Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 5, No. 1, 2023

Large-scale Passenger Behavior Learning and Prediction in Airport Terminals based on Multi-Agent Reinforcement Learning

Yue Li *, Guokang Gao

School of Software Engineering, Chengdu University of Information Technology, Chengdu, Sichuan, 610225, China
*** Corresponding author:** Yue Li (Email: 13880226883@163.com)

Abstract: For the problem of predicting passenger flow in airport terminals, multi-agent reinforcement learning is applied to airport terminals simulation. Multi-Agent Reinforcement Learning based on Group Shared Policy with Mean-field and Intrinsic Rewards (GQ-MFI) is proposed to predict passenger behavior in order to simulate the distribution of flow in different areas of the terminal at different time periods. Independent learning of multi-agent may lead to environmental instability and long convergence time. To improve the adaptability of agents in non-stationary environments and accelerate learning time, a multi-agent grouping learning strategy is proposed. Clustering is used to group multi-agent, and a shared Q-table is set within each group to improve the learning efficiency of multi-agent. Meanwhile, in order to simplify the interaction information among the agent after grouping, the idea of average field is used to transmit partial global information among the agent within the group. Intrinsic rewards are added to make the agent closer to human cognition and behavioral patterns. By conducting the airport terminal simulations using Anylogic, the experimental results show that the training speed of this algorithm is 17% higher than that of Q-learning algorithm, and it achieves good prediction accuracy in predicting the number of security check passengers with a time scale of 10 minutes.

Keywords: Reinforcement Learning; Multi-agent; Airport Passengers; Anylogic.

1. Introduction

With the development of the times, China has shifted towards a stage of high-quality development. The economy has been consistently strong, and the size and proportion of the middle-income group have increased. The aviation market has tremendous potential, and the development of civil aviation is still in a growth phase. People have higher expectations for the convenience, fairness, diversity, and quality of aviation services. Civil aviation needs to further improve its capacity, enhance service quality, and strengthen passenger satisfaction. To enhance service quality, it is crucial to accurately grasp the passenger flow in airport terminals. Only by adopting scientific and appropriate methods to accurately understand the flow of airport terminals can we avoid large-scale queues and chaos caused by excessive passenger flow. This, in turn, helps passengers choose the right travel time, reduce waiting time in queues, improve the phenomenon of long queues in airport terminals, and enhance the level of service in airport terminals.

Multi-agent reinforcement learning is an important branch of reinforcement learning that extends to multi-agent systems, characterized by autonomy, coordination, and distribution [1]. In a multi-agent system, each agent collects data by receiving environmental information, selects actions to interact with the environment, and learns to improve its strategies based on the rewards obtained, thereby obtaining the optimal strategy in that environment.

Existing methods for predicting airport terminal passenger flow rely on a large amount of statistical data, and some data is difficult to obtain in reality. Additionally, current research has focused more on the temporal distribution of passengers in airport terminals, overlooking the spatial distribution. Therefore, this study focuses on the behavioral actions of passengers in airport terminals. Utilizing the Anylogic software, which is based on agent-based modeling, we simulate the internal layout of airport terminals and the boarding process of passengers. By combining multi-agent reinforcement learning, we predict the behavioral actions of passengers in airport terminals, thereby simulating the flow in different areas of the terminal at different time periods. The primary challenges encountered in this study include:

1) Addressing the issue of environmental instability by grouping agents with similar state and action spaces and conducting group training to enhance their adaptability to non-stationary environments;
2) Overcoming the limitation of individual independent learning results by implementing shared Q-tables within each group, facilitating experience sharing and information exchange among agents, enabling them to learn and optimize their strategies based on the experiences of other agents;
3) Simplifying the complex interaction among agents by introducing the concept of average field, reducing interaction complexity and learning costs, and achieving a more stable algorithm learning process, leading to the acquisition of superior cooperative strategies by the agents;
4) Tackling the problem of sparse rewards by incorporating intrinsic rewards, which work in conjunction with external rewards defined for the task, helping agents achieve goals in the complex airport terminal scenarios.

2. Related Work

With the increasing global air passenger volume and the growing demands for airport development, artificial intelligence (AI) technology has gradually become a crucial support for airport terminals. There is a growing number of researchers both domestically and internationally focusing on airport terminals, and the research scope is expanding. There have been abundant

research findings in areas such as flow prediction, service facility planning, service process optimization, and airport terminals resource optimization. Xia Feng *et al.* [2] reconstructed the phase space of security check passenger flow time series and used the Wolf method to determine the chaotic nature of the time series, followed by BP neural network prediction for the chaotic time series. Xinglong Wang *et al.* [3] conducted historical data statistics on the operations of a target airport under different weather conditions, constructed a similarity matrix, and established a grey cluster model to select similar days at the airport. Then, they employed a particle swarm optimization-based support vector machine method to train the selected similar day samples and predict airport traffic flow. Xiang Zhong *et al.* [4] considered four factors, including time period, visual distance of the area, flight quantity, and check-in passenger flow, that affect security check passenger flow, and used the BP neural network algorithm to establish a prediction model for airport security check passenger flow. Álvaro *et al.* [5] proposed a queuing behavior model based on real flight data, which can be used for real-time prediction of waiting times in queues.

Multi-agent reinforcement learning applies reinforcement learning techniques and game theory to multi-agent systems, enabling multiple agents to interact and make decisions to complete complex tasks in higher-dimensional and dynamic real-world scenarios. Scalability is currently a key focus. Chenghao Li *et al.* [6] introduced diversity into shared multi-agent reinforcement learning and proposed a regularization based on information theory to maximize the mutual information between the agent's identity and its trajectory, addressing the problem of similar behaviors resulting from sharing and limiting their coordination ability. Yang *et al.* [7] first proposed the use of mean-field to simplify interactions among agents, replacing all other agents within an individual's scope with a mean value. This algorithm reduces the multi-agent problem to a two-agent problem and effectively solves the non-stationarity and scale issues in multi-agent environments. Dianxi Shi *et al.* [8] addressed the credit assignment problem in heterogeneous multi-agent systems with different roles, which makes it difficult to learn effective cooperative strategies. They proposed an end-to-end cooperative adaptive reward method based on multi-agent reinforcement learning to guide agents in generating cooperative strategies based on the situation on the field. Andres *et al.* [9] presented a collaborative learning framework for heterogeneous agents in the same environment, combining intrinsic motivation with transfer learning to achieve more effective exploration and learning.

3. Problem Description

The goal of this paper is to predict the spatial behavior of travelers in an airport terminal, thus enabling a more accurate simulation of the density of people in different areas of the terminal. In order to simplify the description of the problem, time is discretized in the time dimension with equal-length time slices.

Based on the above, in a time slot t, the behavior of a single passenger can be described as: 1) Passenger arrives at the terminal and selects his activity according to the situation; 2) Passenger who go to the event location receive rewards, and complete the event after Δt ; 3) Passenger select the next activity starting from the previous activity location until boarding the flight. As shown in Figure 1.

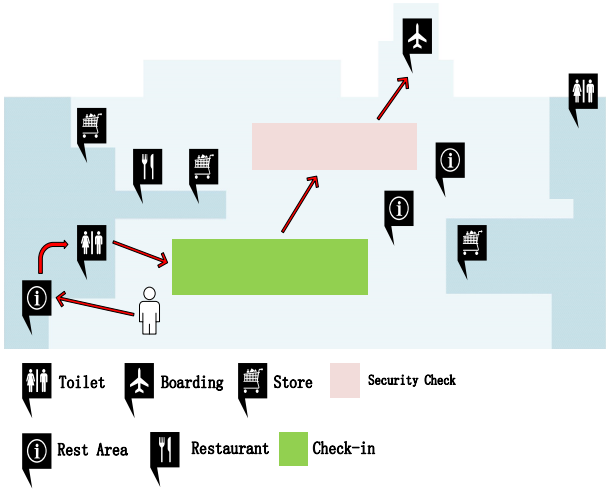

Fig 1. Individual passenger behavior at the airport terminal

Considering that in the real world, passengers' activities last for varying lengths of time, the span of actions is adapted from one time slice to multiple time slices. A Semi-Markov Decision Process (SMDP) is built from the perspective of a single passenger (local view), whereby each passenger is considered as an intelligence, each of which still follows the goal of reinforcement learning, which is to maximize the cumulative reward that can be obtained. Semi-Markov Decision can be represented by the tuple $\langle S, A, P, R \rangle$, where S denotes the state space of the intelligent, and A denotes the action space of the intelligent, and P is the state transfer probability, and P denotes the reward function.

Mapping to the real world, the relevant definitions are as follows:

1) State space S. The state of a passenger at time t is denoted by $s_t \in S$ denoted by three parts. The first part, $freedom$, is the difference between the passenger's departure time from the flight at time t. If $freedom$ is small, the passenger may forgo additional activities that he had planned, such as shopping or eating at a restaurant, and choose to go through the necessary check-in process to ensure that he can board the plane on time. If $freedom$ is larger, the passenger usually chooses some extra activities to pass the time according to his/her preference. The second part is the distance d_t between the geographic location of the passenger at time t and the nearest various service facilities. The third part, n_t, is the number of passengers queuing at the nearest various service facilities at time t. The environment is also one of the important factors affecting passenger behavior. The various service facilities in the terminal attract passengers, but the distance between passengers and the facilities, as well as the phenomenon of congestion and queuing at the facilities, will affect passengers' activity decisions [10]. For example, when travelers find the target security checkpoint travelers may first go to other service facilities nearby, and then go to the security checkpoint when the security checkpoint is no longer congested with queues. To summarize, the state can be represented by a ternary group $s_t := (freedom, d_t, n_t)$.

2) Action space A, the actions that can be executed by the agent. The process of passengers arriving at the airport until boarding the plane is completed in the terminal. Starting from reality, we consider the various facilities and services provided by the terminal building. The actions that passengers choose to perform in s_t are represented by a_t, and are set to six types: check-in, security check, shopping, rest, catering, and bathroom.Because of the consideration of discrete action

space, one-hot coding method is used to encode the six actions using 6-bit state registers, and each action has its own independent coding, the $A \in (100000, 010000, 001000, 000100, 000010, 000001)$.

3) State transfer probability P , the probability of the passenger's state transitioning from s_t to s_{t+1} after executing action a_t at time t.

4) Reward function R , the reward received by the agent after performing an action. $R(s_t, a_t)$ indicates that the passenger is in the state s_t performs an action a_t, when the state changes from s_t transitions to s_{t+1} the reward received by the agent.

This transforms the problem into a locally observable Semi-Markov problem with multi-agent.

4. Algorithm Design

4.1. Group Shared Policy

Q-learning is a value-based reinforcement learning algorithm that utilizes Q-functions to find optimal action-selection policies[11] . Its formula is:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[R(s_t, a_t) + \gamma \max Q(s_{t+1}, a) - Q(s_t, a_t)]$$

Among them, $Q(s_t, a_t)$denotes the value of taking action a in state s at time t, which is the estimated action value function; $\alpha(0 \le \alpha \le 1)$ denotes the learning rate, which determines the speed of updating; $R(s_t, a_t)$ denotes the immediate reward obtained after taking action a in the current state s; $\gamma(0 \le \gamma \le 1)$ denotes the discount factor, which measures the importance attached to the goal; $maxQ(s_{t+1}, a)$ denotes selecting the action with the highest value among all possible actions in state s_{t+1}. By continuously iterating and updating the Q-value, a converged Q-value function can be obtained to guide agents in making optimal decisions in the environment.

The action strategy of the agent is ε-greedy strategy, ε is a policy parameter, the agent randomly selects actions with probability ε, and with probability$1 - \varepsilon$select the action that will result in the max Q-value:

$$a_t = \begin{cases} \text{Random actions in A}, P(\varepsilon) \\ \text{argmax}\left(Q(s_t, a_t)\right), P(1-\varepsilon) \end{cases}$$

To address the problems raised by challenges 1) and 2) in the introduction, i.e., the environment of multi-intelligence reinforcement learning is non-stationary and the learning results are limited. In this paper, we propose a group shared policy based on Q-learning, as shown in Figure 2. In non-stationary environments, this policy enables the agent to obtain information about others from sharing, enabling the agent to better adapt to environmental instability and changes, and promoting learning and collaboration between the agent.

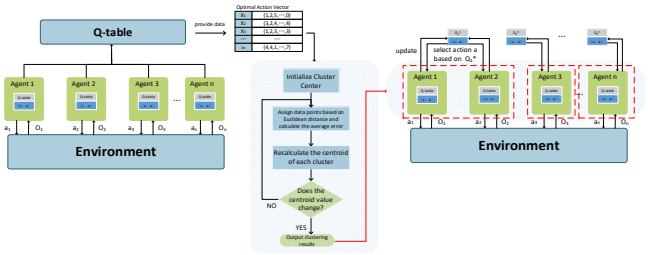

Fig 2. Group shared policy

Firstly, a round of training is conducted on the agent to obtain the set of optimal actions. The optimal actions are clustered by K-means, and the agent is divided into H clusters by minimizing the sum of error squares and SSE. The detailed definition is given below.

Definition 1. Optimal action x_j is the vector consisting of the actions that maximize the Q-value in each state of the agentj, denoted as follows:

$$x_j = \{\text{argmax}_1 Q(s, a), \text{argmax}_2 Q(s, a), \ldots, \text{argmax}_n Q(s, a)\}$$

Definition 2. Euclidean distance dist is the shortest distance in a straight line between two points in space. In n-dimensional space, given two optimal action vectors $x_1$and x_2, the Euclidean distance between x_1 and x_2 is denoted as:

$$dist\,(x_1, x_2) = \| X - Y \|_2 = \sqrt{\sum_{i=1}^{n} (x_{1n} - x_{2n})^2}$$

Definition 3. Cluster C_h is a collection of data points, where data points within the same cluster have similar characteristics.

Definition 4. Center of mass c_h is the center point of a cluster, which can be regarded as the average position of the data points within the cluster, i.e., the average of the eigenvalues of all the data points within the cluster in each dimension, expressed as follows:

$$c_h = \frac{1}{|C_h|} \sum_{x_j \in C_h} x_j$$

Where $|C_h|$ denotes the cluster C_h the number of data points in the x_j denotes the number of data points in the cluster C_h for each data point in the cluster.

Definition 5. The sum of squared errors (SSE), the sum of the squared values of the Euclidean distances from each data point to the center of mass within its cluster, is expressed as follows:

$$SSE = \sum_{h=1}^{H'} \sum dist\,(x_j, c_h)^2$$

A smaller value of SSE indicates that the data points are closer to their center of mass.

Algorithm 1. Grouping algorithm

Input: number of clusters H, optimal action xj

Output: grouping results

1 Randomly initialize H-cluster cluster centers c1 ,... , cH
2 Repeat:
3 for j = 1: N do
4 for h = 1: H do
5 $h_j = \arg\min_h \|x_j - c_h\|_2^2$
6 end for
7 end for
8 for h = 1: H do
9 Updating the center of mass $c_h = \frac{1}{|C_h|} \sum_{x_j \in C_h} x_j$
10 end for
11 Until: No change in c_h, return clustering results

The optimal number of clusters H in Algorithm 1 is difficult to obtain, and here the elbow method is used to determine the H value. The basic idea is to select different H values to perform K-means clustering on the dataset. When H is less than the true number of clusters in the data, every unit increase in H will significantly increase the degree of aggregation of each cluster, and the decrease in SSE will be significant; When H approaches the true number of clusters in the data, the degree of aggregation return obtained by increasing K will quickly decrease, and the decrease in SSE

will also decrease; As K continues to increase, the change in SSE tends to be gradual, and the inflection point of SSE value change, i.e. the corresponding H value at the elbow, is considered the optimal number of clusters for the data.

After obtaining the cluster labels for each optimal action from Algorithm 1, the agent is divided into H groups. Set up a shared table Q^* for each group, using a continuous sharing method. Every time there is a state transition, the intelligent agent will update the learned Q-value to Q^*. The agent j updates in the state as follows:

$$Q^*_{c_h}(s_t, a_t) \leftarrow Q^*_{c_h}(s_t, a_t) + \mathrm{Q}^{\mathrm{j}}(s_t, a_t) \qquad \mathrm{j} \in \mathrm{c_h}$$

The purpose of using cumulative sum is to gather the estimated values of multi-agent on the same state action pair in the Q^*table, thereby generating more comprehensive and comprehensive information. This can avoid inaccurate estimation caused by the limitations of a single agent, and improve the agent's ability to understand the environment.

4.2. Mean-field Theory

In 3.1, the agent with similar characteristics is grouped together and their Q-values are shared. However, the interaction between agents is still quite complex, as each agent must consider the action information of others in the group and adjust their decisions based on this information, which is a complex and computationally intensive process. Therefore, drawing on the Mean-field by Yang *et al.* [6], the interaction is further simplified, transforming the joint actions of agents into the average actions formed by agents through the mean field, and serving as parameters for the value function update function in Q-learning. As shown in Figure 3, each agent is represented as a point in the airport terminal. Taking the black dot as an example, this point is only affected by the average effect of the red dots in its neighboring (orange area). The interaction between multi-agent is effectively transformed into the interaction between two agents, simplifying the scale of the interaction.

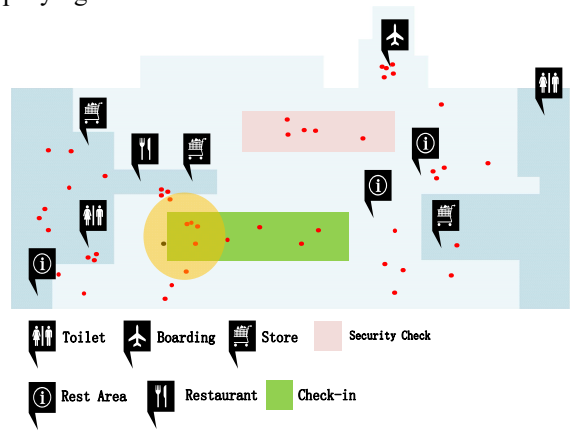

Fig 3. Mean-field theory in the airport terminal

Therefore, redefine the Q function of Q-learning:

$$Q^j(s,a) = \frac{1}{N^j} \sum_{k \in N(j)} Q^j(s, a^j, a^k)$$

N(j) denotes the set of the neighboring agent of agent j; $\mathrm{N^j} = |\mathrm{N(j)}|$ denotes the number of the neighboring agent of the agent j. The neighboring agent of agent j is defined as an agent in the same group as agent j that has arrived at the airport and has not yet boarded the plane.

The action $a^k$of each neighboring agent can be calculated by summing and averaging the action codes of the neighboring agent of agent j:

$$\begin{cases} a^k = \bar{a}^j + \delta a^{j,k} \\ \bar{a}^j = \dfrac{1}{N^j} \sum_k a^k \end{cases}$$

the $\delta\, \mathrm{a^{j,k}}$ is the difference between the encoding of the action of the neighbouring agent of agent j and the encoding of their average action.

The above two formulas can be derived:

$$\begin{aligned} & Q^j(s,\mathrm{a}) = \tfrac{1}{N^j} \textstyle\sum_k Q^j(s, a^j, a^k) \\ & = \tfrac{1}{N^j} \textstyle\sum_k \left[Q^j(s, a^j, \bar{a}^j) + \nabla_{a^j} Q^j(s, a^j, \bar{a}^j) \cdot \delta a^{j,k} + \tfrac{1}{2} \delta a^{j,k} \cdot \nabla^2_{\tilde{a}^{j,k}} Q^j(s, a^j, \tilde{a}^{j,k}) \cdot \delta a^{j,k} \right] \\ & = Q^j(s, a^j, \bar{a}^j) + \nabla_{a^j} Q^j(s, a^j, \bar{a}^j) \cdot \left[\tfrac{1}{N^j} \textstyle\sum_k \delta a^{j,k} \right] + \tfrac{1}{2N^j} \textstyle\sum_k \left[\delta a^{j,k} \cdot \nabla^2_{\tilde{a}^{j,k}} Q^j(s, a^j, \tilde{a}^{j,k}) \cdot \delta a^{j,k} \right] \\ & = Q^j(s, a^j, \bar{a}^j) + \tfrac{1}{2N^j} \textstyle\sum_k R^j_{s,a^j}(a^k) \approx Q^j(s, a^j, \bar{a}^j) \end{aligned}$$

Therefore, the update formula for the value function $Q(s_t, a_t)$ of agent j at time t is:

$$\begin{aligned} Q^j_t(s,a) &= Q^j_t(s, a^j, \bar{a}^j) \\ &= (1-\alpha) Q^j_t(s, a^j, \bar{a}^j) \\ &+ \alpha \left[\mathrm{R}(\mathrm{s_t}, \mathrm{a_t}) + \gamma * max Q^j_{t+1}(s, \mathrm{a^j}, \bar{a}^j) \right] \end{aligned}$$

4.3. Intrinsic Rewards

Traditional reinforcement learning usually only considers its extrinsic motivation, that is, designing a specialized external reward function for a specific target task to guide agents in learning behavioral strategies, in order to maximize long-term cumulative rewards [12]. But when in an environment with sparse rewards, the agent is unable to achieve the expected behavior of the target task. In order to enhance the autonomy of intelligent agents in learning in complex airport terminal scenarios, to assist in improving the learning efficiency of specific tasks in reinforcement learning [13], and to endow intelligent agents with behavior patterns closer to those of humans in the real world, intrinsic motivation is considered in the design of reward functions.

Intrinsic motivation originates from the concept of self-motivation and psychology, and its discovery and discussion were first seen in Harlow's 1950 explanation of the phenomenon of macaques solving mechanical puzzles for hours without any external reward conditions. It originates from individual internal needs, interests, personal values, sense of achievement, etc. [15].

The number of people queuing and distance can be seen as intrinsic motivations that affect passengers' behavior choices at the terminal. Passengers tend to choose service facilities with fewer queues to shorten waiting time and improve their efficiency and comfort. Distance refers to the distance that passengers need to walk, and passengers will consider the distance and time consumption of walking, and then choose a service facility closer to their destination to improve convenience and save time and energy. By mapping the perceived queue size and distance status of passengers into intrinsic reward signals, the agent is incentivized to prioritize options with fewer queue sizes and closer distances when choosing behavior, thereby promoting the agent's decision-making in the airport terminal environment to be closer to the behavior of real-world terminal passengers.

Therefore, the reward function for Q-learning is redefined, with different rewards given according to the different destinations of passengers, consisting of two parts: external rewards $r_{\mathrm{ext}}(\mathrm{s_t}, \mathrm{a_t})$ and internal rewards $r_{\mathrm{int}}(\mathrm{s_t}, \mathrm{a_t})$:

$$R(s_t, a_t) = \mathrm{r}_{ext}(s_t, a_t) + \mathrm{r}_{int}(s_t, a_t)$$

External rewards $r_{\mathrm{ext}}(\mathrm{s_t}, \mathrm{a_t})$ are sparse rewards that passengers receive when go to activity location i. As long as

the pedestrian density x_{obj} of the activity location i meets the given true value x_{real}, it returns 1. Otherwise, it returns 0.

$$r_{\text{ext}}(s_t, a_t) = \begin{cases} 1, \text{ if } x_{\text{real}} = x_{\text{obj}} \\ 0, \text{ else} \end{cases}$$

Intrinsic rewards $r_{\text{int}}(s_t, a_t)$ consists of two parts:

$$r_{\text{int}}(s_t, a_t) = -[r_n(s_t, a_t) + r_d(s_t, a_t)]$$

Where $r_n(s_t, a_t)$ is the number of people in the queue at activity location i, and $r_d(s_t, a_t)$ is the distance passengers to go to activity location i.

4.4. GQ-MFI

Algorithm 2. GQ-MFI

Input: collection of agents, output of Algorithm 1

Output: collection of agents

1 Load the airport terminal simulation environment, initialize the agent parameters

2 Set up and initialize a shared Q^* table for each group and initialize the Q-value for each agent

3 Run the airport terminal simulation environment and obtain observations

4 Learning process of the agent

for agent i $(i \leq n)$ do

state ← computes the state of the agent

actions ← Optional actions of the agent obtained based on the state

$N_{(j)}$ ← Get the neighboring agent of the agent

a^k ← calculates the average action of the neighboring agent

with probability ε : action=random(actions)

else

if agent i has updated $Q^*(state)$: action = $\max Q^*(state, actions)$

else: action = $\max Q^*(state, actions, a^k)$

select actions to act on the environment and collect rewards

feed reward back to the agent: $reward = r_{ext}(s_t, a_t) + r_{int}(s_t, a_t)$

Update the Q_i and Q^*_{ch}

end for

5. Experiment and Analysis

5.1. Experimental Setup

This experiment uses Anylogic software as the terminal simulation platform to achieve passenger behavior simulation. Anylogic is based on three main modeling methods: discrete event, agent based, and system dynamics, and can provide modeling of the internal layout of the terminal and passenger boarding process required for experiments. In order to ensure the authenticity of the experiment, the layout of the T2 terminal of Chengdu Shuangliu Airport and the flight information of a day from 6:00 to 12:00, a total of 6 hours, and the data on the density of people in each area of the terminal were used to carry out simulation in the Anylogic software.

1) Environmental data settings. The environmental data mainly includes the layout information of the terminal, the number and service duration of each service facility, the corresponding airline type data of manual check-in windows and boarding gates, and the opening hours of security checks. These data were collated from historical data information provided by the airport terminal.

2) Passenger data settings. 130 flight information totaling 25051 passengers were obtained from the flight information provided by the airport, and then a passenger aggregation model dominated by flight information was established using the logarithmic normal distribution and regression analysis proposed by Zhiwei Xing et al. [16] to generate passengers.

The 2D simulation effect of Anylogic is shown in Figure 4. Among them, the red area represents the passenger rest area, the blue area represents the bathroom, and the green area represents various service areas, including catering and retail areas. There are a total of 70 service facilities, 60 check-in counters, 24 regular security checkpoints, and 4 VIP security checkpoints inside the terminal. The modeling of the passenger boarding process is shown in Figure 5. Passengers are generated at the portal on a regular basis based on flight information through the Passenger arrival component. After completing a series of activities such as check-in and security checks, it ultimately leaves the airport through the boarding gate.

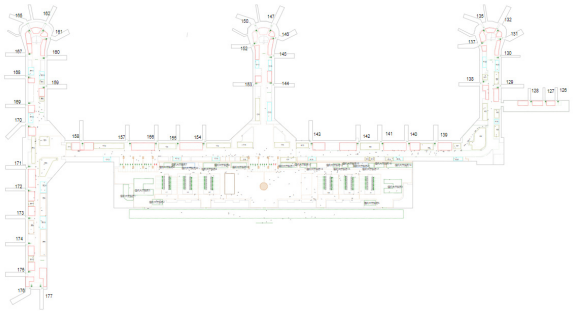

Fig 4. 2D rendering of Anylogic simulation

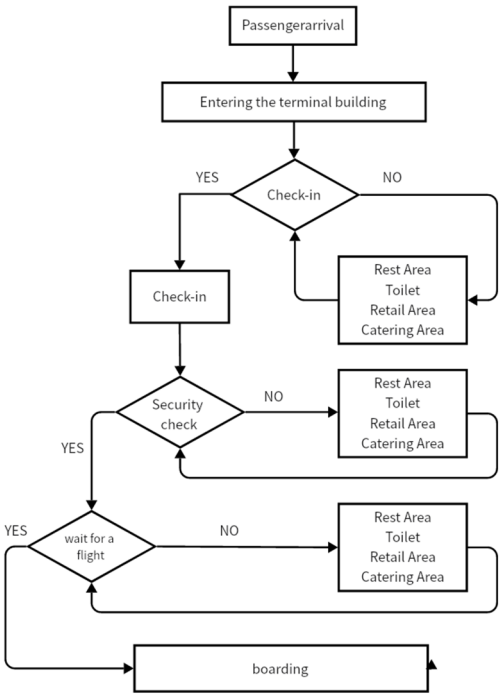

Fig 5. Passenger boarding flow chart

In this paper, five groups of experimental environments are set up, in which the first group is the GQ-MFI algorithm and the other four groups are the comparison groups, and the detailed settings of each group are shown in the following table 1:

Table 1. The detailed settings of each group

Algorithm type	Reward type	Acronyms
GQ-MFI	$r_{ext} + r_{int}$	GQ-MFI
Q-learning	$r_{ext} + r_{int}$	Q-learning
Q-learning using the group shared Q-table policy	$r_{ext} + r_{int}$	GQ-learning
Mean-field Q-learning	$r_{ext} + r_{int}$	MFQ
Mean-field Q-Learning using the group shared Q-Table policy	r_{ext}	GQ-MF

5.2. Result

5.2.1. Accuracy of Pedestrian Flow Density

Crowd density accuracy is used as an evaluation metric, defined as the ratio of the number of zones f in which the prediction of crowd density at a certain time period in each round of training matches the true situation to the total number of zones F.

$$\mathrm{APFD} = \frac{f}{F}$$

Where APFD denotes the accuracy of pedestrian flow density and takes a value ranging from 0 to 1, with closer to 1 indicating closer to the real situation.

The experimental results of comparing the accuracy of pedestrian flow density are shown in Figure 6. It can be seen that in the initial stage of training, the accuracy rates of the five are not very different. With the increase of learning cycles, obvious differences begin to appear. Q-learning has a relatively low accuracy rate, its learning effect is poor, and there is no obvious improvement with the increase of learning cycles. GQ-learning, on the other hand, has an increasing accuracy rate with the increase of learning cycles, which indicates that in the policy of group shared Q-table agent can make full use of the experience that has been learned by the others, which accelerates the overall learning process. The overall learning process is accelerated, and the agent begin to gradually approach the best behavioral patterns. MFQ accuracy rate fluctuates more significantly. GQ-MFI begins to exceed the other four after the learning cycle reaches a certain level and maintains a higher level of accuracy. Other things being equal, after adding intrinsic rewards, GQ-MFI converges about 13 rounds earlier than GQ-MF and reaches the optimal accuracy rate right after convergence. This is due to the fact that the GQ-MF merely encourages the agent to go to the activity sites and ignores the psychological factor of choosing the activity sites, which leads to the fact that the agent will try various activity sites several times for meaningless exploration during training, which ultimately leads to slowing down the training progress.

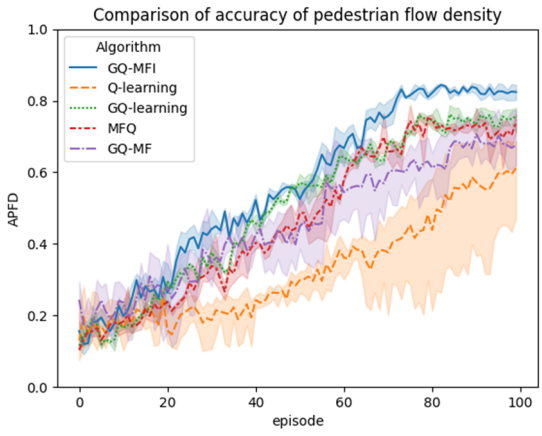

Fig 6. Comparison of the accuracy of crowd density

5.2.2. Prediction of the Number of Security Check Passengers

The number of passengers passing through the security checkpoint in a certain time reflects the crowd density situation from the side. Import the training results of each algorithm into Anylogic, keep the environment data settings unchanged, change the passenger data settings and input new flight information. Run the simulation model and statistically get 36 predicted values of the number of security check passengers in 6 hours at 10min intervals. Compare this result with the obtained real value of the number of security check passengers in the airport terminal, as shown in Figure 7.

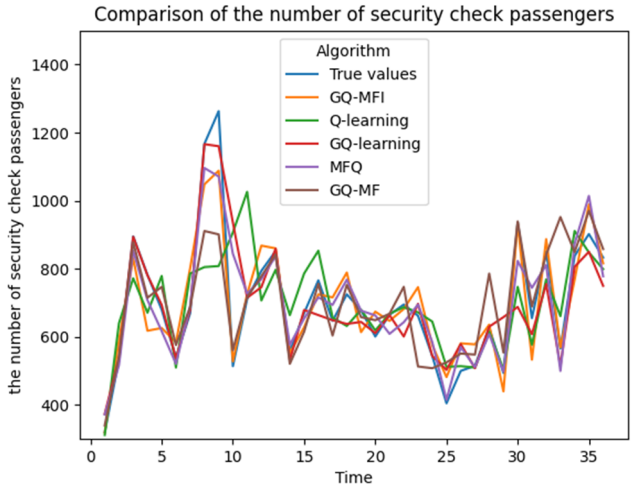

Fig 7. Comparison of the number of security check passengers

The Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) of the six algorithms are calculated as shown in the following equation:

$$\mathrm{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \tilde{y}_i)^2}$$

$$\mathrm{MAPE} = \frac{1}{m}\sum_{i=1}^{m}\frac{|y_i - \tilde{y}_i|}{y_i}$$

Where y_i is the true value of the number of security check passengers, and $\tilde{y}_i$ is the predicted value of the number of security check passengers.

The results of the calculations were counted and are shown in Table 2. Table 2 shows that the GQI-MFC outperforms the others and has a better performance in the time scale of 10min.

Table 2. RMSE and MAPE

Algorithm	RMSE	MAPE
GQ-MFI	71.94	8.19
Q-learning	148.73	13.66
GQ-learning	99.06	9.04
MFQ	80.85	8.12
GQ-MF	111.71	9.63

6. Conclusion

In the context of the large-scale airport terminal, passengers are treated as the agent for reinforcement learning. This paper proposes a group shared Q-table framework for a large number of multi-agent systems, introduces the Mean-field to simplify the interaction information of multi-agent systems, and combined with the corresponding intrinsic reward function. It improves the convergence speed of the algorithm and achieves a more stable algorithm learning process. The experimental results validate the applicability and effectiveness of the algorithm.

References

[1] Dewey, Ding Shifei. Review of multi-agent reinforcement learning [J]. Computer Science, 2019,46 (08): 1-8.

[2] Feng Xia, Zhao Liqiang. Prediction of Terminal Security Check Passenger Flow Based on Time Series Analysis [J]. Modern Electronic Technology, 2023,46 (06): 135-142. DOI: 10.16652/j.issn.1004-373x.2023.06.024.

[3] Wang Xinglong, Shi Zongbei, He Min. Airport traffic prediction based on similar day PSO-SVM [J]. Computer Simulation, 2022, 39 (07): 86-90+123.

[4] Zhong Xiang, Zhu Caiyun, Han Xu. Airport security passenger flow prediction model based on BP neural network [J]. Aviation Engineering Progress, 2019,10 (05): 655-663.

[5] Rodríguez-Sanz Á, de Marcos A F, Pérez-Castán J A, et al. Queue behavioural patterns for passengers at airport terminals: A machine learning approach[J]. Journal of Air Transport Management, 2021, 90: 101940.

[6] Li C, Wang T, Wu C, et al. Celebrating diversity in shared multi-agent reinforcement learning[J]. Advances in Neural Information Processing Systems, 2021, 34: 3991-4002.

[7] Yang Y, Luo R, Li M, et al.Mean field multi-agent reinforcement learning [C]// ICML 2018:Thirty-fifth International Conference on Machine Learning.2018,5567-5576.

[8] Shi Dianxi, Zhao Chenran, Zhang Yaowen, et al. Adaptive reward method for end-to-end cooperation based on multi-agent reinforcement learning [J]. Computer Science, 2022,49 (08): 247-256.

[9] Andres A, Villar-Rodriguez E, Ser J D. Collaborative training of heterogeneous reinforcement learning agents in environments with sparse rewards: what and when to share? [J]. Neural Computing and Applications, 2022: 1-28.

[10] Ma W. Agent-based model of passenger flows in airport terminals. (PhD)[J]. 2013.

[11] Clifton J, Laber E. Q-learning: Theory and applications[J]. Annual Review of Statistics and Its Application, 2020, 7: 279-301.

[12] Aubret, Matignon A, Hassas L, etal. An Information-Theoretic Perspective on Intrinsic Motivation in Reinforcement Learning: A Survey[J]. ENTROPY, 2023, 25(2):327.

[13] Barto A G. Intrinsic motivation and reinforcement learning[J]. Intrinsically motivated learning in natural and artificial systems, 2013: 17-47.

[14] Harlow H F.Learning and satiation of response in intrinsically motivated complex puzzle performance by monkeys[J]. Journal of Comparative and Pnysiological Psychology, 1950, 43(4): [6]289-294.

[15] Shahid S, Paul J. Intrinsic motivation of luxury consumers in an emerging market[J]. Journal of Retailing and Consumer Services, 2021, 61: 102531.

[16] Xing Zhiwei, Feng Wenxing, Luo Qian, et al. A Single Flight Departure Passenger Aggregation Model Based on Flight Departure Time Domination [J]. Journal of University of Electronic Science and Technology, 2015,44 (05): 719-724.