Complex & Intelligent Systems (2024) 10:3025–3039
https://doi.org/10.1007/s40747-023-01278-y

ORIGINAL ARTICLE

Load balancing of multi-AGV road network based on improved Q-learning algorithm and macroscopic fundamental diagram

Xiumei Zhang[1] · Wensong Li[1] · Hui Li[1] · Yue Liu[1] · Fang Liu[1]

Received: 12 April 2023 / Accepted: 4 November 2023 / Published online: 10 January 2024
© The Author(s) 2024

Abstract
To address the challenges of traffic congestion and suboptimal operational efficiency in the context of large-scale applications like production plants and warehouses that utilize multiple automatic guided vehicles (multi-AGVs), this article proposed using an Improved Q-learning (IQL) algorithm and Macroscopic Fundamental Diagram (MFD) for the purposes of load balancing and congestion discrimination on road networks. Traditional Q-learning converges slowly, which is why we have proposed the use of an updated Q value of the previous iteration step as the maximum Q value of the next state to reduce the number of Q value comparisons and improve the algorithm's convergence speed. When calculating the cost of AGV operation, the traditional Q-learning algorithm only considers the evaluation function of a single distance and introduces an improved reward and punishment mechanism to combine the operating distance of AGV and the road network load, which finally equalizes the road network load. MFD is the basic property of road networks and is based on MFD, which is combined with the Markov Chain (MC) model. Road network traffic congestion state discrimination method was proposed to classify the congestion state according to the detected number of vehicles on the road network. The MC model accurately discriminated the range near the critical point. Finally, the scale of the road network and the load factor were changed for several simulations. The findings indicated that the improved algorithm showed a notable ability to achieve equilibrium in the load distribution of the road network. This led to a substantial enhancement in AGV operational efficiency.

Keywords Multi-AGVs · Improved Q-learning · Macroscopic fundamental diagram · Congestion state discrimination · Load balancing

Introduction

The expansion of logistics and warehousing, manufacturing, and unmanned transportation has been characterized by its rapid rise. There is a pressing need for the implementation of intelligent warehouse and distribution systems [1]. Among them, an automated guided vehicle (AGV) is one of the key elements to realize an intelligent warehousing and distribution system [2]. In large-scale AGV distribution systems, factors such as site constraints and path conflicts render AGV distribution less efficient, and the resulting traffic congestion problems are becoming more apparent [3]. Therefore, traffic congestion can be reduced by optimizing vehicles' routes or making connections between vehicles. This will make the road network more efficient and load-balanced [4].

The algorithms used to solve AGV road network loading are generally classified as classical, swarm intelligence, and machine learning algorithms [5]. Traditional algorithms such as the A* algorithm and the Dijkstra algorithm are commonly utilized in path planning problems. Traditional algorithms have a long running time and are rarely used for road network loading problems [6]. Swarm intelligence algorithms such as genetic algorithms and the Artificial Bee Colony algorithm are often employed for path planning, vehicle scheduling, and load balancing problems. The disadvantage of the algorithms is that they can only be used for single AGV or small-scale AGV scheduling problems, take a longer time, and operate less efficiently in solving load balancing problems [7]. Machine learning algorithms can be categorized into three distinct types, namely supervised learning, unsupervised learning, and reinforcement learning

✉ Hui Li
lihui@ccut.edu.cn

Wensong Li
2202104055@stu.ccut.edu.cn

[1] School of Electrical and Electronic Engineering, Changchun University of Technology, Changchun 130012, China

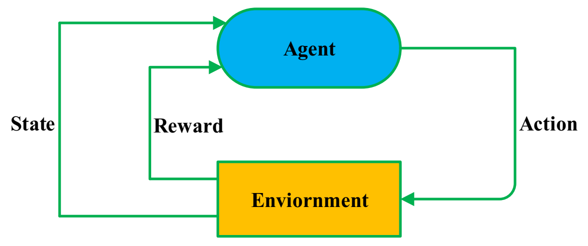


Fig. 1 Reinforcement learning model

tasks. Training data is essential for both supervised and unsupervised machine learning approaches to simulate the job [8]. Reinforcement learning tasks rely on real-time interaction between the system and the environment, which has tremendous advantages in handling path planning, cluster scheduling, and road network load balancing for large-scale AGVs [9].

As a type of classical reinforcement learning algorithm, as shown in Fig. 1, Q-learning algorithms have problems such as high computational effort and slow convergence in discrete states [10]. Devraj et al. proposed a matrix gain algorithm designed to address the problem of slow convergence by continuously updating the equation through two-time scales of the matrix gain sequence to optimize its asymptotic variance [11]. Low et al. introduced a guided concept to Q-learning that used a flower pollination algorithm (FPA) to enhance the initialization of Q-learning, which used FPA to initialize the Q value to accelerate the convergence of Q-Learning that verified the effectiveness of the algorithm [12]. After optimizing the Q-learning algorithm, many researchers have applied the improved algorithm in path planning for AGVs, while some have applied the improved algorithm in load balancing of road networks for AGVs [13]. Roh et al. proposed Q-learning-based load balancing routing to estimate the network by intermediate nodes in road networks obtained by queue states from each ground vehicle node using low overhead technique for load; they showed an improvement in packet delivery ratio, network utilization, and delay [14]. Sethi et al. proposed an online Federated Deep Q-learning-based Offloading technique for vehicular fog computing that reduces load factor by finding the connection between the unit and the server through global information. This provides an excellent concept for our project [15]. However, when multiple vehicles pass through the road network, that queue state can be complicated and difficult to obtain.

In addition to the traditional Q-learning algorithms, the research of many cutting-edge algorithms also brings important references to the research work in this paper [16].Xiao et al. present a deep contrastive representation learning with self-distillation (DCRLS) for the time series domain. DCRLS gracefully combines data augmentation, deep contrastive learning, and self-distillation [17]. Song et al. Evolutionary Multi-Objective RL (EMORL) algorithm to solve trajectory control and task offloading problems. They improve the multi-task multi-objective proximal policy optimization of the original EMORL by retaining all new learning tasks in the offspring population, which can preserve promising learning tasks [18].

Within a region characterized by an intricate road network, there exists a functional relationship that establishes a correlation between the quantities of vehicles present on the road network. This relationship correlates with the overall aggregate number of cars. The MFD is a key characteristic of road networks that visually and accurately represents dynamic traffic patterns [19]. This relationship serves as a means to establish the MFD. Ambühl et al. proposed a technically feasible traffic state-based upper bound smooth approximation form of the MFD function, and the MFD curve is estimated analytically by adjusting the function parameters [20]. To address the multi-scale and uncertain distribution of vehicles on the road, Shen et al. used the K-means-GIoU algorithm to enhance the network's focus on the main regions, and experiments have shown that the algorithm can detect vehicles quickly [21]. Geroliminis et al. proposed a three-dimensional vehicle MFD model, which clusters the network into a few regions with similar mode composition, congestion levels, and integrated traffic strategy for a bimodal multi-regional urban network that divides the priority between multiple vehicles [22]. In urban road networks, the interaction between different modes significantly affects overall travel efficiency. MFDs can be fitted when traffic data are complete, but updated functional forms need to be designed to estimate MFDs when scanty or no data is available [23]. Loder et al. introduced a novel functional form for determining operational features of a given topology by utilizing lower envelope parameters. The estimation of smoothing parameters in this process is achieved by utilizing traffic data, ensuring that the values consistently maintain similarity. Approximation can also be achieved for many networks [24]. Halakoo et al. analyzed the accuracy of the proposed e-MFD for directed traffic demand using a synthetic grid network and a real city-level network. The proposed model can be employed for emission measurement in large-scale networks and hierarchical traffic control systems for more homogeneous congestion distribution and emission control [25].

The proposed MFD model in this study differs from existing approaches by avoiding the use of approximation techniques. Instead, it solves the critical point with a cubic function derived from the fitting result. This method offers improved accuracy in assessing the road network status. For AGV load balancing, road network conflict or congestion is resolved by optimization algorithms and verified with constructed maps through simulation experiments. When a road network congestion problem is solved, rarely is the evaluation metric studied, the MFD [26]. In the study of MFD, the functional relationship is often studied to fit or estimate the MFD curve from acquired vehicle data, from which specific data are derived by simulation and the data are analyzed [27]. There are few studies where multiple AGVs operate on a road network to derive data and use one evaluation metric to analyze the data accurately.

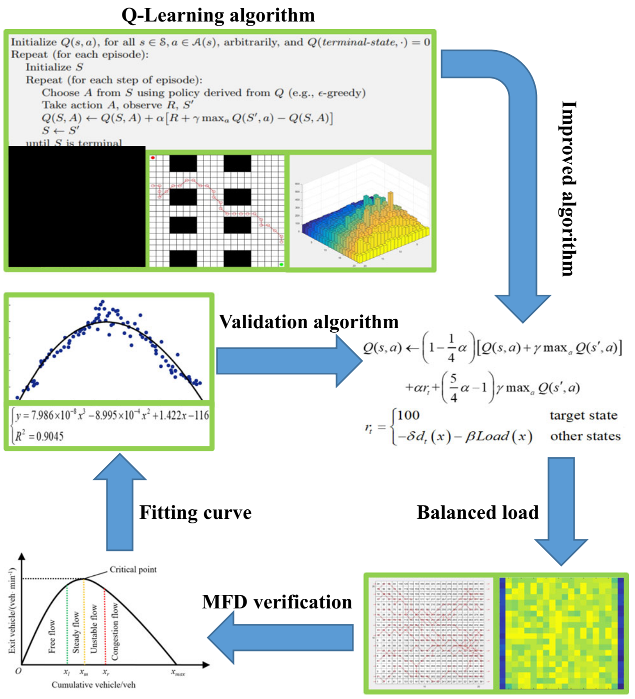
The improvement of Q-learning algorithm by existing articles, although all of them finally converge, the problem of too high computation is still not well solved. The reward and punishment functions are also fixed surrogate value rewards and do not optimize the road network as it changes in real time. In the study of load balancing, the problem is solved when the algorithm converges at the end of the simulation. However, there is hardly any mention of the analysis of the simulation data. This is, i.e., whether the state of the road network at this time can be accurately determined according to the number of AGVs.

Based on the above issues, we propose an Improved Q-learning algorithm applied to the field of multi-AGV load balancing for the problems of slow convergence of Q-learning algorithm and AGV road network load congestion. The research can bring inspiration and theoretical experience to the construction of smart logistics factories. The improved algorithm can improve transportation efficiency and equalize road network load when multiple AGVs are actually operating.

The proposed method in this paper differs from other methods by replacing the learning rate of the Q-learning update criterion with a larger learning rate, so that the algorithm's convergence speed is improved. In action selection, partial sampling in the space of optimal joint actions $Q^*(s, a)$ can effectively reduce computation in the learning phase. The length coefficient and load coefficient are innovatively introduced to optimize road network load in real time. This is done by combining AGV running distance and the regional load on the road network. The MFD based on MC model is also established to accurately identify the congestion state of the road network. This lays a theoretical foundation for the study of AGV road network load.

After obtaining data on the road network, curve fitting of MFD based on MC model can accurately discern different congestion states of roads according to the number of AGVs on the road network, allowing technicians to grasp the field state more intuitively, as shown in Fig. 2.

The IQL algorithm proposed in this paper has a poor strategy for agents to update the Q value at the beginning of training. The use of the current estimation function as the historical estimation function will have a large error. This will make the algorithm learn a poor lesson at the beginning of training.



```
Initialize $Q(s,a)$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(terminal\text{-}state, \cdot) = 0$
Repeat (for each episode):
    Initialize $S$
    Repeat (for each step of episode):
        Choose $A$ from $S$ using policy derived from $Q$ (e.g., $\epsilon$-greedy)
        Take action $A$, observe $R$, $S'$
        $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$
        $S \leftarrow S'$
    until $S$ is terminal
```

$$\begin{cases} y = 7.986\times10^{-8}x^3 - 8.995\times10^{-4}x^2 + 1.422x - 116 \\ R^2 = 0.9045 \end{cases}$$

$$Q(s,a) \leftarrow \left(1-\frac{1}{4}\alpha\right)[Q(s,a)+\gamma \max_a Q(s',a)] + \alpha r_t + \left(\frac{5}{4}\alpha - 1\right)\gamma \max_a Q(s',a)$$

$$r_t = \begin{cases} 100 & \text{target state} \\ -\delta d_t(x) - \beta Load(x) & \text{other states} \end{cases}$$

Fig. 2 General framework diagram

Therefore, this paper proposes to use the Q value updated by the joint action after the agent's previous iteration step as the maximum Q value of the next state. This will reduce the number of Q value comparisons and improve the algorithm's convergence speed of the algorithm. Instead of exhaustively exploring all possible joint action Q values, this improvement uses a partial sampling approach to explore the space of optimal joint actions $Q^*(s, a)$ and identify the greatest Q value. This can effectively reduce computational effort in the learning phase.

When multiple AGVs run on the road, the computation is still large and convergence is slow.

The reward and punishment function in the Q-learning algorithm is improved by replacing the purely numerical reward value with the form of an algebraic function, so that the AGV road network load decreases with the change of the algebraic function. To avoid local excessive traffic flow when multiple AGVs operate on a road network, the operating distance of AGVs and the area load on the road network are combined. The length factor and load factor are introduced to obtain the optimal coefficient value by experiment. MFD is introduced to accurately discriminate the congestion of the road network. It is combined with MC model and MFD curve fitting using data obtained from AGV operation on the road network. According to the different area divisions in MFD, the road network stage was determined. For the critical point attachment range, the MC model was used to make accurate judgments.

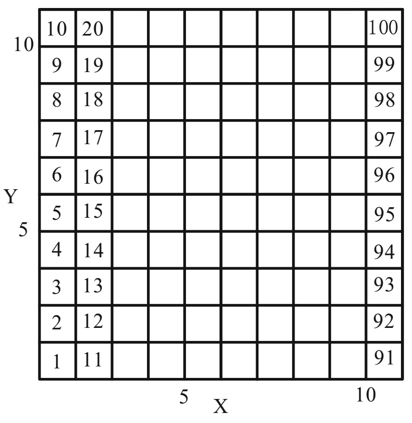


Fig. 3 Raster serial number map

Finally, the IQL algorithm was simulated and verified on a constructed raster map and compared experimentally with other optimization algorithms. AGV road network operation data are used to fit MFD curves using simulation software based on AGV road network operation data.

Environmental modeling

The map modeling of the AGV path planning environment was modeled by the raster construction method [28]. In the raster map, raster grids that are in the obstacle area are represented in black and are considered obstacle areas. Other grids are represented in white and considered passable areas. The coordinates of the first grid in the lower left corner of the coordinate system are actually (1.5, 1.5), which after rounding are (1, 1), corresponding to the ordinal number 1 in the raster map, the ordinal number of the grid with coordinates (1, 2) is 2, and so on. The coordinates of the grid (x, y) and the ordinal number i are one-to-one mapped to each other, with the mapping formula shown in Eq. (1).

$$\begin{cases} x_i = l * [\mathrm{mod}(i, n_x) - l] \\ y_i = l * \left[n - \mathrm{int}\left(\frac{i}{n_y} \right) + l \right] \end{cases}, \tag{1}$$

where n_x is the mapped row, n_y denotes the mapped column, mod and int are the remainders and integer operations, respectively, i denotes the label of the grid, and l denotes the side length of the grid square.

According to the above mapping rules, grid 5 corresponds to the coordinates of (1, 5). The grid serial number corresponds to Fig. 3.

Improvement of the Q-learning algorithm

Principle of Q-learning algorithm

The Q-learning algorithm is a model-free value function reinforcement learning algorithm [29]. The algorithm can be described formally by the Markov Decision Process (MDP) framework [30]. The MDP can be defined by a five-tuple (A, S, P, R, γ), where $a \in A$, $s \in S$, $r \in R$. A denotes the action space, the set of all legal actions that can be taken. S denotes the state space. P denotes the state transfer probability matrix, which defines the probability that the agent generates state s‘ after interacting with the environment and transfers to the new state s' after acting a according to state.

R is the reward generated after the agent (in this paper, the agent is AGV) interacts with the environment to produce action a, the environment state changes from state s to the new state s'. The evaluation of a strategy often relies on the cumulative reward acquired by an agent through a series of actions. A higher cumulative reward is indicative of a more favorable strategy, with the sum of the cumulative reward values of the states defined as presented in Eq. (2).

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{2}$$

Equation (2) represents the cumulative return, r_{t+1} was the reward of environmental feedback after the agent selected and makes an action at the moment $t + 1$. $\gamma \in (0,1)$ was the discount factor. When the value of γ was equal to 0, the agent only considered the next step reward. As the value ofγ tended to 1, more future rewards were considered. In some situations, the present reward was more significant, while in others, the future reward was more important. Depending on what was needed, theγ value could be adjusted.

According to the current environment state, the agent generates the corresponding actions. The generated states and actions are stored in a Q table. If the state–action pair receives positive environmental rewards, its corresponding Q value will keep increasing, while in a negative environment, it will keep decreasing. After each training round, $Q(s, a)$ gradually approaches $Q^*(s, a)$. The Q table formula updates with each round based on the Bellman equation [31] (Eq. 3), as shown in Fig. 4.

$$Q(s, a) \leftarrow Q(s, a) + \alpha\left[r_t + \gamma \max_a Q(s', a) - Q(s, a)\right]. \tag{3}$$

As in Eq. (3), $\alpha \in (0, 1)$ was the learning rate, r_t was the reward the agent received when choosing an action.

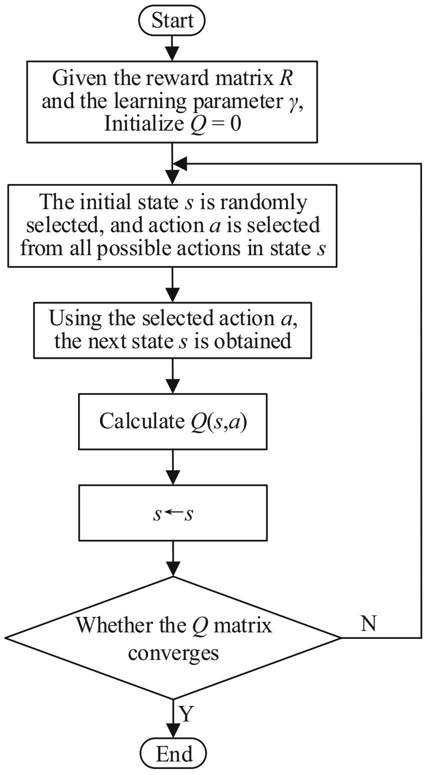

Start
Given the reward matrix R and the learning parameter γ, Initialize $Q = 0$
The initial state s is randomly selected, and action a is selected from all possible actions in state s
Using the selected action a, the next state s is obtained
Calculate $Q(s,a)$
$s \leftarrow s$
Whether the Q matrix converges
N
Y
End

Fig. 4 Q-learning flow chart

Maximum Q update mechanism

The strategy π was the probability of the agent taking action a at state s, which was defined by Eq. (4):

$$\pi(a|s) = P[A = a|S = s] \tag{4}$$

From Eq. (4), it could be seen that π described the agent's action, which only related to the current state s', independent of the other states, from which π was independent of time. $Q^*(s, a)$ denotes the value of Q corresponding to the optimal joint action $a^* = (a_1^*, a_2^*, \cdots, a_n^*)$ in state s. $\pi^*(a|s) = (\pi_1^*(a|s), \pi_2^*(a|s), \cdots, \pi_n^*(a|s))$ denotes the optimal joint strategy corresponding to the optimal joint action in the current state.

The AGV performs the selection of action a at state s. The optimal joint strategy $\pi^*(a|s)$ was selected with probability η, other actions were selected with probability $1 - \eta$. At this point, the AGV received a reward r_t, $Q^*(s, a)$ synchronized with the Q value update, where the AGV received the reward as the global reward under the optimal joint action.

Based on Eq. (3), the updated formula of the Q-learning algorithm can be rewritten as:

$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha\left[r_t + \gamma \max_a Q(s', a)\right] \tag{5}$$

Let $\max_a{}' Q(s', a')$ be the maximum Q value of the next state when the Q value of the state action (s, a) is updated in the previous iteration step. Then the Q value update formula of the algorithm at this point is:

$$\begin{aligned} Q(s, a) \leftarrow\ & (1 - \alpha)Q(s, a) + \alpha\left[r_t + \gamma \max_a' Q(s', a')\right] \\ & +(1 - \alpha)\gamma\left[\max_a Q(s', a) - \max_a' Q(s', a')\right] \end{aligned} \tag{6}$$

According to Eqs. (5) and (6), it can be seen that the main difference between the two is that the learning rate before $[\max_a Q(s', a) - \max_a{}' Q(s', a)']$ is different, and the learning rate of the Improved Q-learning algorithm is $(1 - \alpha)$, which improves the agent's convergence speed.

When training the AGV, the algorithm did not iterate through all joint action Q values after updating the Q values in the previous iteration step. Instead, a partial sample was taken from the space of optimal joint actions $Q^*(s, a)$ to find the maximum Q value. This could effectively reduce the computational effort in the learning phase [32]. According to the design requirements of this algorithm, the equivalence of Eq. (6) was updated, so that the updated equation was shown in Eq. (7).

$$\begin{aligned} Q(s, a) \leftarrow\ & (1 - \alpha)\left[Q(s, a) + \gamma \max_a Q(s', a)\right] \\ & +\alpha r_t + (2\alpha - 1)\gamma \max_a' Q(s', a') \end{aligned} \tag{7}$$

To facilitate the algorithm description, Eq. (7) is equivalently deformed, and to avoid the algorithm's learning rate being too adventurous, the learning rate for obtaining the maximum Q value of the next state is improved from $(2\alpha - 1)$ to $(3/2\alpha - 1)$ to obtain Eq. (8):

$$\begin{aligned} Q(s, a) \leftarrow & \left(1 - \frac{1}{2}\alpha\right)\left[Q(s, a) + \gamma \max_a Q(s', a)\right] \\ & + \alpha r_t + \left(\frac{3}{2}\alpha - 1\right)\gamma \max_a' Q(s', a') \end{aligned} \tag{8}$$

Let $Q_n(s, a)$ be the value of Q at the n^{th} iteration,and replace the learning rate α_n with $1/n(s,a)$. The updated formula is shown in Eq. (9):

$$\begin{aligned} Q_{n+1}(s, a) \leftarrow & \left(1 - \frac{1}{n(s, a)}\right)Q_n(s, a) + \frac{1}{n(s, a)} \\ & \times \left[r_n + \gamma \max_{n-1}' Q(s', a')\right] + \left(1 - \frac{1}{n(s, a)}\right)\gamma \\ & \times \left[\max_n Q(s', a) - \max_{n-1}' Q(s', a')\right] \end{aligned} \tag{9}$$

An equivalent update to the above equation:

$$Q_{n+1}(s,a) \leftarrow \left(1-\frac{1}{n(s,a)}\right) \times \sum_{i=1}^{n}\left[r_i+\gamma\left(1-\frac{1}{n(s,a)}\right)\max_n Q(s',a)\right] \tag{10}$$

Proof: When $n = 1$, $Q_2 = r_1 + \gamma \max_0 Q(s', a) = r_1$.

Assuming that Eq. (10) holds when $n = k$, we have:

$$Q_k(s,a) \leftarrow \left(\frac{1}{k(s,a)-1}\right) \times \sum_{i=1}^{k-1}\left[r_i+\gamma\left(1-\frac{1}{k(s,a)-1}\right)\max_{k-1} Q(s',a)\right] \tag{11}$$

When $n = k + 1$, then we have:

$$Q_{k+1}(s,a) \leftarrow \left(1-\frac{1}{k(s,a)}\right)Q_k(s,a)+\frac{1}{k(s,a)} \times \left[r_k+\gamma \max_{k-1} Q(s',a)\right]+\left(1-\frac{1}{k(s,a)}\right)\gamma \times \left[\max_k Q(s',a)-\max'_{k-1} Q(s',a')\right] \tag{12}$$

Organized and finally available:

$$Q_{k+1}(s,a) \leftarrow \frac{1}{k(s,a)}\sum_{i=1}^{k-1} r_i+\left[\frac{1}{k(s,a)}(k(s,a)-1) -\left(1-\frac{1}{k(s,a)}\right)\right]\gamma \max'_{k-1} Q(s',a') +\left(1-\frac{1}{k(s,a)}\right)\gamma \max_k Q(s',a) \tag{13}$$

The proof is complete.

It can be seen in Eq. (10) that by replacing the function of historical Q estimation with the estimation function of current Q, the agent continuously optimizes the strategy based on learned experience. This improves the algorithm's convergence speed.

Improving the reward and punishment function

After improving the reward and punishment function, the number of AGV runs is set to $100u$ ($u \in N^{+}$) task times. The path length of each AGV running on the road network is recorded. In addition, the amount of road network load at each point on the raster map is extracted. The essence of AGV load balancing is based on the traditional Q-learning algorithm, which takes load factor into account in the actual path cost [33]. That is, the operating distance and the road network load are combined. The setting of the reward and punishment function will directly affect the operating efficiency of AGVs on the road network.

Set the path planning reward and punishment function of the traditional Q-learning algorithm as shown in Eq. (14).

$$r_t = \begin{cases} 100 & \text{target state} \\ -1 & \text{other states} \end{cases} \tag{14}$$

The Q-learning algorithm was used to allow a single AGV to perform path planning in a road network. According to the design of the reward and punishment function, during the interaction between the AGV and the environment, the AGV received a reward of 100 when it reached the target state and a penalty of − 1 when in other states. Finally, the Q-matrix was judged to be convergent. The training ended if it converged, otherwise it continued.

For multi-AGVs, the reward of only the target state and other states is not enough to complete the training complete.Node conflicts arise when multi-AGVs operate, so the effectiveness of IQL algorithm is verified by adding trap barriers to the road network. The multi-AGV reward and punishment function is set as shown in Eq. (15).

$$r_t = \begin{cases} 100 & \text{target state} \\ -50 & \text{same position penalty} \\ -90 & \text{trap punishment} \\ -1 & \text{other states} \end{cases} \tag{15}$$

The multi-AGV reward and punishment function was enhanced by incorporating the same position penalty and trap penalty. When two AGVs ran to the same node position on the road network, a node conflict would occur. A − 50 colocation penalty would be given to AGVs when they fell into a trap. A trap penalty of − 90 was also given when an AGV fell into a trap. Although the four-state reward and punishment function improved the effectiveness of the single AGV operation, multiple AGVs would continue to proceed to this location after the punishment. This was due to the road network map limitation when they passed through the same location, which was not a perfect condition restriction.

For a better solution to road network congestion, path length and load are set in the reward and punishment function. It allows AGVs to effectively avoid local high load areas on the road network. It also improves the operational efficiency of the road network while avoiding conflicts between multiple AGVs.

The reward and punishment function based on multi-AGV load balancing is set as shown in Eq. (16).

$$r_t = \begin{cases} 100 & \text{target state} \\ -\delta d_t(x) - \beta Load(x) & \text{other states} \end{cases} \quad (16)$$

The $\delta \in (0, 1)$ in Eq. (8) is the path length coefficient, $d_t(x)$ is the sum of the paths taken by the AGV, $d_t(x) = d_1 + d_2 + \cdots + d_t$, $t \in (1, n)$, $\beta \in (0, 1)$, is the load factor. *Load* (x) is the number of vehicles passed by the current node on the road network. When δ equals 0, the function only takes into account the load factor as a penalty value. Whenβ equals 0, the function only considers the path length as a penalty value. When both are 0, the function does not consider load balancing. Setting the path length and load in the reward and penalty function can make AGVs operating on the road network choose the optimal path according to the size of the load in the sub-region or the length of the travel length. This approach allows AGVs to avoid areas with high local loads, ultimately leading to road network optimization.

The Improved Q-learning algorithm can be summarized as the following pseudocode:

Algorithm: Improved Q-learning algorithm

```
Input: episodes, $\alpha, \gamma, \delta, \beta$
Output: $Q$
Initialize: set $Q(s, a)$ arbitrarily, for each $s$ in $S$ and $a$ in $A(s)$, set $Q$(terminal state, ) = 0
Repeat for each episode in episodes
    Initialize: $S \leftarrow$ first state of episode
    Repeat for each step of episode
        $A$ = policy($Q$, $S$) (e.g. $\varepsilon$-greedy policy)
        $R = -\delta d_t(x) - \beta Load(x)$, $S'$ = perform_action($S$, $A$)
        $Q(S, A) \leftarrow (1-1/2\alpha)\,[Q(S, A)+\gamma\, max_a\, Q(S', A)]$
                $+\alpha R + (3/2\alpha-1)\,\gamma\, max_a\, Q(S', A)$
        $S \leftarrow S'$
    Until S is terminal state.
Until all episodes are visited.
```

Macroscopic fundamental diagram drawing

The MFD is a graphical representation that depicts the correlation between the cumulative quantities of cars present on a road network. It also depicts the quantity of vehicles exiting the road network. In a general sense, the horizontal axis is utilized to denote the cumulative quantity of cars present on the road network. The vertical axis denotes the quantity of cars exiting the road network. After acquiring the operational data of AGVs on the road network with the IQL algorithm, it is necessary to preprocess the data to enhance the performance of subsequent tests [34]. According to the congestion status of vehicles on the road network, we divided the MFD into four states in this paper [35].

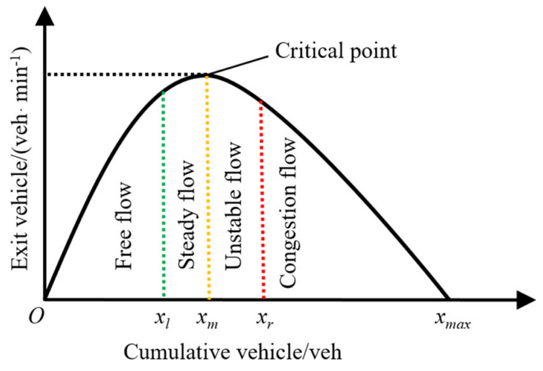


Fig. 5 Macroscopic fundamental diagram

The graph depicted in Fig. 5 illustrates a notable and swift increasing trajectory in correlation with the escalating influx of automobiles into the road network. As the influx of automobiles into the road network persists, the volume of vehicles present on the road network will progressively grow. At this time, the road network reaches its maximum critical point. After reaching the critical point, there is a tendency for the road network load to become saturated, resulting in a steady decrease in the number of vehicles traveling inside the road network. This decrease is seen by a noticeable curve decline. The MFD of the whole road network can be plotted using function fitting with AGV operational data [36].

Determination of the range near the critical point

How to quickly identify the graphical critical points of traffic congestion after drafting the basic macroscopic map of the road network is the key to solving the problem [20]. The collected data are utilized to fit the MFD curve, resulting in the equation $y = f(x)$ for the curve. The first-order derivative $y' = f'(x)$ is found for the curve equation to analyze the curve change trend and derive the curvature change turning point. The curvature is expressed in c. To facilitate calculation, MFD steady and unsteady flows are considered intermediate flows. The rising, middle, and falling segments of the curve can be judged according to the curvature magnitude [37].

The important values x_l and x_r in Fig. 5 can be obtained by solving the equations. Since the vehicles change with time, there will be small fluctuations near the critical point. Define the fluctuation value as x'. The range near the critical point x_l is $(x_l - x', x_l + x')$, the range near the critical point x_r is $(x_r - x', x_r + x')$. MC models are used in this paper to accurately discriminate road network status when road network vehicles are near the critical point. When road network vehicles are not in the range near the critical point, the road network congestion status is directly judged.

Table 1 Congestion state discriminative method based on MC model

State at time T	Y_{t+1}	Set state	P_{pq}	Final state discrimination
Free flow	$Y_{t+1} > c$	Free flow	1	Free flow
	$-c \le Y_{t+1} \le c$	Intermediate flow	0	Free flow
	$Y_{t+1} < -c$	Congestion flow	0	Intermediate flow
Intermediate flow	$Y_{t+1} > c$	Free flow	0	Intermediate flow
	$-c \le Y_{t+1} \le c$	Intermediate flow	1	Intermediate flow
	$Y_{t+1} < -c$	Congestion flow	0	Intermediate flow
Congestion flow	$Y_{t+1} > c$	Free flow	0	Intermediate flow
	$-c \le Y_{t+1} \le c$	Intermediate flow	0	Congestion flow
	$Y_{t+1} < -c$	Congestion flow	1	Congestion flow

Determination of congestion status within the vicinity of the critical point

When the road network vehicles are in the range near the critical point, the MC model congestion discrimination method is established. The critical point state is divided, as shown in Eq. (17), as the equation for the rate of change in time is $t + 1$.

$$Y_{t+1} = \frac{f_{t+1}(x) - f_t(x)}{x_{t+1} - x_t}, \tag{17}$$

where f_t (x) is the number of vehicles leaving the road network at time t, f_{t+1} (x) is the number of vehicles leaving the road network at time $t + 1$. x_t is the number of vehicles on the road network at time t, and $x_t + 1$ is the number of vehicles on the road network at time $t + 1$. The MFD state could be judged according to the magnitude of the rate of change Y in Eq. (9). When the rate of change of Y was low, it was close to the horizontal segment when the state of the road network was an intermediate segment composed of stable and unstable flows. When the value of Y is too large or too small, it corresponds to free flow and congested flow in MFD, respectively.

The MC model depends only on its previous state on the probability of state transfer at a given moment. The utilization of MC models has been extensively employed in various time series models [38]. The core of the MC model is the state transfer matrix $P = [P_{pq}]_{n\times n}$, where P_{pq} is the probability that the system is in state p at the moment t and in state q at the moment $t + 1$. n is the number of all possible states of the system [39]. The number of n is the number of all possible states of the system, while the state transfer matrix for determining the congestion state of the road network is shown in Eq. (18).

$$P = \begin{pmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{18}$$

The transfer probability was represented by 0 and 1 integers, with 0 representing that the p state could not be transferred to the q state and 1 representing that the p state can be transferred to the q state [40]. The state transfer matrix could judge the traffic state near the critical point, with the discrimination method shown in Table 1.

Judgment of congestion status outside the range near the critical point.

When the vehicles on the road network are outside the range and near the critical point, steady flow and unsteady flow are first grouped into one category, i.e., intermediate flow, for the convenience of calculation. The congestion state is discriminated as shown in Eq. (19).

$$M = \begin{cases} 0 \le n_t < x_l & \text{Free flow} \\ x_l \le n_t < x_r & \text{Intermediate flow}, \\ n_t \ge x_r & \text{Congestion flow} \end{cases} \tag{19}$$

where M denotes the congestion level of the road network at time t, n_t is the number of vehicles detected on the road network at time t, as well as n_t did not belong to the range within the vicinity of the critical point.

Simulation experiments

To verify the effectiveness of the IQL algorithm, AGV road network load balancing simulation experiments were conducted in MATLAB 2022a. Based on the data obtained from the experiments, the MFD curve simulation was performed in PTV Vissim 8. First, the convergence speed of IQL algorithm was verified, comparing the Improved Artificial Bee Colony (IABC) algorithm, Q-learning algorithm, and IQL algorithm. Second, IQL algorithm was applied to load balance the AGVs on the road network in raster maps of different sizes. Finally, the gathered data were used to simulate the AGV road network in the simulation software. The obtained data points were used to fit the MFD curve to complete the evaluation index of the road network. Table 2 shows experiment-related parameters.

Table 2 Related parameter settings

Parameters	Value
α	0.2
γ	0.9
ε	0.8
δ	0.5
β	0.95
c	0.15
n	3
η	0.75
u	50

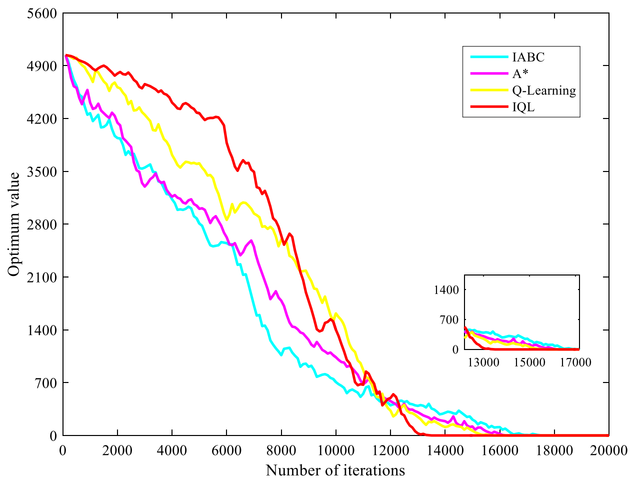


Fig. 6 Comparison of convergence speed

Comparison of convergence speed of improved algorithms

The task volume was set to 100u times and $u = 50$, three different algorithms were used to perform road network simulation experiments for multi-AGVs, respectively. An Improved Artificial Bee Colony (IABC), A*, Q-learning, and IQL algorithms were compared in terms of convergence speed to verify the effectiveness of the algorithms.

By comparing the four different algorithms, Fig. 6 shows that the IQL algorithm reaches convergence first. However, at the beginning of the algorithm, the IQL algorithm converges poorly at the beginning of the iterations. This is due to the fact that the IQL algorithm replaces the estimation of the historical Q value with the estimation of the current Q value. The number of AGVs in the road network is small, and the reward and penalty functions are more complex than the algebraic form. By partial sampling of the Q value to select the maximum value, the reward and penalty functions work together with the road network load, and the algorithm rapidly converges.

Table 3 Number of iterations for convergence of the algorithm

Algorithm	Number of iterations
IABC	17,504
A*	16,226
Q-Learning	15,731
IQL	13,418

The Q-learning algorithm uses a reward in the form of a surrogate value for the simulation. At the beginning, there is no "drag" between the reward and punishment functions, so it converges faster than the IQL algorithm at the beginning. As the number of AGVs in the road network increases, the Q-learning algorithm decreases slowly when faced with more and more data, especially when the traffic volume in the road network increases, and the convergence speed of the algorithm decreases rapidly.

A* algorithm belongs to one of the heuristic algorithms, which obtains the optimal route by finding the shortest path between the start point and the end point. IABC is an Improved Artificial Bee Colony algorithm. It can converge faster by introducing Euclidean distance to make the bee colony find the honey source location more accurately. Both algorithms have more iterations than reinforcement learning algorithms and converge slower than the previous two. The reason for this is that it does not allow road network load optimization in real time like the IQL algorithm.

The number of iterations for each algorithm is shown in Table 3. Through simulation and analysis, it is concluded that the IQL algorithm has the fastest convergence, Q-learning is the second fastest, and the A* and IABC algorithms have the slowest convergence.

IQL algorithm-based road network load balancing

In the AGV load balancing experiment on the road network, the starting and ending points of 5000 AGV tasks were kept constant. The road network load before and after load balancing was compared. The δ and β values were tested several times in the experiment to find the optimal value. When both variables were equal to zero, as was the situation prior to load balancing implementation.

Initially, let one of the AGVs ran on the road network, which would then update the load data for this ran. The later AGVs used the same method until all AGVs had updated the road network situation. The AGVs ran on the road network route so that the left and right sides could enter and exit at the same time. The AGVs were in operation status as shown in Fig. 7.

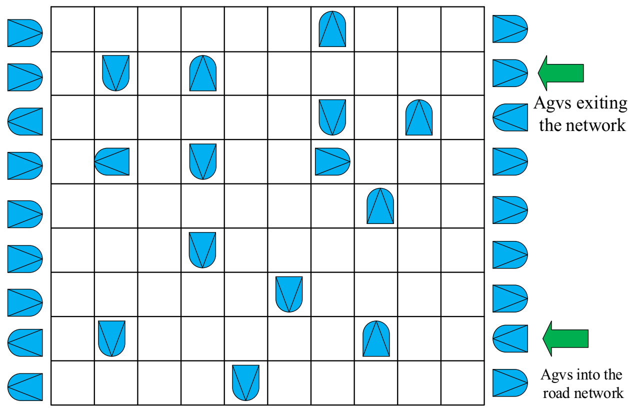


Fig. 7 AGV operation on the road network

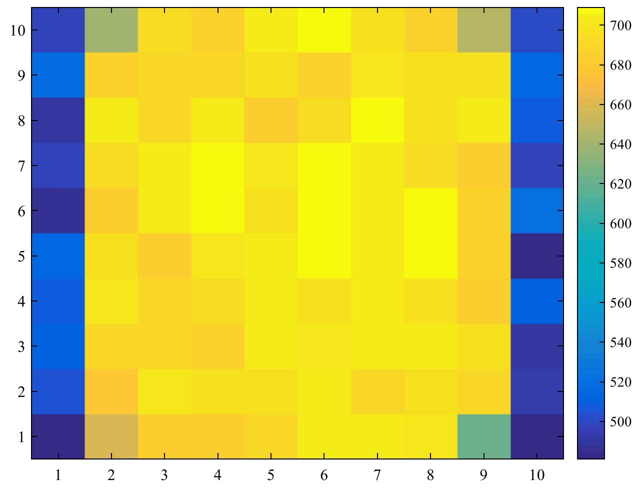


Fig. 9 10 × 10 road network load when $\delta = 0.5$, $\beta = 0.95$

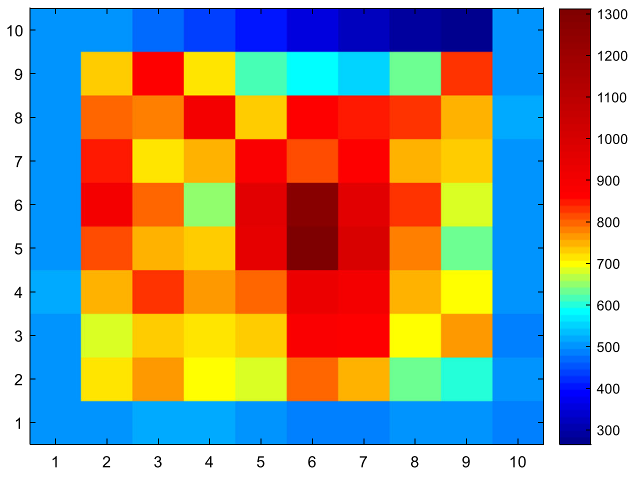


Fig. 8 10 × 10 road network load when $\delta = \beta = 0$

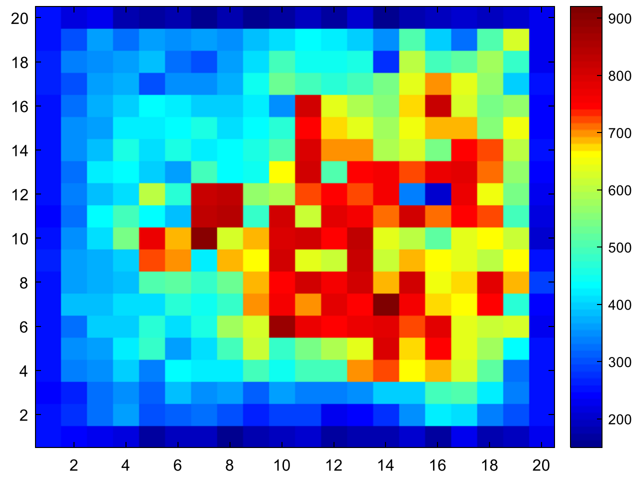


Fig. 10 20 × 20 road network load when $\delta = \beta = 0$

Road network simulation experiments were conducted on 10 × 10 road networks for AGVs before and after load balancing, respectively. The situation before and after load balancing is shown in Figs. 8 and 9.

The road network simulation experiments were conducted on AGVs before and after load balancing in 20 × 20 road networks. The situation before and after load balancing is shown in Figs. 10 and 11.

It could be seen that after improving the reward and punishment function, that IQL algorithm could effectively balance the road network load. In addition, it could also test the values of δ and β to find the optimal values.

In the 10 × 10 road network, the top area load reaches 1311. After lapping, the overall network load is around 700, it could be seen that the total load increases proportionally with the area load. In the 20 × 20 road network, the highest area load reached 910; after load balancing, the overall road

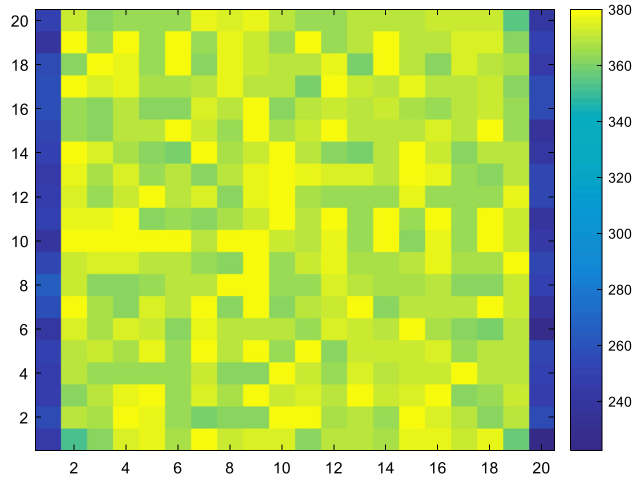


Fig. 11 20 × 20 road network load when $\delta = 0.5$, $\beta = 0.95$

Table 4 Standard deviation of road network load

Map size	Pre-load balancing	After load balancing	Decrease/%
5 × 5	445.63	230.61	48.25
10 × 10	261.08	96.57	63.01
15 × 15	150.58	40.35	73.20
20 × 20	120.75	28.34	76.53

Table 5 Average load of the road network

Map size	Pre-load balancing	After load balancing	Increase/%
5 × 5	1321.87	1329.31	0.56
10 × 10	708.64	710.18	0.22
15 × 15	435.39	450.73	3.5
20 × 20	350.76	375.92	7.1

network load was around 370. The ingress and egress load of the road network was relatively stable. Because the ingress and egress were determined by randomly generated tasks, which had little or no change in the load with and without considering load balancing.

With load balancing implemented, AGVs had the ability to navigate strategically through the road network to avoid high load areas. As a result, AGVs will choose paths with lower load levels rather than strictly adhering to the shortest path. To study the impact of the load reward and punishment function on the overall road network, this paper conducted several experiments on different size maps. Comparison of standard deviation variations of various road network models and their relationship with the average load on the road network was made. This is shown in Tables 4 and 5.

The analysis of Tables 4 and 5 reveals a noteworthy reduction in the standard deviation of the load following load balancing implementation, facilitated by the introduction of a reward and punishment system. However, it is worthwhile to note that the average load of the road network experienced only a marginal rise. So, the IQL algorithm could effectively balance the load of the road network and improve the operation efficiency of the road network.

MC model-based MFD fitting

VISSIM simulation software was used to create the road network map, as shown in Fig. 12. Set the road as a one-way road, where AGVs enter and leave the road network in both directions. Establish 5 × 5 10 × 10 15 × 15 and 20 × 20 road network models with vehicle detectors at road

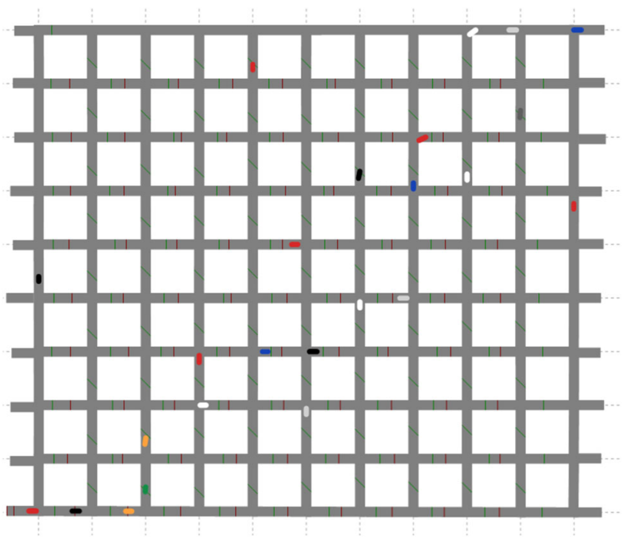


Fig. 12 10 × 10 road network AGV simulation operation diagram

intersections and entrances and exits, respectively, recording the number of AGVs driving into and out of the road network at time t. The simulation time was set to 100 min, with data being recorded at 1-*min* intervals, and many trials were conducted. The scatter correspondence between the AGVs driving out of the road network and the AGVs on the road network in the two-dimensional coordinate system was derived from the simulation data. The first-order function $y' = f'(x)$ was derived from the derivation of the equation, while the MFD critical point was obtained from solving the first-order function to complete the discrimination of the road network congestion status.

After establishing the 5 × 5 10 × 10 15 × 15 and 20 × 20 road network models, the simulations were carried out with VISSIM simulation software, respectively.

$$\begin{cases} y = 7.226 \times 10^{-7}x^3 - 2.329 \times 10^{-3}x^2 + 1.873x - 98.36 \\ R^2 = 0.9145 \end{cases} \tag{20}$$

$$\begin{cases} y = 7.986 \times 10^{-8}x^3 - 8.995 \times 10^{-4}x^2 + 1.422x - 116 \\ R^2 = 0.9045 \end{cases} \tag{21}$$

$$\begin{cases} y = 1.141 \times 10^{-7}x^3 - 9.911 \times 10^{-4}x^2 + 1.706x - 129.4 \\ R^2 = 0.9697 \end{cases} \tag{22}$$

$$\begin{cases} y = -2.281 \times 10^{-7}x^3 + 5.066 \times 10^{-4}x^2 + 0.226x + 59.24 \\ R^2 = 0.9419 \end{cases} \tag{23}$$

The MFD is shown in Figs. 13, 14, 15, and 16: the horizontal coordinates indicate vehicles on the road network and the vertical coordinates indicate vehicles driving out of the road network. The critical point was in the middle forward area

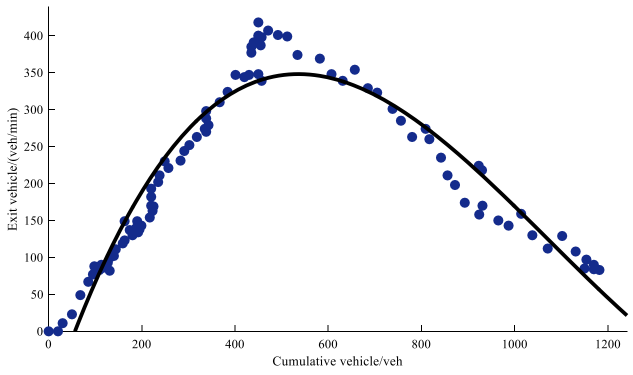


Fig. 13 MFD fitting of 5 × 5 road networks

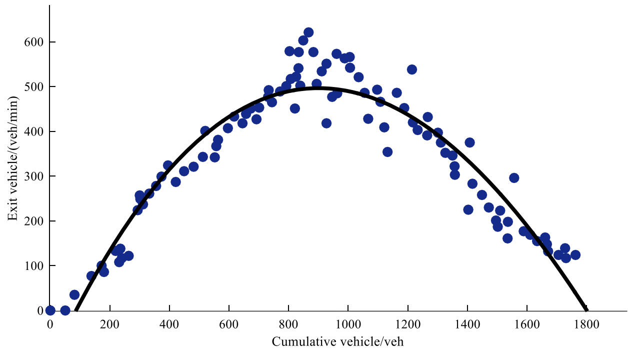


Fig. 14 MFD fitting of 10 × 10 road networks

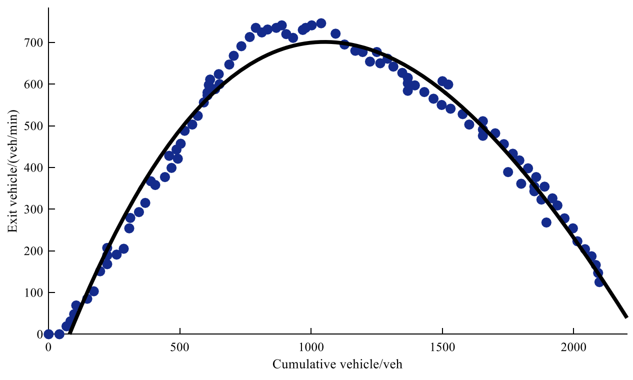


Fig. 15 MFD fitting of 15 × 15 road networks

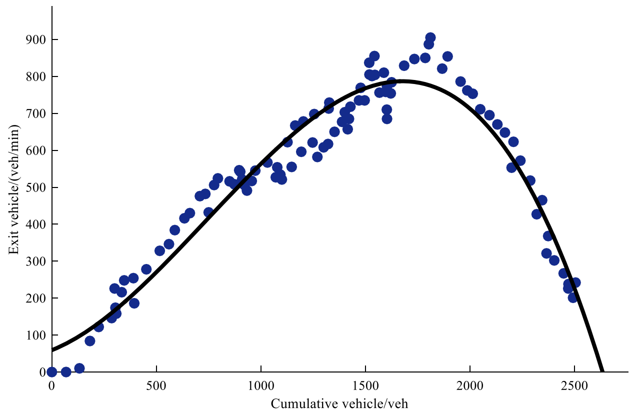


Fig. 16 MFD fitting of 20 × 20 road networks

of the 5 × 5 MFD, since the small road network map causes the AGV to reach the congestion very quickly after entering a part. 10 × 10 and 15 × 15 MFDs had critical points located between the steady flow and unsteady flow. It was shown that the vehicles of AGVs in the two road networks were almost symmetrically distributed on both sides of the critical point. In the 20 × 20 MFD, the critical point was backward compared to the 10 × 10 road network, whereas the left side of the critical point of the 20 × 20 MFD rose more slowly, indicating that the road network was in free and steady flow most of the time. At the same time, the 20 × 20 road network could accommodate more vehicles and the critical point would be generated at a relatively later time. The final simulation results of all four road networks successfully fitted the MFD curves.

The fitted curve equation of MFD is shown in Eqs. 20, 21, 22, and 23: considering the complexity of calculating higher order functions, after experimentally observing that the higher order functions and cubic functions did not change much in the adapted curve, this paper fitted the MFD curve with cubic functions. R^2 indicates the affect or bad degree of the fitted result. The closer R^2 was to 1, the better the adjusted result. In the four road network map experiments, the 15 × 15 road network was the most accurate fit. This indicates that the AGVs entering and exiting the road network can reach a balance when given the same number of AGVs. The 10 × 10 road network was a poor fit compared to the first three road networks. However, the fit was within the experimental effect expectation.

The assessment of the road network congestion situation is determined once the curve equation had been fitted. The first-order derivative of the curve equation of the two road networks was obtained, the first-order function $y' = f'(x)$ was obtained so that $y' = f'(x) = 0.15$ and $y' = f'(x) = -0.15$, the critical points of each road network state were identified as shown in Table 6.

Solving the first-order derivative function determined the left and right critical points of the two road networks, while the intermediate critical points were determined using the third-order function to find the extreme. The critical points in the 5 × 5 network were $(x_{l1}, x_m, x_{r2}) = (474, 535, 604)$. The critical points in the 10 × 10 network were $(x_{l1}, x_m, x_{r2}) = (790, 897, 1010)$. The critical points in the 15 × 15 network were $(x_{l1}, x_m, x_{r2}) = (936, 1051, 1175)$. The critical points in the 20 × 20 network were $(x_{l1}, x_m, x_{r2}) = (1552,$

Table 6 MFD critical point division

Map size	First derivative function $y' = f'(x)$	$y' = f'(x) = 0.15$ x_{l1}	$y' = f'(x) = 0.15$ x_{r1}	$y' = f'(x) = -0.15$ x_{l2}	$y' = f'(x) = -0.15$ x_{r2}	$y' = f'(x) = 0$ Critical point x_m
5 × 5	$y' = 2.1678 \times 10^{-6}x^2 - 4.658 \times 10^{-3}x + 1.873$	474	1673	1544	604	535
10 × 10	$y' = 2.3958 \times 10^{-7}x^2 - 1.799 \times 10^{-3}x + 1.422$	790	6718	6499	1010	897
15 × 15	$y' = 3.423 \times 10^{-7}x^2 - 1.9822 \times 10^{-3}x + 1.706$	936	4854	4616	1175	1051
20 × 20	$y' = 6.843 \times 10^{-7}x^2 + 1.0132 \times 10^{-3}x + 0.226$	1552	− 71	− 407	1787	1678

Table 7 Range around the critical point

Map size	Nearby range x_l $(x_l - x', x_l + x')$	Nearby range x_m $(x_m - x', x_m + x')$	Nearby rangex_r $(x_r - x', x_r + x')$
5 × 5	(455, 494)	(516, 555)	(585, 624)
10 × 10	(757, 823)	(864, 930)	(977, 1043)
15 × 15	(900, 972)	(1015, 1087)	(1139, 1211)
20 × 20	(1517, 1587)	(1643, 1713)	(1752, 1822)

1678, 1787). At this point, the three critical points divided the MFD into four regions: free flow, steady flow, unsteady flow, and congested flow. The nearby range of critical points could be judged according to the curvature c. According to MFD experience, the fluctuation value $x' = c\,(x_r - x_l)$. From this, the range near the three critical points could be determined, as shown in Table 7 for the range near the critical points.

The state of the road network could be discriminated by the number of vehicles on the road network, by improving Eq. (19) and extending the intermediate flows into stable and unstable flows. The congestion level M of the 10 × 10 road network and 20 × 20 road network at time t was judged as shown in Eqs. 24, 25, 26, and 27, respectively.

$$M = \begin{cases} 0 \le n_t < 474 & \text{Free flow} \\ 474 \le n_t < 535 & \text{Steady flow} \\ 535 \le n_t < 604 & \text{Unstable flow} \\ n_t \ge 604 & \text{Congestion flow} \end{cases} \tag{24}$$

$$M = \begin{cases} 0 \le n_t < 790 & \text{Free flow} \\ 790 \le n_t < 897 & \text{Steady flow} \\ 897 \le n_t < 1010 & \text{Unstable flow} \\ n_t \ge 1010 & \text{Congestion flow} \end{cases} \tag{25}$$

$$M = \begin{cases} 0 \le n_t < 936 & \text{Free flow} \\ 936 \le n_t < 1051 & \text{Steady flow} \\ 1051 \le n_t < 1175 & \text{Unstable flow} \\ n_t \ge 1175 & \text{Congestion flow} \end{cases} \tag{26}$$

$$M = \begin{cases} 0 \le n_t < 1552 & \text{Free flow} \\ 1552 \le n_t < 1678 & \text{Steady flow} \\ 1678 \le n_t < 1787 & \text{Unstable flow} \\ n_t \ge 1787 & \text{Congestion flow} \end{cases} \tag{27}$$

When the vehicles on the road network were outside the range near the critical point, the road network status was directly assessed. Based on the MC model, the judgment method was used when the vehicles were near the critical point. After obtaining vehicle data on the road network, the state of the road network in the current time period was judged based on Eq. (19).

Conclusion

In this paper, the algorithm was further enhanced by improving the traditional Q-learning algorithm in terms of the number of iterations and the reward and punishment functions. Load balancing experiments were conducted on different road networks. It was proved that the IQL algorithm could effectively balance road network load and improve AGV operation efficiency. To judge the operation status of the road network more accurately, MFD was used to judge the operation status of the road network.

The fitted MFD and curve equations were derived by fitting the MFD curve based on the MC model. In addition, we conducted experiments on different road networks. The equations were derived to find the critical points of the MFD. Three critical points were found to divide the MFD into four regions. Using the established MC model, AGVs within the critical points were discriminated, while those outside the critical points were discriminated directly.

In summary, the IQL algorithm effectively equalizes the road network load, as well as accurately determines the road network operation status using MFD and curve equations, which demonstrates the superiority of the improved method.

Author contributions We declare that this manuscript entitled “Load Balancing of Multi-AGV Road Network Based on Improved Q-learning Algorithm and Macroscopic Fundamental Diagram” is original, has not been published before and is not currently being considered for publication elsewhere. We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.

Funding This study was supported by Jilin Province Major Science and Technology Special Project "Research on Repeat Positioning Accuracy Technology of AGV", 20210301028GX, Hui Li

Data availability The data that support the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest We declare that we have no known competing financial interests or personal relationships or organizations that could have appeared to influence the work reported in this paper.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Tao H, Cheng L, Qiu J et al (2022) Few shot cross equipment fault diagnosis method based on parameter optimization and feature metric. Meas Sci Technol 33(11):115005
2. Löffler M, Boysen N, Schneider M (2022) Picker routing in AGV-assisted order picking systems. INFORMS J Comput 34(1):440–462. https://doi.org/10.1287/ijoc.2021.1060
3. Pan F, Sun Q (2019) A traffic control strategy of the heavy-duty AGVS in a square topology. IEEE International Conference on Mechatronics and Automation (ICMA). IEEE, pp 263–268
4. Chen Y, Jiang Z (2022) Multi-AGVs scheduling with vehicle conflict consideration in ship outfitting items warehouse. J Shanghai Jiaotong Univ (Science). https://doi.org/10.1007/s12204-022-2561-z
5. Moser BR (2022) Machine learning and digital twin-sed path planning for AGVs at automated container terminals. Adv Transdisciplinary Eng. https://doi.org/10.3233/ATDE220672
6. Zheng T, Xu Y, Zheng D (2019) AGV path planning based on improved A-star algorithm. IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC). IEEE, pp 1534–1538
7. Chen C, Tiong LK, Chen IM (2019) Using a genetic algorithm to schedule the space-constrained AGV-based prefabricated bathroom units manufacturing system. Int J Prod Res 57(10):3003–3019. https://doi.org/10.1080/00207543.2018.1521532
8. Chen C, Hu ZH, Wang L (2021) Scheduling of AGVs in automated container terminal based on the deep deterministic policy gradient (DDPG) using the convolutional neural network (CNN). J Marine Sci Eng 9(12):1439. https://doi.org/10.3390/jmse9121439
9. Hu H, Jia X, He Q et al (2020) Deep reinforcement learning based AGVs real-time scheduling with mixed rule for flexible shop floor in industry 40. Comput Ind Eng 149:106749. https://doi.org/10.1016/j.cie.2020.106749
10. Wei Q, Lewis FL, Sun Q et al (2016) Discrete-time deterministic $ Q $-learning: a novel convergence analysis. IEEE Trans Cybern 47(5):1224–1237. https://doi.org/10.1109/TCYB.2016.2542923
11. Devraj AM, Meyn SP (2017) Fastest convergence for Q-learning. arXiv preprint arXiv:1707.03770. https://doi.org/10.48550/arXiv.1707.03770. Accessed 23 Mar 2018
12. Low ES, Ong P, Cheah KC (2019) Solving the optimal path planning of a mobile robot using improved Q-learning. Robot Auton Syst 115:143–161. https://doi.org/10.1016/j.robot.2019.02.013
13. Yu N, Li T, Wang B (2021) Multi-load AGVs scheduling algorithm in automated sorting warehouse. 14th International Symposium on Computational Intelligence and Design (ISCID). IEEE. 126–129. https://doi.org/10.1109/ISCID52796.2021.00037
14. Roh BS, Han MH, Ham JH et al (2020) Q-LBR: Q-learning based load balancing routing for UAV-assisted VANET. Sensors 20(19):5685. https://doi.org/10.3390/s20195685
15. Sethi V, Pal S (2023) FedDOVe: a federated deep q-learning-based offloading for vehicular fog computing. Futur Gener Comput Syst 141:96–105. https://doi.org/10.1016/j.future.2022.11.012
16. Chen J, Xing H, Xiao Z et al (2021) A DRL agent for jointly optimizing computation offloading and resource allocation in MEC. IEEE Internet Things J 8(24):17508–17524. https://doi.org/10.1109/JIOT.2021.3081694
17. Xiao Z, et al. (2023) Deep Contrastive Representation Learning With Self-Distillation. In: IEEE transactions on emerging topics in computational intelligence. https://doi.org/10.1109/tetci.2023.3304948
18. Song F, Xing H, Wang X, et al. (2022) Evolutionary multi-objective reinforcement learning based trajectory control and task offloading in UAV-assisted mobile edge computing. arXiv e-prints. DOI:https://doi.org/10.48550/arXiv.2202.12028
19. Ji Y, Daamen W, Hoogendoorn S et al (2010) Investigating the shape of the macroscopic fundamental diagram using simulation data. Transp Res Rec 2161(1):40–48. https://doi.org/10.3141/2161-05
20. Ambühl L, Loder A, Bliemer MCJ et al (2020) A functional form with a physical meaning for the macroscopic fundamental diagram. Transp Res Part B: Methodol 137:119–132. https://doi.org/10.1016/j.trb.2018.10.013
21. Shen L, Tao H, Ni Y et al (2023) Improved YOLOv3 model with feature map cropping for multi-scale road object detection. Meas Sci Technol. https://doi.org/10.1088/1361-6501/acb075
22. Geroliminis N, Zheng N, Ampountolas K (2014) A three-dimensional macroscopic fundamental diagram for mixed bi-modal urban networks. Transp Res Part C Emerg Technol 42:168–181. https://doi.org/10.1016/j.trc.2014.03.004
23. Gayah VV, Gao XS, Nagle AS (2014) On the impacts of locally adaptive signal control on urban network stability and the macroscopic fundamental diagram. Transp Res Part B Methodol 70:255–268. https://doi.org/10.1016/j.trb.2014.09.010
24. Loder A, Dakic I, Bressan L et al (2019) Capturing network properties with a functional form for the multi-modal macroscopic fundamental diagram. Transp Res Part B Methodol 129:1–19. https://doi.org/10.1016/j.trb.2019.09.004
25. Halakoo M, Yang H, Abdulsattar H (2023) Heterogeneity aware emission macroscopic fundamental diagram (e-MFD). Sustainability 15(2):1653. https://doi.org/10.3390/su15021653
26. He F, Yan X, Liu Y et al (2016) A traffic congestion assessment method for urban road networks based on speed performance index. Proc Eng 137:425–433. https://doi.org/10.1016/j.proeng.2016.01.277

27. Ambühl L, Loder A, Menendez M, et al. (2017) Empirical macroscopic fundamental diagrams: new insights from loop detector and floating car data. TRB 96th Annual Meeting Compendium of Papers. Transportation Research Board, pp 17–03331
28. Zhao X, Liu Y, Wang Y (2016) Automatic extraction and construction algorithm of overpass from raster maps. Pacific Rim conference on multimedia. Springer, Cham, pp 479–489. https://doi.org/10.1007/978-3-319-48896-7_47
29. Oh J, Hessel M, Czarnecki WM et al (2020) Discovering reinforcement learning algorithms. Adv Neural Inform Process Syst. 33:1060–1070. https://doi.org/10.48550/arXiv.2007.08794
30. Puterman ML (1990) Markov decision processes. Handbooks Oper Res Manag Sci 2:331–434. https://doi.org/10.1002/9780470316887
31. Liu J, Qi W, Lu X (2017) Multi-step reinforcement learning algorithm of mobile robot path planning based on virtual potential field. International Conference of Pioneering Computer Scientists. Engineers and Educators. Springer, Singapore, pp 528–538
32. Tao H, Qiu J, Chen Y et al (2023) Unsupervised cross-domain rolling bearing fault diagnosis based on time-frequency information fusion. J Franklin Inst 360(2):1454–1477. https://doi.org/10.1016/j.jfranklin.2022.11.004
33. Shang Y, Liu F, Qin P et al (2023) Research on path planning of autonomous vehicle based on RRT algorithm of Q-learning and obstacle distribution. Eng Comput. https://doi.org/10.1108/EC-11-2022-0672
34. Song X, Wu C, Stojanovic V et al (2023) 1 bit encoding–decoding-based event-triggered fixed-time adaptive control for unmanned surface vehicle with guaranteed tracking performance. Control Eng Pract 135:105513. https://doi.org/10.1016/j.conengprac.2023.105513
35. Hu G, Lu W, Whalin RW et al (2021) Analytical approximation for macroscopic fundamental diagram of urban corridor with mixed human and connected and autonomous traffic [J]. IET Intel Transport Syst 15(2):261–272. https://doi.org/10.1049/itr2.12020
36. Qu X, Wang S, Zhang J (2015) On the fundamental diagram for freeway traffic: a novel calibration approach for single-regime models. Transp Res Part B Methodol 73:91–102. https://doi.org/10.1016/j.trb.2015.01.001
37. Ji K, Tang J, Li M et al (2023) Distributed traffic control based on road network partitioning using normalization algorithm. Sustainability 15(14):11378. https://doi.org/10.3390/su151411378
38. Ching WK, Ng MK (2006) Markov chains. Models, algorithms and applications. Kluwer Academic Publishers, Boston. https://doi.org/10.1007/0-387-29337-X
39. Sun Z, Wang G, Jin L et al (2022) Noise-suppressing zeroing neural network for online solving time-varying matrix square roots problems: a control-theoretic approach. Expert Syst Appl 192:116272. https://doi.org/10.1016/j.eswa.2021.116272
40. Bellec E, Doudard C, Facchinetti ML et al (2023) Loading classification proposal for fatigue design of automotive chassis-parts: a relevant process for variable amplitude and multi-input load cases. Int J Fatigue 166:107284. https://doi.org/10.1016/j.ijfatigue.2022.107284

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.