in a shared environment, significantly increasing the computational time and complexity as the number of agents (packets) increases. Furthermore, conflict resolution is embedded directly in the RL process, which makes the learning phase very sensitive to collision scenarios and does not accommodate handling packets of different sizes.

In contrast, [26] proposed a robust motion control framework for omnidirectional conveyors by integrating Fuzzy Sliding-Mode Tracking Control (FSTC) with a Fuzzy Inter-package Collision Avoidance (FICA) scheme. This hybrid control method provided real-time collision handling by generating deflection angles and force amplification to reroute packages away from potential collisions. Even under disturbances, the approach demonstrated strong trajectory tracking performance, offering fast convergence and smooth movement. However, the method relied heavily on predefined fourth-order Bézier curves to generate motion trajectories. This dependence limited the system’s ability to dynamically adjust to varying gate positions or object sizes, as each planned path was fixed in both shape and length. Additionally, while FICA offers real-time adjustments, the system lacks an autonomous path learning mechanism that can be generalized to different environments with limited routes.

In the case of path planning, RL provides high flexibility for path determination in varying environments. In the domain of mobile robotics, Q-learning is capable of managing several mobile robots within a single operational environment [42], [43], [44], [45]. In this step, a modular architecture is used. This architecture has separate learning modules for each agent and a mediator module that uses the Q value and state information to decide the final action. Experiments on soccer mobile robots prove that this method can determine the robot's fastest time to kick the ball to optimize overall team performance.

The Q-learning strategy for multi-agent system navigation in complex environments is also proposed by [46], [47], [48]. Applying the Compressed State Space (CCSS) approach with the controller effectively decreases memory requirements by as much as 70% relative to traditional methods [48]. It is achieved by compressing the state space through non-overlapping features and diminishing the degree of discretization. In order to improve the performance, a modification of Q-Learning for mobile robot path planning is proposed by [49]. Improved Q-Learning (IQL) algorithm combined with enhanced Ant Colony Optimization algorithm (IAC-IQL) for global path planning of search and rescue (SAR) robots using Bessel curves. Simulation experiments were conducted in three scenarios: a traditional grid environment, a complex grid environment, and a real-world image-based environment. The results demonstrate that the IAC-IQL algorithm significantly improves search efficiency, produces smoother paths, and outperforms other path-planning methods such as ACO, Q-Learning, and the Sparrow Search Algorithm. Therefore, this method has great potential for real-world SAR robot applications.

A modification of QL is also proposed according to similar environmental conditions, as discussed in [50]. The grid method is used to simulate the environment and represent obstacles. The IQL algorithm incorporates a priority weight to address the issue of action selection. Based on the test results, the IQL algorithm can find the shortest path in the shortest time compared to the A* algorithm, the Probabilistic Roadmap Algorithm (PRM), the Bidirectional Rapidly Exploring Random Tree Algorithm (BRRT), and RRT. To further improve the performance of training results, the combination of Q-learning with the Artificial Potential Field (APF) method is proposed by [18]. The QAPF algorithm is used for path planning on mobile robots in offline and online obstacle environments. QAPF consists of three primary operations: exploration, exploitation, and APF weighting. The offline testing phase demonstrated notable improvements in path length, path smoothness, and computation time compared to CQL. Similarly, during the online testing phase, the QAPF algorithm outperformed CQL regarding path efficiency, smoothness, and overall work time.

III. METHOD

This section outlines the systematic approach used in implementing the Q-RCR framework to ensure collision-free, simultaneous package delivery in the Four-Wheeled Omnidirectional Cellular Conveyor (FOCC) system. By combining Q-Learning-based artificial intelligence with rule-based conflict management. The framework offers an effective path-planning solution, enabling adaptive and efficient collision avoidance strategies. This section comprehensively explains the FOCC environment initialization, the selection and application of path planning algorithms, the rule-based conflict detection and resolution mechanisms, and the testing scenarios designed to evaluate the framework’s performance under various operational configurations. The modular approach optimizes the efficiency of delivery time and navigation accuracy in complex and dynamic environments.

A. *FOCC Environment*

This study develops the Omniconveyor, which utilizes a four-wheel omnidirectional drive system [51], [52]. The initialization of the FOCC environment is similar to previous research [24]. FOCC accommodates eight transfer directions used as actions in path planning. The FOCC environment consists of the OCC size, the inbound station, the outbound station, and obstacles. For RL training, the cell configuration on the FOCC is depicted as a grid map.

This study uses a cube-shaped package. δ is the size of the cube side, d is the diagonal distance between wheels, and l is the size of the cell side. In FOCC, the diagonal distance between wheels in one cell is the same as that between adjacent wheels in the cell. The depiction of the FOCC environment is shown in Fig. 1. The cell numbering is determined based on the positive axis position in Cartesian coordinates (x, y). The cell numbering (1, 1) starts from the bottom left point. The FOCC shown in Fig. 4 is 5×4 in size with an entry coordinate (2, 5) and an exit coordinate (0, 3). Next, path planning will be used to determine the most efficient route to transport the package from the entry gate to the exit gate. The algorithm will assess the surrounding environment, considering obstacles and potential paths.

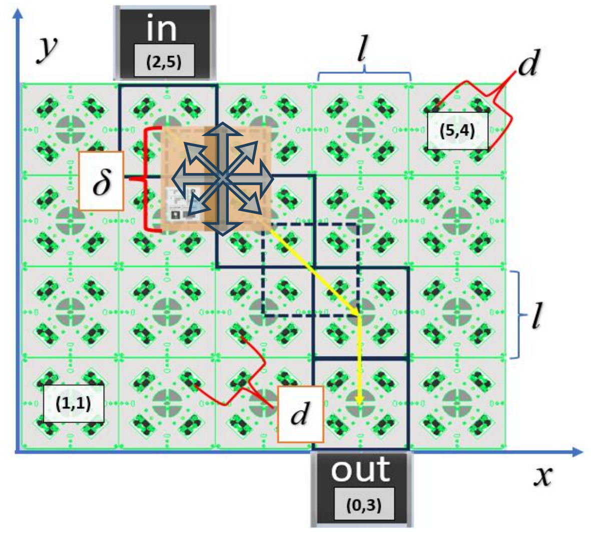

Fig. 1. FOCC environment

B. *Path Planning Algorithm*

In this study, five types of path planning algorithms are used, specifically: Q-Learning (QL), Tabular Q-Learning (TQL), Double Q-Learning (DoQL), RRT, and A*. All path planning uses eight possible delivery directions in the Moor Neighborhood configuration.

1) Q-Learning

Q-learning is a reinforcement learning algorithm utilized to determine the optimal policy in scenarios characterized by the Markov Decision Process (MDP) [53]. The QL algorithm employs an off-policy method to differentiate between the action and learning policies. As a result, when the action selected in the next state is suboptimal, the corresponding information is excluded from the Q-function update of the current state. Q-learning enables agents to acquire the ability to select actions that maximize the total future rewards [54]. Agents can learn without needing a model from the environment.

The first step in Q-learning is the initialization of the Q-table, which connects the state and action. The action is subsequently chosen at random based on the epsilon value. The Q-table is updated following the chosen activities and received rewards [55]. The pattern repeats itself based on the number of episodes used. Equation (1) is the Classic Q-Learning (CQL) algorithm. α is a parameter that controls how much the value of $Q(s,a)$ is updated when a new experience occurs [56], [57], [58], [59]. If the value of α is high, the algorithm learns faster from new experiences. On the other hand, low alpha values make learning slower. The discount factor γ quantifies how much future reward values impact present decisions. This value reflects the preference for long-term rewards over short-term rewards. Epsilon is a parameter utilized in the ϵ-greedy exploration strategy, determining the proportion between exploration and exploitation during training.

$$Q_{\text{new}}(s,a) = Q(s,a) + \alpha \,\&r + \gamma \, max \quad {}_{a'} \, Q(s',a') \\ - [Q(s,a)] \tag{1}$$

2) Double Q-Learning

This study also assesses Double Q-Learning as a path-planning method. DoQL [60], [61], [62], [63], [64], attempts to reduce the overestimation bias inherent to Q-learning by utilizing two distinct sets of Q values that are independently updated. DoQL uses two distinct value functions: (2) and (3). $Q1$ and $Q2$ denote distinct collections of Q values. The action a' that optimizes the value of $Q1(s`, a')$ is employed to update $Q2$, and vice versa.

$$Q1(s,a) \leftarrow Q1(s,a) + \ldots \\ \alpha\left[r + \gamma Q2(s', \text{argmax}_{a'} Q1(s',a')) - Q1(s,a)\right] \tag{2}$$

$$Q2(s,a) \leftarrow Q2(s,a) + \ldots \\ \alpha\left[r + \gamma Q1(s', \text{argmax}_{a'} Q2(s',a')) - Q2(s,a)\right] \tag{3}$$

3) Tabular Q-Learning

Tabular Q-Learning is a basic RL method used by [25] for path planning and sorting packages in a three-wheeled omnidirectional conveyor system. This method stores the Q value (action quality) in a discrete table for each pair of states and actions. Collision avoidance is handled implicitly through the design of a reward function in the training process. The TQL approach uses QL development with multi-agent learning [37], [38] which is carried out in the same environment. Rewarding for two types of collisions is also applied in this training.

4) RRT

RRT, or Rapidly-exploring Random Tree, is a widely used probabilistic technique for solving motion planning problems [65], [66], [67], [68], [69], [70]. The basic process of the RRT algorithm is carried out iteratively through the following steps: a) Selecting a random point in space, which is then referred to as a sample point x_{rand}, b) determining the closest point, namely x_{near}, the node closest to x_{rand} in the tree, c) determining a new point based on a step size h, using (4). This equation shows how to create a new point x_{new} that is closer to x_{rand} based on x_{near} using a step size h.

$$\boldsymbol{x}_{new} = \boldsymbol{x}_{near} + h \cdot \frac{\boldsymbol{x}_{rand} - \boldsymbol{x}_{near}}{\|\boldsymbol{x}_{rand} - \boldsymbol{x}_{near}\|} \tag{4}$$

5) A*

A* represents an algorithm in the field of path planning, applicable to both metric and topological configuration spaces. This algorithm combines heuristic search with shortest path search [3], [71], [72], [73], [74], [75], [76]. A* is categorized as a best-first search algorithm due to the evaluation of each cell in the configuration space using (5). Each neighboring cell of the cell being evaluated will have its $f(v)$ value calculated. The cell with the smallest $f(v)$ value is selected as the next step in the sequence.

$$f(v) = g(v) + h(v) \tag{5}$$

Where, $h(v)$ is the heuristic distance from the cell to the goal, and $g(v)$ is the path length from the starting point to the cell.

C. Collision Detection Algorithm

Collision detection [77], [78], [79] is performed by determining the position of each package within the global frame over the delivery time span t = 0:$t_{outbound}$. During the initial stage, the Q-Learning algorithm will sequentially generate packet pathways based on the inbound and outbound coordinates inside the same FOCC environment. The trajectory produced by Q-Learning is interpolated to acquire points within a narrower range [80] through (6)-(9). This aims to enhance the precision of collision detection.

$$d_i = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \tag{6}$$

$$N_i = \left\lceil \frac{d_i}{\Delta s} \right\rceil \tag{7}$$

$$x_{\text{interp}}(k) = x_i - 0.5 + k \cdot \frac{(x_{i+1} - x_i)}{N_i - 1}, \quad k \\ = 0,1,\ldots,N_i - 1 \tag{8}$$

$$y_{\text{interp}}(k) = y_i - 0.5 + k \cdot \frac{(y_{i+1} - y_i)}{N_i - 1}, \quad k \\ = 0,1,\ldots,N_i - 1 \tag{9}$$

Where, d_i is the distance between lines, Ds is desired interpolation distance, k is the interpolation step and x_{inter} and y_{inter} is the X & Y values from interpolation.

At each positional change, the function utilizes Algorithm I to verify the intersection of the (x_{interp}, y_{interp}) packages. If the function detects a collision, it will return a value of 1, and if it doesn't, it will return a value of 0. Additionally, the coordinates of the cell identified as the collision location will be designated as the collision coordinate C (x_c, y_c).

Algorithm I: Collision Detection

```
Function st_collision = c_overlap(x1, y1, x2, y2, s1, s2)
  If:  s1==1
    w1 = 1;  h1 = 1;
  else: s2 = 2;
    w1 = 2;  h1 = 2;

  x_overlap_start = max(x1, x2);
  y_overlap_start = max(y1, y2);
  x_overlap_end = min(x1 + w1, x2 + w2);
  y_overlap_end = min(y1 + h1, y2 + h2);

  overlap_width = x_overlap_end - x_overlap_start;
  overlap_height = y_overlap_end - y_overlap_start;

   If: overlap_width >= 0 && overlap_height >= 0
     st_collision = 1;
   Else:
     st_collision = 0;
End
```

D. Rule-based Conflict Resolution

Rule-based Conflict Resolution (RCR) is a collision avoidance rule for simultaneous multi-package transfer. The scheme adopts the Modular Q-learning in the study [81], [82], [83]. In this paper, four approaches are used for collision conflict management. This aims to analyze the most appropriate technique for the case of multi-package transfer in FOCC. The conflict management modes used are Sequential (Sq-mode), Insert Path (IP-mode), Reroute (Rr-mode), and Hybrid (Hb-mode). The packet handling order for collision avoidance is based on the number of possible collisions, the path length, and the initialization order. The packet that will be handled first is the packet with the highest number of possible collisions. If the number of possible collisions is the same, then the packet with the shorter path will be handled. If the paths are the same, the path initialized last is handled in the order D-C-B-A.

1) Sequential Mode (Sq-Mode)

In the sequential process, if a collision is detected, the first package is sent first. Then, the second package is sent when the first packet has reached the goal. This approach is the most straightforward technique to avoid collisions because it does not require additional algorithms for path modification. Fig. 2 illustrates the sequential transfer process flow.

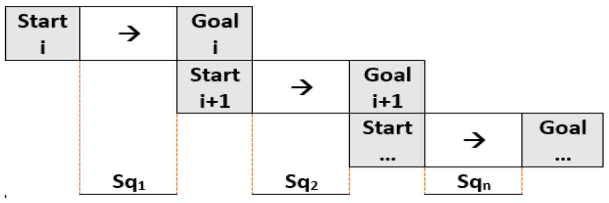

Fig. 2. Sequential Mode

2) Insert Path Mode (IP-mode)

During the IP-mode procedure, an insert path will be executed if a collision occurs. Path$_{i+1}$ will be appended to the identical cell position as path$_i$ The processed path is the one with the minimum distance. The package containing the subsequent initialization sequence will be handled if the distance is identical. After the execution of the insert delay, a collision potential assessment will be conducted once more. Upon detecting a collision, the algorithm will execute an insert delay repeatedly until the collision ceases to be detected. Fig. 3 depicts the flow of path addition in IP-mode.

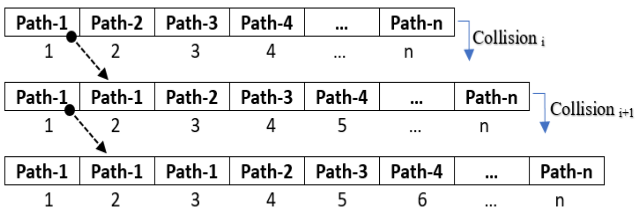
Path-1 Path-2 Path-3 Path-4 ... Path-n
1 2 3 4 ... n
Collision i
Path-1 Path-1 Path-2 Path-3 Path-4 ... Path-n
1 2 3 4 5 ... n
Collision i+1
Path-1 Path-1 Path-1 Path-2 Path-3 Path-4 ... Path-n
1 2 3 4 5 6 ... n

Fig. 3. Insert-path mode

3) Reroute Mode (Rr-Mode)

Once the collision coordinate C is determined, the reward value is changed at that point. The value of r determines the number of neighboring cells that are considered obstacles. The collision point and neighboring cells with a radius of r will be considered obstacles. Suppose $r = 1$, then one neighboring cell will be considered an obstacle, as shown in Fig. 4. Determining the cells that are considered obstacles is explained in Algorithm II. The algorithm will retrain the i+1 package and find the optimal route by considering the updated obstacles.

Algorithm II: Generate Radius Obstacles

```
Function  obstacles = add_obs(xc, yc, r)
  obstacles=[];
  For dx = -r:r
    For dy = -r:r
      new_point = [xc + dx, yc + dy];
      obstacles = [obstacels; new_point];
    End For
  End For
End Function
```

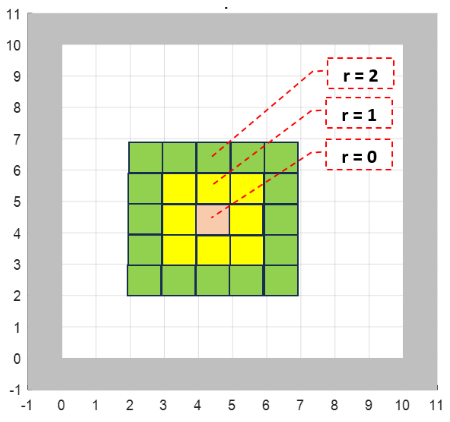

Fig. 4. Illustration of cell-radius obstacles

4) Hybrid Mode (Hb-M)

The 'hybrid' mode combines delay and obstacles. If the number of paths added to IP mode exceeds 2 times, a reroute process will occur with a value of r = 1 at the collision point.

Fig. 5 is a flowchart for the combination transfer process. A rule-based system [84], [85], [86] is established to determine whether the agent must regenerate pathways for multi-package delivery.

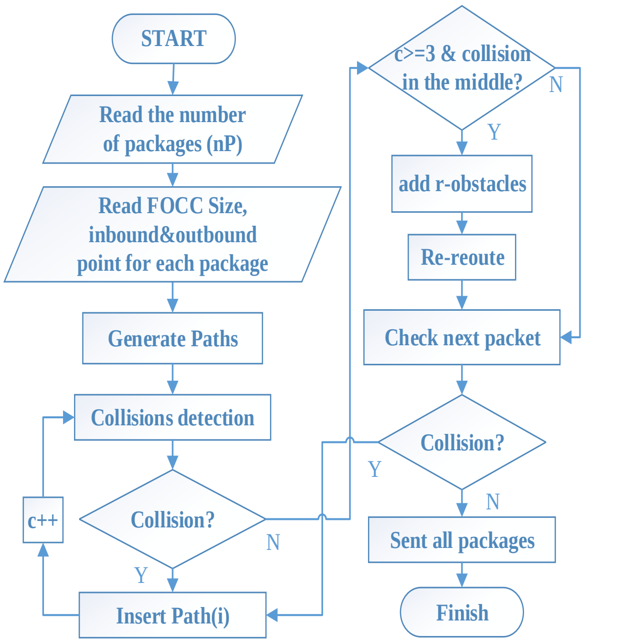

Fig. 5. Hybrid-mode flowchart

E. Testing Scenarios

In actual implementation, OCC can be configured with different cell sizes and gate numbers as shown in Fig. 6. Packets can be sent to different exit points randomly, according to the sorting data stored in the database. In this study, four test scenarios are created in scenarios I and II, FOCC with size 8×11 is used, and scenarios III and IV with size 12×12. In scenarios I and II, there are two entry points with coordinates (2,0) and (7,0), and four exit points are located at coordinates (0,8), (2,12), (7,12), and (8,10). In scenarios III and IV, there are four inbound points with coordinates (4,0), (9,0), (0,4), and (0,9), and four outbound points are located at coordinates (4,12), (9,12), (12,4), and (12,9), respectively. The four scenarios are created to accommodate several possible collisions in the FOCC system. There are two types of packet sizes used in this study, namely packet types S and M. Type S has a size of $\delta = l$ or equal to one cell. In contrast, type M has a size of $\delta = 2.l$, which is equal to the size of 4 cells.

Learning algorithm for path planning, each packet is considered a different agent in the same environment. The entry point is highlighted in green, the exit point in blue, and the gray line indicates the boundary of the omnidirectional cellular conveyor, as shown in Fig. 7. This experimental setup is designed to evaluate the flexibility and efficiency of the Q-Learning algorithm in various package sizes, entry and exit configurations, and multi-packet transfer scenarios.

The scenarios are created with several conditions not found in the collision case [25]. Collision is considered to occur if: I, there is the same destination status, and II, there is the next status A = status B and the next status B = status A. Meanwhile, in the FOCC system, collision can also occur on intersecting paths without associated cells, as in case I. In case II, TQL will change the direction of the packet, but it does not accommodate the difference in packet size. Furthermore, this study also addresses collision handling for transfers directed to the same outbound point, ensuring reliable multi-package coordination even under identical destination conditions.

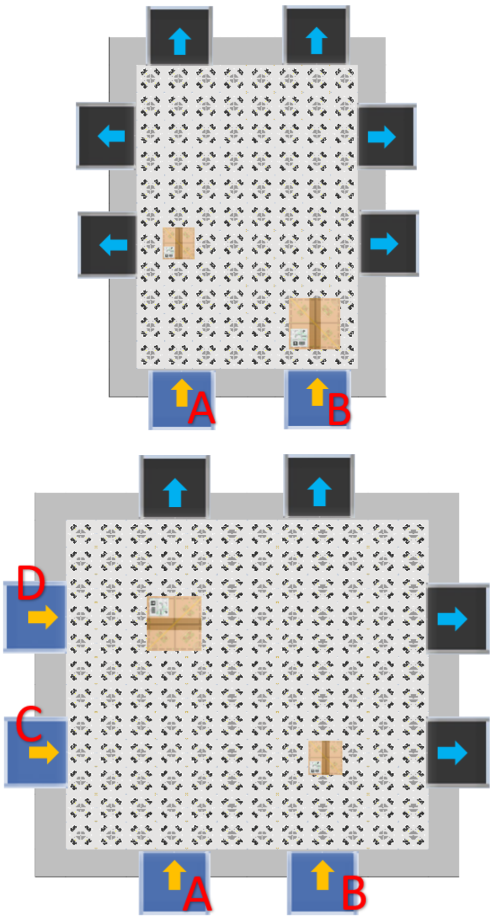

Fig. 6. FOCC configuration for testing

IV. Simulation and Analysis

The simulation utilized MATLAB software on a computer with a Core i5 processor operating at 2.4 GHz and 16GB of RAM. Several parameters are used in the path planning algorithm according to the data in Table I. For the RL method, five types of parameter pairs α, γ, and € are used for testing. In QL, TQL, and DoQL, each parameter is tested with four types of episodes to determine the effect of each parameter on RL training. QL and DoQL training is carried out with each agent alternately, while in TQL, all agents learn in the same cycle. In RRT, 2000 maxNodes are used with a step value = 1 and a goal threshold of 0.5.

Meanwhile, in the A* algorithm, the Euclidean distance calculation is used for the heuristic function. The performance of the path planning algorithm is analyzed based on the computation time (t_c), path smoothness (P_s), and the number of activated cells (a_c). t_c is calculated from the program's initialization until the path is generated. P_s is calculated based on the number of changes in direction on the generated path. The smaller the P_s value, the smoother the resulting path. a_c is the number of cells used on the path. The path planning algorithm uses the midpoint reference of the cell as a reference for movement. The fewer cells are activated, the smaller the energy consumption used to move the actuator.

TABLE I. PARAMETERS FOR THE PATH PLANNING ALGORITHM

Algorithm	Indeks	Parameters
QL, DoQL, TQL	I	α = 0.3, β0.7 =, € = 0.3, episode: a) 500, b) 1000, c) 2000, d) 3000
	II	α = 0.5, β0.7 =, € = 0.5, episode: a) 500, b) 1000, c) 2000, d) 3000
	III	α = 0.8, β0.7 =, € = 0.5, episode: a) 500, b) 1000, c) 2000, d) 3000
	IV	α = 0.5, β0.7 =, € = 0.8, episode: a) 500, b) 1000, c) 2000, d) 3000
	V	α = 0.8, β0.8 =, € = 0.8, episode: a) 500, b) 1000, c) 2000, d) 3000
RRT	maxNodes = 20000; step = 1; goalThreshold = 0.5;	
A*	Heuristic Function → Euclidean distance	

A. *Path Planning Result*

Based on the test results, all algorithms can reach the goal point. However, there are differences in the path patterns produced by each method. Fig. 7 to Fig. 10 is the path planning results for scenarios I, II, III, and IV. The inbound point is indicated in green, while the outbound point is indicated in blue. The path produced for package A is indicated in red. The path produced for package B is indicated in blue. The path produced for package C is indicated in magenta. The path produced for package D is indicated in orange. Furthermore, the performance of the path planning algorithm is analyzed based on three criteria, namely, a_c, P_s, and t_c values.

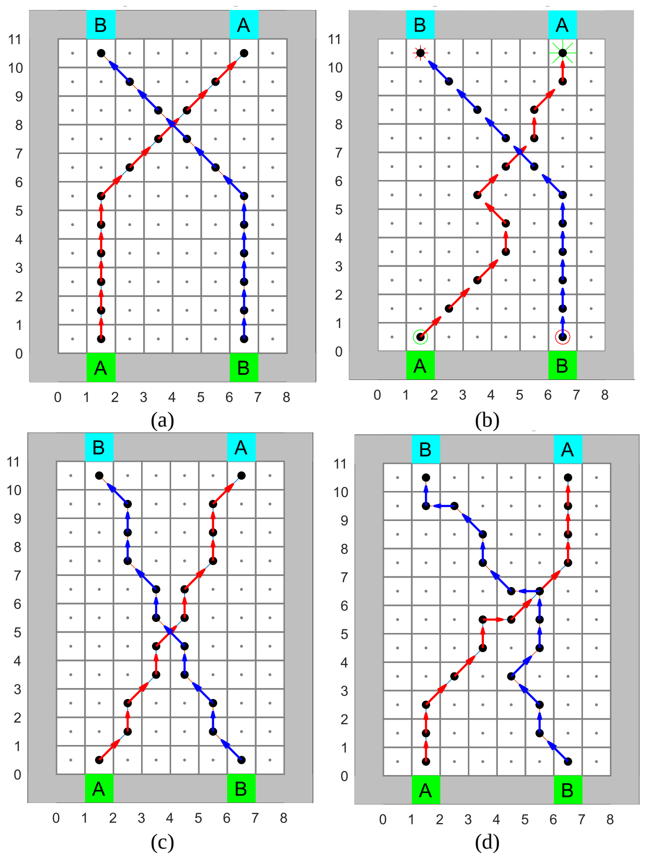

Fig. 7. Path planning results using (a) QL&DoQL-Vb, (b) TQL-Vd, (c) A*, and (d) RRT for scenario I

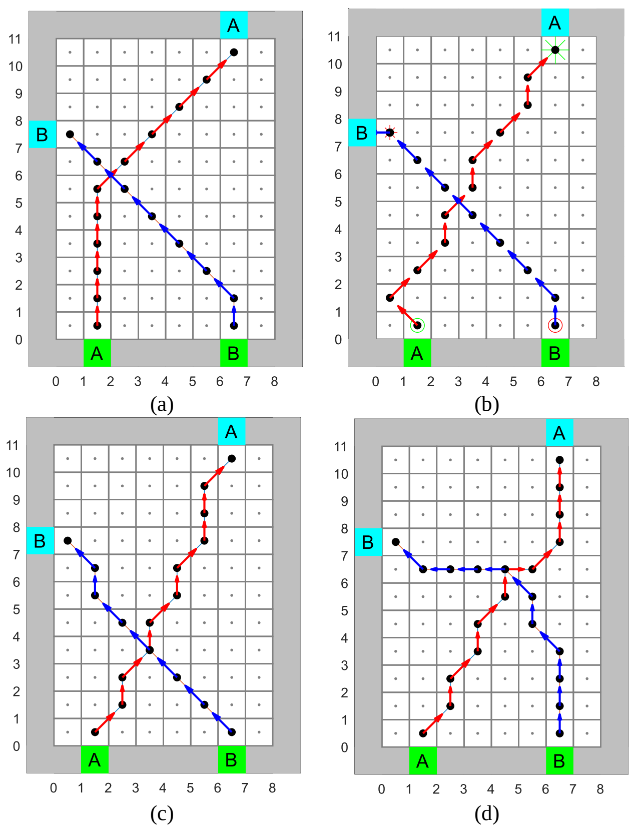

Fig. 8. Path planning results (a) QL&DoQL-Vb, (b) TQL-Vd, (c) A*, and (d) RRT for scenario II

1) *Number of activated cells*

In the scenario I, QL, DoQL, and A* produce paths with consistent and relatively the same a_c values, specifically 11 cells for packages A and B, as shown in Fig. 7(a) and Fig. 7(c). This indicates that learning-based algorithms converge to efficient paths with limited search space after training. Concurrently, the RRT algorithm exhibits the most suboptimal performance in path planning, as shown in Fig. 7(d). The random nature of RRT often activates more cells, especially when it does not directly obtain an optimal solution.

In the scenario II, QL, DoQL, and A* yield pathways with consistent and comparable a_c values, specifically 11 cells for package A and eight cells for package B, as illustrated in Fig. 8(a) and Fig. 8(c). Concurrently, TQL yields an increased a_c value for package B, specifically 11 cells, as illustrated in Fig. 8(b). The RRT algorithm again shows a higher a_c value, as shown in Fig. 8(d). In scenarios III & IV, RRT also shows a higher a_c value, as shown in Fig. 9 and Fig. 10. Learning QL & DoQL with low α, γ values and number of episodes can cause less than optimal a_c values in package B, as shown in the graphs in Fig. 11 and Fig. 15.

2) *Path Smoothness*

Although A* has relatively the same ac performance as DQL and DoQL, A* has a high value in path smoothness (P_s). In scenario I, A* produces a high P_s value for both packages A & B, which is 8x the change in direction as shown in Fig. 7(c). In scenario II, A* produces a smoother path for package B, as shown in Fig. 8(c). Because it uses deterministic calculations, A* will produce a path that always remains the same for each training session. Unlike the QL algorithm, DoQL and TQL show performance that tends to improve as the number of training episodes increases. Initially, in lower episodes, P_s for packages A and B varied (above 5), reflecting the presence of sharp maneuvers or non-smooth routes as shown in Fig. 12. However, in higher episodes (2000–3000), QL and DoQL produced the best P_s values as shown in Fig. 7(a).

The epsilon values α and γ also affect the smoothness of the path. Even with fewer episodes, the higher the epsilon value, the smoother the resulting path. With an epsilon value approaching 1, the opportunity for the exploration process occurs more often, so the Q-Table has an optimal value when the exploitation process is carried out. The α parameter will accelerate the process of updating the Q-table value during learning, so the resulting path tends to be smoother even with fewer episodes. Based on the simulations that have been carried out, a γ value below 0.7 causes failure in the learning process because a small γ value makes the agent prioritize immediate rewards and ignore long-term results. So that it fails to find the optimal path. A comparison of P_s' performance against various learning parameter variations is shown in Fig. 12 and Fig. 16.

In TQL, the resulting P_s value remains high even though the learning parameters (α, γ, and number of episodes) are significant, as shown in the graphs in Fig. 12 and Fig. 16. This may be due to the collision avoidance algorithm on the multi-agent system finding suboptimal paths during the learning phase. Meanwhile, RRT produces the highest and most inconsistent P_s values (up to 10 in all scenarios) due to its random nature and lack of emphasis on path smoothness. The resulting paths are often snaky with sharp changes in direction, as shown in Fig. 7(d), Fig. 8(d), Fig. 9(d), and Fig. 10(d).

3) *Computation Time*

Generally, A* and RRT exhibit minimal computation times, remaining below 0.05 seconds across all scenarios. In contrast, the computation time for RL is significantly higher, with a t_c value exceeding 0.1 seconds at 500 episodes. Unlike A* and RRT, the iterative learning process in RL causes a significant computational burden. The number of episodes also significantly affects the computation time. The greater the number of episodes, the greater the t_c value as shown in Fig. 13 and Fig. 17. The epsilon value also affects the computation time. Based on the test results, the greater the epsilon value, the higher the computation time.

TQL has the highest computation time compared to all algorithms, as shown in Fig. 14 and Fig. 18. In scenarios I and II, the total learning time t_c for packages A and B in QL and DoQL is still lower than TQL. For instance, in the V-d parameter, the t_c value for QL is 0.6883 seconds, three times lower than that observed for TQL at 1.675 seconds. The result indicates that the computation time of individual inter-agent learning remains lower than that of simultaneous multi-agent learning. Based on the overall test, DoQL gives the same path results as QL but with larger learning parameters. A* can be used for path planning in the FOCC system with reduced computational needs, but at the expense of path smoothness.

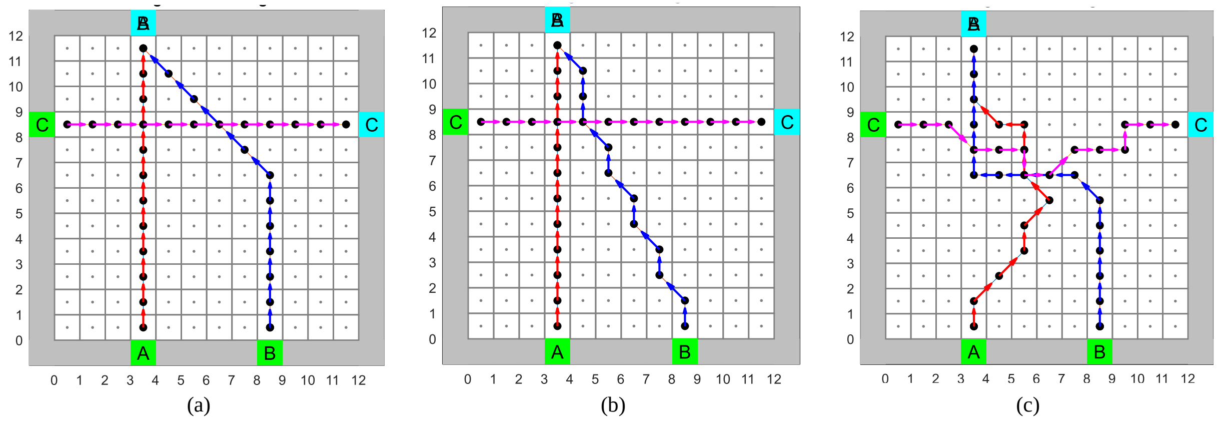

Fig. 9. Path planning results using (a) QL&DoQL-Vb, (b) A*, and (c) RRT for scenario III

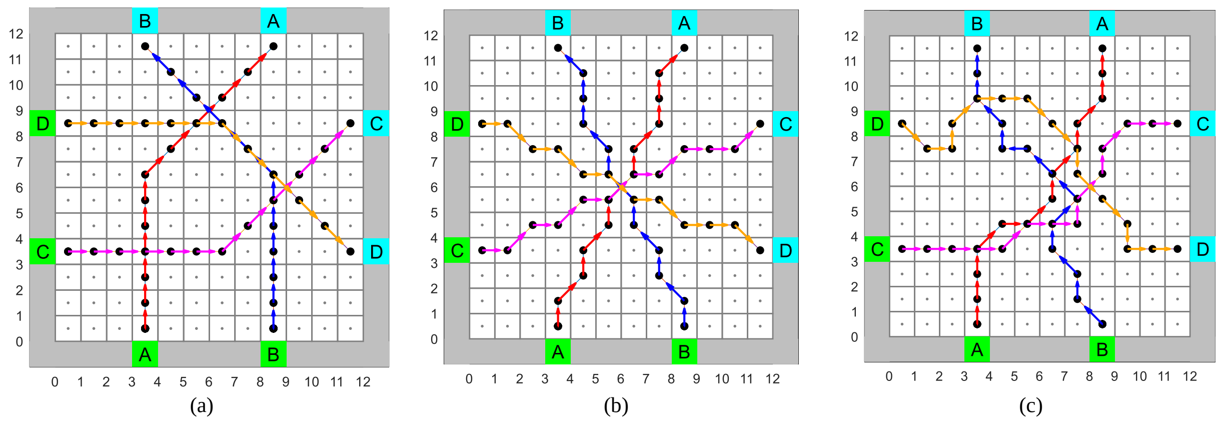

Fig. 10. Path planning results using (a) QL&DoQL-Vb, (b) A*, and (c) RRT for scenario IV

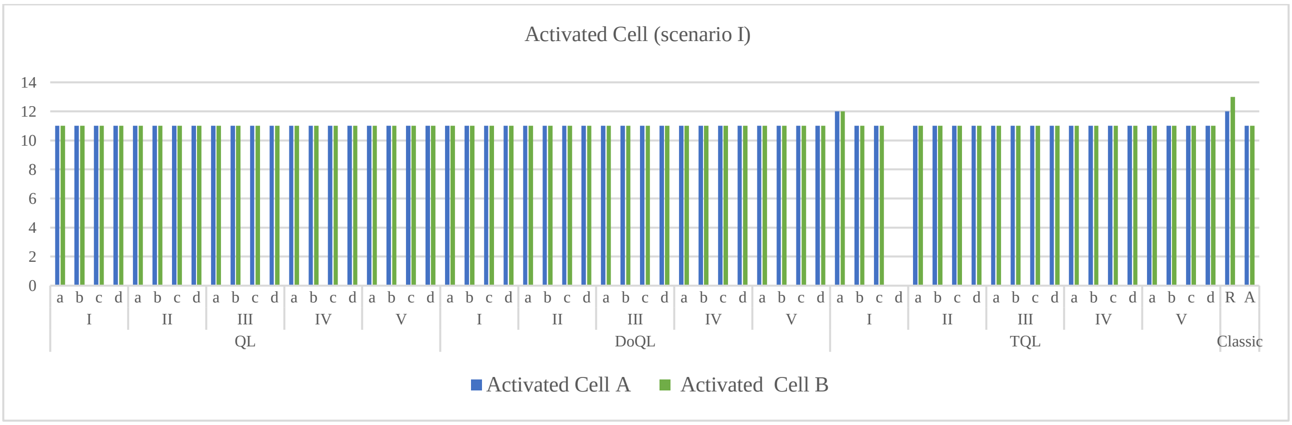

Fig. 11. Number of activated cell for packages A and B in scenario I

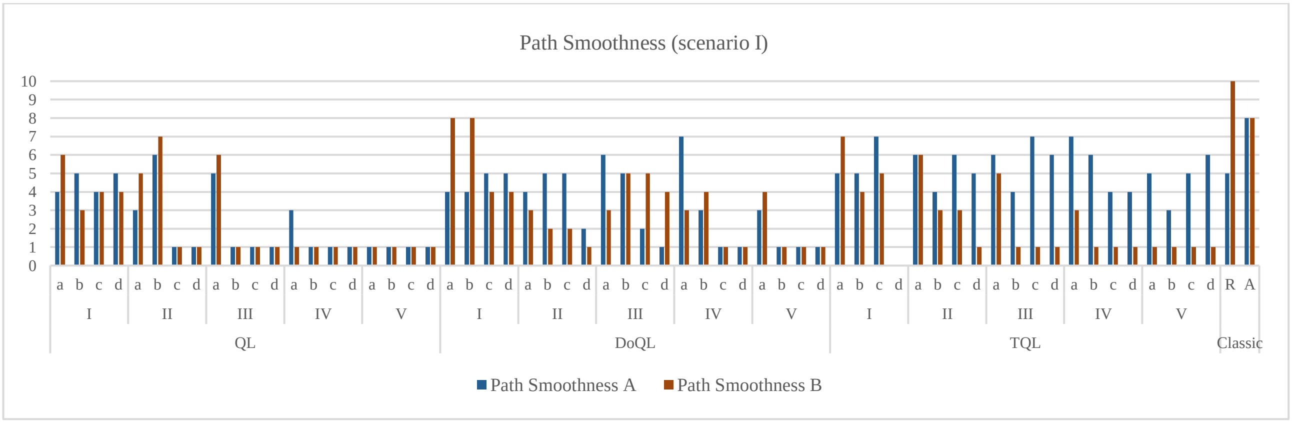

Fig. 12. Path smoothness values for packages A and B in scenario I

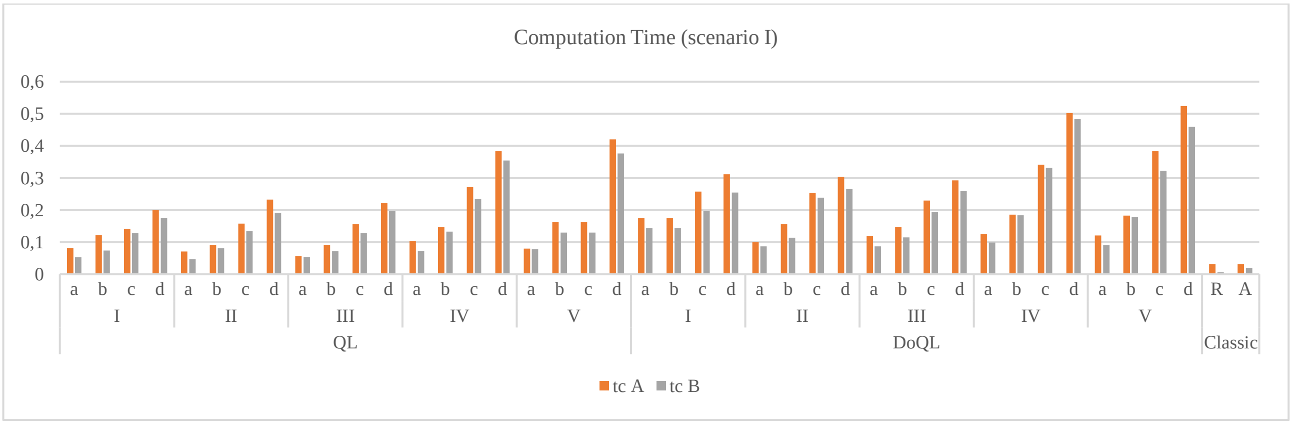

Fig. 13. Computation time for packages A and B in the scenario I

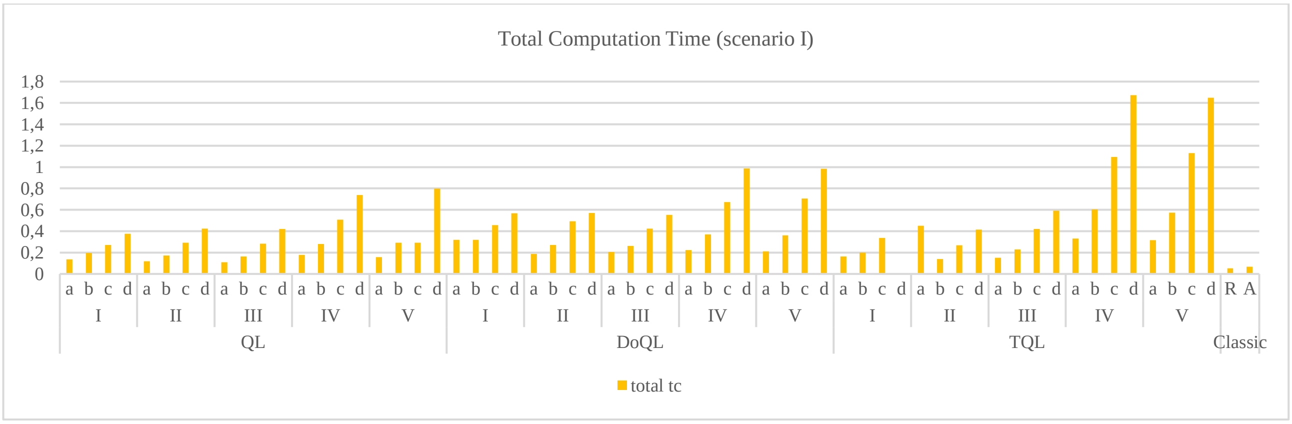

Fig. 14. Total computation time for packages A and B in scenario I

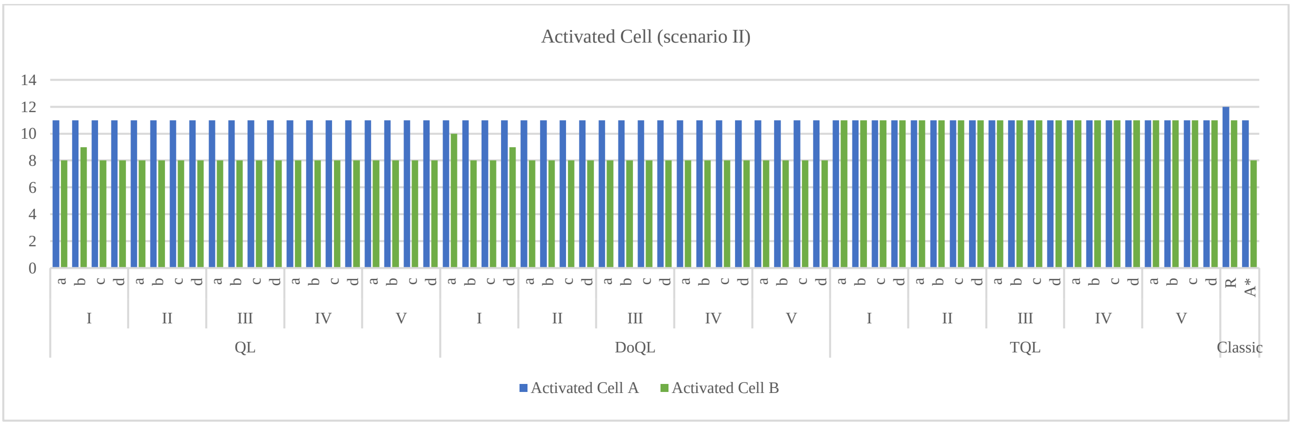

Fig. 15. Number of activated cell for packages A and B in scenario II

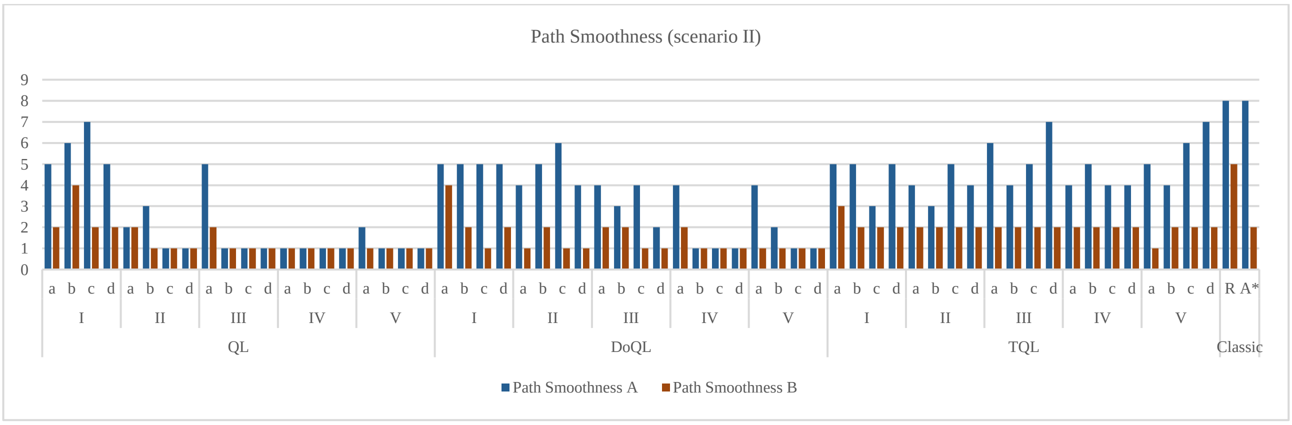

Fig. 16. Path smoothness values for packages A and B in scenario II

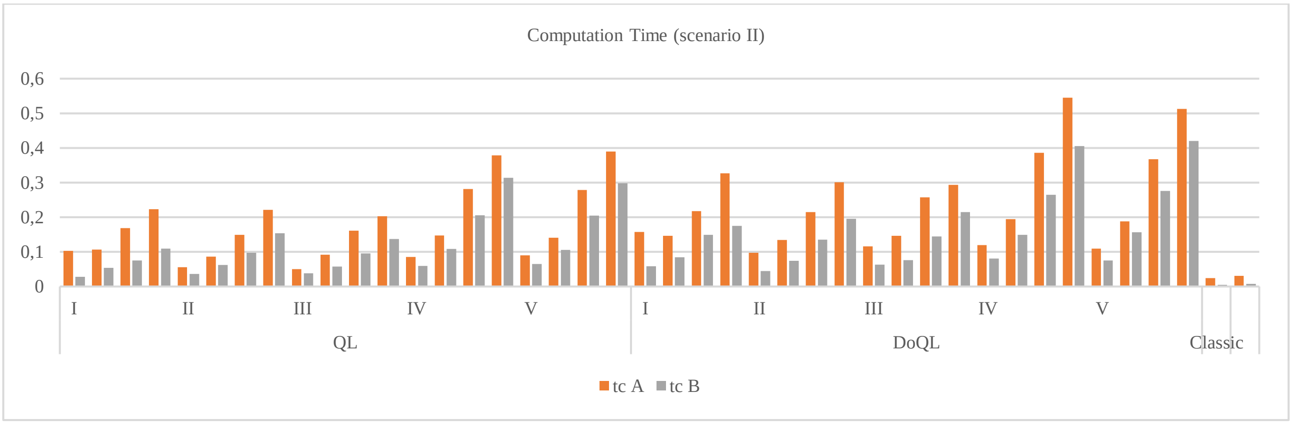

Fig. 17. Computation time for packages A and B in scenario II

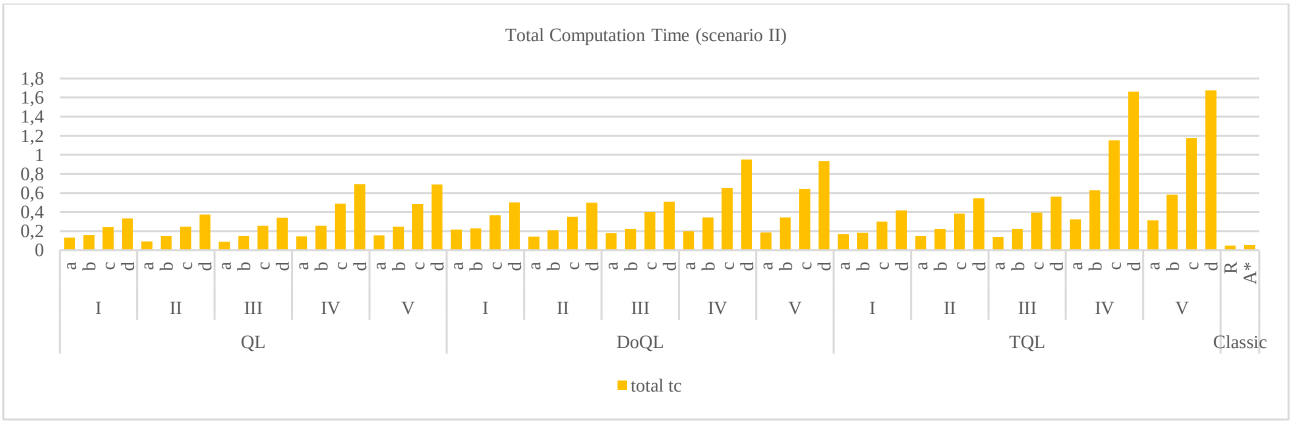

Fig. 18. Total computation time for packages A and B in scenario II

B. *Rule-Based Conflict Resolution (RCR) Test Result*

At this stage, collision avoidance scenario testing is carried out with the following modes: Sq, IP, Rr, and Hb. Testing is carried out on all scenarios I, II, III, and IV. When all paths have been generated, the algorithm will perform collision analysis. If there is more than one packet to be sent, the algorithm will calculate all packets' collision probability. Furthermore, the program will run the collision avoidance algorithm if a collision is detected. Based on the results of the performance analysis from the previous test, because DoQL gives the same path results as QL, and TQL is not suitable for use in this FOCC system, QL with V-b parameters is used for the RCR test, and A* and RRT are used as comparison methods. Collision avoidance performance is analyzed based on computation time (t_c), path effectiveness (P_s and a_c), and total delivery time (t_f). Various collision conditions are shown in Fig. 10 on the path generated by the QL method. Fig. 20(a) is a collision condition for scenario I. Package B will collide with package A at points 5,8. For scenario II, if the package size is the same (using size S), then no collision occurs. Meanwhile, if package B uses size II, a collision will be detected at point (4,5) as shown in Fig. 20(b).

In scenario III, three packets are used for testing. Packages A and B are size S, while package C is size M. Package B has two possible collisions, specifically at point (8,7) with package C and at point (5,11) with package A. Fig. 20(c) shows the first collision's position for scenario III. In scenario IV, there are four packages, where packages A and C use size M, and packages B and D use size S. In this scenario, all four packets have more than one possible collision. Package A can collide with package C at point (4,3), with package B at point (4,7). Package B has the possibility of colliding with package A at point (7,9) and package D at point (8,7) as shown in Fig. 20(d). Packages B and D have three cells that overlap during the delivery process at the same time. Package C has a potential collision with package A at point (2,4) and package D at point (8,6).

1) *Sq-Mode*

The sequential transfer technique does not require additional computing time because it utilizes existing paths. Fig. 21 is an example of delivery with Sq-mode in scenario I. Package A will move first, followed by package B after package A has finished arriving at the goal point. The sequential flow was successfully implemented for packet delivery in all scenarios.

2) *IP-Mode*

In IP mode, the path pattern does not change. The number of points at the beginning of the path increases so that the packet will remain still for some time. Fig. 22 is an example of sending with IP mode in scenario II. Packet A will move first, followed by packet B after packet A passes point (2,2), indicating 1x additional paths to avoid collisions in scenario II. Based on the experiment, the average time for the execution time of IP-mode is 5.5mS.

3) *Rr-Mode*

In reroute mode, there is a change in the path pattern due to the addition of obstacles with r = 1, as shown in Fig. 19. In scenario III, there is a change in the path of packet B due to the obstacle at point (8,7), compared to the initial route in Fig. 9 (a). All packets can move simultaneously in Rr mode since t = 1 from the entry point, as shown in Fig. 23. Because of re-planning the path, the execution time of the IP mode is the same as the time of 1x training on each algorithm.

4) *Hb-Mode*

The hybrid mode is tested in scenario IV. Because three cells intersect between packages B and D, using IP-mode will cause a delivery delay that is too long. Combining IP-mode and Rr-Mode for several transfer conditions can save the total transfer time. Fig. 24 is an example of the results of using Hb-mode. Package B experiences a route change compared to the initial route. Packages A and C get delayed, while packages B and D move simultaneously. Package A only moves after packages B and D have passed four cells. Finally, package C moves after package A has passed five cells. It is different from IP-mode, which causes all packages to experience delays. In the case of a collision between C&A packets, rerouting is impossible because the collision point is too close to the inbound point, and the packet size is large.

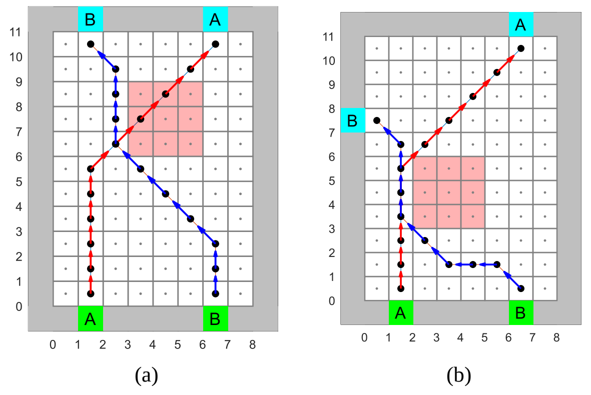

Fig. 19. QL reroute obstacles (a) scenario I, (b) scenario II

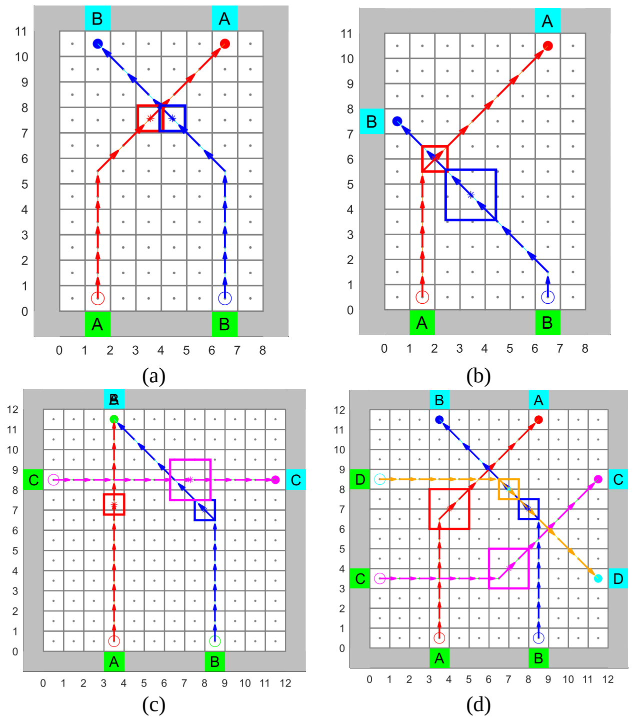

Fig. 20. Crash point in scenario (a) I, (b) II, (c) III, (d) IV

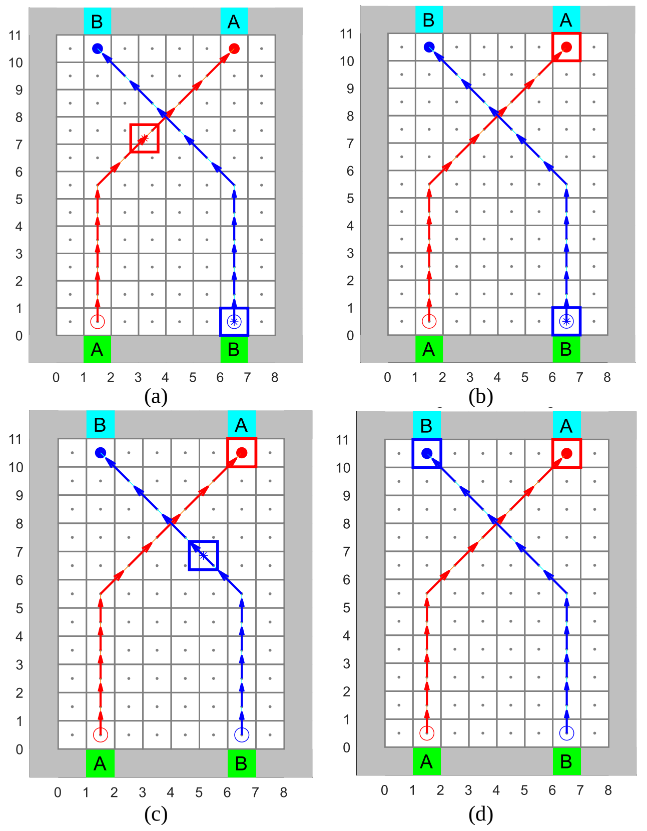

Fig. 21. Packages transfer simulation using QL Sq-mode at *t:* (a) 3.45s, (b) 5.8s, (c) 9s, (d) 11.6s

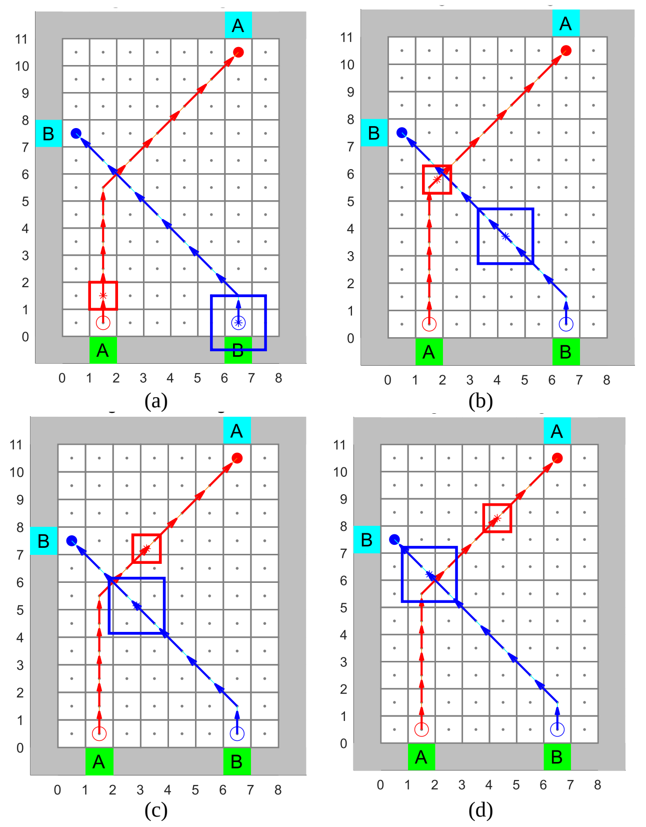

Fig. 22. Packages transfer simulation using QL IP-mode at *t*: (a) 0.5s, (b) 2.5s, (c) 3.5s, (d) 4.25s

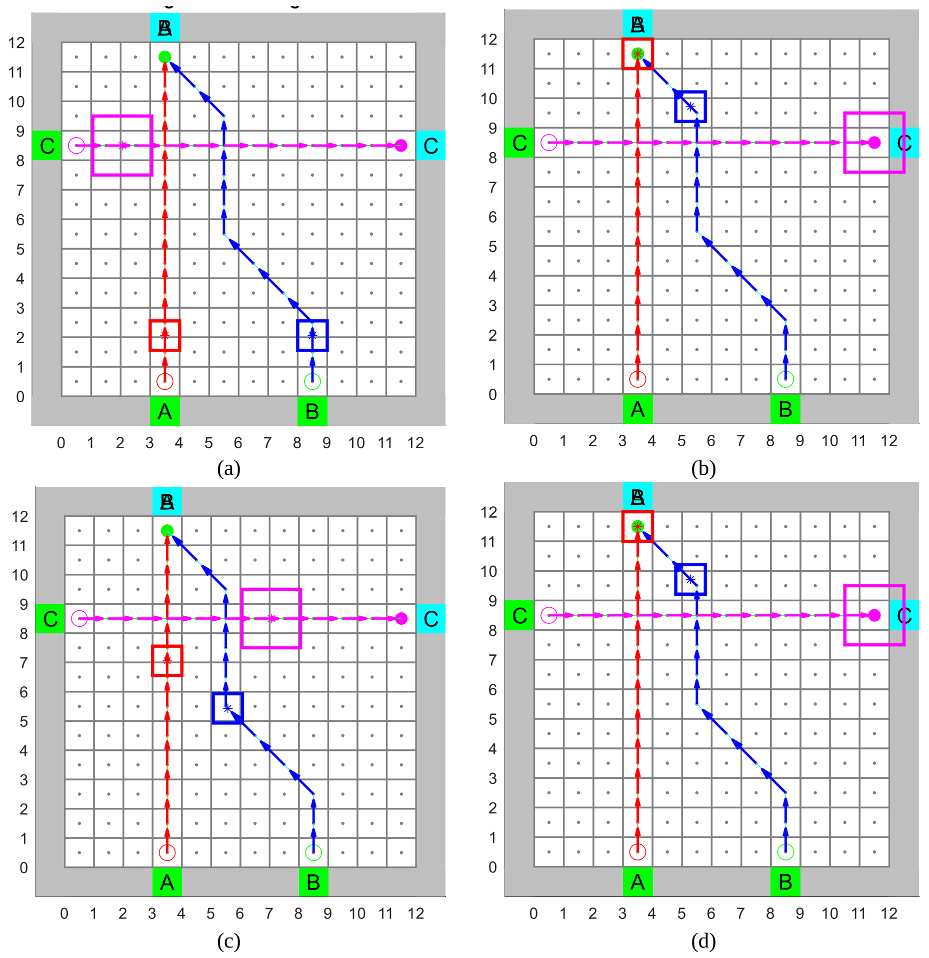

Fig. 23. Package transfer simulation using QL Rr-mode at *t* (a) 0.75s, (b) 1.5s, (c) 3s, (d) 5s

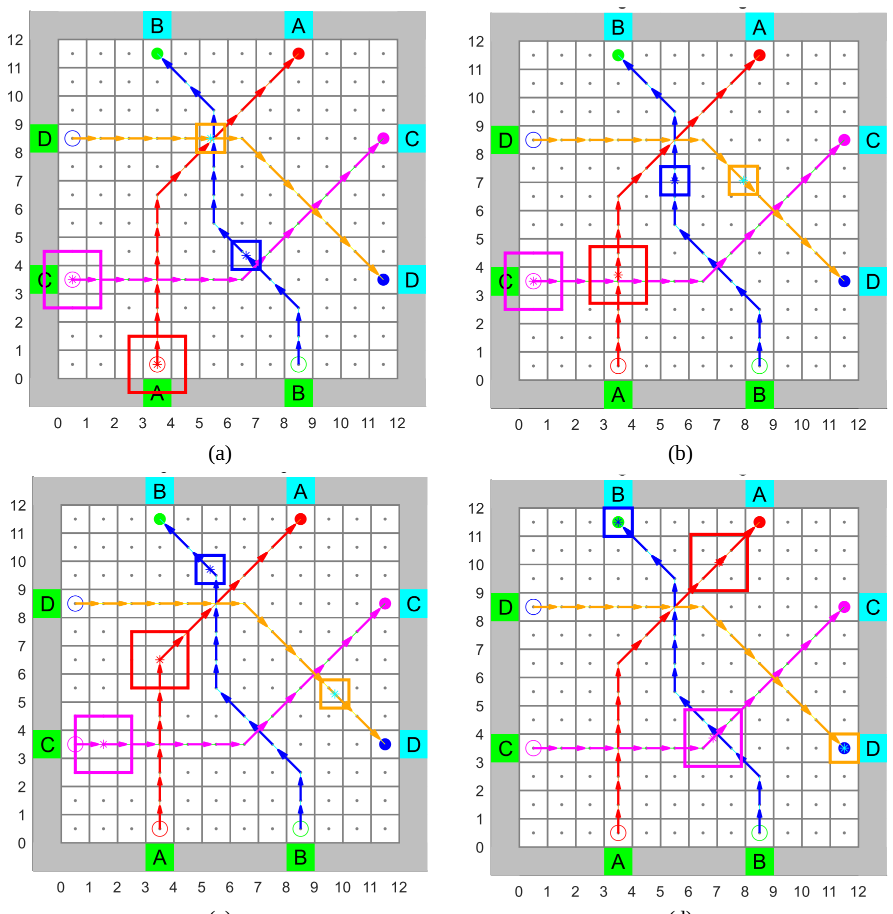
Fig. 24. Package transfer simulation using QL Hb-mode at *t*: (a) 2.2s, (b) 3.7s, (c) 4.95s, (d) 7.45s

C. *Multi-Package Transfer Analysis*

The package transfer rate during the transfer process is assumed to be constant, namely $v = 2.l$ / second. Based on the simulation results, the QL algorithm achieves the fastest transfer time in scenario I in Rr-mode with a time of 5.8 seconds. In terms of path smoothness, A* provides the same value as QL for path changes in packet B, as shown in Fig. 26. However, there are two additional activated cells in the A* algorithm, as shown in Fig. 25. To avoid collisions in Rr-mode, a value of $r = 2$ is required in the A* algorithm. This has implications for increasing the path length to reach the outbound point. RRT shows the lowest performance for the three modes tested. The path generated by RRT has high a_c and P_s values, as shown in Fig. 25 and Fig. 26. Paths with many changes in direction can increase the risk of delivery errors in real implementations. The number of cells used also affects the path length so that the transfer time increases in all modes, as shown in Table II.

In scenario II, QL also performs well in Rr and IP-mode, with the best package transfer time of 5.8 seconds for both modes. Based on the experiment, the path generated in Rr-mode with $r = 1$ has the same total delivery time as IP-mode with 1x additional path, as shown in Table II. In the case of ac, QL with Rr-mode adds one activated cell compared to the original path, as shown in Fig. 25. In terms of path smoothness, even though there is an increase in the P_s value compared to the original path, QL still shows the best performance, as shown in Fig. 26. In this case, A* in Rr mode fails to reach its destination. The observation results indicate that a collision point near the inbound point will terminate the search path, preventing the algorithm from achieving the final result. RRT again shows the lowest performance in this scenario for the three modes tested, as shown in Fig. 25 and Fig. 26.

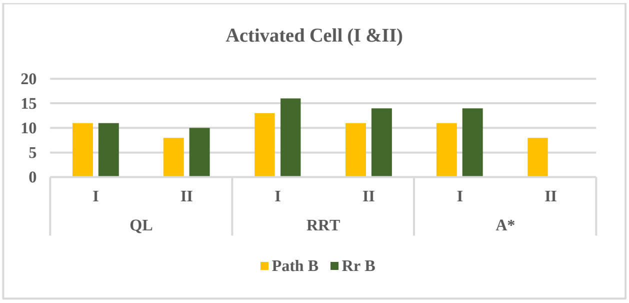

Fig. 25. Activated cell data for QL, RRT, A* algorithms in scenario I&II

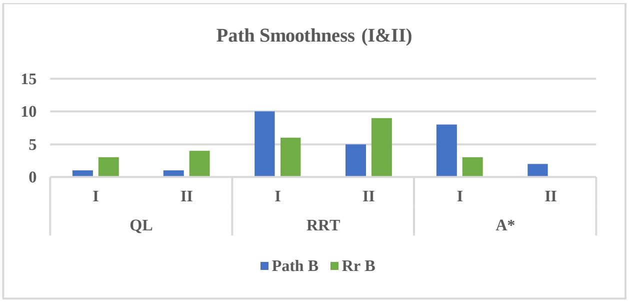

Fig. 26. Path smoothness data for QL, RRT, A* algorithms in scenario I&II

TABLE II. COMPARISON OF TRANSFER TIMES BASED ON PATH PLANNING ALGORITHMS

Algorithm	Mode	Packages Transfer Time (s)			
		I	*II*	*III*	*IV*
QL	Seq	11.6	10.5	16.25	25
	IP	6.7	5.8	7.6	12.1
	Rr/Hb	5.8	5.8	6.25	10.75
RRT	Seq	12.7	11.3	20.15	27.9
	IP	8.05	7.1	12.25	11.45
	Rr/Hb	7.3	6.4	-	11.45
A*	Seq	11.6	10.5	16.25	25
	IP	6.7	6.7	6.25	12.1
	Rr/Hb	6.4	-	6.25	10.3

In scenario III, QL in Rr-mode has the same t_f performance as A*, which is 6.25 seconds. In Rr-mode, QL uses one additional activated cell, as shown in Fig. 27. Meanwhile, since A* does not detect collisions in scenario III, the generated path remains the same as in Sq-mode. Although A* provides more stable a_c values, QL is far superior in terms of P_s, as shown in Fig. 28. RRT again has the weakest performance in terms of delivery time, path smoothness, and number of activated cells, as shown in Fig. 27 and Fig. 28. For package C (which has inbound and outbound points aligned with the X axis) and package A (which has inbound and outbound points aligned with the Y axis), the QL and A* algorithms generate straight paths (P_s = 0) as shown in Fig. 28.

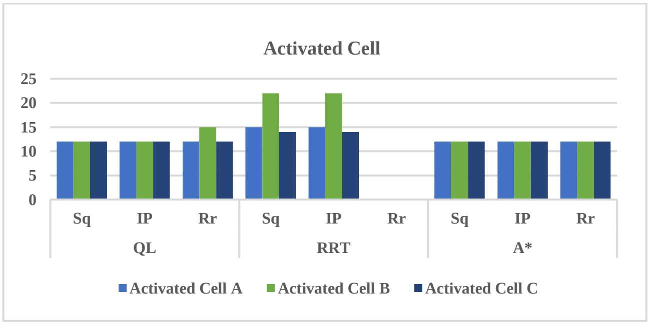

Fig. 27. Activated cell data for Sq, IP and Rr-mode in scenario III

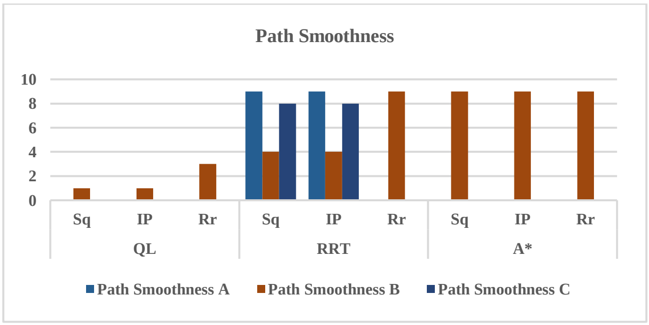

Fig. 28. Path smoothness data for Sq, IP and Rr-mode in scenario III

In simulation IV, Hb-mode replaces Rr-mode because reroute mode cannot handle high-density collisions. Using Hb-mode shortens the delivery time from 6.7 seconds (in IP mode) to 5.8 seconds. The path between packets B and D has three overlapping cell points. This causes the algorithm to add more than 3x insert paths, which has an impact on increasing the total delivery time, as shown in Table II. The movement time of packet B can be accelerated by changing the route. However, it affects increasing the P_s and a_c value, as shown in Fig. 29 and Fig. 30. In addition to Hb-mode, QL records a P_s value = 1, far superior to the A* and RRT algorithms, as shown in Fig. 30. From the overall simulation, it can be concluded that the RRT algorithm is less suitable for the case of path planning in FOCC. A* can be an alternative but with the consequence of a sneaky path. Q-learning is the optimal approach for path planning in the FOCC scenario, noting that the α, γ, € and episode parameters must be considered to minimize computational time.

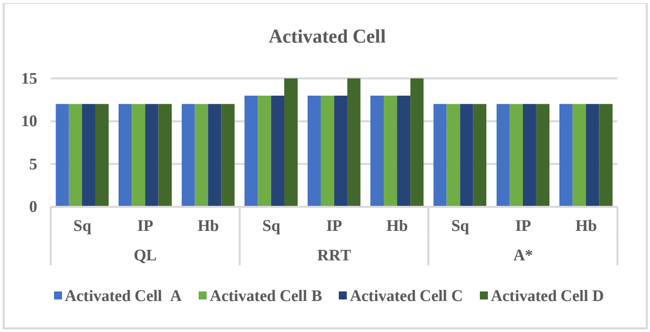

Fig. 29. Activated cell data for Sq, IP and Hb-mode in scenario IV

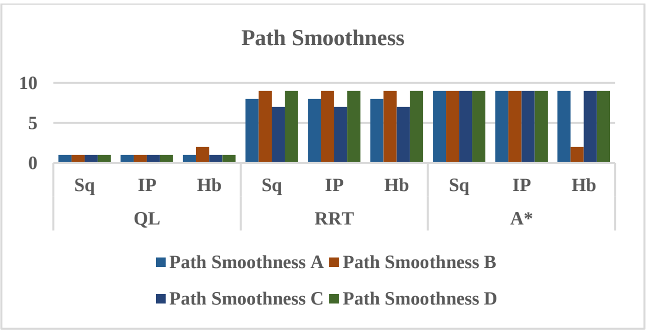

Fig. 30. Path smoothness data for Sq, IP and Hb-mode in scenario IV

Based on the applied RCR framework, the sequential transfer technique has the longest delivery time, as each packet is transmitted individually. Although this method is the safest way to send many packets and does not require additional computation for path modification, a delivery time that is too long can cause packets to pile up on the inbound path, which has the potential to cause deadlock. IP mode can also be applied to all scenarios to shorten the total transfer time. As seen in Table II, IP mode can save the total transfer time up to 2x compared to sequential.

Based on testing results in scenarios I–III, Rr-mode provides an option to increase the efficiency of t_f compared to the insert path technique, especially if the added delay is too much and extends the package waiting time. However, the Rr-mode does not work effectively if the collision point is too close to the entry or exit point. In addition, Rr-mode can also produce non-convergent training results when applied to a large number of packets because the added obstacle positions are too wide. Rr-mode also has a higher computation time because it requires path recalculation. Therefore, for sending many packets, such as in scenario IV, combining IP and Rr-modes (Hybrid mode) can perform better than implementing each mode separately. When the Hb-mode algorithm is applied to Scenarios I–III, it produces the same output as the IP-mode. This is because, in these scenarios, collisions do not require more than two iterations.

Integrating Q-Learning with the proposed RCR framework may provide a solution for simultaneous package transfer in the OCC system.

V. Conclusion

This study introduces the Q-RCR framework, a modular and scalable solution for collision-free multi-package transfer on Four-Wheeled Omnidirectional Cellular Conveyor (FOCC) systems. The primary theoretical contribution lies in decoupling the path learning and collision resolution processes by integrating Q-learning with rule-based conflict management. This architecture reduces computational overhead, accelerates convergence, and enhances adaptability in dynamic and high-variability environments. Experimental results demonstrate that Q-RCR outperforms baseline methods, such as Double Q-Learning, Tabular Q-Learning, RRT, and A*, regarding path smoothness, activated cell count, and total delivery time. The hybrid conflict resolution mode (Hb-mode) performs best under high-density collision scenarios, effectively balancing efficiency and computational cost. These findings affirm Q-RCR's potential to improve operational flow and mitigate deadlocks in modern logistics systems.

However, the current study has several limitations. The reroute (Rr) mode becomes less effective when collisions occur near entry or exit points due to limited space for path regeneration. The system also shows sensitivity to obstacle density, grid size, and the number of simultaneous packets, which may affect scalability and generalization. While the Insert Path (IP) mode applies to all scenarios, its repeated delays can significantly increase total delivery time in densely populated environments. Additionally, the absence of speed control mechanisms limits the system's responsiveness in dynamic or unpredictable conditions. To address these issues, future research should explore adaptive inter-package speed regulation as a more fluid alternative to delay-based or reroute strategies, potentially enhancing overall transfer performance. Other promising directions include learning-based dynamic rerouting, agent prioritization schemes, and real-world validation via integration with physical control systems and IoT-enabled innovative logistics platforms.

Acknowledgment

This research was funded by the Indonesian Education Scholarship (Beasiswa Pendidikan Indonesia/BPI) managed by the Center for Higher Education Funding and Assessment (Pusat Pembiayaan dan Asesmen Pendidikan Tinggi/PPAPT) under the Ministry of Higher Education, Science, and Technology (Kemendiktisaintek), and the Indonesian Endowment Fund for Education (Lembaga Pengelola Dana Pendidikan/LPDP).

References

[1] A. W. Youssef, N. M. Elhusseiny, O. M. Shehata, L. A. Shihata, and E. Azab, "Kinematic modeling and control of omnidirectional wheeled cellular conveyor," *Mechatronics*, vol. 87, Nov. 2022, doi: 10.1016/j.mechatronics.2022.102896.

[2] Z. Zhang, T. Sun, Z. Wang, and X. Zhang, "Trajectory Planning and Performance Atlases of a New Omnidirectional Conveyor," *Actuators*, vol. 13, no. 11, p. 441, Nov. 2024, doi: 10.3390/act13110441.

[3] Z. Lin, K. Wu, R. Shen, X. Yu, and S. Huang, "An Efficient and Accurate A-Star Algorithm for Autonomous Vehicle Path Planning," *IEEE Trans Veh Technol*, vol. 73, no. 6, pp. 9003–9008, Jun. 2024, doi: 10.1109/TVT.2023.3348140.

[4] B. Wang, L. Zhang, and J. Kim, "Fault Detection and Diagnosis of Three-Wheeled Omnidirectional Mobile Robot Based on Power Consumption Modeling," *Mathematics*, vol. 12, no. 11, p. 1731, Jun. 2024, doi: 10.3390/math12111731.

[5] M. Eyuboglu and G. Atali, "A novel collaborative path planning algorithm for 3-wheel omnidirectional Autonomous Mobile Robot," *Rob Auton Syst*, vol. 169, p. 104527, Nov. 2023, doi: 10.1016/j.robot.2023.104527.

[6] E. Rubies, R. Bitriá, and J. Palacín, "A Parcel Transportation and Delivery Mechanism for an Indoor Omnidirectional Robot," *Applied Sciences*, vol. 14, no. 17, p. 7987, Sep. 2024, doi: 10.3390/app14177987.

[7] M. Ramírez-Neria, R. Madonski, E. G. Hernández-Martínez, N. Lozada-Castillo, G. Fernández-Anaya, and A. Luviano-Juárez, "Robust trajectory tracking for omnidirectional robots by means of anti-peaking linear active disturbance rejection," *Rob Auton Syst*, vol. 183, p. 104842, Jan. 2025, doi: 10.1016/j.robot.2024.104842.

[8] L. Wen, B. Liang, B. Li, and L. Zhang, "Power Balance Control Strategy of Permanent Magnet Synchronous Motor of Belt Conveyor," *IEEE Access*, vol. 10, pp. 117045–117052, 2022, doi: 10.1109/ACCESS.2022.3219088.

[9] P. Zhou *et al.*, "A New Embedded Condition Monitoring Node for the Idler Roller of Belt Conveyor," *IEEE Sens J*, vol. 24, no. 7, pp. 10335–10346, Apr. 2024, doi: 10.1109/JSEN.2024.3363905.

[10] D. Miao, Y. Wang, L. Yang, and S. Wei, "Coal Flow Detection of Belt Conveyor Based on the Two-Dimensional Laser," *IEEE Access*, vol. 11, pp. 82294–82301, 2023, doi: 10.1109/ACCESS.2023.3301768.

[11] S. Gaiardelli, D. Carra, S. Spellini, and F. Fummi, "Dynamic Job and Conveyor-Based Transport Joint Scheduling in Flexible Manufacturing Systems," *Applied Sciences*, vol. 14, no. 7, p. 3026, Apr. 2024, doi: 10.3390/app14073026.

[12] M. Sperling, T. Kurschilgen, and P. Schumacher, "Concept of a Peripheral-Free Electrified Monorail System (PEMS) for Flexible Material Handling in Intralogistics," *Inventions*, vol. 9, no. 3, p. 52, Apr. 2024, doi: 10.3390/inventions9030052.

[13] K.-J. Wang and T.-L. Lee, "Designing a digital-twin based dashboard system for a flexible assembly line," *Comput Ind Eng*, vol. 196, p. 110491, Oct. 2024, doi: 10.1016/j.cie.2024.110491.

[14] K. Li, T. Liu, P. N. Ram Kumar, and X. Han, "A reinforcement learning-based hyper-heuristic for AGV task assignment and route planning in parts-to-picker warehouses," *Transp Res E Logist Transp Rev*, vol. 185, p. 103518, May 2024, doi: 10.1016/j.tre.2024.103518.

[15] M. Jeon, J. Lee, and S.-K. Ko, "Modular Reinforcement Learning for Playing the Game of Tron," *IEEE Access*, vol. 10, pp. 63394–63402, 2022, doi: 10.1109/ACCESS.2022.3175299.

[16] X. Li, Z. Lv, S. Wang, Z. Wei, and L. Wu, "A Reinforcement Learning Model Based on Temporal Difference Algorithm," *IEEE Access*, vol. 7, pp. 121922–121930, 2019, doi: 10.1109/ACCESS.2019.2938240.

[17] B. An, S. Sun, and R. Wang, "Deep Reinforcement Learning for Quantitative Trading: Challenges and Opportunities," *IEEE Intell Syst*, vol. 37, no. 2, pp. 23–26, 2022, doi: 10.1109/MIS.2022.3165994.

[18] U. Orozco-Rosas, K. Picos, J. J. Pantrigo, A. S. Montemayor, and A. Cuesta-Infante, "Mobile Robot Path Planning Using a QAPF Learning Algorithm for Known and Unknown Environments," *IEEE Access*, vol. 10, pp. 84648–84663, 2022, doi: 10.1109/ACCESS.2022.3197628.

[19] L. Chen *et al.*, "Transformer-Based Imitative Reinforcement Learning for Multirobot Path Planning," *IEEE Trans Industr Inform*, vol. 19, no. 10, pp. 10233–10243, Oct. 2023, doi: 10.1109/TII.2023.3240585.

[20] Z. Bai, H. Pang, Z. He, B. Zhao, and T. Wang, "Path Planning of Autonomous Mobile Robot in Comprehensive Unknown Environment Using Deep Reinforcement Learning," *IEEE Internet Things J*, vol. 11, no. 12, pp. 22153–22166, Jun. 2024, doi: 10.1109/JIOT.2024.3379361.

[21] A. A. N. Kumaar and S. Kochuvila, "Mobile Service Robot Path Planning Using Deep Reinforcement Learning," *IEEE Access*, vol. 11, pp. 100083–100096, 2023, doi: 10.1109/ACCESS.2023.3311519.

[22] W. Liu, L. Dong, J. Liu, and C. Sun, "Knowledge transfer in multi-agent reinforcement learning with incremental number of agents,"

Journal of Systems Engineering and Electronics, vol. 33, no. 2, pp. 447–460, Apr. 2022, doi: 10.23919/JSEE.2022.000045.

[23] I. Han, S. Oh, H. Jung, I. Chung, and K. J. Kim, "Monte Carlo and Temporal Difference Methods in Reinforcement Learning [AI-eXplained]," *IEEE Comput Intell Mag*, vol. 18, no. 4, pp. 64–65, Nov. 2023, doi: 10.1109/MCI.2023.3304145.

[24] S. Kautsar, A. S. Aisjah, M. Syai'in, K. Indriawati, and T. R. Biyanto, "Path Planning for 4-Wheeled Omnidirectional Cellular Conveyor using Q-Learning Algorithm," in *2024 International Electronics Symposium (IES)*, pp. 466–472, 2024, doi: 10.1109/IES63037.2024.10665817.

[25] W. Zaher, A. W. Youssef, L. A. Shihata, E. Azab, and M. Mashaly, "Omnidirectional-Wheel Conveyor Path Planning and Sorting Using Reinforcement Learning Algorithms," *IEEE Access*, vol. 10, pp. 27945–27959, 2022, doi: 10.1109/ACCESS.2022.3156924.

[26] M. Q. Zaman and H. M. Wu, "Intelligent Motion Control Design for an Omnidirectional Conveyor System," *IEEE Access*, vol. 11, pp. 47351–47361, 2023, doi: 10.1109/ACCESS.2023.3275962.

[27] R. V. Subrahmanyam, M. Kiran Kumar, S. Nair, S. Warrier, and B. N. Prashanth, "Design and development of automatic warehouse sorting rover," *Mater Today Proc*, vol. 46, pp. 4497–4503, 2021, doi: 10.1016/j.matpr.2020.09.688.

[28] S. G. Ozden, A. E. Smith, and K. R. Gue, "A computational software system to design order picking warehouses," *Comput Oper Res*, vol. 132, p. 105311, Aug. 2021, doi: 10.1016/j.cor.2021.105311.

[29] V. Simic, S. Dabic-Miletic, E. B. Tirkolaee, Ž. Stević, A. Ala, and A. Amirteimoori, "Neutrosophic LOPCOW-ARAS model for prioritizing industry 4.0-based material handling technologies in smart and sustainable warehouse management systems," *Appl Soft Comput*, vol. 143, p. 110400, Aug. 2023, doi: 10.1016/j.asoc.2023.110400.

[30] Z. Qiu, J. Long, Y. Yu, and S. Chen, "Integrated task assignment and path planning for multi-type robots in an intelligent warehouse system," *Transp Res E Logist Transp Rev*, vol. 194, p. 103883, Feb. 2025, doi: 10.1016/j.tre.2024.103883.

[31] V. Gupta, R. Mitra, F. Koenig, M. Kumar, and M. K. Tiwari, "Predictive maintenance of baggage handling conveyors using IoT," *Comput Ind Eng*, vol. 177, p. 109033, Mar. 2023, doi: 10.1016/j.cie.2023.109033.

[32] Z. Seibold, K. Furmans, and K. R. Gue, "Using Logical Time to Ensure Liveness in Material Handling Systems with Decentralized Control," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 1, pp. 545–552, Jan. 2022, doi: 10.1109/TASE.2020.3029199.

[33] S. Li, L. Fan, and S. Jia, "A hierarchical solution framework for dynamic and conflict-free AGV scheduling in an automated container terminal," *Transp Res Part C Emerg Technol*, vol. 165, p. 104724, Aug. 2024, doi: 10.1016/j.trc.2024.104724.

[34] S. Zhou, Q. Liao, C. Xiong, J. Chen, and S. Li, "A novel metaheuristic approach for AGVs resilient scheduling problem with battery constraints in automated container terminal," *J Sea Res*, vol. 202, p. 102536, Dec. 2024, doi: 10.1016/j.seares.2024.102536.

[35] X. Fan, H. Sang, M. Tian, Y. Yu, and S. Chen, "Integrated scheduling problem of multi-load AGVs and parallel machines considering the recovery process," *Swarm Evol Comput*, vol. 94, p. 101861, Apr. 2025, doi: 10.1016/j.swevo.2025.101861.

[36] X. Yang, H. Hu, and C. Cheng, "Collaborative scheduling of handling equipment in automated container terminals with limited AGV-mates considering energy consumption," *Advanced Engineering Informatics*, vol. 65, p. 103133, May 2025, doi: 10.1016/j.aei.2025.103133.

[37] Z. Ali *et al.*, "Design and development of a low-cost 5-DOF robotic arm for lightweight material handling and sorting applications: A case study for small manufacturing industries of Pakistan," *Results in Engineering*, vol. 19, p. 101315, Sep. 2023, doi: 10.1016/j.rineng.2023.101315.

[38] J. Palacín, E. Rubies, R. Bitrià, and E. Clotet, "Non-Parametric Calibration of the Inverse Kinematic Matrix of a Three-Wheeled Omnidirectional Mobile Robot Based on Genetic Algorithms," *Applied Sciences*, vol. 13, no. 2, p. 1053, Jan. 2023, doi: 10.3390/app13021053.

[39] C.-H. Hsu and J.-G. Juang, "Using a Robot for Indoor Navigation and Door Opening Control Based on Image Processing," *Actuators*, vol. 13, no. 2, p. 78, Feb. 2024, doi: 10.3390/act13020078.

[40] A. G. Rojas-López, M. G. Villarreal-Cervantes, A. Rodríguez-Molina, and J. A. Paredes-Ballesteros, "Efficient Online Controller Tuning for Omnidirectional Mobile Robots Using a Multivariate-Multitarget Polynomial Prediction Model and Evolutionary Optimization," *Biomimetics*, vol. 10, no. 2, p. 114, Feb. 2025, doi: 10.3390/biomimetics10020114.

[41] J. G. Pérez-Juárez *et al.*, "Kinematic Fuzzy Logic-Based Controller for Trajectory Tracking of Wheeled Mobile Robots in Virtual Environments," *Symmetry (Basel)*, vol. 17, no. 2, p. 301, Feb. 2025, doi: 10.3390/sym17020301.

[42] W. Chen *et al.*, "A survey of autonomous robots and multi-robot navigation: Perception, planning and collaboration," *Biomimetic Intelligence and Robotics*, vol. 5, no. 2, p. 100203, Jun. 2025, doi: 10.1016/j.birob.2024.100203.

[43] Z. Lu *et al.*, "A reinforcement learning-based optimization method for task allocation of agricultural multi-robots clusters," *Computers and Electrical Engineering*, vol. 120, p. 109752, Dec. 2024, doi: 10.1016/j.compeleceng.2024.109752.

[44] X. Li, J. Ren, and Y. Li, "Multi-mode filter target tracking method for mobile robot using multi-agent reinforcement learning," *Eng Appl Artif Intell*, vol. 127, p. 107398, Jan. 2024, doi: 10.1016/j.engappai.2023.107398.

[45] L. Zhang, Z. Cai, Y. Yan, C. Yang, and Y. Hu, "Multi-agent policy learning-based path planning for autonomous mobile robots," *Eng Appl Artif Intell*, vol. 129, p. 107631, Mar. 2024, doi: 10.1016/j.engappai.2023.107631.

[46] Z. Tang, F. Fu, G. Lu, and D. Chen, "Reinforcement Learning for Autonomous Agents: Scene-Specific Dynamic Obstacle Avoidance and Target Pursuit in Unknown Environments," *IEEE Access*, vol. 12, pp. 145496–145510, 2024, doi: 10.1109/ACCESS.2024.3463732.

[47] X. Zhang, H. Zong, and W. Wu, "Cooperative Obstacle Avoidance of Unmanned System Swarm via Reinforcement Learning Under Unknown Environments," *IEEE Trans Instrum Meas*, vol. 74, pp. 1–15, 2025, doi: 10.1109/TIM.2024.3522370.

[48] V. B. Ajabshir, M. S. Guzel, and E. Bostanci, "A Low-Cost Q-Learning-Based Approach to Handle Continuous Space Problems for Decentralized Multi-Agent Robot Navigation in Cluttered Environments," *IEEE Access*, vol. 10, pp. 35287–35301, 2022, doi: 10.1109/ACCESS.2022.3163393.

[49] W. Fang, Z. Liao, and Y. Bai, "Improved ACO algorithm fused with improved Q-Learning algorithm for Bessel curve global path planning of search and rescue robots," *Rob Auton Syst*, vol. 182, p. 104822, Dec. 2024, doi: 10.1016/j.robot.2024.104822.

[50] C. Wang, X. Yang, and H. Li, "Improved Q-Learning Applied to Dynamic Obstacle Avoidance and Path Planning," *IEEE Access*, vol. 10, pp. 92879–92888, 2022, doi: 10.1109/ACCESS.2022.3203072.

[51] D. U. Rijalusalam and I. Iswanto, "Implementation Kinematics Modeling and Odometry of Four Omni Wheel Mobile Robot on The Trajectory Planning and Motion Control Based Microcontroller," *Journal of Robotics and Control (JRC)*, vol. 2, no. 5, 2021, doi: 10.18196/jrc.25121.

[52] H. Taheri and C. X. Zhao, "Omnidirectional mobile robots, mechanisms and navigation approaches," *Mech Mach Theory*, vol. 153, p. 103958, Nov. 2020, doi: 10.1016/j.mechmachtheory.2020.103958.

[53] S. Chu, M. Lin, D. Li, R. Lin, and S. Xiao, "Adaptive reward shaping based reinforcement learning for docking control of autonomous underwater vehicles," *Ocean Engineering*, vol. 318, p. 120139, Feb. 2025, doi: 10.1016/j.oceaneng.2024.120139.

[54] P. Chintala, R. Dornberger, and T. Hanne, "Robotic Path Planning by Q Learning and a Performance Comparison with Classical Path Finding Algorithms," *International Journal of Mechanical Engineering and Robotics Research*, pp. 373–378, 2022, doi: 10.18178/ijmerr.11.6.373-378.

[55] Y. Shi and Z. Rong, "Analysis of Q-Learning Like Algorithms Through Evolutionary Game Dynamics," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 5, pp. 2463–2467, May 2022, doi: 10.1109/TCSII.2022.3161655.

[56] A. Nadeem, A. Ullah, and W. Choi, "Social-Aware Peer Selection for Energy Efficient D2D Communications in UAV-Assisted Networks: A Q-Learning Approach," *IEEE Wireless Communications Letters*, vol. 13, no. 5, pp. 1468–1472, May 2024, doi: 10.1109/LWC.2024.3375235.

[57] M. Muzammul, M. Assam, Y. Y. Ghadi, N. Innab, M. Alajmi, and T. J. Alahmadi, "IR-QLA: Machine Learning-Based Q-Learning

Algorithm Optimization for UAVs Faster Trajectory Planning by Instructed- Reinforcement Learning," *IEEE Access*, vol. 12, pp. 91300–91315, 2024, doi: 10.1109/ACCESS.2024.3420169.

[58] Y. Shi and Z. Rong, "Analysis of Q-Learning Like Algorithms Through Evolutionary Game Dynamics," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 5, pp. 2463–2467, May 2022, doi: 10.1109/TCSII.2022.3161655.

[59] G. Zheng, J. Zhang, S. Deng, W. Cai, and L. Chen, "Evolution of cooperation in the public goods game with Q-learning," *Chaos Solitons Fractals*, vol. 188, p. 115568, Nov. 2024, doi: 10.1016/j.chaos.2024.115568.

[60] H. Jiang, G. Li, J. Xie, and J. Yang, "Action Candidate Driven Clipped Double Q-Learning for Discrete and Continuous Action Tasks," *IEEE Trans Neural Netw Learn Syst*, 2022, doi: 10.1109/TNNLS.2022.3203024.

[61] J. Fan, X. Zhang, Y. Zou, Y. Li, Y. Liu, and W. Sun, "Improving policy training for autonomous driving through randomized ensembled double Q-learning with Transformer encoder feature evaluation," *Appl Soft Comput*, vol. 167, p. 112386, Dec. 2024, doi: 10.1016/j.asoc.2024.112386.

[62] K. Xie and A. Szolnoki, "Reputation in public goods cooperation under double Q-learning protocol," *Chaos Solitons Fractals*, vol. 196, p. 116398, Jul. 2025, doi: 10.1016/j.chaos.2025.116398.

[63] G. Liu *et al.*, "Weighted double Q-learning based eco-driving control for intelligent connected plug-in hybrid electric vehicle platoon with incorporation of driving style recognition," *J Energy Storage*, vol. 86, p. 111282, May 2024, doi: 10.1016/j.est.2024.111282.

[64] M. Ben-Akka, C. Tanougast, C. Diou, and A. Chaddad, "An Efficient Hardware Implementation of the Double Q-Learning Algorithm," in *International Conference on Electrical, Computer, Communications and Mechatronics Engineering, ICECCME 2023*, 2023, doi: 10.1109/ICECCME57830.2023.10252988.

[65] N. Li and S. I. Han, "Adaptive Bi-Directional RRT Algorithm for Three-Dimensional Path Planning of Unmanned Aerial Vehicles in Complex Environments," *IEEE Access*, vol. 13, pp. 23748–23767, 2025, doi: 10.1109/ACCESS.2025.3537697.

[66] Z.-J. Ding, X.-J. Meng, L.-X. Zhang, Y.-B. Guo, M.-Q. Chen, and S.-H. Zhen, "Variable Sampling Area RRT Algorithm Based on Limiting Joint Angle for Robotic Arm Path Planning," *IEEE Access*, vol. 13, pp. 30665–30677, 2025, doi: 10.1109/ACCESS.2025.3541645.

[67] Z. Sun, B. Xia, P. Xie, X. Li, and J. Wang, "NAMR-RRT: Neural Adaptive Motion Planning for Mobile Robots in Dynamic Environments," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 13087–13100, 2025, doi: 10.1109/TASE.2025.3551464.

[68] D. Uwacu, A. Yammanuru, K. Nallamotu, V. Chalasani, M. Morales, and N. M. Amato, "HAS-RRT: RRT-Based Motion Planning Using Topological Guidance," *IEEE Robot Autom Lett*, vol. 10, no. 6, pp. 6223–6230, Jun. 2025, doi: 10.1109/LRA.2025.3560878.

[69] Z. Yao, Z. Liu, and C. Han, "The Improved RRT Integrated With the Artificial Potential Field Path Planning Algorithm," *IEEE Access*, vol. 13, pp. 68398–68409, 2025, doi: 10.1109/ACCESS.2025.3561348.

[70] S. Mohammad Langari, F. Vahdatikhaki, and A. Hammad, "Improving the performance of RRT path planning of excavators by embedding heuristic rules," *Advanced Engineering Informatics*, vol. 62, p. 102724, Oct. 2024, doi: 10.1016/j.aei.2024.102724.

[71] R. Li *et al.*, "Emperor Yu Tames the Flood: Water Surface Garbage Cleaning Robot Using Improved A* Algorithm in Dynamic Environments," *IEEE Access*, vol. 13, pp. 48888–48903, 2025, doi: 10.1109/ACCESS.2025.3551088.

[72] Z. Liu, H. Liu, Z. Lu, and Q. Zeng, "A Dynamic Fusion Pathfinding Algorithm Using Delaunay Triangulation and Improved A-Star for Mobile Robots," *IEEE Access*, vol. 9, pp. 20602–20621, 2021, doi: 10.1109/ACCESS.2021.3055231.

[73] G. Tang, C. Tang, C. Claramunt, X. Hu, and P. Zhou, "Geometric A-Star Algorithm: An Improved A-Star Algorithm for AGV Path Planning in a Port Environment," *IEEE Access*, vol. 9, pp. 59196–59210, 2021, doi: 10.1109/ACCESS.2021.3070054.

[74] Y. Ma, Y. Zhao, Z. Li, X. Yan, H. Bi, and G. Królczyk, "A new coverage path planning algorithm for unmanned surface mapping vehicle based on A-star based searching," *Applied Ocean Research*, vol. 123, p. 103163, Jun. 2022, doi: 10.1016/j.apor.2022.103163.

[75] A. Allus and M. Unel, "Angle-based multi-goal ordering and path-planning using an improved A-star algorithm," *Rob Auton Syst*, vol. 190, p. 105001, Aug. 2025, doi: 10.1016/j.robot.2025.105001.

[76] J. Huang, C. Chen, J. Shen, G. Liu, and F. Xu, "A self-adaptive neighborhood search A-star algorithm for mobile robots global path planning," *Computers and Electrical Engineering*, vol. 123, p. 110018, Apr. 2025, doi: 10.1016/j.compeleceng.2024.110018.

[77] X. Zhang, G. Xiong, Y. Wang, S. Teng, and L. Chen, "D-PBS: Dueling Priority-Based Search for Multiple Nonholonomic Robots Motion Planning in Congested Environments," *IEEE Robot Autom Lett*, vol. 9, no. 7, pp. 6288–6295, Jul. 2024, doi: 10.1109/LRA.2024.3402183.

[78] T. P. Nguyen, H. Nguyen, and H. Q. T. Ngo, "Towards sustainable scheduling of a multi-automated guided vehicle system for collision avoidance," *Computers and Electrical Engineering*, vol. 120, p. 109824, Dec. 2024, doi: 10.1016/j.compeleceng.2024.109824.

[79] X. Zhou, X. Wang, Z. Xie, J. Gao, F. Li, and X. Gu, "A Collision-free path planning approach based on rule guided lazy-PRM with repulsion field for gantry welding robots," *Rob Auton Syst*, vol. 174, p. 104633, Apr. 2024, doi: 10.1016/j.robot.2024.104633.

[80] T. S. Khan, D. Pfoser, S. Ruan, and A. Züfle, "Simplifying traffic simulation - from Euclidean distances to agent-based models," *Computational Urban Science*, vol. 4, no. 1, p. 32, Nov. 2024, doi: 10.1007/s43762-024-00145-x.

[81] M. Jeon, J. Lee, and S.-K. Ko, "Modular Reinforcement Learning for Playing the Game of Tron," *IEEE Access*, vol. 10, pp. 63394–63402, 2022, doi: 10.1109/ACCESS.2022.3175299.

[82] R. Trauth, K. Moller, G. Würsching, and J. Betz, "FRENETIX: A High-Performance and Modular Motion Planning Framework for Autonomous Driving," *IEEE Access*, vol. 12, pp. 127426–127439, 2024, doi: 10.1109/ACCESS.2024.3436835.

[83] T. Lin, C. Yue, Z. Liu, and X. Cao, "Modular Multi-Level Replanning TAMP Framework for Dynamic Environment," *IEEE Robot Autom Lett*, vol. 9, no. 5, pp. 4234–4241, May 2024, doi: 10.1109/LRA.2024.3377556.

[84] Y. Zhu, Z. Wang, C. Chen, and D. Dong, "Rule-Based Reinforcement Learning for Efficient Robot Navigation With Space Reduction," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 2, pp. 846–857, Apr. 2022, doi: 10.1109/TMECH.2021.3072675.

[85] M. I. Radaideh and K. Shirvan, "Rule-based reinforcement learning methodology to inform evolutionary algorithms for constrained optimization of engineering applications," *Knowl Based Syst*, vol. 217, p. 106836, Apr. 2021, doi: 10.1016/j.knosys.2021.106836.

[86] Y. Wang *et al.*, "Enhancing Closed-Loop Performance in Learning-Based Vehicle Motion Planning by Integrating Rule-Based Insights," *IEEE Robot Autom Lett*, vol. 9, no. 9, pp. 7915–7922, Sep. 2024, doi: 10.1109/LRA.2024.3433170.