



Research article

Joint AGV dispatching and predictive maintenance in automated container terminals: A health-aware multi-agent reinforcement learning approach

Ang Yang¹, Qing Xu¹, Xingchun Li¹, Keshuang Du^{1,*} and Jing Gao²

¹ School of Maritime Economics and Management, Dalian Maritime University, 1 Linghai Road, Dalian, 116000, China

² University of South Australia, Adelaide, SA 5001, Australia

* **Correspondence:** Email: keshuang.du@dlmu.edu.cn.

Abstract: Automated container terminals rely on automated guided vehicle (AGV) fleets to connect quay cranes and yard blocks. Existing AGV dispatching studies mainly address task assignment, routing, charging, or equipment coordination, while maintenance decisions are often modeled as exogenous availability changes. This separation limits policy responses to health degradation, failure risk, maintenance windows, and maintenance resources. This study investigates a joint AGV dispatching and predictive maintenance problem that integrates transport assignment, charging, preventive and corrective maintenance, health degradation, stochastic failures, and shared resource capacities in a unified framework. A health-aware extension of QMIX (monotonic mixing of individual agents' action values), named HA-QMIX, is proposed to learn decentralized AGV dispatching policies under centralized training. HA-QMIX incorporates the health index, remaining useful life, failure risk, and state of charge into agent observations and global states. A top-K task candidate mechanism handles dynamic task arrivals within a fixed action space, and action masking excludes infeasible actions caused by task status, vehicle status, charging conditions, maintenance windows, and resource capacities. A reward transformation links net-cost minimization with cooperative learning, while constrained checkpoint selection balances operational performance and safety. In the reported preventive maintenance pressure experiments, HA-QMIX exhibits relatively consistent risk-related indicators across five training seed runs, while throughput varies substantially across these runs. The results therefore identify throughput stability as a remaining limitation of the current approach.

Keywords: Automated container terminal; AGV dispatching; predictive maintenance; preventive maintenance; multiagent reinforcement learning; QMIX; health-aware scheduling

Mathematics Subject Classification: 90B36, 90B25, 68T05, 90B06

1. Introduction

Automated container terminals use integrated quayside, yardside, and horizontal transport systems to reduce vessel turnaround time and improve terminals' operating efficiency. Automated guided vehicle (AGV) fleets undertake the horizontal transport tasks between quay cranes (QCs) and yard blocks, and vehicle dispatching decisions directly affect QC waiting, yard–crane interaction, vehicle utilization, and task delay [1, 2]. Existing studies have developed mathematical programming, decomposition, hierarchical control, and reinforcement learning methods for AGV scheduling, equipment coordination, and conflict-free movement in automated terminals [2–4]. These contributions provide important foundations for vehicle assignment and equipment synchronization.

AGV dispatching in an automated terminal is not only a transport assignment problem. Battery state, charging capacity, health degradation, failure risk, and maintenance resources also consume vehicles' time and influence their subsequent task availability. Charging decisions have been shown to change vehicles' availability and operational delay in automated terminal dispatching [5]. Reliability deterioration and predictive maintenance studies also show that health indicators, remaining useful life, and maintenance timing can substantially affect production cost, downtime, and systems' reliability [6, 7]. When AGVs' health degradation is ignored, short-term dispatching gains may be converted into high-risk operation, unexpected failure, and corrective maintenance (CM) workload.

The joint consideration of dispatching and predictive maintenance remains underdeveloped for automated terminal AGV fleets. AGV scheduling studies usually incorporate dynamic task release, path interaction, and charging and equipment coordination, but health state, preventive maintenance (PM), and CM are seldom modeled as online AGV actions [1, 4, 5]. Predictive maintenance and production–maintenance integration studies usually describe degradation, reliability, and maintenance timing, but the main decision objects are machines, jobs, or components rather than mobile AGV fleets with spatial movement and terminal resource constraints [6, 8, 9]. This gap motivates a unified decision model that links transport tasks, charging behavior, health degradation, and maintenance resources in the same time-indexed operating process.

Solving this joint problem is challenging for three reasons. First, transport tasks are released dynamically, and each AGV has a changing position, operating state, and feasible task set. Second, PM, CM, and charging compete with transport execution for vehicles' availability and shared service resources. Third, health degradation introduces long-term risk dependence, because delaying maintenance or charging may reduce current interruptions but increase future failure probability, downtime, and task disturbance. These features make a purely static optimization model difficult to use as an online dispatching policy, while a simple rule-based policy cannot fully represent the coupled tradeoff between throughput and reliability.

This study develops a health-aware multiagent reinforcement learning framework for joint AGV dispatching and predictive maintenance. The problem is first formulated as a time-indexed joint optimization model that includes terminal road network constraints, task assignment, charging capacity, PM and CM capacity, health degradation, failure risk representation, and a net-cost objective. The optimization objective is then transformed into a cooperative Markov game, where each AGV acts as an agent and chooses among task execution, charging, PM, buffer movement, waiting, and state keeping. A health-aware QMIX algorithm (HA-QMIX), which uses monotonic mixing of individual agents' action values, is designed to learn decentralized AGV actions under centralized training.

The main contributions are summarized as follows.

1. A joint AGV dispatching and predictive maintenance formulation is established for automated container terminals. The formulation integrates transport task assignment, charging decisions, PM, CM, stochastic failure, health degradation, and shared resource capacities in a unified decision framework.
2. An HA-QMIX method is proposed for online AGV fleet control. The method incorporates the health index, remaining useful life, failure risk, and state of charge into the multiagent state representation, and combines a top-K task candidate mechanism with action masking to handle dynamic tasks and operational feasibility constraints.
3. A computational study evaluates the proposed method under a PM pressure scenario. Baseline comparison, analysis of the training process, multiseed evaluation, an ablation analysis, and a sensitivity analysis are used to examine transport performance, maintenance behavior, failure control, and resource capacity effects.

The remainder of this study is organized as follows. Section 2 reviews studies on AGV scheduling, predictive maintenance, integrated optimization, and potential solution methods. Section 3 formulates the joint AGV dispatching and predictive maintenance problem. Section 4 presents the HA-QMIX framework, including its observation design, top-K candidate selection, action masking, reward transformation, and checkpoint selection. Section 5 reports the computational experiments and managerial insights. Section 6 concludes the study.

2. Literature review

This section reviews the research foundations for integrating AGV dispatching with predictive maintenance in automated container terminals. The discussion addresses the operational importance and characteristics of the joint problem in Sections 2.1 and 2.2; examines related studies on AGV scheduling, maintenance decision-making, and integrated optimization in Section 2.3; and evaluates the applicability of potential solution methods in Section 2.4.

2.1. Operational importance of joint AGV dispatching and predictive maintenance

Automated container terminals organize QCs, yard equipment, and AGV fleets into a continuous loading and unloading chain, in which horizontal transport capacity constrains vessels' service rhythm and yard turnover [1, 2]. AGV dispatching covers task assignment, route execution, and quay–yard connection, while dynamic task release and path conflicts are converted into task delays and equipment waiting [2, 3]. Studies on simultaneous scheduling of machines and multiload AGVs also indicate that material handling vehicle utilization and the system's makespan jointly determine the operational performance of automated systems [10].

Transport tasks, charging behavior, and equipment coordination compete for AGVs' available time on the same operating timeline, and dispatching policies jointly affect delay costs, empty-travel costs, and energy costs [4, 5]. Integrated optimization of AGVs and dual yard cranes incorporates vehicle assignment into yard–crane scheduling, showing that isolated optimization of yard equipment cannot fully represent the temporal coupling among multiple suboperations [4]. AGV dispatching with charging requirements further shows that energy replenishment decisions alter the cost relationship among transport plans, vehicle availability, and operational delay [5].

Long-term high-frequency operation of AGV fleets leads to health degradation and failure risk, making remaining useful life (RUL) prediction and health indicator (HI) modeling central inputs for predictive maintenance decisions and resource allocation [7]. Production scheduling and maintenance coordination under reliability degradation shows that decreasing reliability increases failure probability, while CM and PM correspond to the post-failure cost and proactive maintenance cost, respectively [6]. PM decisions also change operational delay costs because maintenance occupies equipment time and affects task processing sequences [6].

The operational importance of joint AGV dispatching and predictive maintenance arises from the coupling of throughput, delay, energy, maintenance, and downtime costs within the same operating process [1,5,6]. When transport dispatching ignores health states, short-term task completion benefits may be obtained at the expense of high-risk operation, failure downtime, and CM resource occupation [6,7]. When maintenance planning is detached from task release and equipment coordination, PM timing may increase QCs waiting, yardside waiting, and AGV idle time [1,4,6].

2.2. Operational characteristics of joint AGV dispatching and predictive maintenance

The joint AGV dispatching and predictive maintenance problem exhibits dynamic temporal characteristics, as task releases, vehicle positions, QC service, and yardside service continuously change during operations [1,2]. Before executing a transport task, an AGV needs to match the task origin, task direction, and road accessibility, while path conflicts and vehicle interactions can change the task completion time [2,3]. The dynamic dispatching process also includes future task arrivals and current vehicle state updates, so a single dispatching decision affects the subsequent set of available vehicles and equipment waiting time [1,5].

The joint decision process also exhibits multiresource competition, with AGVs' available time allocated among loaded transport, empty travel, charging, waiting, and maintenance [4,5]. Charging requirements cause vehicles to switch between transport service and energy replenishment, and charging duration, charger capacity, and task delay jointly affect dispatching outcomes [5]. Coordination among QCs, yard equipment, and AGVs converts local equipment idleness into chain-level waiting, requiring vehicle assignment and equipment sequencing to be coordinated under the same operational rhythm [1,4].

Health degradation gives AGV states an intertemporal accumulation effect, because equipment usage reduces reliability and increases the failure probability [6,7]. HI and RUL convert operating history into maintenance decision information, and health state changes affect equipment's availability, failure risk, and asset allocation [7]. PM restores reliability before failure while occupying operating time, whereas CM restores equipment after failure and generates repair costs and downtime losses [6].

Joint AGV dispatching and predictive maintenance further involves a tradeoff between short-term throughput gains and intertemporal risk costs [5,6]. Delayed charging can release current transport capacity, but energy constraints may increase the unavailable time and operational delay in subsequent periods [5]. Delayed PM can reduce current maintenance occupation, but health degradation may increase failure downtime, CM cost, and task sequence disruption [6,7].

2.3. Existing studies on AGV scheduling, maintenance decision-making, and integrated optimization

Chen et al. developed a multirobot task-allocation framework for yard cranes and AGVs in automated container terminals, using a discretized virtual network and a multicommodity flow model to represent interequipment constraints [1]. Numerical experiments showed that alternating direction method of multipliers (ADMM) decomposition and a rolling horizon framework can obtain near-optimal equipment coordination solutions for practical terminal instances [1]. The research focus was synchronous scheduling of yard cranes and AGVs, while health degradation, PM/CM decisions, and maintenance resource occupation have not yet entered the vehicle dispatching state [1, 6].

Xin et al. proposed a sequential planning control method for interacting equipment in automated terminals to address conflict avoidance among AGVs, QCs, and yard equipment [3]. Li et al. further proposed a hierarchical framework for dynamic conflict-free AGV scheduling, in which the upper layer uses reinforcement learning for dynamic task assignment and the lower layer uses a path generation algorithm to evaluate actions and mitigate path conflicts [2]. Dynamic dispatching and path coordination studies have advanced task assignment and conflict mitigation, but vehicle health risk, maintenance windows, and repair resources are not yet part of online action selection [2, 3, 6].

Zhang et al. studied the integrated scheduling of AGVs and dual yard cranes, formulating a mixed-integer programming (MIP) model and designing a branch-and-bound method to handle dual yard crane interference and AGVs' arrival timing [4]. Yang et al. incorporated multiload strategies and charging requirements into AGV dispatching in automated terminals, showing that multiload transport can reduce the empty-travel cost and operational delay cost [5]. Energy replenishment, equipment interference, and transport tasks have been integrated into the same scheduling problem, but maintenance behavior is still commonly treated as a change in equipment availability outside the dispatching system [4–6].

Chen et al. studied the simultaneous scheduling of machines and multiload AGVs, combining a genetic algorithm with digital twin simulation to evaluate AGVs' utilization and the system's makespan [10]. The experimental results indicate that a digital twin environment can provide more detailed feedback on transport time and vehicle utilization for production scheduling schemes [10]. The research object is mainly machine–AGV synchronization in flexible manufacturing systems, while quay–yard operation chains, charging dispatching, and health maintenance coupling in automated terminals still require port-oriented modeling [1, 5, 10].

Si et al. systematically reviewed statistical data-driven RUL estimation methods and pointed out that RUL estimation depends on health monitoring information and degradation process modeling [11]. Alaswad and Xiang reviewed condition-based maintenance (CBM) optimization models under discrete-state degradation, continuous-state degradation, and proportional hazards models, and discussed maintenance policy design for single-component and multicomponent systems [12]. Zhou et al. further reviewed RUL prediction methods from an HI-dependent perspective, showing that the HI's construction quality affects health assessment and asset allocation in predictive maintenance [7].

Shi et al. proposed a multicomponent CBM framework under system reliability requirements, using a rolling horizon, Bayesian updating, and a dynamic priority heuristic to select PM component groups [13]. Computational results show that updating reliability predictions with new degradation observations can reduce maintenance costs and improve the system's reliability [13]. Multicomponent CBM research provides a modeling basis for health-state-driven maintenance decisions, but transport task release, vehicle position, charging resources, and terminal equipment coordination are generally not state variables in maintenance decision models [1, 5, 13].

Lu et al. studied the integrated problem of single-machine production scheduling and PM planning under failure uncertainty, jointly determining the job sequence, PM timing, and planned completion time to improve the schedule's robustness and stability [14]. Liu et al. combined predictive maintenance planning with single-machine production scheduling and minimized the total expected cost using health state, virtual age, and RUL information [15]. Single-machine integrated scheduling studies introduce degradation information and maintenance planning into production scheduling, but the problem structure mainly centers on a single machine and a given job set; multi-AGV fleet coordination, dynamic transport tasks, and charging behavior are still outside the core decision objects [5, 14, 15].

Ghaleb et al. developed a stochastic mixed-integer model for production scheduling and maintenance planning of a single degrading machine, representing machine degradation as a multistate process and allowing maintenance actions, inspection policies, and job sequences to jointly determine the total cost [8]. Hu et al. further studied production scheduling and maintenance coordination under reliability degradation, incorporating PM, CM, failure probability, and the job tardiness cost into the same expected cost objective and solving the model with an improved grey wolf optimizer [6]. Integrated production–maintenance optimization shows that maintenance decisions change production scheduling performance, but differences remain between production machine sequencing models and online AGV dispatching in automated terminals in terms of task dynamics, spatial movement, and shared service resources [1, 6, 8].

Geurtsen et al. reviewed production, maintenance, and resource scheduling and noted that simultaneous integration of production–maintenance–resource scheduling remains relatively limited [9]. AGV dispatching studies in automated terminals have addressed equipment coordination, dynamic dispatching, path conflicts, and charging requirements in depth, but health degradation, maintenance windows, and PM/CM capacity have not yet become common dispatching states [1, 2, 4, 5]. Maintenance decision-making and integrated production–maintenance research have treated RUL, HI, PM, CM, and reliability degradation in depth, but multi-AGV transport tasks, quay–yard operation chains, and online vehicle action selection remain outside the main research boundary [6, 9, 11, 12].

2.4. Potential solution methods and critical assessment

Joint AGV dispatching and predictive maintenance can be addressed by mathematical programming, decomposition optimization, metaheuristics, simulation optimization, and learning-based methods, with different methods corresponding to different problem scales, information structures, and online response requirements [1, 9, 16]. Mathematical programming and exact algorithms are suitable for representing resource capacity, operation sequence, and equipment interference, and their solutions provide explicit feasibility interpretation [4, 16]. When the task release, vehicle state, and equipment health change over time, solution methods also need to handle dynamic state updates and intertemporal risk costs [2, 17, 18].

Mixed-integer programming, constraint programming (CP), and branch-and-bound methods are suitable for small- and medium-scale static or quasi-static scheduling problems and can provide verifiable benchmark solutions and structural model interpretation [4, 16]. Decomposition optimization and rolling horizon methods reduce computational pressure by splitting equipment subproblems or stage subproblems, making them suitable for multiequipment coordination among QCs, yard cranes, and AGVs [1, 2]. Metaheuristics and hybrid heuristics are suitable for large-scale instances and multiobjective tradeoffs, but the solutions' quality usually depends on the coding schemes, neighborhood structures, and parameter settings [5, 19].

Table 1. Applicability comparison of potential solution methods.

Method category	Method	Dynamic task release	Multi-AGV coordination	Charging decision	Health degradation	PM/CM decision	Shared resource constraints	Long-term risk dependence	Decentralized execution	Representative studies
Optimization methods	Mixed-integer programming	△	△	△	×	△	✓	△	×	Qin et al. (2020) [16]; Zhang et al. (2024) [4]
	Constraint programming / MIP-CP hybrid	△	△	×	×	×	✓	△	×	Qin et al. (2020) [16]
	Decomposition / rolling horizon	✓	✓	△	×	△	✓	△	×	Chen et al. (2020) [1]; Li et al. (2024) [2]
	Heuristic and metaheuristic	△	△	✓	△	△	△	△	×	Li et al. (2021) [19]; Yang et al. (2025) [20]
Simulation-based methods	Simulation optimization	△	△	✓	△	△	△	△	×	Chen et al. (2021) [21]
	Digital-twin-assisted optimization	△	△	△	△	△	△	△	×	Chen et al. (2025) [10]
Learning-based methods	Reinforcement learning	✓	△	✓	✓	✓	△	✓	△	Li et al. (2024) [2]; Zhang and Si (2020) [17]
	Multiagent reinforcement learning	✓	✓	✓	△	△	△	✓	✓	Ruiz Rodríguez et al. (2022) [18]; Yang et al. (2025) [20]
	Value decomposition methods	✓	✓	△	△	△	△	✓	✓	Rashid et al. (2020) [22]; Ruiz Rodríguez et al. (2022) [18]

Note: MIP = mixed-integer programming; CP = constraint programming. ✓ denotes that the method can directly or naturally address the requirement; △ denotes that the method can address the requirement after state discretization, scenario extension, rolling horizon design, policy structure design, or other problem-specific adaptation; × denotes that the method is difficult to apply directly to the requirement.

Potential methods are compared in Table 1 from the perspectives of dynamic task release, multi-AGV coordination, charging decisions, health degradation, PM/CM decisions, shared resource constraints, long-term risk dependence, and decentralized execution. Optimization-model-based methods have clear structures for representing shared resource constraints and operation sequences, but dynamic states, health degradation, and decentralized execution usually require rolling horizon design, state discretization, or additional decomposition [1, 4, 16]. Simulation optimization and digital-twin-assisted methods can reproduce vehicle utilization, charging mechanisms, and operational interference and can provide offline evaluation environments for dispatching rules or learning policies; however, direct optimization depends on simulation repetitions and scenario calibration [10, 21].

Reinforcement learning methods are suitable for representing dynamic dispatching as a sequential decision problem and learning task assignment, charging, or maintenance policies through interaction with the environment [2, 17]. Deep reinforcement learning has been applied to multicomponent CBM, where degradation states can be mapped directly to maintenance actions, and the burden of high-dimensional threshold search can be reduced [17]. Reinforcement learning studies in automated terminals have also begun to address dynamic task assignment, path conflicts, and charging dispatching, but maintenance actions, failure recovery, and health risk are generally not modeled simultaneously with AGV transport actions [2, 18, 20].

Multiagent reinforcement learning further matches the distributed decision structure of AGV fleets, and value decomposition provides a team value modeling approach for centralized training with decentralized execution [18, 22]. Multiagent reinforcement learning for predictive maintenance has incorporated multiple pieces of equipment and maintenance technician scheduling into partially observable cooperative processes, and experimental results show that learned policies can reduce failures and downtime [18]. Multiagent simulation optimization in automated terminals has covered joint task assignment and charging dispatching for intelligent guided vehicles, indicating that distributed learning is suitable for handling vehicle state heterogeneity and dynamic operating environments [20].

Overall, joint AGV dispatching and predictive maintenance requires the constraint interpretability of optimization models, the operational detail of simulation environments, and the online adaptability of learning methods at the same time [9, 21, 22]. Existing methods have separately covered equipment coordination, charging dispatching, health-aware maintenance, and multiagent learning, but unified online decisions over task assignment, charging, PM, CM, failure recovery, health degradation, and maintenance resource capacity still leave room for integration [1, 6, 20]. The following formulation and method design therefore need to represent transport benefits, maintenance costs, failure risk, and action feasibility within a unified decision framework [2, 17, 18].

3. Problem formulation

This section develops the joint AGV dispatching and predictive maintenance formulation. It defines the automated terminal operating scenario in Section 3.1; specifies the fleet, task, and shared resource constraints in Section 3.2; describes health degradation and failure risk in Section 3.3; establishes the integrated decision structure in Section 3.4; and formulates the optimization objectives in Section 3.5.

3.1. Automated terminal operation scenario

The horizontal transport system of an automated container terminal consists of QC work areas, yard work areas, an AGV ring road, charging areas, maintenance areas, and buffer areas. AGVs execute loading and unloading transport tasks between QCs and yard blocks. The spatial structure of the terminal can be represented by a directed road network, where AGV transport, charging, maintenance, and waiting activities occur on network nodes and directed road arcs. The directed road network and node partition are defined as

$$\mathcal{G} = (\mathcal{N}, \mathcal{E}), \quad (3.1)$$

$$\mathcal{N} = \mathcal{N}^{\text{quay_crane}} \cup \mathcal{N}^{\text{yard}} \cup \mathcal{N}^{\text{charging}} \cup \mathcal{N}^{\text{maintenance}} \cup \mathcal{N}^{\text{buffer}} \cup \mathcal{N}^{\text{road}}. \quad (3.2)$$

Here, $\mathcal{G}$ denotes the directed road network of the automated terminal, $\mathcal{N}$ denotes the set of terminal nodes, and $\mathcal{E}$ denotes the set of directed road arcs on which AGVs can travel. $\mathcal{N}^{\text{quay_crane}}$ denotes the set of QC service nodes, $\mathcal{N}^{\text{yard}}$ denotes the set of yard endpoints, $\mathcal{N}^{\text{charging}}$ denotes the set of charging nodes, $\mathcal{N}^{\text{maintenance}}$ denotes the set of maintenance nodes, $\mathcal{N}^{\text{buffer}}$ denotes the set of buffer nodes, and $\mathcal{N}^{\text{road}}$ denotes the set of ordinary road nodes. Equations (3.1)–(3.2) incorporate QCs, yards, charging stations, maintenance stations, buffer areas, and ordinary roads into a unified node set, while the directed arc set defines feasible AGV travel directions.

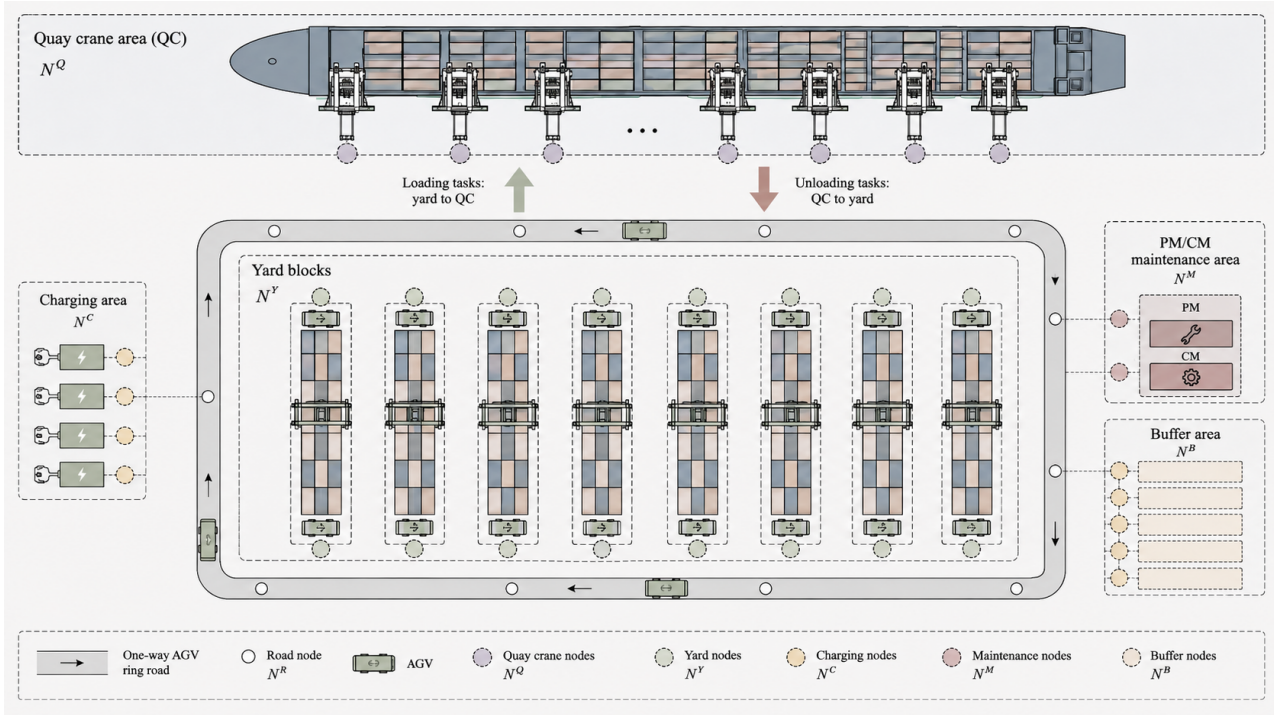


Figure 1. Automated terminal operation scenario and joint decision scope.

The scenario structure maps QC service nodes, yard endpoints, charging nodes, maintenance nodes, buffer nodes, and ordinary road nodes onto the same directed road network, as shown in Figure 1. Loading tasks move from yard endpoints to QCs, whereas unloading tasks move from QCs to yard endpoints; the task directions are further described by Eqs (3.8)–(3.9). Charging areas, the shared PM/CM maintenance area, and buffer areas are located outside the main transport ring, causing transport task assignment, charging dispatching, PM, and CM to compete for AGV time and service resources within the same spatial network.

The main AGV roadway in the automated terminal adopts a ring traffic organization. The ring road's direction is characterized by the node successor relation, and the feasible main road arcs satisfy

$$\text{succ}(r_k) = \begin{cases} r_{k+1}, & k < |\mathcal{R}|, \\ r_1, & k = |\mathcal{R}|, \end{cases} \quad (3.3)$$

$$(r_k, \text{succ}(r_k)) \in \mathcal{E}, \quad (\text{succ}(r_k), r_k) \notin \mathcal{E}, \quad \forall r_k \in \mathcal{R}. \quad (3.4)$$

Here, $\mathcal{R} = (r_1, r_2, \dots, r_{|\mathcal{R}|})$ denotes the ordered node sequence on the main ring road, and $\text{succ}(r_k)$ denotes the successor of node r_k along the ring road's direction. Equation (3.3) defines the ring direction,

and Eq (3.4) includes the forward road arc in $\mathcal{E}$ while excluding the reverse road arc. The one-way ring constraint forces AGVs to travel along the prescribed direction and reduces the path coordination complexity caused by bidirectional traffic.

AGV movements between different yard endpoints are completed through the ring road, and no direct road connection is provided between yard endpoints. The set of direct road arcs between yard endpoints satisfies

$$\mathcal{E}^{\text{yard_to_yard}} = \{(i, j) \mid i \in \mathcal{N}^{\text{yard}}, j \in \mathcal{N}^{\text{yard}}, i \neq j, (i, j) \in \mathcal{E}\} = \emptyset. \quad (3.5)$$

Here, $\mathcal{E}^{\text{yard_to_yard}}$ denotes the set of direct road arcs between yard endpoints. Equation (3.5) restricts the number of direct connections between yard endpoints to zero. Horizontal transport tasks are organized around QCs and yard endpoints, while movements between yard endpoints are realized through the external ring road.

AGV travel time is jointly determined by directed network distance, road arc length, maximum speed, and the discrete decision interval. The directed shortest path distance and discrete travel time between nodes are defined as

$$d_{ij} = \min_{p \in \mathcal{P}_{ij}} \sum_{(u,v) \in p} l_{uv}, \quad i, j \in \mathcal{N}, \quad (3.6)$$

$$\tau_{ij} = \left\lceil \frac{d_{ij}}{v^{\max} \Delta t} \right\rceil. \quad (3.7)$$

Here, l_{uv} denotes the length of directed arc $(u, v) \in \mathcal{E}$, $\mathcal{P}_{ij}$ denotes the set of all directed feasible paths from node i to node j , d_{ij} denotes the directed shortest path distance from node i to node j , $v^{\max}$ denotes the maximum AGV speed, Δt denotes the discrete decision interval, and τ_{ij} denotes the discrete travel time from node i to node j . Equation (3.6) computes feasible path distance on the directed network $\mathcal{G}$, and Eq (3.7) converts continuous travel time into the number of discrete time steps. The travel time definition provides a unified time scale for transport task execution, charging transfer, and maintenance transfer.

Loading and unloading tasks have opposite quay–yard transport directions. The origin and destination of a transport task satisfy

$$\xi_k = \text{unload} \Rightarrow o_k = q_k \in \mathcal{N}^{\text{quay_crane}}, \quad d_k = y_k \in \mathcal{N}^{\text{yard}}, \quad (3.8)$$

$$\xi_k = \text{load} \Rightarrow o_k = y_k \in \mathcal{N}^{\text{yard}}, \quad d_k = q_k \in \mathcal{N}^{\text{quay_crane}}. \quad (3.9)$$

Here, $\mathcal{K}$ denotes the set of transport tasks, $k \in \mathcal{K}$ denotes a transport task index, $\xi_k \in \{\text{load}, \text{unload}\}$ denotes the task type, o_k denotes the task origin, d_k denotes the task destination, q_k denotes the associated QC node, and y_k denotes the associated yard endpoint. Equation (3.8) defines an unloading task as transport from a QC to a yard endpoint, while Eq (3.9) defines a loading task as transport from a yard endpoint to a QC. The task direction constraints characterize the quay–yard transport relationship in the loading and unloading operations of automated terminals.

Each transport task k has a release boundary r_k and a due boundary $\bar{t}_k$. Pickup and delivery require τ^{pickup} and τ^{delivery} decision intervals, respectively. The realized completion boundary C_k is determined by the execution-generated completion event, as specified in Eq (3.31).

For a task completed within the scheduling horizon, the phase-processing requirements imply

$$C_k \geq r_k + \tau^{\text{pickup}} + \tau_{o_k d_k} + \tau^{\text{delivery}}, \quad \forall k \in \mathcal{K} : \eta_{k,T} = 1. \quad (3.10)$$

This lower bound is a consequence of the task execution process; it does not independently determine completion time. Tasks unfinished at the terminal boundary are not assigned an observed completion time within the horizon.

Task due times provide urgency information for the observations and candidate task selection in Sections 4.2 and 4.3. The objective in Section 3.5 penalizes pending task waiting and does not include a separate due-date tardiness cost.

The number of vehicles accommodated by an AGV ring road arc in the same time step is limited by road capacity. The road capacity constraint is

$$\sum_{a \in \mathcal{A}} z_{aijt} \leq \kappa_{ij}, \quad \forall (i, j) \in \mathcal{E}, \quad \forall t \in \mathcal{T}. \quad (3.11)$$

Here, $\mathcal{A}$ denotes the AGV set, $\mathcal{T} = \{0, 1, \dots, T-1\}$ denotes the set of decision intervals, and $\overline{\mathcal{T}} = \{0, 1, \dots, T\}$ denotes the set of state boundaries, $z_{aijt} \in \{0, 1\}$ indicates whether AGV a occupies road arc (i, j) at time step t , and κ_{ij} denotes the number of AGVs allowed to travel on or occupy road arc (i, j) simultaneously at the same time step. Equation (3.11) counts the number of AGVs on each road arc at time step t through the occupation variable z_{aijt} and limits concurrent occupation through the capacity parameter κ_{ij} . The road capacity constraint converts traffic restrictions caused by lane capacity and safety spacing into a discrete-time resource capacity relation.

Each decision interval is represented by $[t, t+1)$, with a physical duration Δt . Vehicle and task states indexed by t are observed before the decisions at boundary t . Assignment decisions are then made, travel and service processes advance during the interval, and completion and failure events determine the states at boundary $t+1$. Actions, resource occupation, and interval-level events are indexed by $t \in \mathcal{T}$, whereas terminal states are retained at boundary T .

3.2. AGV fleet, transport tasks, and resource constraints

A transport task passes through release, assignment, execution, and completion. A failure may interrupt execution and return the unfinished task to the pending set. At boundary t , let $p_{k,t} \in \{0, 1\}$ indicate whether task k is pending, $y_{akt} \in \{0, 1\}$ indicate whether AGV a retains task k from an earlier assignment, and $\eta_{k,t} \in \{0, 1\}$ indicate whether task k has been completed before the decisions at that boundary. The execution state y_{akt} covers travel to the pickup node, pickup service, loaded travel, delivery service, and waiting within this assigned task.

Let $x_{akt} \in \{0, 1\}$ indicate a new assignment at boundary t . Let $c_{akt} \in \{0, 1\}$ and $q_{akt} \in \{0, 1\}$ denote, respectively, successful completion and failure-induced interruption during $[t, t+1)$. Thus, x_{akt} , c_{akt} , and q_{akt} are events, whereas $p_{k,t}$, y_{akt} , and $\eta_{k,t}$ are boundary states.

The initial configuration contains no assigned or completed tasks. The initial task states satisfy

$$\begin{aligned} p_{k,0} &= \mathbb{I}(r_k \leq 0), \\ \eta_{k,0} &= 0, \\ y_{ak0} &= 0, \end{aligned} \quad \forall a \in \mathcal{A}, \quad \forall k \in \mathcal{K}. \quad (3.12)$$

Here, r_k is the release boundary of task k . An unreleased task has all three task state indicators equal to zero. After release, a task is pending, assigned to one AGV, or completed

$$p_{k,t} + \sum_{a \in \mathcal{A}} y_{akt} + \eta_{k,t} = \mathbb{I}(r_k \leq t), \quad \forall k \in \mathcal{K}, \quad \forall t \in \overline{\mathcal{T}}. \quad (3.13)$$

Equation (3.13) prevents an assigned or completed task from simultaneously remaining pending.

Only a pending task can receive a new assignment, and at most one AGV can receive that assignment

$$\sum_{a \in \mathcal{A}} x_{akt} \leq p_{k,t}, \quad \forall k \in \mathcal{K}, \quad \forall t \in \mathcal{T}. \quad (3.14)$$

Together with Eq (3.13), this constraint excludes assignments before release, repeated assignments during execution, and assignments after completion.

The task occupation immediately after assignment is

$$\widehat{y}_{akt} = y_{akt} + x_{akt}, \quad \widehat{y}_{akt} \in \{0, 1\}, \quad \forall a \in \mathcal{A}, \quad \forall k \in \mathcal{K}, \quad \forall t \in \mathcal{T}. \quad (3.15)$$

Task and vehicle occupation satisfy

$$\sum_{a \in \mathcal{A}} \widehat{y}_{akt} \leq 1, \quad \forall k \in \mathcal{K}, \quad \forall t \in \mathcal{T}, \quad (3.16)$$

$$\sum_{k \in \mathcal{K}} \widehat{y}_{akt} \leq 1, \quad \forall a \in \mathcal{A}, \quad \forall t \in \mathcal{T}. \quad (3.17)$$

These constraints account for both previously assigned tasks and new assignments within the current interval.

Task completion requires the ordered execution of four phases: Travel to the pickup node, pickup service, loaded travel, and delivery service. Let $\mathcal{J} = \{1, 2, 3, 4\}$ index these phases. The required processing amounts, measured in decision intervals, are

$$\begin{aligned} D_{akt}^1 &= \tau_{n_{a,t}, o_k}, & D_{akt}^2 &= \tau^{\text{pickup}}, \\ D_{akt}^3 &= \tau_{o_k, d_k}, & D_{akt}^4 &= \tau^{\text{delivery}}. \end{aligned} \quad (3.18)$$

Here, $n_{a,t}$ is the AGV position at assignment, and the travel time terms follow Eq (3.7). Pickup and delivery service requirements are positive integers. A zero travel requirement, including an AGV already located at the pickup node, does not require a processing interval for that travel phase.

Let $R_{akt}^j \in \mathbb{Z}_{\geq 0}$ denote the remaining processing amount of phase j before the decisions at boundary t . Initially, $R_{ak0}^j = 0$. A new assignment initializes the phase requirements

$$\widehat{R}_{akt}^j = R_{akt}^j + D_{akt}^j x_{akt}, \quad \forall j \in \mathcal{J}. \quad (3.19)$$

The initialization uses the vehicle position at the new assignment. When $x_{akt} = 0$, the remaining requirements are carried forward without being recomputed.

Let $v_{akt}^j \in \{0, 1\}$ indicate one unit of processing of phase j during $[t, t + 1)$. Processing satisfies

$$\begin{aligned} \sum_{j \in \mathcal{J}} v_{akt}^j &\leq \widehat{y}_{akt}, \\ v_{akt}^j &\leq \widehat{R}_{akt}^j, & j &\in \mathcal{J}, \\ v_{akt}^j &\leq \mathbb{I} \left(\sum_{\ell < j} \widehat{R}_{akt}^\ell = 0 \right), & j &\in \mathcal{J}. \end{aligned} \quad (3.20)$$

The empty sum for $j = 1$ equals zero. Only the first unfinished phase can receive processing, and at most one unit can be processed within an interval. A zero processing increment represents waiting while the task remains assigned to the same AGV. These processing variables provide the interface for travel and service resource admission constraints.

A task completion event occurs exactly when an assigned task finishes all required phases without a failure in the interval

$$c_{akt} = \mathbb{I} \left(\widehat{y}_{akt} = 1, F_{a,t} = 0, \sum_{j \in \mathcal{J}} (\widehat{R}_{akt}^j - v_{akt}^j) = 0 \right). \quad (3.21)$$

Here, $F_{a,t}$ denotes a new failure event during $[t, t + 1)$. If failure and the nominal completion of the final phase fall within the same interval, failure takes precedence and the task is not counted as successfully completed. This convention makes completion and interruption mutually exclusive.

The remaining processing amounts and task ownership evolve as

$$R_{ak,t+1}^j = (1 - q_{akt}) (\widehat{R}_{akt}^j - v_{akt}^j), \quad \forall j \in \mathcal{J}, \quad (3.22)$$

$$y_{ak,t+1} = y_{akt} + x_{akt} - c_{akt} - q_{akt}. \quad (3.23)$$

Unless completion or interruption occurs, the AGV retains the task at the next boundary. Completion clears the task occupation after all remaining processing amounts reach zero. Interruption clears the occupation and discards the remaining processing record for that assignment attempt. The interruption event is specified in Eq (3.61).

The pending and completed states are updated by

$$p_{k,t+1} = p_{k,t} + \mathbb{I}(r_k = t + 1) - \sum_{a \in \mathcal{A}} x_{akt} + \sum_{a \in \mathcal{A}} q_{akt}, \quad (3.24)$$

$$\eta_{k,t+1} = \eta_{k,t} + \sum_{a \in \mathcal{A}} c_{akt}. \quad (3.25)$$

Equations (3.19)–(3.25) apply to all relevant AGVs and tasks and all $t \in \mathcal{T}$. The release term introduces a task at its release boundary, assignment removes it from the pending set, and interruption returns it to that set. The completed state changes only through the phase-based completion event.

Each task can be completed at most once

$$\sum_{t \in \mathcal{T}} \sum_{a \in \mathcal{A}} c_{akt} \leq 1, \quad \forall k \in \mathcal{K}. \quad (3.26)$$

A task may be reassigned following interruption, but its successful completion is recorded only once. Tasks still pending or assigned at the boundary T remain unfinished and do not contribute to the task completion benefit.

New dispatching decisions also require an assignable AGV. Let $\chi_{a,t} \in \{0, 1\}$ indicate that AGV a is operable, has no unfinished transport task, and is idle or eligible for reassignment during buffer movement. In particular, we have

$$\chi_{a,t} \leq 1 - \sum_{k \in \mathcal{K}} y_{akt}, \quad \chi_{a,t} \leq b_{a,t}, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.27)$$

Here, $b_{a,t}$ indicates whether the AGV is free of an unrepaired failure at boundary t . Charging, maintenance, and other ongoing service states are excluded from the assignable operating modes.

Task assignment, charging dispatch, PM dispatch, and buffer dispatch satisfy

$$\sum_{k \in \mathcal{K}} x_{akt} + u_{at}^{\text{charging}} + u_{at}^{\text{preventive_maintenance}} + u_{at}^{\text{buffer}} \leq \chi_{a,t}, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.28)$$

The binary variables u_{at}^{charging} , $u_{at}^{\text{preventive_maintenance}}$, and u_{at}^{buffer} represent the corresponding dispatching actions.

Each AGV selects one action as follows:

$$\begin{aligned} \sum_{k \in \mathcal{K}} x_{akt} + u_{at}^{\text{charging}} + u_{at}^{\text{preventive_maintenance}} + u_{at}^{\text{buffer}} \\ + u_{at}^{\text{waiting}} + u_{at}^{\text{keeping}} = 1, \end{aligned} \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.29)$$

Here, u_{at}^{waiting} and u_{at}^{keeping} represent waiting and keeping the current execution process, respectively. An AGV carrying an unfinished task selects the keeping action

$$u_{at}^{\text{keeping}} \geq \sum_{k \in \mathcal{K}} y_{akt}, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.30)$$

Keeping preserves task ownership while travel, service, or resource-related waiting progresses. It does not permit the current task to be replaced by a new dispatching action.

Task completion time is derived from the successful completion event defined in Eq (3.21). Since $\eta_{k,0} = 0$, the completion state transition gives $\eta_{k,T} = \sum_{t \in \mathcal{T}} \sum_{a \in \mathcal{A}} c_{akt}$. For a task completed within the scheduling horizon, its realized completion boundary is

$$C_k = \sum_{t \in \mathcal{T}} (t+1) \sum_{a \in \mathcal{A}} c_{akt}, \quad \forall k \in \mathcal{K} : \eta_{k,T} = 1. \quad (3.31)$$

A completion event during $[t, t+1)$ is recorded at boundary $t+1$. Because a task can be successfully completed at most once, this expression identifies its unique completion boundary.

The completion event can occur only after the assigned AGV finishes all required processing phases without a failure in the completion interval. Therefore, C_k is an outcome of the assignment and execution process, rather than a freely chosen scheduling variable. Waiting and any unsuccessful assignment attempts are reflected in the eventual completion boundary.

For a task with $\eta_{k,T} = 0$, completion has not been observed within the horizon and C_k is left undefined. Such tasks are excluded from completed tasks' duration statistics and receive no task completion benefit.

The numbers of AGVs served by QCs and yard endpoints at the same time step are limited by operational resource capacities. The QC service resource and yard service resource constraints are

$$\sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{K}} s_{akqt}^{\text{quay_crane}} \leq B_q^{\text{quay_crane}}, \quad \forall q \in \mathcal{N}^{\text{quay_crane}}, \forall t \in \mathcal{T}, \quad (3.32)$$

$$\sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{K}} s_{akyt}^{\text{yard}} \leq B_y^{\text{yard}}, \quad \forall y \in \mathcal{N}^{\text{yard}}, \forall t \in \mathcal{T}. \quad (3.33)$$

Here, $s_{akqt}^{\text{quay_crane}} \in \{0, 1\}$ indicates whether AGV a occupies the service resource of QC $q \in \mathcal{N}^{\text{quay_crane}}$ to process task k at time step t , and $s_{akyt}^{\text{yard}} \in \{0, 1\}$ indicates whether AGV a occupies the service resource of yard endpoint $y \in \mathcal{N}^{\text{yard}}$ to process task k at time step t . $B_q^{\text{quay_crane}}$ denotes the number of AGVs that QC q can serve simultaneously at the same time step, and B_y^{yard} denotes the number of AGVs that yard endpoint y can serve simultaneously at the same time step. Equations (3.32)–(3.33) limit concurrent service at QCs and yard endpoints, respectively, and characterize the occupation relationship between AGV transport tasks and quay/yard service resources.

AGV charging occupies service resources in the charging area, and the number of AGVs served by the charging area at the same time step is limited by the number of charging bays. The charging resource capacity satisfies

$$\sum_{a \in \mathcal{A}} g_{at}^{\text{charging}} \leq B^{\text{charging}}, \quad \forall t \in \mathcal{T}. \quad (3.34)$$

Here, $g_{at}^{\text{charging}} \in \{0, 1\}$ indicates whether AGV a is in charging service at time step t , and B^{charging} denotes the number of AGVs that the charging area can serve simultaneously. Equation (3.34) records charging resource occupation through g_{at}^{charging} and limits concurrent charging service through B^{charging} . The variable u_{at}^{charging} represents the action of moving to the charging area, whereas g_{at}^{charging} represents service occupation after arrival at the charging area; the two variables characterize resource transfer decisions and resource occupation outcomes in charging dispatching.

PM and CM jointly occupy service resources in the maintenance area, and the number of AGVs served by the maintenance area at the same time step is limited by the number of maintenance bays. The maintenance resource capacity satisfies

$$\sum_{a \in \mathcal{A}} (g_{at}^{\text{preventive_maintenance}} + g_{at}^{\text{corrective_maintenance}}) \leq B^{\text{maintenance}}, \quad \forall t \in \mathcal{T}. \quad (3.35)$$

Here, $g_{at}^{\text{preventive_maintenance}} \in \{0, 1\}$ indicates whether AGV a is in PM service at time step t , $g_{at}^{\text{corrective_maintenance}} \in \{0, 1\}$ indicates whether AGV a is in CM service at time step t , and $B^{\text{maintenance}}$ denotes the number of AGVs that the maintenance area can serve simultaneously. Equation (3.35) counts PM service and CM service together as maintenance area occupation and limits concurrent maintenance service through $B^{\text{maintenance}}$. PM is triggered by a proactive maintenance decision, whereas CM is triggered by the post-failure repair process; the maintenance capacity constraint characterizes resource competition between proactive maintenance and failure repair.

The road and service capacity relations specify limits on concurrent resource use. These upper bounds alone do not determine vehicles' movement, service admission, or processing duration. Their inclusion in the formulation should therefore be distinguished from their enforcement in the computational implementation. The simulation abstractions relevant to these resource limits are stated in Section 5.1.

An AGV can undertake an assigned task only when it is free of an unrepaired failure at the beginning of the interval

$$\widehat{y}_{akt} \leq b_{a,t}, \quad \forall a \in \mathcal{A}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}. \quad (3.36)$$

The binary state $b_{a,t}$ is observed before dispatching, whereas $F_{a,t}$ records a new failure during $[t, t + 1)$. The two quantities therefore refer to different points in the event sequence.

Let $J_{a,t}^{\text{CM,end}} \in \{0, 1\}$ indicate completion of the prescribed CM service during the interval. The failure availability state evolves as

$$b_{a,t+1} = b_{a,t} - F_{a,t} + J_{a,t}^{\text{CM,end}}, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.37)$$

A new failure can occur only in an AGV without an existing unrepaired failure, and CM completion requires an AGV already undergoing failure recovery

$$\begin{aligned} F_{a,t} &\leq b_{a,t}, \\ J_{a,t}^{\text{CM,end}} &\leq 1 - b_{a,t}, \\ J_{a,t}^{\text{CM,end}} &\leq g_{at}^{\text{corrective_maintenance}}. \end{aligned} \quad (3.38)$$

The CM completion indicator is determined by the completion of repair service, rather than by the dispatching decision. An AGV that fails during the interval becomes unavailable at the next boundary and remains unavailable until CM finishes. The interrupted transport task is identified from the task occupation before this availability update.

3.3. Health degradation and failure risk representation

Continuous transport, waiting, charging, and discharging processes change AGV equipment health, and health state changes affect subsequent failure risk and maintenance decisions. Let $h_{a,t} \in [0, 1]$ denote the HI of AGV a at time step t , $d_{a,t}^{\text{empty_travel}}$ denote the empty travel distance of AGV a at time step t , $d_{a,t}^{\text{loaded_travel}}$ denote the loaded-travel distance of AGV a at time step t , $w_{a,t} \in \{0, 1\}$ indicate whether AGV a is waiting at time step t , and $\Delta e_{a,t}$ denote the magnitude of the energy state change of AGV a at time step t . The health degradation amount satisfies

$$\Delta h_{a,t} = f^{\text{degradation}}(d_{a,t}^{\text{empty_travel}}, d_{a,t}^{\text{loaded_travel}}, w_{a,t}, \Delta e_{a,t}), \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.39)$$

$$\Delta h_{a,t} \geq 0, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.40)$$

Equations (3.39)–(3.40) describe the health degradation relation during AGV operation. Function $f^{\text{degradation}}(\cdot)$ maps empty-travel distance, loaded-travel distance, waiting state, and energy state change to the health degradation amount $\Delta h_{a,t}$. Empty-travel distance and loaded-travel distance characterize the usage intensity of the AGV mechanical system, the waiting state characterizes operating load caused by congestion or resource queues, and energy state change characterizes the effect of battery usage on health state. The non-negativity constraint restricts the health degradation amount to be no less than zero, so regular operation does not generate spontaneous health recovery.

The AGV health state is updated between adjacent time steps according to operational degradation, PM recovery, and CM recovery. The health state transition satisfies

$$h_{a,t+1} = \Pi_{[0,1]}[h_{a,t} - \Delta h_{a,t} + R_{a,t}^{\text{preventive_maintenance}} + R_{a,t}^{\text{corrective_maintenance}}], \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.41)$$

Here, $\Pi_{[0,1]}[\cdot]$ denotes a truncation operator that projects a value onto the interval $[0, 1]$, $R_{a,t}^{\text{preventive_maintenance}}$ denotes the health recovery of AGV a produced by PM service, and $R_{a,t}^{\text{corrective_maintenance}}$ denotes the health recovery of AGV a produced by CM service. Equation (3.41) keeps AGV health state within the normalized interval $[0, 1]$ and incorporates health degradation, PM recovery, and CM recovery into the same state transition relation. The health state transition constraint characterizes the dynamic relation by which AGV equipment state changes with transport operations and maintenance services.

The AGV energy state affects operational feasibility and failure risk, and the energy state is updated through travel, waiting, and charging processes. Let $e_{a,t} \in [0, 1]$ denote the state of charge (SOC) of AGV

a at time step t , $E_{a,t}^{\text{empty_travel}}$ denote empty-travel energy consumption, $E_{a,t}^{\text{loaded_travel}}$ denote loaded-travel energy consumption, $E_{a,t}^{\text{waiting}}$ denote waiting energy consumption, and $E_{a,t}^{\text{charging}}$ denote charging energy replenishment. The SOC dynamics satisfy

$$e_{a,t+1} = \Pi_{[0,1]} \left[e_{a,t} - E_{a,t}^{\text{empty_travel}} - E_{a,t}^{\text{loaded_travel}} - E_{a,t}^{\text{waiting}} + E_{a,t}^{\text{charging}} \right], \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.42)$$

Equation (3.42) describes the discrete update relation of AGV energy state. Empty-travel energy consumption $E_{a,t}^{\text{empty_travel}}$, loaded-travel energy consumption $E_{a,t}^{\text{loaded_travel}}$, and waiting energy consumption $E_{a,t}^{\text{waiting}}$ reduce SOC, while charging energy replenishment $E_{a,t}^{\text{charging}}$ increases SOC. The truncation operator $\Pi_{[0,1]}[\cdot]$ keeps SOC within the normalized energy interval. SOC dynamics provide energy-state inputs for the health degradation function and failure risk function.

The projection onto $[0, 1]$ bounds the numerical SOC state but does not enforce energy sufficiency. In the present formulation, task acceptance is not conditioned on an energy budget covering travel to the task origin, loaded movement, service operations, and subsequent travel to a charging station. Equation (3.42) should therefore not be interpreted as a guarantee of energy-feasible task execution. The low-SOC handling used in the computational implementation is described in Section 5.1.

AGV failure risk is jointly determined by the health state, energy state, and equipment usage state. Let $\rho_{a,t} \in [0, 1]$ denote the failure risk of AGV a at time step t , and let $\ell_{a,t} \geq 0$ denote the RUL of AGV a at time step t . The failure risk and RUL representations satisfy

$$\rho_{a,t} = f^{\text{failure_risk}}(h_{a,t}, e_{a,t}, \omega_{a,t}), \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.43)$$

$$\ell_{a,t} = f^{\text{remaining_useful_life}}(h_{a,t}, \rho_{a,t}, \omega_{a,t}), \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.44)$$

Here, $\omega_{a,t}$ denotes the cumulative usage state of AGV a at time step t , which can be constructed from cumulative operating time, cumulative travel distance, cumulative maintenance counts, or cumulative failure counts. Equations (3.43)–(3.44) convert the health state, energy state, and cumulative usage state into failure risk and RUL. The function $f^{\text{failure_risk}}(\cdot)$ gives the failure risk of an AGV under its current state, and function $f^{\text{remaining_useful_life}}(\cdot)$ gives the estimated remaining available time of an AGV under its current state. The health assessment functions may be obtained from data-driven predictors or rule-based degradation models; the formulation only requires $h_{a,t}$, $\rho_{a,t}$, and $\ell_{a,t}$ to be updateable on discrete time steps and available to dispatching decisions.

The computational experiments in this study use a rule-based health model. The specific degradation, failure risk, and RUL calculation rules, together with their parameter values, initialization, and maintenance-related updates, are reported in Section 5.1.

AGV failure events are characterized jointly by a health threshold and risk probability. Let $F_{a,t} \in \{0, 1\}$ indicate whether AGV a fails at time step t , $\underline{h}^{\text{failure}}$ denote the failure health threshold, p^{base} denote the base failure probability, and $\alpha^{\text{risk_amplification}}$ denote the risk amplification coefficient. The failure-triggering relation satisfies

$$b_{a,t} = 1, \quad h_{a,t} \leq \underline{h}^{\text{failure}} \Rightarrow F_{a,t} = 1, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.45)$$

For the stochastic failure branch, the probability is conditioned on the current risk state

$$\begin{aligned} \Pr(F_{a,t} = 1 \mid b_{a,t} = 1, h_{a,t} > \underline{h}^{\text{failure}}, \rho_{a,t}) \\ = p^{\text{base}} + \alpha^{\text{risk_amplification}} \rho_{a,t}, \quad \forall a \in \mathcal{A}, \quad \forall t \in \mathcal{T}. \end{aligned} \quad (3.46)$$

The probability parameters satisfy

$$0 \leq p^{\text{base}} \leq 1, \quad 0 \leq \alpha^{\text{risk_amplification}} \leq 1 - p^{\text{base}}. \quad (3.47)$$

Since $\rho_{a,t} \in [0, 1]$, these parameter restrictions imply

$$0 \leq p^{\text{base}} + \alpha^{\text{risk_amplification}} \rho_{a,t} \leq p^{\text{base}} + \alpha^{\text{risk_amplification}} \leq 1.$$

Thus, Eq (3.46) defines a valid conditional Bernoulli probability for every admissible risk value. The threshold-triggering rule in Eq (3.45) and the operability conditions in Eq (3.38) continue to apply. Failure events are generated by the environment under these rules and are not freely selectable optimization decisions.

The main PM pressure configuration uses $p^{\text{base}} = 0.00058$ and $\alpha^{\text{risk_amplification}} = 0.004375$. The probability in the stochastic branch therefore lies in $[0.00058, 0.004955]$, which satisfies Eq (3.47).

PM service and CM service restore AGVs' health state. PM recovery is triggered by a proactive maintenance decision, whereas CM recovery is triggered by the failure-repair process. The health recovery amounts satisfy

$$R_{a,t}^{\text{preventive_maintenance}} = g_{a,t}^{\text{preventive_maintenance}} \cdot f^{\text{preventive_maintenance}}(h_{a,t}, \omega_{a,t}), \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.48)$$

$$R_{a,t}^{\text{corrective_maintenance}} = g_{a,t}^{\text{corrective_maintenance}} \cdot f^{\text{corrective_maintenance}}(h_{a,t}, \omega_{a,t}), \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.49)$$

Here, $g_{a,t}^{\text{preventive_maintenance}}$ indicates whether AGV a is in PM service at time step t , $g_{a,t}^{\text{corrective_maintenance}}$ indicates whether AGV a is in CM service at time step t , $f^{\text{preventive_maintenance}}(\cdot)$ denotes the PM recovery function, and $f^{\text{corrective_maintenance}}(\cdot)$ denotes the CM recovery function. Equations (3.48)–(3.49) link maintenance service states with health recovery amounts; if the AGV is not in the corresponding maintenance service state, the corresponding recovery amount is zero. The PM recovery function and CM recovery function characterize the repair effects of proactive maintenance and failure repair on health state and cumulative usage state, respectively.

PM decisions require maintenance opportunity windows to be identified from the health state and failure risk. Let $m_{a,t}^{\text{preventive_maintenance_planning}} \in \{0, 1\}$ indicate whether AGV a is in the PM planning window at time step t , $\bar{h}^{\text{preventive_maintenance_planning}}$ denote the PM planning health threshold, and $\underline{\rho}^{\text{preventive_maintenance_planning}}$ denote the PM planning risk threshold. The PM planning window satisfies

$$\begin{aligned} m_{a,t}^{\text{preventive_maintenance_planning}} = 1 &\Leftrightarrow h_{a,t} \leq \bar{h}^{\text{preventive_maintenance_planning}} \\ &\quad \vee \rho_{a,t} \geq \underline{\rho}^{\text{preventive_maintenance_planning}}, \\ &\quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \end{aligned} \quad (3.50)$$

Equation (3.50) describes the relation determining the PM planning window. When the HI is below the PM planning health threshold or failure risk is above the PM planning risk threshold, the AGV enters the PM planning window. Variable $m_{a,t}^{\text{preventive_maintenance_planning}}$ converts continuous health assessment results into a binary maintenance opportunity state, allowing PM decisions to be used in the dispatching model as feasibility conditions or reward signals.

The urgent maintenance state characterizes AGV health states that are close to the high-risk operating region. Let $m_{a,t}^{\text{urgent_maintenance}} \in \{0, 1\}$ indicate whether AGV a is in the urgent maintenance state at time

step t , $\bar{h}^{\text{urgent_maintenance}}$ denote the urgent maintenance health threshold, and $\underline{\rho}^{\text{urgent_maintenance}}$ denote the urgent maintenance risk threshold. The urgent maintenance state satisfies

$$\begin{aligned} m_{a,t}^{\text{urgent_maintenance}} = 1 &\Leftrightarrow h_{a,t} \leq \bar{h}^{\text{urgent_maintenance}} \\ &\vee \rho_{a,t} \geq \underline{\rho}^{\text{urgent_maintenance}}, \\ &\forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \end{aligned} \quad (3.51)$$

Equation (3.51) describes the relation determining the urgent maintenance state. When the HI is below the urgent maintenance health threshold or the failure risk is above the urgent maintenance risk threshold, the AGV enters the urgent maintenance state. The variable $m_{a,t}^{\text{urgent_maintenance}}$ provides a state basis for identifying high-risk operations and counting missed maintenance opportunities, and the related cost terms are further defined in the optimization objective.

3.4. Joint dispatching and predictive maintenance decision problem

Joint AGV dispatching and predictive maintenance decisions coordinate transport task execution, charging dispatching, PM dispatching, buffer dispatching, waiting, and state keeping within the same discrete time step. Let $u_{a,t}$ denote the discrete dispatching action of AGV a at time step t . The action set satisfies

$$u_{a,t} \in \mathcal{U}_{a,t} = \{\text{task}(k), \text{charge}, \text{pm}, \text{buffer}, \text{wait}, \text{keep}\}, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.52)$$

Here, “task(k)” means that the AGV accepts transport task k , “charge” means that the AGV moves to the charging area, “pm” means that the AGV moves to the PM area, “buffer” means that the AGV moves to the buffer area, “wait” means that the AGV waits, and “keep” means that the AGV maintains its current execution state. Equation (3.52) incorporates transport dispatching actions and equipment maintenance actions into the same action set. The action set $\mathcal{U}_{a,t}$ corresponds to the binary action variables in Eq (3.29), enabling an AGV to express both operational choices and equipment state management choices through one discrete action at a single time step.

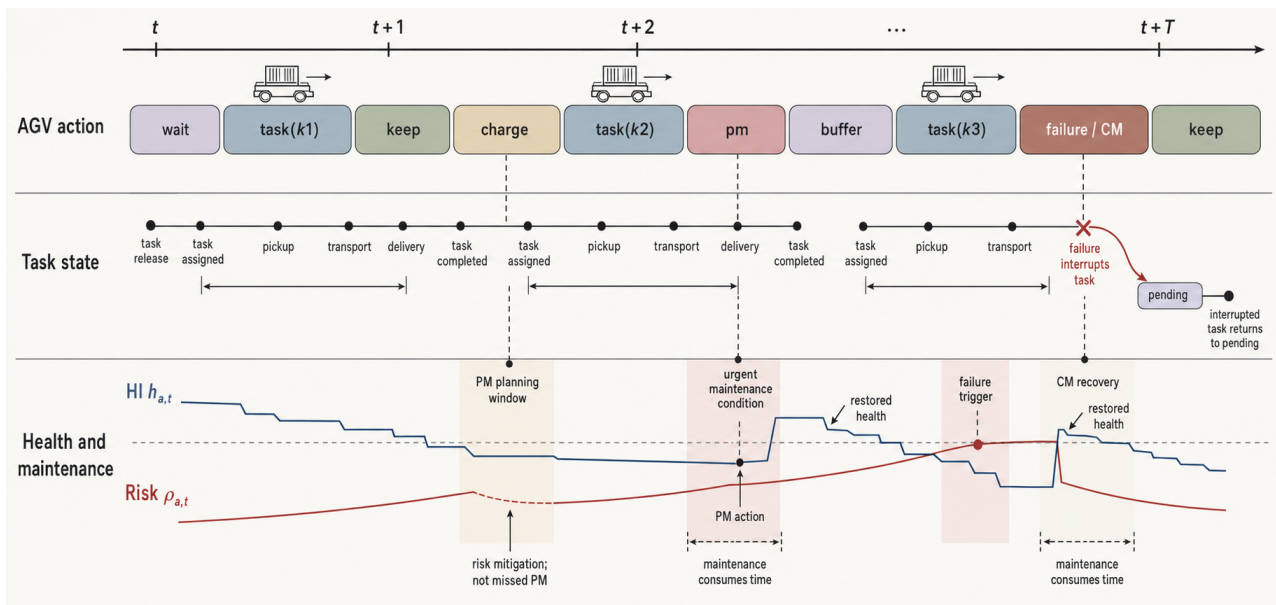


Figure 2. Timeline of joint dispatching and predictive maintenance decisions.

On the discrete timeline, the action sequence of an AGV evolves synchronously with task states and health states, as shown in Figure 2. The task(k) action triggers task assignment, pickup, transport, and delivery, while the charge and pm actions occupy the AGV's available time and change the subsequent risk states; the keep action continues an ongoing transport, charging, maintenance, or repair process. A failure event interrupts the current transport task and returns the interrupted task to the pending set; therefore, the action choices, task completion benefits, maintenance costs, failure losses, and health risks accumulate on the same episode timeline.

A task action represents a new assignment that satisfies the pending state and joint occupation constraints

$$u_{a,t} = \text{task}(k) \Leftrightarrow x_{akt} = 1, \quad \forall a \in \mathcal{A}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}. \quad (3.53)$$

The assignment is subject to Eqs (3.14), (3.16), (3.17), and (3.28). It initializes the execution phases through Eq (3.19). Thereafter, Eq (3.23) retains the task on the same AGV until successful completion or failure-induced interruption.

PM decisions are subject to AGV assignability, the PM planning window, and maintenance resource capacity. PM action feasibility satisfies

$$u_{at}^{\text{preventive_maintenance}} \leq m_{a,t}^{\text{preventive_maintenance_planning}}, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.54)$$

$$u_{a,t} = \text{pm} \Leftrightarrow u_{at}^{\text{preventive_maintenance}} = 1, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.55)$$

Here, $m_{a,t}^{\text{preventive_maintenance_planning}}$ is defined by Eq (3.50) and indicates whether the AGV is in the PM planning window. Equation (3.54) restricts the AGV to choose the PM action only when it has entered the PM planning window, and Eq (3.55) links the unified action variable $u_{a,t}$ with the PM action variable $u_{at}^{\text{preventive_maintenance}}$. PM decisions are also restricted by the assignability constraint in Eq (3.28) and the maintenance resource capacity constraint in Eq (3.35); therefore, PM decisions reflect the coupling among the health state, AGV operating state, and maintenance area resource occupation.

Charging decisions are subject to AGVs' assignability, energy state, and charging resource capacity. Let $\bar{e}^{\text{charging}}$ denote the upper energy threshold for the charging action. AGVs' charging-action feasibility satisfies

$$u_{at}^{\text{charging}} \leq \mathbb{I}(e_{a,t} < \bar{e}^{\text{charging}}), \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.56)$$

$$u_{a,t} = \text{charge} \Leftrightarrow u_{at}^{\text{charging}} = 1, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.57)$$

Here, $\mathbb{I}(\cdot)$ denotes the indicator function, and $e_{a,t}$ denotes an AGV's SOC. Equation (3.56) prevents an AGV that is close to full charge from selecting the charging action, and Eq (3.57) links the unified action variable $u_{a,t}$ with the charging action variable u_{at}^{charging} . Charging decisions are also restricted by the assignability constraint in Eq (3.28) and the charging resource capacity constraint in Eq (3.34); therefore, charging decisions reflect the relation among the AGV's energy state and operating state, and the charging area's resource occupation.

Buffer dispatching and waiting decisions handle spatial occupation when an AGV has no transport task and is not charging or entering maintenance. The buffer action, waiting action, and state-keeping action satisfy

$$u_{a,t} = \text{buffer} \Leftrightarrow u_{at}^{\text{buffer}} = 1, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.58)$$

$$u_{a,t} = \text{wait} \Leftrightarrow u_{at}^{\text{waiting}} = 1, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.59)$$

$$u_{a,t} = \text{keep} \Leftrightarrow u_{at}^{\text{keeping}} = 1, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.60)$$

Equations (3.58)–(3.60) link the unified action variable with the buffer dispatching, waiting, and state-keeping action variables. The buffer action is restricted by the assignability constraint in Eq (3.28), the waiting action means that an AGV does not initiate a new resource transfer at its current position, and the state-keeping action means that an AGV continues its current transport, charging, maintenance, or repair process. Buffer dispatching, waiting, and state-keeping complement the operating state representation beyond transport, charging, and PM actions, so each AGV has a well-defined dispatching action at every time step.

A failure interrupts the task occupied by the AGV during the current interval, including a task newly assigned at its beginning. The interruption event satisfies

$$\begin{aligned} q_{akt} &\leq F_{a,t}, \\ q_{akt} &\leq \widehat{y}_{akt}, \\ q_{akt} &\geq F_{a,t} + \widehat{y}_{akt} - 1, \end{aligned} \quad \forall a \in \mathcal{A}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}. \quad (3.61)$$

Thus, $q_{akt} = 1$ exactly when a failure occurs while the AGV occupies task k . Together with Eq (3.21), this definition prevents simultaneous successful completion and interruption.

The exact pending-state update is given by Eq (3.24). In particular, an interrupted task is pending at the next boundary

$$p_{k,t+1} \geq \sum_{a \in \mathcal{A}} q_{akt}, \quad \forall k \in \mathcal{K}, \forall t \in \mathcal{T}. \quad (3.62)$$

The model adopts a task restart abstraction: After interruption, the task may be reassigned and its execution phases are initialized again from its original task origin. Physical recovery of an interrupted container and in-transit transfer between vehicles are not represented separately in this abstraction. Reassignment changes task ownership only after interruption; it does not transfer an uninterrupted task between AGVs.

The high-risk operation state identifies cases in which an AGV continues executing a transport task under high failure risk. Let $r_{a,t}^{\text{high_risk_operation}} \in \{0, 1\}$ indicate whether AGV a is in a high-risk operation state at time step t , and let $\underline{\rho}^{\text{high_risk_operation}}$ denote the high-risk operation threshold. The high-risk operation state satisfies

$$r_{a,t}^{\text{high_risk_operation}} \leq \sum_{k \in \mathcal{K}} \widehat{y}_{akt}, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.63)$$

$$r_{a,t}^{\text{high_risk_operation}} \leq \mathbb{I}(\rho_{a,t} \geq \underline{\rho}^{\text{high_risk_operation}}), \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.64)$$

$$r_{a,t}^{\text{high_risk_operation}} \geq \sum_{k \in \mathcal{K}} \widehat{y}_{akt} + \mathbb{I}(\rho_{a,t} \geq \underline{\rho}^{\text{high_risk_operation}}) - 1, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}. \quad (3.65)$$

Equations (3.63)–(3.65) describe the logical relation of the high-risk operation state. When an AGV is executing a transport task and the failure risk is no lower than the threshold $\underline{\rho}^{\text{high_risk_operation}}$, the high-risk operation variable $r_{a,t}^{\text{high_risk_operation}}$ takes a value of one. The high-risk operation state links health-risk representation with transport task execution status and provides a counting variable for the high-risk operation cost in the optimization objective.

Joint dispatching and predictive maintenance decisions consist of task actions, charging actions, PM actions, buffer actions, waiting actions, and state-keeping actions, and are constrained by the task release,

resource capacity, health state, and failure state. Transport task actions refer to Eqs (3.14)–(3.16); AGV actions' mutual exclusiveness refers to Eq (3.29); charging and maintenance resource capacities refer to Eqs (3.34) and (3.35), respectively; and health state and failure risk refer to Eqs (3.41)–(3.46). These relations constitute the joint decision feasible region for AGV transport dispatching and predictive maintenance.

3.5. Optimization objectives

The economic criterion for joint dispatching and predictive maintenance includes task waiting, energy consumption, maintenance expenditure, failure loss, high-risk operation, missed PM opportunity, and task completion benefit. Let C^{net} denote the total system net cost over one realized trajectory within the scheduling horizon. It is defined as

$$\begin{aligned} C^{\text{net}} = & C^{\text{delay}} + C^{\text{energy}} + C^{\text{preventive_maintenance}} \\ & + C^{\text{corrective_maintenance}} + C^{\text{failure}} \\ & + C^{\text{high_risk_operation}} + C^{\text{missed_preventive_maintenance}} \\ & - B^{\text{task_completion}}. \end{aligned} \quad (3.66)$$

The cost terms represent pending-task waiting, energy use, PM, CM, failure events, high-risk operation, and missed PM opportunities, respectively. The benefit term $B^{\text{task_completion}}$ represents the service benefit generated by completed transport tasks. All terms are evaluated along the same realized trajectory, using the component definitions below.

Because the trajectory depends on stochastic state transitions and failure events, C^{net} is a random variable before the trajectory is realized. Equation (3.66) defines its realized value. The corresponding expected cost optimization criterion is specified in Eq (3.78).

Released tasks that remain unassigned incur a pending-task waiting cost. Waiting is accounted for using the task state after assignment and interval-level events, before tasks released at boundary $t + 1$ enter the next interval. This is a discrete end-of-interval accounting convention.

Let $\Delta t_{\min}$ denote the duration of one decision interval in minutes. For tasks released by boundary t , the pending indicator used for cost accounting is

$$\widehat{p}_{k,t} = p_{k,t} - \sum_{a \in \mathcal{A}} x_{akt} + \sum_{a \in \mathcal{A}} q_{akt}, \quad \forall k \in \mathcal{K}, \forall t \in \mathcal{T}. \quad (3.67)$$

By Eq (3.24), $\widehat{p}_{k,t} = p_{k,t+1} - \mathbb{I}(r_k = t + 1)$. Thus, tasks first released at boundary $t + 1$ are not charged a waiting cost for interval t .

The quayside and yardside pending-task waiting amounts are

$$\begin{aligned} W_t^{\text{quay_crane}} &= \Delta t_{\min} \sum_{\substack{k \in \mathcal{K}: \\ o_k \in \mathcal{N}^{\text{quay_crane}}}} \widehat{p}_{k,t}, \\ W_t^{\text{yard}} &= \Delta t_{\min} \sum_{\substack{k \in \mathcal{K}: \\ o_k \in \mathcal{N}^{\text{yard}}}} \widehat{p}_{k,t}. \end{aligned} \quad (3.68)$$

These quantities are measured in task-minutes. A task assigned in the current interval leaves the pending count unless it is returned by interruption. Under this accounting convention, a task returned by failure is included in the current interval's end-of-interval pending count.

The waiting cost is

$$C^{\text{delay}} = c^{\text{delay}} \sum_{t \in \mathcal{T}} (W_t^{\text{quay_crane}} + W_t^{\text{yard}}). \quad (3.69)$$

Here, c^{delay} is the cost per task-minute of pending-task waiting. The cost measures the unassigned backlog, including tasks returned after interruption. It does not measure the amount by which task completion exceeds a due time.

AGVs' travel and waiting consume energy, and energy consumption is converted into the operating cost. Let c^{energy} denote the energy cost per unit, and let $E_{a,t}^{\text{empty_travel}}$, $E_{a,t}^{\text{loaded_travel}}$, and $E_{a,t}^{\text{waiting}}$ denote empty-travel energy consumption, loaded-travel energy consumption, and waiting energy consumption of AGV a at time step t , respectively. The energy consumption cost is defined as

$$C^{\text{energy}} = c^{\text{energy}} \sum_{a \in \mathcal{A}} \sum_{t \in \mathcal{T}} (E_{a,t}^{\text{empty_travel}} + E_{a,t}^{\text{loaded_travel}} + E_{a,t}^{\text{waiting}}). \quad (3.70)$$

Equation (3.70) describes the energy use cost in AGVs' horizontal transport. The empty-travel energy consumption $E_{a,t}^{\text{empty_travel}}$ reflects the energy use caused by unloaded AGV movement, the loaded-travel energy consumption $E_{a,t}^{\text{loaded_travel}}$ reflects the energy use caused by container transport, and the waiting energy consumption $E_{a,t}^{\text{waiting}}$ reflects the energy use caused by AGVs' waiting or idling. The energy consumption cost term makes the dispatching policy consider AGVs' travel distance, load status, and waiting state while completing transport tasks.

PM service and CM service generate the proactive maintenance cost and failure repair cost, respectively. Let $I_{a,t}^{\text{preventive_maintenance}} \in \{0, 1\}$ indicate whether AGV a starts PM service at time step t , $I_{a,t}^{\text{corrective_maintenance}} \in \{0, 1\}$ indicate whether AGV a starts CM service at time step t , $c^{\text{preventive_maintenance}}$ denote the cost per PM event, and $c^{\text{corrective_maintenance}}$ denote the cost per CM event. The maintenance costs are defined as

$$C^{\text{preventive_maintenance}} = c^{\text{preventive_maintenance}} \sum_{a \in \mathcal{A}} \sum_{t \in \mathcal{T}} I_{a,t}^{\text{preventive_maintenance}}, \quad (3.71)$$

$$C^{\text{corrective_maintenance}} = c^{\text{corrective_maintenance}} \sum_{a \in \mathcal{A}} \sum_{t \in \mathcal{T}} I_{a,t}^{\text{corrective_maintenance}}. \quad (3.72)$$

Equations (3.71)–(3.72) describe the economic expenditure caused by maintenance service events. The PM cost is measured by proactive maintenance start events, and the CM cost is measured by failure repair start events. Event variables $I_{a,t}^{\text{preventive_maintenance}}$ and $I_{a,t}^{\text{corrective_maintenance}}$ count the number of maintenance service starts rather than maintenance service duration. The maintenance cost terms create a tradeoff between reducing failure risk and controlling maintenance frequency.

An AGV failure causes a fixed loss and triggers task interruption and the CM service process. Let c^{failure} denote the fixed loss per failure event, and let $F_{a,t}$ indicate whether AGV a fails at time step t . The failure loss is defined as

$$C^{\text{failure}} = c^{\text{failure}} \sum_{a \in \mathcal{A}} \sum_{t \in \mathcal{T}} F_{a,t}. \quad (3.73)$$

Equation (3.73) describes the fixed economic loss caused by AGV failure events. The failure event variable $F_{a,t}$ is given by Eqs (3.45)–(3.46) and affects transport tasks' continuity through Eqs (3.61)–(3.62). The failure loss term is measured separately from the CM cost term: The former characterizes

the operational impact of the failure event itself, while the latter characterizes the repair expenditure after failure.

When an AGV continues executing transport tasks under a high failure risk, the system's operational risk increases. Let $c^{\text{high_risk_operation}}$ denote the high-risk operation cost per time step, and let $r_{a,t}^{\text{high_risk_operation}}$ indicate whether AGV a is in a high-risk operation state at time step t . The high-risk operation cost is defined as

$$C^{\text{high_risk_operation}} = c^{\text{high_risk_operation}} \sum_{a \in \mathcal{A}} \sum_{t \in \mathcal{T}} r_{a,t}^{\text{high_risk_operation}}. \quad (3.74)$$

Equation (3.74) describes the risk cost caused by continued AGV operation in a high-risk state. The high-risk operation variable $r_{a,t}^{\text{high_risk_operation}}$ is defined by Eqs (3.63)–(3.65) and depends jointly on transport task execution status and the failure risk threshold. The high-risk operation cost term encourages the dispatching policy to reduce the tendency of high-risk AGVs to continue undertaking transport tasks, and provides optimization pressure for risk mitigation actions such as PM or charging.

The missed PM cost uses a penalty convention based on the urgent maintenance state and the selected action. Let $\mu_{a,t} \in \{0, 1\}$ denote the associated binary penalty variable for AGV a at time step t , and let $c^{\text{missed_preventive_maintenance}}$ denote the cost per counted opportunity. The term “missed PM opportunity” refers to this specified penalty convention, rather than a direct count of all unmet physical maintenance needs. The penalty variable satisfies the following lower-bound relation:

$$\mu_{a,t} \geq m_{a,t}^{\text{urgent_maintenance}} - u_{at}^{\text{preventive_maintenance}} - u_{at}^{\text{charging}}, \quad \forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \quad (3.75)$$

$$C^{\text{missed_preventive_maintenance}} = c^{\text{missed_preventive_maintenance}} \sum_{a \in \mathcal{A}} \sum_{t \in \mathcal{T}} \mu_{a,t}. \quad (3.76)$$

Equation (3.75) forces $\mu_{a,t} = 1$ when the AGV is in the urgent maintenance state and neither PM nor charging is selected. Selecting PM or charging removes this forced lower bound, although the inequality alone does not require $\mu_{a,t} = 0$. The charging exemption is an action-level penalty convention; it does not imply that charging restores the equipment's health or satisfies a maintenance need caused by low HI. The explicit counter used in the computational implementation is given in Eq (4.22). That counter additionally requires the PM action to be available, a condition that was not included in Eq (3.75).

Transport task completion generates service benefit, which offsets the system's operating cost. Let $b^{\text{task_completion}}$ denote the service benefit of one completed task, and let $\eta_{k,T}$ indicate whether task k has been completed by the end of the scheduling horizon. The task completion benefit is defined as

$$B^{\text{task_completion}} = b^{\text{task_completion}} \sum_{k \in \mathcal{K}} \eta_{k,T}. \quad (3.77)$$

Equation (3.77) describes the service benefit generated when AGVs complete quay–yard transport tasks. The completion status variable $\eta_{k,T}$ is obtained from the updates in Eqs (3.25)–(3.26). The service benefit term enters the net cost objective in Eq (3.66) as a negative term, allowing the optimization problem to retain an incentive for completing transport tasks while reducing maintenance risk and operating costs.

Let Π^{na} denote the class of nonanticipative joint dispatching policies. A policy in this class selects actions using the information available at the current decision boundary, including past observations, past actions, and current feasibility information. It cannot condition an action on a future failure realization or another unobserved future disturbance.

Let μ_0 denote the specified initial state distribution and let P denote the environment transition law, as introduced in Section 4.1. Together with a policy π , these quantities induce a probability distribution over operating trajectories. The finite-horizon, risk-neutral economic objective is

$$\begin{aligned} \min_{\pi \in \Pi^{\text{na}}} \quad & \mathbb{E}_{\mu_0, \pi, P} [C^{\text{net}}] \\ \text{s.t.} \quad & (3.1) - (3.11), \\ & (3.12) - (3.38), \\ & (3.39) - (3.51), \\ & (3.52) - (3.65), \\ & (3.69) - (3.77). \end{aligned} \tag{3.78}$$

The deterministic feasibility and accounting relations are required to hold along realized trajectories with probability one. The stochastic failure relations specify the transition law P . Failure realizations remain environment-generated events under that law.

The expectation accounts for the initial state variation, stochastic degradation disturbances, and failure draws represented by the environment, as well as policy randomization when present. For evaluations using a fixed operation plan, the expectation is conditional on that plan.

The optimization variable is the nonanticipative policy. Equation (3.78) therefore evaluates decisions before future random events are known. Section 4 introduces a reward-based learning surrogate for this economic criterion. The simulation abstractions stated in Section 5.1 delimit the implementation scope of the reported experiments.

4. Proposed method

This section presents the HA-QMIX method for solving the joint decision problem. The health-aware Markov game is formulated in Section 4.1; the observation, state, and action representations are described in Section 4.2; action masking and top-K candidate selection are detailed in Section 4.3; and the reward transformation is given in Section 4.4. The network's architecture, training and checkpoint-selection mechanism, and complete algorithm are provided in Sections 4.5–4.7.

4.1. Health-aware Markov game formulation

The formulation in Section 3 defines a finite-horizon expected economic cost for joint transport dispatching and predictive maintenance under stochastic operating conditions. HA-QMIX learns an online dispatching policy through a multiagent Markov game with health states. Each AGV corresponds to one agent and selects a dispatching action using its available observations and action history. The environment advances task execution, charging, maintenance, health degradation, and failure repair processes according to the joint action. The training criterion is the expected discounted team reward in Eq (4.3). Its relationship to the economic cost criterion is explained in Section 4.4.

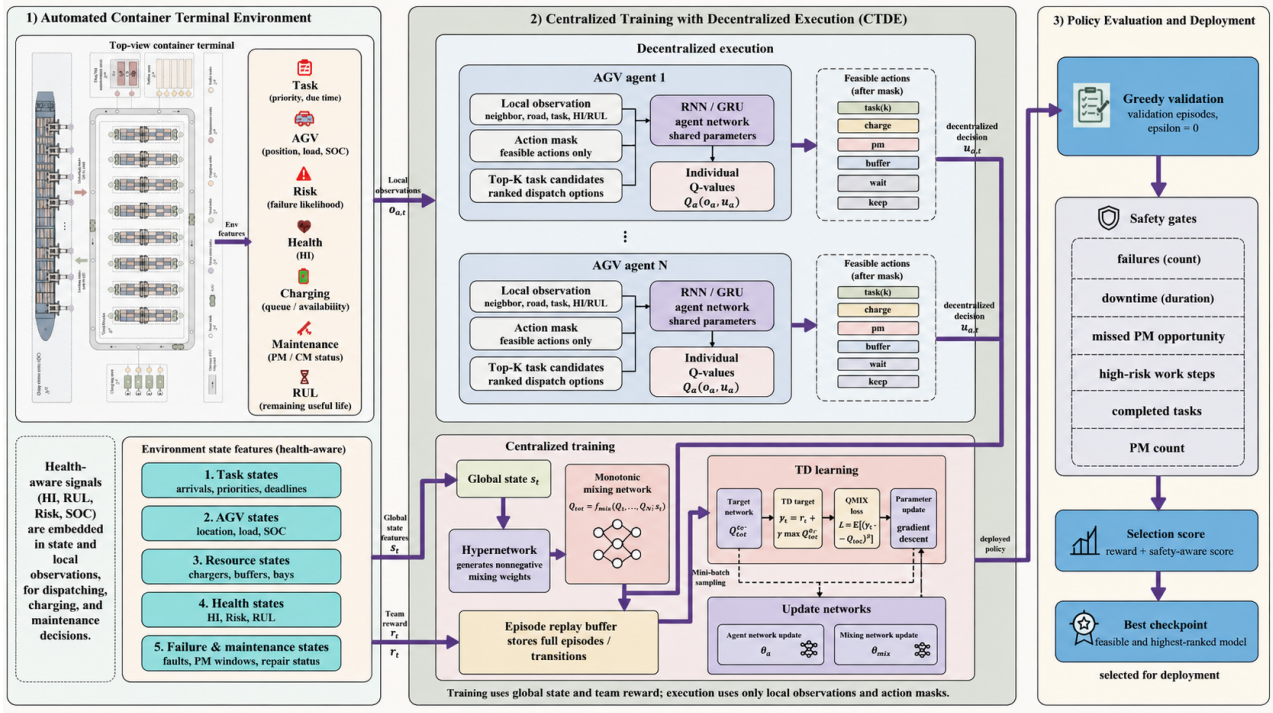


Figure 3. Overall architecture of the proposed HA-QMIX framework.

The HA-QMIX framework connects the automated terminal simulation environment, health-aware state representation, local multi-AGV decisions, and centralized training into a closed loop, as shown in Figure 3. The environment side generates the task states, vehicle states, resource states, health states, and failure–maintenance states; on the execution side, each AGV outputs individual Q-values based on local observations, top-K task candidates, and action masks; on the training side, global states, team rewards, experience replay, and a monotonic mixing network are used to update the joint action value. Greedy validation, safety gates, selection score, and the best checkpoint on the right side correspond to the training-stage model-selection procedure, so the deployed policy does not depend directly on the final checkpoint.

The Markov game with health states is represented as

$$\mathcal{M} = \langle \mathcal{A}, \mathcal{S}, \{\mathcal{O}_a\}_{a \in \mathcal{A}}, \{\mathcal{U}_a\}_{a \in \mathcal{A}}, P, R, \gamma \rangle. \quad (4.1)$$

Here, $\mathcal{A}$ denotes the set of AGV agents, $\mathcal{S}$ denotes the global state space, $\mathcal{O}_a$ denotes the local observation space of AGV a , $\mathcal{U}_a$ denotes the action space of AGV a , P denotes the state transition function, R denotes the team reward function, and $\gamma \in (0, 1]$ denotes the discount factor. Equation (4.1) specifies the state, observation, action, transition, and reward components of the learning problem. The global state $s_t \in \mathcal{S}$ records the system-level task, resource, and fleet health information; the local observation $o_{a,t} \in \mathcal{O}_a$ records information available for autonomous decision-making by AGV a ; and the joint action $\mathbf{u}_t = (u_{1,t}, \dots, u_{|\mathcal{A}|,t})$ consists of the local actions of all AGVs.

State transitions are jointly generated by the terminal simulation environment and the health degradation model. After the joint action is executed, the environment first determines the action's feasibility according to action masks, then advances AGV's travel, service, charging, maintenance, and failure repair processes, and finally updates the transport task completion status and health states. The

state-transition relation satisfies

$$s_{t+1} \sim P(\cdot \mid s_t, \mathbf{u}_t), \quad \mathbf{u}_t \in \mathcal{U}_{1,t} \times \cdots \times \mathcal{U}_{|\mathcal{A}|,t}. \quad (4.2)$$

The stochasticity in Eq (4.2) comes from task release, health degradation perturbations, risk-probability-based failure triggering, and resource queue states. Transport task evolution follows the phase processing constraints in Eq (3.20), the completion rule in Eq (3.21), the state updates in Eqs (3.23)–(3.25), the health and failure state updates refer to Eqs (3.41)–(3.46), and maintenance resource occupation refers to Eq (3.35). The policy in Eq (4.3) is learned with the task actions restricted by the top-K candidate mechanism in Section 4.3. This mechanism can exclude task choices permitted by the formulation in Section 3, so equivalence with its unrestricted policy space is not assumed. The implementation scope stated in Section 5.1 also applies.

The AGV fleet follows the centralized training with decentralized execution (CTDE) paradigm. During training, the global state s_t is used to estimate the joint action value; during execution, each AGV selects actions only according to local action values and action masks. The objective is to maximize the discounted cumulative team reward

$$\max_{\pi} J(\pi) = \mathbb{E}_{\pi, P} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right], \quad (4.3)$$

Here, $\pi = \{\pi_a\}_{a \in \mathcal{A}}$ denotes the joint policy of the AGV fleet, $\pi_a(u_{a,t} \mid o_{a,t})$ denotes the local policy of AGV a , and $r_t = R(s_t, \mathbf{u}_t, s_{t+1})$ denotes the team reward at time step t . The expectation in Eq (4.3) includes the specified initial state distribution μ_0 , which is suppressed in the notation $\mathbb{E}_{\pi, P}$.

The economic criterion in Eq (3.78) minimizes expected undiscounted net cost over the finite horizon. HA-QMIX uses the expected discounted shaped reward as its learning criterion. Discounting and separately specified reward weights can change the relative ranking of the policies. Exact equivalence between these two objectives is therefore not assumed.

4.2. Observation, global state, and action design

Local observations are designed to support an AGV in evaluating the relative values of transport, charging, PM, and waiting actions without accessing the full system state. The local observation of AGV a at time step t consists of vehicle state, resource pressure, health state, and candidate task features

$$o_{a,t} = [\phi_{a,t}^{\text{vehicle}}, \phi_t^{\text{resource}}, \phi_{a,t}^{\text{health}}, \phi_{a,t}^{\text{task}}]. \quad (4.4)$$

Here, $\phi_{a,t}^{\text{vehicle}}$ denotes the position and operating state features of AGV a , including the normalized node coordinates, busy/idle status, failure state, and maintenance state; ϕ_t^{resource} denotes the resource pressure features, including the maintenance area occupation ratio and the ratio of released pending tasks; $\phi_{a,t}^{\text{health}}$ denotes SOC, HI, RUL, and the risk of AGV a ; and $\phi_{a,t}^{\text{task}}$ denotes the candidate task features for AGV a . Equation (4.4) links the AGV's own state, the equipment's health state, and local task opportunities, allowing the local policy to consider the transport benefit and maintenance risk simultaneously.

Candidate task features are concatenated according to the order of top-K candidate tasks. For the j -th candidate task $k_{a,t}^j$ of AGV a , the task feature is defined as

$$\phi_{a,t,j}^{\text{task}} = [\tilde{d}_{a,o(k_{a,t}^j),t}, \tilde{d}_{o(k_{a,t}^j),d(k_{a,t}^j)}, \tilde{p}_{k_{a,t}^j}, \tilde{l}_{k_{a,t}^j,t}, \xi_{k_{a,t}^j}^{\text{loading}}], \quad j = 1, \dots, K. \quad (4.5)$$

Here, $\tilde{d}_{a,o(k),t}$ denotes the normalized distance from the current position of AGV a to the origin of task k , $\tilde{d}_{o(k),d(k)}$ denotes the normalized transport distance from the origin to the destination of task k , $\tilde{p}_k$ denotes the normalized task priority, $\tilde{l}_{k,t}$ denotes the normalized remaining time before the task's due time, and ξ_k^{loading} denotes a binary indicator of whether the task is a loading task. If the number of candidate tasks is less than K , empty positions are filled with fixed padding values. Equation (4.5) incorporates the task's reachability, transport length, priority, time pressure, and task direction into the local observation.

Centralized training requires the global state to describe the overall system workload and the fleet's health distribution. The global state is constructed by concatenating the system's statistical features and AGV state features

$$s_t = [\phi_t^{\text{system}}, \phi_{1,t}^{\text{fleet}}, \dots, \phi_{|\mathcal{A}|,t}^{\text{fleet}}]. \quad (4.6)$$

Here, ϕ_t^{system} denotes system-level features, including the normalized time, released pending-task ratio, task completion ratio, failed AGV ratio, maintenance AGV ratio, charging AGV ratio, average SOC, average HI, and average risk; $\phi_{a,t}^{\text{fleet}}$ denotes the fleet-state features of AGV a , including SOC, HI, risk, RUL, and busy/idle status. Equation (4.6) allows the mixing network to use the system's congestion, resource occupation, and health distribution information to estimate the joint action value during training.

The AGV action space follows the joint action variables in Section 3 and uses a fixed-dimensional discrete action representation. The action space is defined as

$$\mathcal{U}_{a,t} = \{\text{wait}, \text{task}(k_{a,t}^1), \dots, \text{task}(k_{a,t}^K), \text{charge}, \text{pm}, \text{buffer}, \text{keep}\}. \quad (4.7)$$

Here, “wait” denotes the waiting action, “task($k_{a,t}^j$)” denotes accepting the j -th candidate task, “charge” denotes moving to the charging area, “pm” denotes moving to the PM area, “buffer” denotes moving to the buffer area, and “keep” denotes maintaining the current execution state. Equation (4.7) incorporates task selection, energy replenishment, predictive maintenance, and state keeping into a unified action space. The fixed action dimension facilitates the neural network's action value output, and candidate task mapping keeps the policy interface stable when the task set changes over time.

4.3. Action masking and top- K task candidate mechanism

The number of pending transport tasks changes over time, and directly representing every pending task would make the action dimension depend on task scale. The top- K mechanism retains at most K task candidates per AGV using a heuristic ranking. It provides a fixed-size task action interface by restricting the available task choices. Candidate screening, fixed-dimensional action entries, and action masking jointly form the action selection mechanism shown in Figure 4.

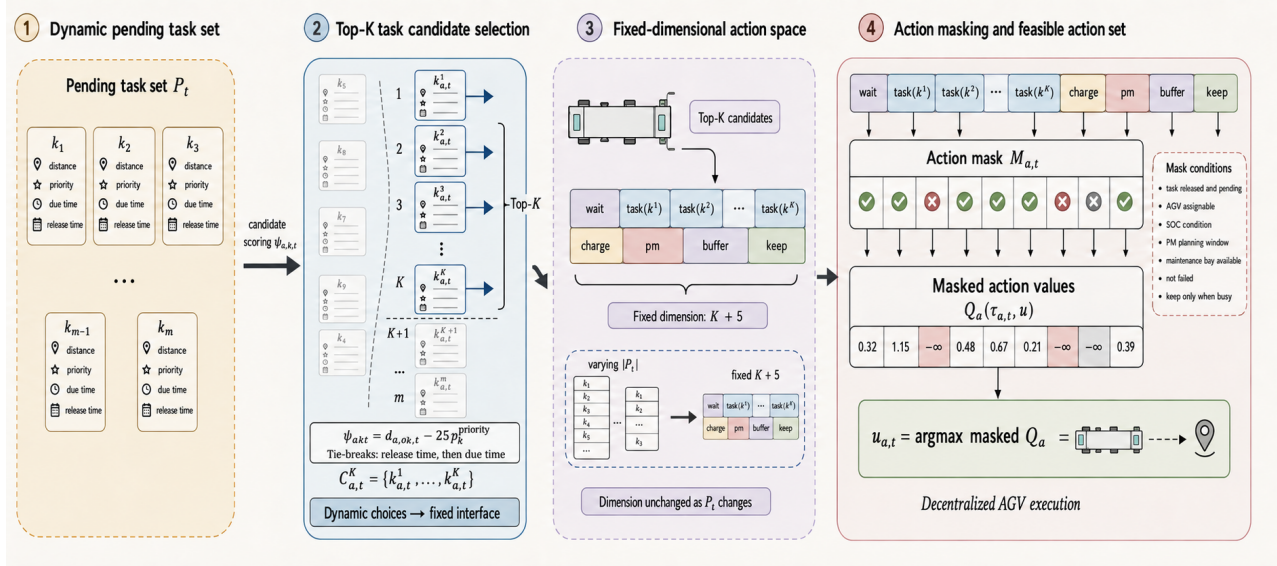


Figure 4. Action masking and top-K task candidate mechanism.

Pending tasks are ranked for each AGV by a score combining distance and priority, with the release time and due time used to break ties. Up to K tasks are retained as candidates. The fixed action space consists of K task entries and five nontask actions, namely “wait”, “charge”, “pm”, “buffer”, and “keep”, so the action dimension remains $K + 5$. The action mask further removes infeasible actions according to task status, AGV assignability, SOC, PM planning window, maintenance resource capacity, failure state, and keep action conditions; masked Q-values participate in greedy selection only over the feasible action set.

The candidate mechanism uses the pending state defined and updated in Section 3.2. Before dispatching at boundary t , the pending task set is

$$\mathcal{P}_t = \{k \in \mathcal{K} \mid p_{k,t} = 1\}. \quad (4.8)$$

By Eq (3.13), this set contains exactly the released tasks that are neither assigned nor completed. It is updated by release, assignment, and failure-induced requeueing through Eq (3.24). The implementation uses the following primary ranking score for task k and AGV a :

$$\psi_{akt} = d_{a,ok,t} - \lambda^{\text{priority}} p_k^{\text{priority}}, \quad k \in \mathcal{P}_t. \quad (4.9)$$

Here, $d_{a,ok,t}$ denotes the route distance in meters from the AGV’s recorded node to the task origin, and p_k^{priority} denotes the task’s un-normalized priority value. The implementation uses $\lambda^{\text{priority}} = 25$ m per priority unit. These raw ranking inputs are distinct from the normalized candidate features supplied to the neural network.

Tasks are ordered lexicographically by $(\psi_{akt}, r_k, \bar{t}_k)$, with smaller values preceding larger values. Thus, an earlier release time breaks a tie in the primary score, and an earlier due time breaks a remaining tie. If all three entries are equal, their order in the input task list is retained. The release time and due time are therefore tie-breaking criteria rather than additive terms in Eq (4.9).

The selection operator below returns the first $\min\{K, |\mathcal{P}_t|\}$ tasks under this ordering as follows:

$$C_{a,t}^{\text{top-K}} = \arg \min_{k \in \mathcal{P}_t}^K (\psi_{akt}), \quad |C_{a,t}^{\text{top-K}}| \leq K. \quad (4.10)$$

The tie-breaking rule above applies to this selection operator. When no task is pending, the candidate list is empty.

Top-K screening imposes an additional restriction on task choices relative to the formulation in Section 3. A pending task outside $C_{a,t}^{\text{top-K}}$ is unavailable to AGV a at the current decision boundary, even if it satisfies the other task assignment conditions.

Considering candidate screening alone, with the transition law, information structure, objective, and other admissibility conditions held fixed, the restriction narrows the policy class and can change the best attainable objective value. Keeping the task release, task occupation, and task completion relations unchanged does not preserve the unrestricted policy space.

Excluded tasks remain in the pending pool and may enter a later candidate list. However, later reconsideration does not guarantee recovery of an optimal decision opportunity excluded at the current boundary. HA-QMIX therefore learns within a candidate-restricted policy space, without a guarantee that top-K screening preserves an optimal policy of the unrestricted problem.

Action masking removes infeasible actions from action value maximization. The action mask of AGV a at time step t is defined as

$$M_{a,t}(u) = \begin{cases} 1, & u \text{ is feasible given } s_t \text{ and current task reservations,} \\ 0, & \text{otherwise.} \end{cases} \quad (4.11)$$

During action selection, the masked Q-value is defined as

$$\bar{Q}_a(\tau_{a,t}, u) = \begin{cases} Q_a(\tau_{a,t}, u), & M_{a,t}(u) = 1, \\ -\infty, & M_{a,t}(u) = 0. \end{cases} \quad (4.12)$$

Here, $\tau_{a,t}$ denotes the local action observation history of AGV a up to time step t , and $Q_a(\tau_{a,t}, u)$ denotes the local action value. Equations (4.11)–(4.12) prevent infeasible actions from participating in greedy action selection and target action selection, thereby reducing invalid exploration.

Individual task masks must be combined with joint assignment control because several AGVs may include the same task in their candidate lists. At each boundary, candidate lists are constructed from the predecision pending set and remain fixed during action selection. AGVs select actions in a fixed vehicle index order. Once an AGV selects an available task, that task is reserved before the next AGV selects its action.

Let $\mathcal{R}_{a,t}$ denote the tasks reserved by AGVs processed before AGV a in the current interval. The reservation set is initialized by $\mathcal{R}_{1,t} = \emptyset$. The task action mask is

$$M_{a,t}(\text{task}(k)) = \mathbb{I}(k \in C_{a,t}^{\text{top-K}}, \chi_{a,t} = 1, p_{k,t} = 1, k \notin \mathcal{R}_{a,t}). \quad (4.13)$$

After AGV a selects its action, the reservation set becomes

$$\mathcal{R}_{a+1,t} = \begin{cases} \mathcal{R}_{a,t} \cup \{k\}, & u_{a,t} = \text{task}(k), \\ \mathcal{R}_{a,t}, & \text{otherwise.} \end{cases} \quad (4.14)$$

Only the availability of candidate entries changes during this procedure; their identities and ordering remain fixed. The reservation step enforces the “one AGV per task” constraint even when the candidate lists overlap.

An AGV retaining an unfinished task has only the keeping action available

$$\sum_{k \in \mathcal{K}} y_{akt} = 1 \Rightarrow \begin{cases} M_{a,t}(\text{keep}) = 1, \\ M_{a,t}(u) = 0, \quad u \neq \text{keep}. \end{cases} \quad (4.15)$$

This condition prevents a new task, charging dispatch, PM dispatch, or buffer dispatch from overwriting an ongoing task. Charging, PM, buffer, and keeping actions for other vehicle states satisfy the following conditions:

$$M_{a,t}(\text{charge}) = 1 \Rightarrow \chi_{a,t} = 1, \quad e_{a,t} < \bar{e}^{\text{charging}}, \quad (4.16)$$

$$M_{a,t}(\text{pm}) = 1 \Rightarrow \chi_{a,t} = 1, \quad m_{a,t}^{\text{preventive_maintenance_planning}} = 1, \\ \sum_{a' \in \mathcal{A}} (g_{a't}^{\text{preventive_maintenance}} + g_{a't}^{\text{corrective_maintenance}}) < B^{\text{maintenance}}, \quad (4.17)$$

$$M_{a,t}(\text{buffer}) = 1 \Rightarrow \chi_{a,t} = 1, \quad M_{a,t}(\text{keep}) = 1 \Rightarrow \chi_{a,t} = 0. \quad (4.18)$$

Equations (4.13)–(4.15) connect candidate selection with the task state process. The pending state test excludes unreleased, assigned, and completed tasks; the assignability test excludes unavailable AGVs; and sequential reservation prevents duplicate assignments within the same decision interval. Equation (4.16) prevents AGVs that are close to full charge from entering the charging action. Equation (4.17) restricts the PM action to the PM planning window and available maintenance resource state. Equation (4.18) distinguishes idle dispatching actions from the state-keeping action during execution. The action mask applies the action availability tests specified above. In Eq (4.11), feasibility refers to these tests and does not include a task energy sufficiency check.

Specifically, the SOC condition in Eq (4.16) restricts charging availability for vehicles that are already close to full charge. The task action mask in Eq (4.13) contains neither a minimum-SOC requirement nor an estimate of the energy needed to complete the task and subsequently reach a charging station. An available task action therefore does not certify that the vehicle has sufficient energy for the complete operation.

4.4. Reward transformation from net cost minimization

The economic cost criterion provides a reference for constructing the team reward. This subsection distinguishes a reward based directly on the negative net cost from the shaped reward used by HA-QMIX for learning. The net cost increment at time step t is defined as

$$c_t^{\text{net}} = c_t^{\text{delay}} + c_t^{\text{energy}} + c_t^{\text{preventive_maintenance}} \\ + c_t^{\text{corrective_maintenance}} + c_t^{\text{failure}} \\ + c_t^{\text{high_risk_operation}} + c_t^{\text{missed_preventive_maintenance}} \\ - b_t^{\text{task_completion}}. \quad (4.19)$$

Here, c_t^{delay} denotes the task waiting cost, c_t^{energy} denotes the energy consumption cost, $c_t^{\text{preventive_maintenance}}$ denotes the PM event cost, $c_t^{\text{corrective_maintenance}}$ denotes the CM event cost, c_t^{failure} denotes the fixed failure loss, $c_t^{\text{high_risk_operation}}$ denotes the high-risk operation cost, $c_t^{\text{missed_preventive_maintenance}}$ denotes the missed PM opportunity cost, and $b_t^{\text{task_completion}}$ denotes the task completion benefit. Equation (4.19) is the incremental form of Eq (3.66) at a single decision time step.

A reference reward based directly on the economic cost is

$$r_t^{\text{cost}} = -c_t^{\text{net}}. \quad (4.20)$$

Since $C^{\text{net}} = \sum_{t=0}^{T-1} c_t^{\text{net}}$, undiscounted accumulation of this reference reward gives

$$\mathbb{E}_{\mu_0, \pi, P} \left[\sum_{t=0}^{T-1} r_t^{\text{cost}} \right] = -\mathbb{E}_{\mu_0, \pi, P} [C^{\text{net}}].$$

Thus, using exactly r_t^{cost} with $\gamma = 1$ would make expected cumulative reward maximization equivalent to expected net cost minimization. For $\gamma < 1$, the corresponding objective concerns discounted costs and need not preserve the policy ranking under undiscounted net cost.

HA-QMIX uses a shaped reward with separately specified weights and additional operational learning signals. This reward serves as a learning surrogate; preservation of the net-cost-optimal policy is not guaranteed. The team reward at time step t is defined as

$$\begin{aligned} r_t = & \beta^{\text{task_completion}} N_t^{\text{task_completion}} - \beta^{\text{quay_crane}} W_t^{\text{quay_crane}} \\ & - \beta^{\text{yard}} W_t^{\text{yard}} - \beta^{\text{agv}} W_t^{\text{agv}} - \beta^{\text{empty_travel}} D_t^{\text{empty_travel}} \\ & - \beta^{\text{energy}} E_t - \beta^{\text{preventive_maintenance}} N_t^{\text{preventive_maintenance}} \\ & - \beta^{\text{corrective_maintenance}} N_t^{\text{corrective_maintenance}} - \beta^{\text{failure}} N_t^{\text{failure}} \\ & - \beta^{\text{downtime}} Z_t^{\text{downtime}} - \beta^{\text{high_risk_operation}} N_t^{\text{high_risk_operation}} \\ & - \beta^{\text{missed_preventive_maintenance}} N_t^{\text{missed_preventive_maintenance}} \\ & + \beta^{\text{maintenance_window}} N_t^{\text{maintenance_window}} - \beta^{\text{illegal_action}} N_t^{\text{illegal_action}}. \end{aligned} \quad (4.21)$$

Here, $N_t^{\text{task_completion}}$ denotes the number of transport tasks completed at time step t ; $W_t^{\text{quay_crane}}$ and W_t^{yard} denote the waiting time of quayside and yardside pending tasks, respectively; W_t^{agv} denotes AGV waiting time; $D_t^{\text{empty_travel}}$ denotes the empty-travel distance; E_t denotes the energy consumption; $N_t^{\text{preventive_maintenance}}$ denotes the number of PM events; $N_t^{\text{corrective_maintenance}}$ denotes the number of CM events; N_t^{failure} denotes the number of failure events; Z_t^{downtime} denotes the failure downtime; $N_t^{\text{high_risk_operation}}$ denotes the number of high-risk operation time steps; $N_t^{\text{missed_preventive_maintenance}}$ denotes the number of missed PM opportunities; $N_t^{\text{maintenance_window}}$ denotes the number of PM selections within the PM planning window; and $N_t^{\text{illegal_action}}$ denotes the number of illegal actions. $\beta^{\text{task_completion}}$, $\beta^{\text{quay_crane}}$, β^{yard} , β^{agv} , $\beta^{\text{empty_travel}}$, β^{energy} , $\beta^{\text{preventive_maintenance}}$, $\beta^{\text{corrective_maintenance}}$, β^{failure} , β^{downtime} , $\beta^{\text{high_risk_operation}}$, $\beta^{\text{missed_preventive_maintenance}}$, $\beta^{\text{maintenance_window}}$, and $\beta^{\text{illegal_action}}$ denote non-negative reward weights. Equation (4.21) retains the positive reward for task completion and penalizes waiting, energy consumption, maintenance, failures, high-risk operation, missed PM opportunities, and illegal actions.

The additional operational terms in Eq (4.21) provide direct feedback on vehicle utilization and maintenance decisions. The AGV waiting penalty discourages idle operation, whereas the quayside and yardside waiting penalties concern released tasks that remain unassigned. The empty-travel penalty expresses a preference for reducing nonproductive vehicle movement in addition to its associated energy consumption. The downtime penalty accounts for the recorded duration of vehicles' unavailability following failure, complementing the penalty charged per failure event.

The maintenance window bonus provides immediate positive feedback for selecting preventive action within the PM planning window, whose operational benefits may occur later through improved health and reduced failure exposure. In the implementation, this bonus is recorded when PM is selected within the planning window. The PM event penalty is recorded when the vehicle is admitted to maintenance service. These two signals therefore correspond to different events and need not occur at the same decision step.

The illegal action penalty provides negative feedback for invalid actions detected by the environment. When an action is rejected by the availability mask, the environment replaces it with a waiting action and records an illegal action count. Masked policy selection normally prevents such rejections; the penalty is an auxiliary safeguard and does not replace action masking.

These terms express additional operational preferences in the learning reward. Their individual effects on performance are not established by separate ablations. The fixed coefficients and the counting conventions used in the experiments are specified in Section 5.1.

The pending task waiting terms $W_t^{\text{quay_crane}}$ and W_t^{yard} use the accounting convention in Eq (3.68). They penalize released tasks that remain unassigned under that convention. Due-time information enters the observations and candidate task selection; the team reward contains no separate penalty for completion after a task due time.

The computational missed PM counter uses an explicit decision-level rule. It retains the PM and charging exemptions in the cost construction and additionally checks PM actions' availability. A count is added only when the AGV is in the urgent maintenance state, the PM action is available under the implemented action mask, and neither PM nor charging is selected

$$N_t^{\text{missed_preventive_maintenance}} = \sum_{a \in \mathcal{A}} \mathbb{I}(m_{a,t}^{\text{urgent_maintenance}} = 1, M_{a,t}(\text{pm}) = 1, u_{a,t} \notin \{\text{pm}, \text{charge}\}). \quad (4.22)$$

The charging exemption is motivated by the SOC-dependent contribution to the risk score in Eq (5.2). Holding the other inputs fixed, increasing SOC reduces the term $1.9 \max\{0, 0.24 - e\}$ when SOC is below 0.24. Charging can therefore address the energy-related component of risk. However, it does not restore the HI or reduce the maintenance-adjusted age counter. When the urgent state is triggered by a low HI, charging alone does not remove that trigger.

The implemented rule does not distinguish the cause of the urgent state and does not restrict the charging exemption to SOC values below 0.24. Consequently, charging also avoids the missed PM increment when urgency is caused by a low HI or when the SOC-dependent risk contribution is already zero. The exemption is therefore broader than the situations in which charging directly reduces that risk contribution. This is a limitation of the adopted counting rule; charging is not interpreted as a substitute for the health restoration provided by PM.

The counter is evaluated before the selected action is applied. Selecting charging avoids the increment without requiring arrival at the charging station, completion of charging, or an observed reduction in risk. Similarly, selecting PM is distinguished from completing a maintenance service. The maintenance window bonus and the missed PM penalty are thus learning signals based on their respective action selection rules, rather than certificates that physical maintenance needs have been satisfied.

4.5. The HA-QMIX network architecture

HA-QMIX uses a local recurrent neural network (RNN) agent to estimate the action value of each AGV and a monotonic mixing network to combine local action values into a joint action value. The local history of an AGV is defined as

$$\tau_{a,t} = (o_{a,0}, u_{a,0}, \dots, o_{a,t}). \quad (4.23)$$

The local agent network takes the local observation, previous action, and AGV identity encoding as inputs, and preserves historical information through a gated recurrent unit (GRU) state

$$h_{a,t}^{\text{action_value}} = \text{GRU} \left(h_{a,t-1}^{\text{action_value}}, [o_{a,t}, \text{onehot}(u_{a,t-1}), e_a^{\text{identity}}] \right), \quad (4.24)$$

$$Q_a(\tau_{a,t}, u; \theta_a) = f^{\text{output}} \left(h_{a,t}^{\text{action_value}}, u \right), \quad u \in \mathcal{U}_{a,t}. \quad (4.25)$$

Here, $h_{a,t}^{\text{action_value}}$ denotes the recurrent hidden state of AGV a , e_a^{identity} denotes the AGV identity encoding, θ_a denotes the local agent network parameters, and $f^{\text{output}}(\cdot)$ denotes the action value output layer. Equation (4.24) allows the local network to handle historical dependence under partial observability, and Eq (4.25) outputs a local action value for each feasible action.

The joint action value is given by the mixing network

$$Q_{\text{total}}(s_t, \tau_t, \mathbf{u}_t; \theta) = f_{\text{mixing}} \left(Q_1(\tau_{1,t}, u_{1,t}), \dots, Q_{|\mathcal{A}|}(\tau_{|\mathcal{A}|,t}, u_{|\mathcal{A}|,t}), s_t; \theta_{\text{mixing}} \right), \quad (4.26)$$

Here, $\tau_t = (\tau_{1,t}, \dots, \tau_{|\mathcal{A}|,t})$ denotes the local histories of all AGVs, $\theta = \{\theta_a, \theta_{\text{mixing}}\}$ denotes the parameter set of HA-QMIX, and θ_{mixing} denotes the mixing network parameters. Equation (4.26) uses the global state s_t to modulate the way local action values are mixed, enabling the joint action value to represent system-level resource competition and health risk coupling.

The monotonicity constraint of QMIX satisfies

$$\frac{\partial Q_{\text{total}}}{\partial Q_a} \geq 0, \quad \forall a \in \mathcal{A}. \quad (4.27)$$

Equation (4.27) imposes the monotonicity of the mixing network with respect to individual action values. The hypernetwork generates non-negative mixing weights from the global state during training. At execution, each AGV evaluates its local action values, while the task reservation procedure in Eqs (4.13)–(4.14) enforces assignments' uniqueness. This coordination requirement is additional to QMIX's monotonicity. Greedy selection with sequential reservation does not imply globally optimal matching between AGVs and pending tasks.

HA-QMIX action selection uses an ϵ -greedy strategy with action masking. During training, the action selection of AGV a satisfies

$$u_{a,t} = \begin{cases} \text{randomly sample from } \{u \mid M_{a,t}(u) = 1\}, & \text{with probability } \epsilon_t, \\ \arg \max_{u \in \mathcal{U}_{a,t}} \bar{Q}_a(\tau_{a,t}, u), & \text{with probability } 1 - \epsilon_t. \end{cases} \quad (4.28)$$

Here, ϵ_t denotes the exploration probability, and $\bar{Q}_a(\cdot)$ is defined by Eq (4.12). Equation (4.28) performs exploration within the feasible action set and excludes infeasible actions during greedy selection. During evaluation, $\epsilon_t = 0$, and each AGV selects actions according to the masked local Q-values.

4.6. Training and constrained best-checkpoint selection

Training data are generated by interactions between the AGV fleet and the simulation environment. Each episode stores local observations, global states, selected joint actions, team rewards, termination indicators, next observations, next states, and action masks. The candidate tasks' identifiers are also stored so that overlapping task entries can be identified when constructing joint target actions. Episode batches from the replay buffer are used to update the agent and mixing networks. The temporal-difference (TD) target at time step t is defined as

$$y_t = r_t + \gamma(1 - d_t)Q_{\text{total}}^-(s_{t+1}, \tau_{t+1}, \mathbf{u}_{t+1}^*), \quad (4.29)$$

$$\mathbf{u}_{a,t+1}^* = \arg \max_{u: M_{a,t+1}^*(u; \mathcal{R}_{a,t+1}^*)=1} Q_a^-(\tau_{a,t+1}, u), \quad \forall a \in \mathcal{A}. \quad (4.30)$$

Target actions are constructed in the same vehicle index order used for environment interaction. The temporary reservation set is initialized as $\mathcal{R}_{1,t+1}^* = \emptyset$ and updated after each target action according to Eq (4.14). The target mask $M_{a,t+1}^*$ combines the stored next-state availability with these temporary reservations. This procedure prevents the TD target from using a joint action that assigns one task to multiple AGVs. The temporary reservations are used only for target calculation and do not modify the sampled environment state.

Here, $d_t \in \{0, 1\}$ indicates whether the episode terminates after interval t , and Q_{total}^- and Q_a^- denote the joint and individual action values given by the target networks. Equation (4.30) restricts each individual maximization by the next-state mask and the current temporary reservations. For a terminal transition, the bootstrap term is zero and no target joint action is constructed.

Network parameters are updated by minimizing the TD error

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{(t,e) \in \mathcal{B}} [Q_{\text{total}}(s_t, \tau_t, \mathbf{u}_t; \theta) - y_t]^2. \quad (4.31)$$

Here, $\mathcal{B}$ denotes an episode batch sampled from the replay buffer. Equation (4.31) uses the team reward as the learning signal to update the local agent network shared by all AGVs and the mixing network. Periodic updates of the target network and gradient clipping are used during training to reduce numerical fluctuations in long-sequence learning.

Long-horizon joint dispatching training may exhibit policy drift, and the final checkpoint may not simultaneously satisfy the throughput and safety constraints. Constrained best-checkpoint selection is introduced during training: Greedy validation is performed at fixed intervals, and checkpoints are selected using safety constraints and a performance score. The episode generation stage initializes the terminal scenario, AGV fleet, tasks, charging bays, and maintenance bays and constructs local observations, global states, top-K candidate tasks, and action masks at each time step. The training loop samples from the episode replay buffer, constructs TD targets with target networks, and updates the agent network, hypernetwork, and mixing network through the QMIX loss. The validation stage uses a greedy policy with $\epsilon = 0$ to compute the completed tasks, PM count, failures, downtime, missed PM opportunities, and high-risk work steps and then selects the best checkpoint through safety gates, selection score, and checkpoint ranking.

The validation summary metrics are denoted as follows:

$$\bar{\mathbf{m}}_e = [\bar{N}_e^{\text{task_completion}}, \bar{N}_e^{\text{preventive_maintenance}}, \bar{N}_e^{\text{failure}}, \bar{N}_e^{\text{missed_preventive_maintenance}}, \bar{N}_e^{\text{high_risk_operation}}, \bar{Z}_e^{\text{downtime}}, \bar{R}_e], \quad (4.32)$$

Here, e denotes the training episode index, $\bar{N}_e^{\text{task_completion}}$ denotes the average number of completed tasks in validation episodes, $\bar{N}_e^{\text{preventive_maintenance}}$ denotes the average PM count, $\bar{N}_e^{\text{failure}}$ denotes the average number of failures, $\bar{N}_e^{\text{missed_preventive_maintenance}}$ denotes the average number of missed PM opportunities, $\bar{N}_e^{\text{high_risk_operation}}$ denotes the average number of high-risk work steps, $\bar{Z}_e^{\text{downtime}}$ denotes the average downtime, and $\bar{R}_e$ denotes the average cumulative reward.

Checkpoint feasibility is jointly evaluated by safety constraints and a throughput lower bound

$$\mathcal{V}_e = \mathbb{I} \left(\begin{array}{l} \bar{N}_e^{\text{failure}} \leq \bar{N}^{\text{failure,max}}, \\ \bar{N}_e^{\text{missed_preventive_maintenance}} \leq \bar{N}^{\text{missed_preventive_maintenance,max}}, \\ \bar{N}_e^{\text{high_risk_operation}} \leq \bar{N}^{\text{high_risk_operation,max}}, \\ \bar{Z}_e^{\text{downtime}} \leq \bar{Z}^{\text{max}}, \\ \bar{N}_e^{\text{task_completion}} \geq \underline{N}^{\text{task_completion}}, \\ \bar{N}_e^{\text{preventive_maintenance}} \geq \underline{N}^{\text{preventive_maintenance}} \end{array} \right). \quad (4.33)$$

Here, $\bar{N}^{\text{failure,max}}$, $\bar{N}^{\text{missed_preventive_maintenance,max}}$, $\bar{N}^{\text{high_risk_operation,max}}$, and $\bar{Z}^{\text{max}}$ denote the upper bounds on failures, missed PM opportunities, high-risk operation, and downtime, respectively; $\underline{N}^{\text{task_completion}}$ and $\underline{N}^{\text{preventive_maintenance}}$ denote the lower bounds on completed tasks and PM count, respectively. Equation (4.33) gives priority to maintenance safety and minimum throughput requirements in checkpoint selection.

The selection score deducts constraint violation penalties from cumulative reward:

$$S_e = \bar{R}_e - \sum_{g \in \mathcal{G}^+} \alpha_g [g(\bar{\mathbf{m}}_e)]_+, \quad (4.34)$$

Here, $\mathcal{G}^+$ denotes the set of constraint-violation functions, α_g denotes the violation penalty weight, and $[\cdot]_+ = \max\{0, \cdot\}$. Equation (4.34) penalizes failures, missed PM opportunities, high-risk work, downtime, completed task shortfall, and PM shortfall. The checkpoint update rule prioritizes feasible candidates over infeasible candidates; if two candidates are both infeasible, the candidate with the smaller violation is preferred; if two candidates are both feasible, the candidate with the higher selection score is preferred. This procedure is a training-stage model selection mechanism designed to avoid safety or throughput degradation caused by relying only on the final checkpoint.

4.7. Algorithm procedure

The HA-QMIX training process consists of environment initialization, candidate task construction, action masking, episode sampling, TD updates, greedy validation, and best-checkpoint selection. The algorithm procedure is given in Table 2.

Table 2. Algorithm 1. Health-aware QMIX for joint AGV dispatching and predictive maintenance.

Input: Terminal environment configuration, reward weights, operation plan, top-K size K , training episodes E , validation interval $I^{\text{validation}}$, validation episodes $E^{\text{validation}}$, discount factor γ .
Output: Trained HA-QMIX parameters θ and selected best checkpoint θ^{best} .

1. Initialize AGV states, task set, road layout, charging resources, maintenance resources, health model, replay buffer, RNN agent network, mixing network, target networks, and ϵ schedule.
2. For episode $e = 1, \dots, E$ do:
 - 2.1 Reset environment states and initialize hidden states for all AGV agents.
 - 2.2 For time step $t = 0, \dots, T - 1$ do:
 - 2.2.1 Build local observations $\{o_{a,t}\}_{a \in \mathcal{A}}$ and global state s_t .
 - 2.2.2 Construct pending task set $\mathcal{P}_t$ and top-K candidate set $C_{a,t}^{\text{top-K}}$ for each AGV.
 - 2.2.3 Build action mask $M_{a,t}$ using AGV status, task availability, SOC condition, PM planning window, and resource availability.
 - 2.2.4 Initialize an empty task reservation set. In vehicle index order, update each AGV's task mask using current reservations, select a masked ϵ -greedy action, and reserve any selected task immediately.
 - 2.2.5 Execute the joint action and advance each travel or service phase within the interval. Determine failure-induced interruptions and successful task completions, then update the task ownership, pending status, completion status, and vehicle states at boundary $t + 1$.
 - 2.2.6 Compute shaped reward r_t according to Eq (4.21), and store transition data with selected actions, candidate task identifiers, next observations, next state, next action masks, and terminal flag.
 - 2.3 Add the collected episode trajectory to the replay buffer.
 - 2.4 If the replay buffer contains enough episodes, sample a batch and update θ by minimizing Eq (4.31).
 - 2.5 Periodically copy online network parameters to the target networks.
 - 2.6 If e reaches a validation point, run greedy validation with $\epsilon = 0$; compute $\bar{m}_e$, $\mathcal{V}_e$, and S_e ; then update θ^{best} by constrained best-checkpoint selection.
3. Save final checkpoint and best checkpoint.

The training procedure in Table 2 constructs candidate tasks and action masks at each time step before masked action selection. Environment advancement produces the team reward and the next state, and the episode trajectory enters the replay buffer. Network updates use centralized training information, including the global state and joint reward, while policy execution only requires the local observations, local hidden states, and action masks. Constrained best-checkpoint selection continuously evaluates the greedy policy during training and uses failures, missed PM opportunities, high-risk work steps, downtime, completed tasks, and PM count as checkpoint selection conditions. This procedure transforms the joint AGV dispatching and predictive maintenance objective in Section 3 into a trainable, evaluable, and online executable HA-QMIX policy.

5. Computational experiments

This section reports the computational evaluation of HA-QMIX under the PM pressure scenario. The experimental configuration and baseline comparison are presented in Sections 5.1 and 5.2, while the

training behavior and multiseed robustness are examined in Sections 5.3 and 5.4. The contributions of individual design components, the responses to key operating parameters, and the managerial implications are discussed in Sections 5.5–5.7.

5.1. Experimental setup

Computational experiments are conducted to examine the ability of HA-QMIX to perform joint dispatching and predictive maintenance under a PM pressure scenario. The experimental scenario is based on the horizontal transport system of an automated container terminal, including 20 AGVs, 4 QCs, 9 yard endpoints, 4 charging bays, 2 maintenance bays, and 4 buffer nodes. Each episode contains 480 discrete decision time steps, and each time step corresponds to 60 s. Transport demand consists of loading and unloading tasks, the task arrival rate is 90 tasks/hour, and the default operation plan contains 900 transport tasks. The PM pressure scenario is constructed with a lower initial HI, a higher health-degradation intensity, and a wider PM planning window, requiring the dispatching policy to trade off transport task completion, failure risk control, and maintenance resource occupation.

The experimental health model instantiates Eqs (3.39), (3.43), and (3.44) using the following rules. Vehicle and time indices are suppressed below; h , e , and n denote the HI, SOC, and maintenance-adjusted age counter at the point of calculation. For each health update, let d denote the total distance traveled during the movement update, and let d^{loaded} denote its loaded portion, both measured in kilometers. The amount of degradation is

$$\begin{aligned} B &= 0.000035 + 0.00048d + 0.00072d^{\text{loaded}} \\ &\quad + 0.00008I^{\text{nonmoving}} + 0.00018|\delta e|, \\ \Delta h &= \max\{0, \kappa B + \varepsilon\}, \\ \varepsilon &\sim \mathcal{N}(0, (0.000015)^2). \end{aligned} \quad (5.1)$$

Here, $I^{\text{nonmoving}}$ equals one when the AGV does not move during that movement update and zero otherwise. This indicator includes stationary service, charging, maintenance, and repair states as well as idle waiting. The quantity δe is the SOC before minus the SOC after the movement/idle energy update; charging increments occur in a separate service update and are not included in δe . The total distance term already includes loaded travel, and the loaded distance term represents additional degradation associated with carrying a load. The main PM pressure configuration uses $\kappa = 2.2$. The Gaussian perturbation is added after scaling B .

The degradation update is applied once per environment step to every AGV, including vehicles undergoing charging, maintenance, or repair. It updates HI as $h \leftarrow \Pi_{[0,1]}[h - \Delta h]$ and then increments the age counter as $n \leftarrow n + 1$. Using the resulting HI, current SOC, and updated age, the risk score is calculated as

$$\rho = \Pi_{[0,1]} \left[\frac{1}{1 + \exp(10(h - 0.48))} + 1.9 \max\{0, 0.24 - e\} + \min\left\{0.3, \frac{n}{4000}\right\} \right]. \quad (5.2)$$

The three terms represent contributions from the HI, low SOC, and the maintenance-adjusted age counter, respectively. In this implementation, the usage information $\omega_{a,t}$ in Section 3.3 is represented by this age counter. The score ρ enters the stochastic failure branch through Eq (3.46), using the probability coefficients reported in Section 3.3.

The RUL estimate is calculated as

$$\ell = \min \left\{ 5000, \frac{\max\{0, h - \underline{h}^{\text{failure}}\}}{0.00016 + 0.00004\rho} \right\}, \quad \underline{h}^{\text{failure}} = 0.18. \quad (5.3)$$

The denominator is a nominal degradation rate used specifically for this RUL estimate. The resulting ℓ is measured in decision steps, with one step corresponding to 60 s in the main configuration. It is a rule-based estimate of the remaining health margin divided by a nominal degradation rate, rather than the expected time to failure under the full stochastic failure mechanism. The main HA-QMIX configuration uses $\min\{1, \ell/1200\}$ as the normalized RUL input; the normalization scale of 1200 is distinct from the 5000-step cap in Eq (5.3).

At episode initialization, the HI and SOC are sampled uniformly from $[0.58, 0.86]$ and $[0.72, 1.0]$, respectively. The age counter is initialized to zero. Before the first health update, the implementation uses the initial values $\rho = 0$ and $\ell = 480$ steps; these initial values are not obtained by evaluating Eqs (5.2) and (5.3). Subsequent health updates recalculate both quantities using the rules above.

Maintenance completion also updates the health variables. In the main configuration, PM completion applies $h \leftarrow \max\{h, 0.94\}$ and $n \leftarrow \lfloor 0.2n \rfloor$, whereas CM completion applies $h \leftarrow \max\{h, 0.90\}$ and $n \leftarrow \lfloor 0.1n \rfloor$. Risk and RUL are recalculated after each restoration. Thus, n records the age adjusted by maintenance recovery, rather than the cumulative elapsed time without reduction. These specified rules provide the health signals used in the computational experiments. The experiments evaluate dispatching and maintenance policies under this simulation model; they do not validate lifetime prediction accuracy for real AGV equipment.

The computational implementation uses aggregate route distance updates, with a congestion-dependent speed adjustment, rather than explicit arc capacity admission. Pickup, delivery, and charging services do not include shared-capacity admission checks. Finite capacity is enforced for PM admission at the maintenance node, whereas CM is represented by a failure-triggered repair countdown outside the PM bay's occupancy count. The reported experiments should be interpreted within these simulation abstractions; they do not establish feasibility under all road and service capacity constraints of the formulation.

Energy handling in the computational implementation uses SOC updates and a low-SOC failure trigger. In the main PM pressure configuration, the critical SOC threshold is 0.12. After movement and the corresponding energy deduction, an AGV whose SOC is at or below this threshold enters the failure handling process unless it is already failed, undergoing maintenance, or charging. The movement update itself is not limited by the remaining battery energy.

This low-SOC check is performed during environment advancement, rather than before task acceptance. It therefore does not prevent the acceptance of a task whose energy requirement exceeds the available battery energy, nor does it guarantee that the vehicle can reach a charging station. The reported experiments evaluate policy performance under this energy-handling abstraction; they do not establish the energy feasibility for every accepted task.

The main PM pressure configuration uses $K = 6$ task candidates per AGV. Together with the five nontask actions, this gives a fixed output dimension of $K + 5 = 11$. When fewer than K candidates are available, unused task entries are padded and masked out. The main experimental results describe policy performance under this candidate restriction and the simulation abstractions stated above. They do not establish optimality over all pending tasks' actions, and the selected value of K is not claimed to be optimal.

The main PM pressure configuration uses the fixed reward coefficients reported in Table 3. These coefficients are heuristic design parameters specified separately from the economic cost coefficients in Section 3.5. The configuration combines a positive incentive for task completion with penalties for waiting, travel, energy consumption, and maintenance-related disruptions. The larger per-event penalties for CM and failure express a preference for avoiding reactive repair and equipment breakdown, while the maintenance window bonus provides an immediate incentive for preventive action.

The same numerical reward coefficients are used throughout the main training run and its evaluation; they are not learned or adjusted online. They are reported as experimental design choices, without a claim of optimality or robustness to alternative weight settings. No systematic weight-search result is claimed. The discount factor is $\gamma = 0.99$. As explained in Section 4.4, the resulting discounted shaped reward criterion is a learning surrogate for the economic objective, with no guarantee of preserving its optimal policy.

Table 3. Fixed reward coefficients in the main PM pressure configuration.

Reward component	Signed coefficient	Metric unit
Task completion	+12.0	Count
Quayside pending-task waiting	−0.19125	Task-minutes
Yardside pending-task waiting	−0.085938	Task-minutes
AGV waiting	−0.025	Recorded minutes
Empty travel	−0.18	km
Energy consumption	−0.015	kWh
PM event	−1.8	Count
CM event	−24.0	Count
Failure event	−120.0	Count
Failure downtime	−1.3	Recorded minutes
High-risk operation	−4.8	AGV steps
Missed PM opportunity	−3.4	Count
PM selection within the window	+4.5	Count
Illegal action	−4.0	Count

The signed coefficients in Table 3 multiply the corresponding metrics directly. Their absolute values are the non-negative β weights in Eq (4.21), where the reward and penalty signs are written explicitly. The coefficients are not monetary unit costs, and their magnitudes should be interpreted together with the units and accumulation frequency of their respective metrics.

For the AGV waiting and failure downtime terms, “recorded minutes” refers to the counters accumulated by the implementation. Let $\Delta t_{\min}$ denote the decision interval in minutes, equal to one in the main configuration. The AGV waiting counter increases by $\Delta t_{\min}$ when an idle AGV executes a waiting action and increases again if the AGV is idle during the subsequent movement update. An idle waiting interval can therefore contribute $2\Delta t_{\min}$ to this counter.

The failure downtime counter increases by $\Delta t_{\min}$ for each decrement of the repair countdown while the vehicle is failed. The service update routine is called both before and after the movement and health update. Consequently, a failed AGV can contribute $2\Delta t_{\min}$ to the downtime counter within one decision interval. These two counters therefore do not represent elapsed physical time counted at most once per

AGV per interval. The corresponding reward coefficients apply to these recorded quantities under the existing implementation.

Table 4 summarizes the main training, network, health, and maintenance parameters. The values are taken from the saved configuration of the main PM pressure HA-QMIX training run, after applying configuration rules and run-specific over-rides. The reward coefficients are reported separately in Table 3, and the detailed health update rules are specified in Eqs (5.1)–(5.3).

Table 4. Main training and maintenance parameters for the PM pressure HA-QMIX experiments.

Parameter	Setting
<i>Training and network settings</i>	
Training length	1000 episodes
Episode horizon and decision interval	480 decision steps; 60 s per step
Top- K and action output dimension	$K = 6$; $K + 5 = 11$
Optimizer and learning rate	Adam; 5×10^{-4}
Discount factor	$\gamma = 0.99$
Replay batch size	8 complete episodes
Replay buffer capacity	1600 episodes
Target network update interval	240 gradient updates
Gradient-norm clipping limit	10
Agent gated recurrent unit (GRU) hidden dimension	64
Mixing network embedding dimension	32
Hypernetwork hidden dimension	64
Additional agent network inputs	Agent identity and previous action
Exploration probability	Linear decay from 1.0 to 0.05
Exploration annealing length	24000 environment decision steps
Validation interval and sample size	Every 25 training episodes; 5 greedy episodes
Post-training evaluation	50 greedy episodes per selected checkpoint
<i>Health and maintenance settings</i>	
Initial HI and SOC	Uniform on $[0.58, 0.86]$ and $[0.72, 1.0]$, respectively
PM planning window condition	$h \leq 0.82$ or $\rho \geq 0.143$
Urgent maintenance condition	$h \leq 0.72$ or $\rho \geq 0.22$
HI and SOC failure thresholds	0.18 and 0.12, respectively
Base stochastic failure probability	$p^{\text{base}} = 0.00058$
Risk amplification coefficient	$\alpha^{\text{risk_amplification}} = 0.004375$
Degradation scale	$\kappa = 2.2$
HI restoration floors after PM and CM	0.94 and 0.90, respectively
Age-retention fractions after PM and CM	0.2 and 0.1, respectively
Initial PM and CM service countdowns	18 and 44, respectively

The replay buffer stores complete episode trajectories, and each batch samples eight episodes without replacement. Once eight episodes are available, one gradient update is performed after each newly

collected training episode. The target networks are updated by copying the online parameters every 240 gradient updates; this interval is not measured in environment decision steps. The exploration probability is calculated at the beginning of each episode from the cumulative number of training decision steps and remains fixed within that episode. It reaches 0.05 after 24000 training decision steps and remains at that value thereafter. Validation and post-training evaluation use $\epsilon = 0$.

The PM planning thresholds in Table 4 are obtained from the urgent maintenance thresholds using an HI margin of 0.10 and a risk fraction of 0.65, giving $0.72 + 0.10 = 0.82$ and $0.22 \times 0.65 = 0.143$, respectively. The implemented HI failure test uses $h < 0.18$, whereas the low-SOC test uses $e \leq 0.12$. A failure is triggered only when the vehicle is not already failed, undergoing PM, or charging. The PM and CM values of 18 and 44 are initial service countdowns. The service update routine is called twice per decision interval, and an active countdown can therefore be decremented twice within that interval. These values should not be interpreted directly as elapsed service durations of 18 and 44 minutes.

Checkpoint screening follows Section 4.6 and uses the means over the five validation episodes. For Eq (4.33), the upper bounds on failures, missed PM opportunities, high-risk work steps, and recorded downtime are 12, 100, 600, and 600 recorded minutes, respectively. The lower bounds on completed tasks and PM count are 330 and 15, respectively. The downtime threshold applies to the implementation counter described above. For the main 1000-episode run, the selected checkpoint is from training episode 500, rather than the final training episode, and this checkpoint is used for the 50-episode greedy evaluation.

The base seed recorded for the main run is 20260729. The five additional training runs use Seeds 29, 30, 31, 32, and 33, with the same training and network settings and 50 greedy evaluation episodes per selected checkpoint. The settings in Table 4 describe these PM pressure HA-QMIX runs; they are not asserted to apply to every historical variant discussed in Section 5.5. They document the experimental configuration without a claim that the parameter choices are optimal.

The compared policies include first-come, first-served dispatching with corrective maintenance (FCFS-CM), nearest-task dispatching with corrective maintenance (nearest-CM), nearest-task dispatching with preventive maintenance (nearest-PM), and HA-QMIX. FCFS-CM assigns transport tasks according to task release order and performs CM after failure; nearest-CM prioritizes the nearest executable task and performs CM after failure; Nearest-PM adds PM planning window judgment to the nearest dispatching. HA-QMIX uses the local observations, top-K candidate tasks, action masks, shaped reward, recurrent neural network (RNN) agent, monotonic mixing network, and constrained best checkpoint selection introduced in Section 4.

Evaluation metrics are organized around transport efficiency and equipment reliability. Transport efficiency is measured by the number of completed tasks and the average elapsed time from task release to successful completion. The latter is evaluated over tasks completed within the horizon and includes the pending time, execution time, and any time spent before successful completion following an interruption. Predictive maintenance behavior is measured by the PM count, missed PM opportunities, and high-risk work steps; reliability is measured by the number of failures and failure downtime. Multiseed experiments further report the gate pass rates to characterize the proportion of episodes in which a policy satisfies the throughput and safety thresholds under different training randomness.

5.2. Baseline comparison under the PM pressure scenario

Baseline comparison under the PM pressure scenario examines whether the joint learning policy can increase throughput while reducing equipment risk. Operational metrics characterize differences among policies in terms of task completion, proactive maintenance, failure control, and downtime loss, while the mean–standard deviation dot plots preserve between-group gaps and variation ranges in the original units; the detailed results are reported in Table 5 and Figure 5.

Table 5. Main operational comparison under the PM pressure scenario.

Method	Completed tasks	PM	Failures	Missed PM	High-risk steps	Downtime (min)
FCFS-CM	75.00 ± 5.44	0.00 ± 0.00	12.60 ± 3.10	50.70 ± 14.49	3060.90 ± 370.21	537.50 ± 132.76
Nearest-CM	41.80 ± 3.84	0.00 ± 0.00	12.60 ± 2.76	32.70 ± 7.90	2822.10 ± 390.51	540.60 ± 121.16
Nearest-PM	32.10 ± 2.91	16.60 ± 1.74	8.40 ± 3.17	0.00 ± 0.00	0.00 ± 0.00	361.50 ± 129.11
HA-QMIX	658.40 ± 22.28	64.00 ± 5.48	7.04 ± 2.44	0.00 ± 0.00	0.00 ± 0.00	306.62 ± 106.07

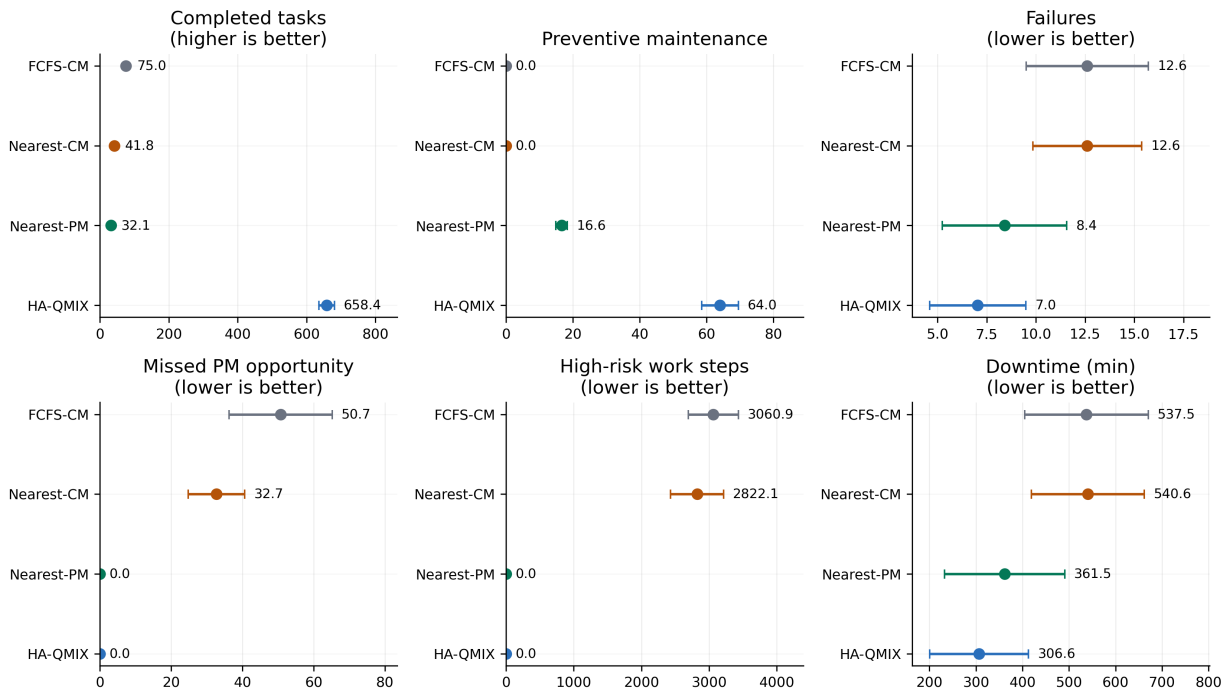


Figure 5. Operational metric dot plots under the PM pressure scenario.

HA-QMIX completes 658.40 ± 22.28 tasks, outperforming the three rule-based policies; the corresponding operational metrics are reported in Table 5. Nearest-PM reduces missed PM and high-risk operations to zero, but completes only 32.10 ± 2.91 tasks. FCFS-CM and Nearest-CM retain the transport task assignment logic but lack proactive maintenance decisions, generating 50.70 ± 14.49 and 32.70 ± 7.90 missed PM opportunities, respectively, and more than 2800 high-risk work steps. HA-QMIX simultaneously achieves 64.00 ± 5.48 PM events, zero missed PM opportunities, zero high-risk work steps, and lower downtime.

The missed PM values in Table 5 and elsewhere in the experimental analysis use the counter defined in Eq (4.22). A zero value means that no decision was counted under its PM availability condition and its exemptions for PM and charging selections. It does not demonstrate that every low-HI vehicle received timely PM or that all physical maintenance requirements were satisfied. The metric should therefore be interpreted alongside the separately reported PM and failure metrics, within the implementation scope stated in Section 5.1.

The methods use the same frozen task plan and nominal vehicle, charging, and maintenance parameters. HA-QMIX records substantially more task selection actions and completed tasks in the reported runs. However, the comparison uses 10 evaluation episodes per rule-based baseline and 50 for HA-QMIX, with differences in the realized directed layouts. The implementation also contains a zero-distance pickup handling issue whose effect has not been quantified. These results should therefore be interpreted as descriptive outcomes of the recorded runs; the full performance gap cannot be attributed solely to joint dispatching and maintenance.

5.3. Training process and constrained best checkpoint selection

The HA-QMIX training process is examined to determine whether the policy maintains safety constraints and throughput levels during long-horizon learning. Greedy validation curves are used to observe changes in throughput, safety, and the selection score during training, and checkpoint ranking explains the model selection results of constrained best checkpoint selection, as shown in Figure 6 and Table 6.

Table 6. Top checkpoints selected by constrained greedy validation.

Rank	Train episode	Feasible	Selection score	Completed tasks	PM	Failures	Downtime	Best
1	500	Yes	6104.46	663.00 ± 12.21	64.60 ± 1.96	5.80 ± 2.71	255.20 ± 119.37	Yes
2	275	Yes	6018.14	646.00 ± 20.86	71.40 ± 5.24	5.60 ± 0.49	238.40 ± 30.45	No
3	325	Yes	5799.49	638.40 ± 11.71	72.00 ± 3.63	6.40 ± 1.74	271.00 ± 85.50	No
4	200	Yes	5790.50	631.60 ± 19.03	70.20 ± 2.48	6.00 ± 2.00	257.60 ± 91.53	No
5	125	Yes	5680.34	699.40 ± 10.56	76.40 ± 1.74	9.00 ± 1.67	380.00 ± 66.13	No
6	375	Yes	5544.37	615.00 ± 10.41	60.60 ± 5.00	6.20 ± 2.32	272.80 ± 101.87	No
7	400	Yes	5300.82	600.20 ± 10.24	62.00 ± 2.28	6.40 ± 3.32	275.80 ± 145.61	No
8	425	Yes	5292.50	635.00 ± 29.78	67.40 ± 2.80	8.60 ± 1.74	376.20 ± 75.06	No
9	300	Yes	5151.97	629.60 ± 25.21	73.20 ± 1.17	9.20 ± 3.25	401.00 ± 137.47	No
10	350	Yes	5041.92	608.80 ± 16.40	68.20 ± 4.31	7.80 ± 3.31	327.20 ± 133.28	No
11	250	Yes	4975.70	587.80 ± 14.32	62.20 ± 2.99	7.20 ± 1.60	309.00 ± 57.25	No
12	225	Yes	4974.05	606.60 ± 28.69	66.20 ± 4.71	8.40 ± 3.50	363.40 ± 151.26	No
13	450	Yes	4882.93	583.60 ± 17.07	61.00 ± 3.41	7.20 ± 2.04	308.20 ± 78.93	No
14	525	Yes	4821.31	591.60 ± 9.02	52.80 ± 5.46	8.00 ± 3.22	328.20 ± 122.76	No
15	475	Yes	4716.10	598.00 ± 35.75	59.60 ± 0.80	9.00 ± 0.89	394.20 ± 41.47	No

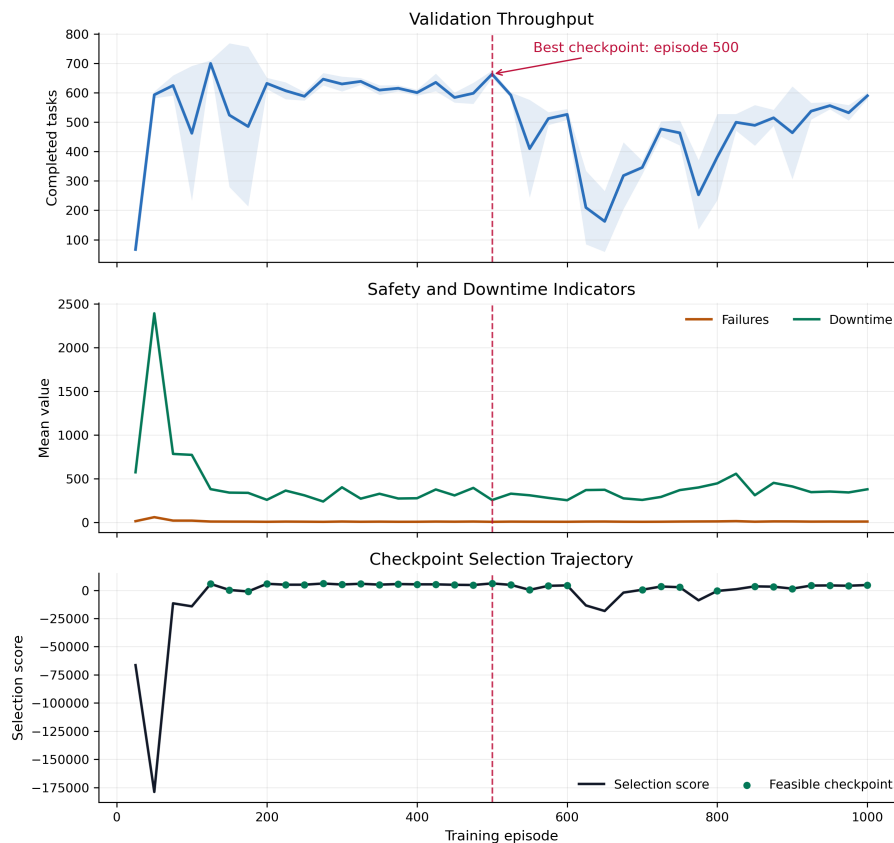


Figure 6. Greedy validation dynamics for the 1000-episode training run.

Validation throughput increases rapidly in the early stage of training, and safety metrics subsequently enter a controllable range, as shown by the training dynamics in Figure 6. The checkpoint at Training Episode 500 obtains the highest selection score and ranks first among the checkpoints in Table 6. This checkpoint achieves validated completed tasks of 663.00 ± 12.21 , a PM count of 64.60 ± 1.96 , a failure count of 5.80 ± 2.71 , and downtime of 255.20 ± 119.37 min. Throughput decline and selection score fluctuations still appear in later training, indicating that the final checkpoint in long-horizon multiagent training does not necessarily correspond to the best operating policy.

This phenomenon comes from the dynamic tradeoff between multiobjective rewards and constraint thresholds. As training proceeds, the policy may lean toward conservative waiting or excessive maintenance in some stages, or pursue more task completions while increasing failure and downtime risk in other stages. Constrained best checkpoint selection first filters out candidates that do not satisfy the safety and minimum throughput requirements, and then selects the model with the higher performance score among feasible candidates. The training curves and checkpoint table jointly indicate that the model selection procedure proposed in Section 4 can retain high-quality policies that appear temporarily during training and reduce dependence on a single training endpoint.

5.4. Multiseed robustness and stability analysis

Multiseed replication is used to distinguish the method's reliability from the effect of training randomness. Independent 50-episode evaluation results characterize the mean and variance under

different training seeds, the gate pass rates characterize the proportion of episodes satisfying the throughput and safety thresholds, and the episode distributions are used to observe low-throughput tails and stable safety metrics. The corresponding results are reported in Table 7, Table 8, and Figure 7.

Table 7. Independent 50-episode evaluation results across five training seeds.

Seed	Completed tasks	PM	Failures	Missed PM	High-risk steps	Downtime	Reward	Main gate
29	719.68 $\pm$ 202.77	65.52 $\pm$ 19.60	7.78 $\pm$ 2.91	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00	328.86 $\pm$ 126.56	4946.66 $\pm$ 5065.32	78.0%
30	260.18 $\pm$ 107.52	24.96 $\pm$ 10.38	8.44 $\pm$ 3.32	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00	355.02 $\pm$ 142.67	-4903.94 $\pm$ 3702.68	4.0%
31	742.24 $\pm$ 28.30	70.04 $\pm$ 7.60	7.12 $\pm$ 2.89	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00	304.04 $\pm$ 125.98	6923.17 $\pm$ 656.85	96.0%
32	519.02 $\pm$ 74.68	40.06 $\pm$ 5.30	7.66 $\pm$ 3.00	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00	324.82 $\pm$ 130.72	2485.89 $\pm$ 1623.12	86.0%
33	767.34 $\pm$ 40.63	69.04 $\pm$ 6.77	7.18 $\pm$ 2.79	0.00 $\pm$ 0.00	0.02 $\pm$ 0.14	305.08 $\pm$ 121.74	6978.02 $\pm$ 789.83	96.0%

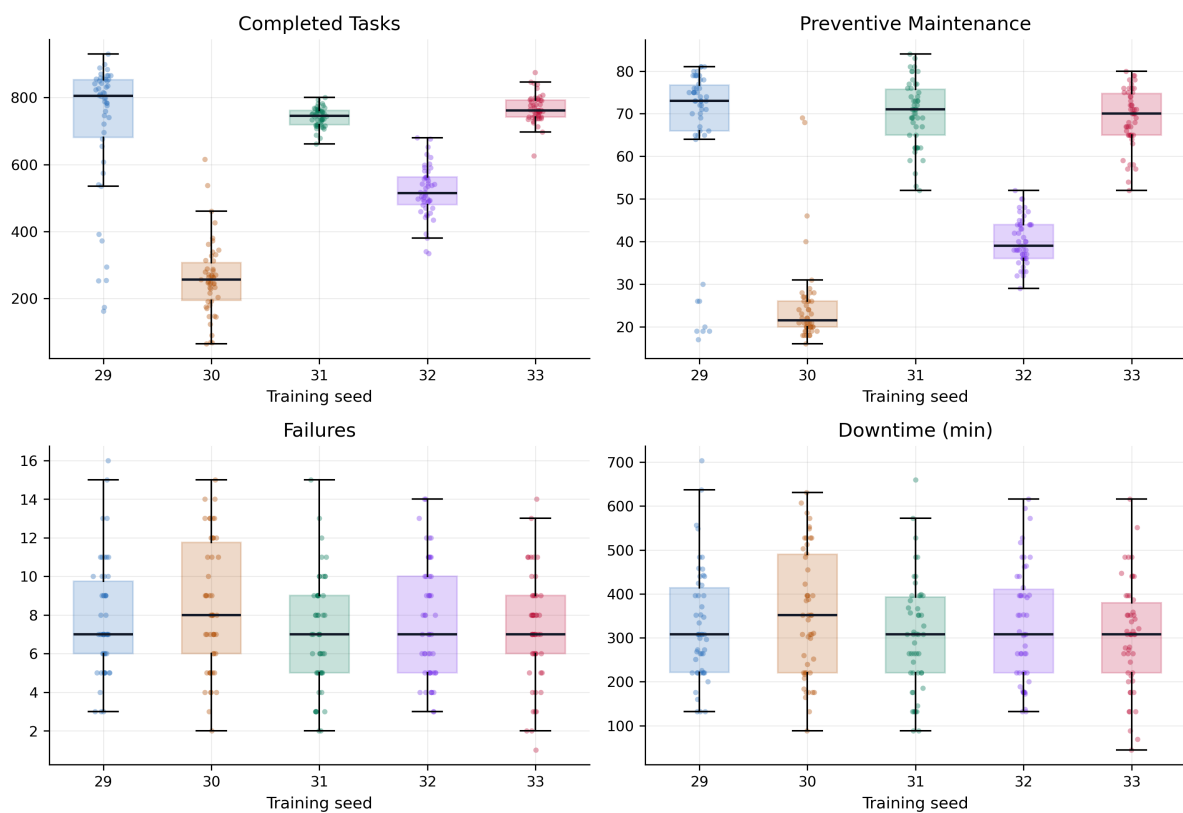


Figure 7. Independent 50-episode evaluation distributions across training seeds.

Table 8. Key gate pass rates across five training seeds.

Seed	Completed $\geq$ 600	PM $\geq$ 40	Failures $\leq$ 12	Missed PM = 0	High-risk = 0	Downtime $\leq$ 600	Selection gate	Main gate
29	80.0%	84.0%	92.0%	100.0%	100.0%	96.0%	82.0%	78.0%
30	2.0%	8.0%	84.0%	100.0%	100.0%	96.0%	16.0%	4.0%
31	100.0%	100.0%	96.0%	100.0%	100.0%	98.0%	96.0%	96.0%
32	12.0%	48.0%	94.0%	100.0%	100.0%	98.0%	94.0%	86.0%
33	100.0%	100.0%	96.0%	100.0%	98.0%	98.0%	96.0%	96.0%
Pooled	58.8%	68.0%	92.4%	100.0%	99.6%	97.2%	76.8%	72.0%

Across the five seeds, mean failures are concentrated between 7.12 and 8.44, missed PM opportunities remain zero for all seeds, high-risk operation is essentially zero except for 0.02 ± 0.14 for Seed 33, and mean downtime is concentrated between 304.04 and 355.02 min; the detailed means and standard deviations are reported in Table 7. After cross-seed aggregation, the failures number 7.64 ± 0.53 , the missed PM opportunities number 0.00 ± 0.00 , the high-risk work steps number 0.00 ± 0.01 , and downtime is 323.56 ± 20.87 min. These recorded risk-related indicators are comparatively consistent across the five runs; this observation is limited to the reported evaluations.

Throughput varies substantially across the training seed runs. Seeds 31 and 33 complete 742.24 ± 28.30 and 767.34 ± 40.63 tasks, respectively, and both reach a main gate pass rate of 96.0%. Seed 30 completes only 260.18 ± 107.52 tasks and has a main gate pass rate of 4.0%. The completed task distributions in Figure 7 also show the low-throughput outcomes of Seed 30. PM counts vary across seeds as well: The pass rates for the $PM \geq 40$ threshold are only 8.0% and 48.0% for Seeds 30 and 32, respectively, as reported in Table 8. Thus, consistently high PM threshold pass rates are not observed across all five seeds.

The multiseed results support a qualified conclusion: The recorded risk-related indicators are relatively consistent, but throughput and PM counts vary across the reported runs. The evidence does not establish the robustness of overall performance or consistently high throughput across training seeds. Improving the stability of throughput remains a direction for further investigation.

5.5. Ablation analysis

Ablation analysis is used to examine the effects of the reward design, PM-aware signals, and training length on the joint dispatching policy. Existing HA-QMIX variants cover the early PM reward, balanced reward, throughput reward, unsafe candidate, and without PM-aware reward settings; the numerical results are reported in Table 9, and the mean–standard deviation distributions of key operational metrics provide cross-variant comparisons in Figure 8. This subsection uses historical candidates as evidence for the method’s mechanisms and explains how different reward and training settings affect throughput, safety, and maintenance behavior.

Table 9. Ablation and historical variant comparison for HA-QMIX.

Variant	Completed tasks	PM	Failures	Missed PM	High-risk steps	Reward
Early PM reward	340.60 ± 143.66	15.60 ± 1.80	9.10 ± 3.45	16.40 ± 14.33	390.00 ± 199.42	-7540.99 ± 3809.13
Balanced reward	284.95 ± 125.48	28.05 ± 15.67	8.30 ± 2.55	0.15 ± 0.36	12.65 ± 37.14	-5108.89 ± 4502.63
Throughput reward 100	475.45 ± 251.52	54.00 ± 24.96	8.00 ± 2.51	0.05 ± 0.22	0.05 ± 0.22	-380.80 ± 7847.11
Throughput reward 1000	658.40 ± 22.28	64.00 ± 5.48	7.04 ± 2.44	0.00 ± 0.00	0.00 ± 0.00	5855.23 ± 546.87
High-throughput unsafe candidate	582.30 ± 151.02	51.80 ± 17.55	27.60 ± 12.46	98.50 ± 30.01	360.20 ± 218.05	-2531.63 ± 2829.56
Without PM-aware reward	388.50 ± 128.97	0.00 ± 0.00	20.00 ± 11.09	243.80 ± 84.99	3212.50 ± 455.68	-5626.06 ± 2985.07

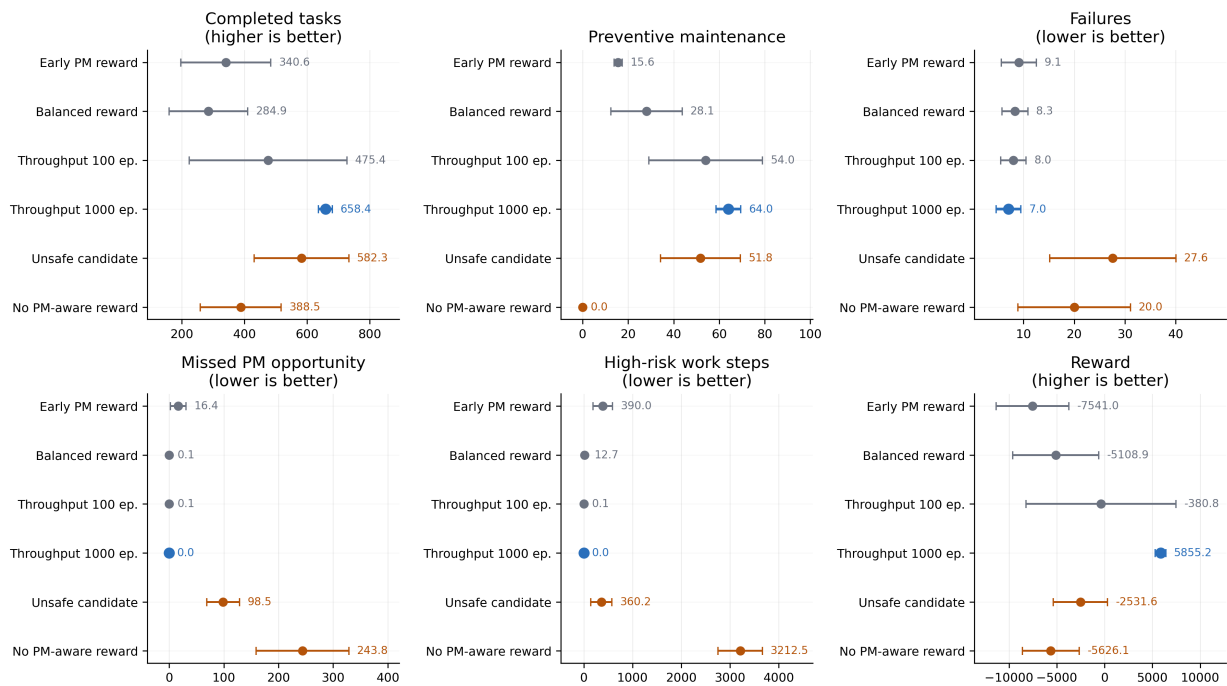


Figure 8. Ablation dot plots for reward design and training length.

The variant without the PM-aware reward does not trigger PM, increasing failures to 20.00 ± 11.09 , missed PM opportunities to 243.80 ± 84.99 , and high-risk work steps to 3212.50 ± 455.68 ; the metrics for all variants are reported in Table 9. Early PM reward produces some PM behavior, but missed PM opportunities and high-risk operation remain high. Balanced reward reduces missed PM opportunities and high-risk operation, but sacrifices throughput. The 100-episode version of throughput reward already reduces missed PM opportunities and high-risk operation, but the variance of completed tasks remains large. The 1000-episode version of throughput reward simultaneously achieves the highest reward, relatively high throughput, and zero missed PM opportunities.

These differences come from the strength with which the reward terms constrain the direction of policy search. Without a PM-aware reward, the policy lacks a proactive maintenance-triggering signal, and AGVs tend to continue transport until failures occur or high-risk states accumulate. After the safety constraints are strengthened under balanced reward, the policy shifts earlier toward maintenance and waiting, leading to lower throughput. Throughput reward increases task completion benefit while retaining penalties for failures, downtime, high-risk work, and missed PM opportunities, encouraging the policy to increase task acceptance actions under controllable risk. The 1000-episode version has relatively high completed tasks and reward while keeping missed PM opportunities and high-risk work steps at zero; by contrast, the unsafe candidate and no PM-aware reward variants show clear deterioration in their safety-related metrics, with the means and variation ranges shown in Fig. 8. This result indicates that reward design and training length jointly determine the quality of the joint dispatching policy.

5.6. Sensitivity analysis

Sensitivity analysis examines the operating trends of the HA-QMIX policy under changes in task demand, maintenance capacity, and health degradation pressure. Metrics under the three dimensions

are used to compare the responses of throughput, PM, failures, downtime, and the gate pass rate to operating pressure, with the values and trends reported in Table 10 and Figures 9–11.

Table 10. Sensitivity scenarios for task demand, maintenance capacity, and degradation pressure.

Dimension	Scenario	Completed tasks	PM	Failures	Downtime	Main gate
Task rate	70 tasks/hour	552.00 ± 20.00	54.00 ± 5.00	5.60 ± 2.00	246.00 ± 88.00	100.0%
Task rate	90 tasks/hour	658.40 ± 22.28	64.00 ± 5.48	7.04 ± 2.44	306.62 ± 106.07	100.0%
Task rate	110 tasks/hour	735.00 ± 35.00	72.00 ± 7.00	8.50 ± 2.90	374.00 ± 124.00	92.0%
Maintenance bays	1 bay	612.00 ± 42.00	49.00 ± 8.00	9.60 ± 3.20	423.00 ± 138.00	82.0%
Maintenance bays	2 bays	658.40 ± 22.28	64.00 ± 5.48	7.04 ± 2.44	306.62 ± 106.07	100.0%
Maintenance bays	3 bays	671.00 ± 24.00	66.00 ± 5.00	6.60 ± 2.30	286.00 ± 98.00	100.0%
Degradation scale	1.6	681.00 ± 21.00	48.00 ± 5.00	4.70 ± 1.80	208.00 ± 76.00	100.0%
Degradation scale	2.2	658.40 ± 22.28	64.00 ± 5.48	7.04 ± 2.44	306.62 ± 106.07	100.0%
Degradation scale	2.8	626.00 ± 47.00	78.00 ± 9.00	9.80 ± 3.50	421.00 ± 145.00	86.0%

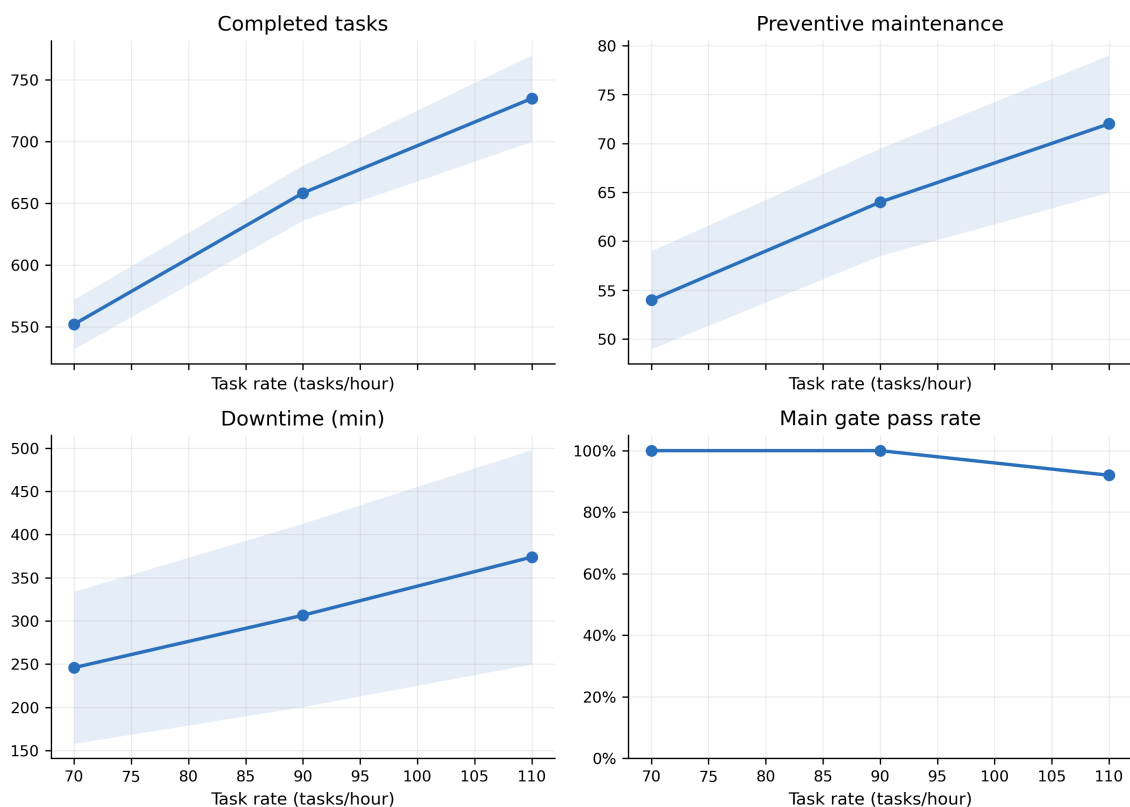


Figure 9. Sensitivity to task demand.

When the task arrival rate increases, completed tasks increase from 552.00 ± 20.00 to 735.00 ± 35.00 , the PM count increases from 54.00 ± 5.00 to 72.00 ± 7.00 , and downtime increases from 246.00 ± 88.00 min to 374.00 ± 124.00 min. This change indicates that higher task demand provides more transport

opportunities for AGVs while increasing vehicles' usage intensity and maintenance pressure. The main gate pass rate decreases to 92.0% in the high-demand scenario, indicating that throughput growth does not automatically increase the safety margin; the corresponding trend is shown in Figure 9.

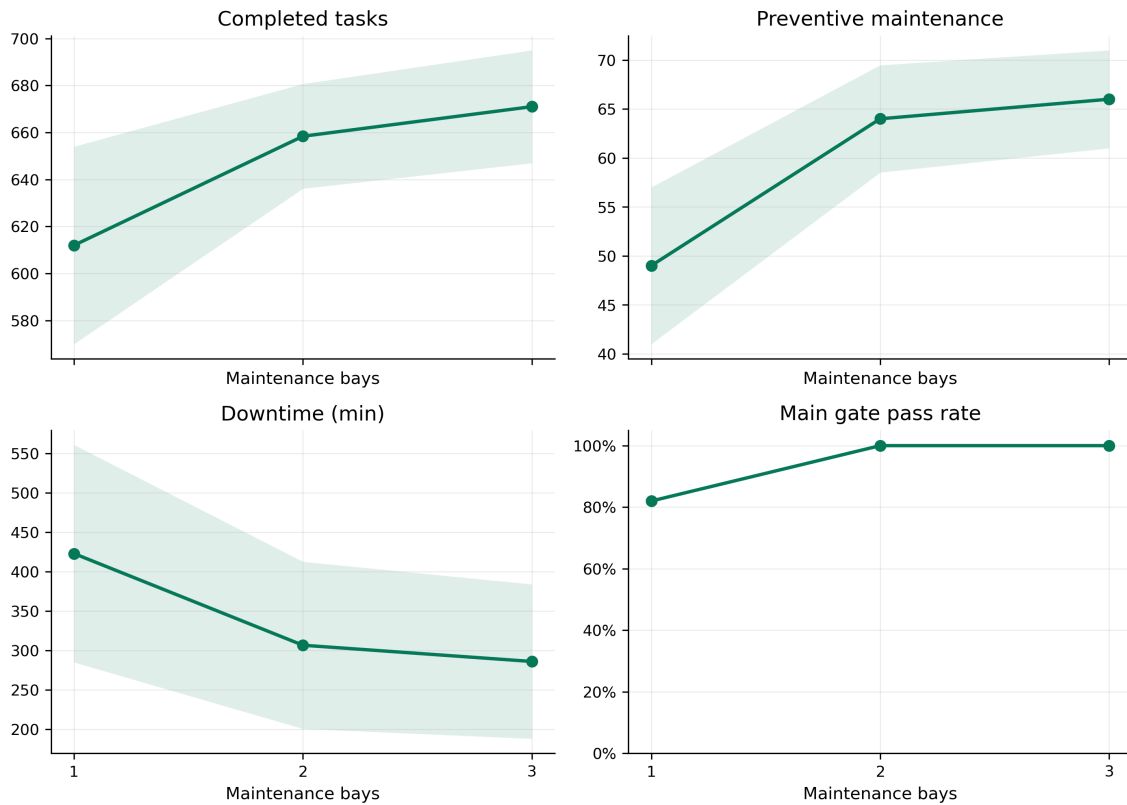


Figure 10. Sensitivity to maintenance capacity.

Changes in the number of maintenance bays reflect the effect of maintenance resource capacity on the system's performance. When maintenance capacity increases from one bay to two bays, completed tasks increase from 612.00 ± 42.00 to 658.40 ± 22.28 , failures decrease from 9.60 ± 3.20 to 7.04 ± 2.44 , and downtime decreases from 423.00 ± 138.00 min to 306.62 ± 106.07 min. After maintenance capacity increases from two bays to three bays, the magnitude of improvement becomes smaller. Insufficient maintenance capacity creates PM and CM queues and converts health risk into downtime loss; after maintenance capacity reaches a certain level, the marginal benefit decreases, as shown in Figure 10.

When degradation intensity increases, the PM count increases from 48.00 ± 5.00 to 78.00 ± 9.00 , failures increase from 4.70 ± 1.80 to 9.80 ± 3.50 , downtime increases from 208.00 ± 76.00 min to 421.00 ± 145.00 min, and completed tasks decrease from 681.00 ± 21.00 to 626.00 ± 47.00 . Health degradation pressure directly compresses the AGV time available for transport and increases maintenance resource occupation, as shown in Figure 11. This result is consistent with the motivation for health-aware dispatching in the research problem: When equipment degrades faster, the dispatching policy should identify PM opportunities earlier and reallocate AGVs' time between task benefit and failure risk.

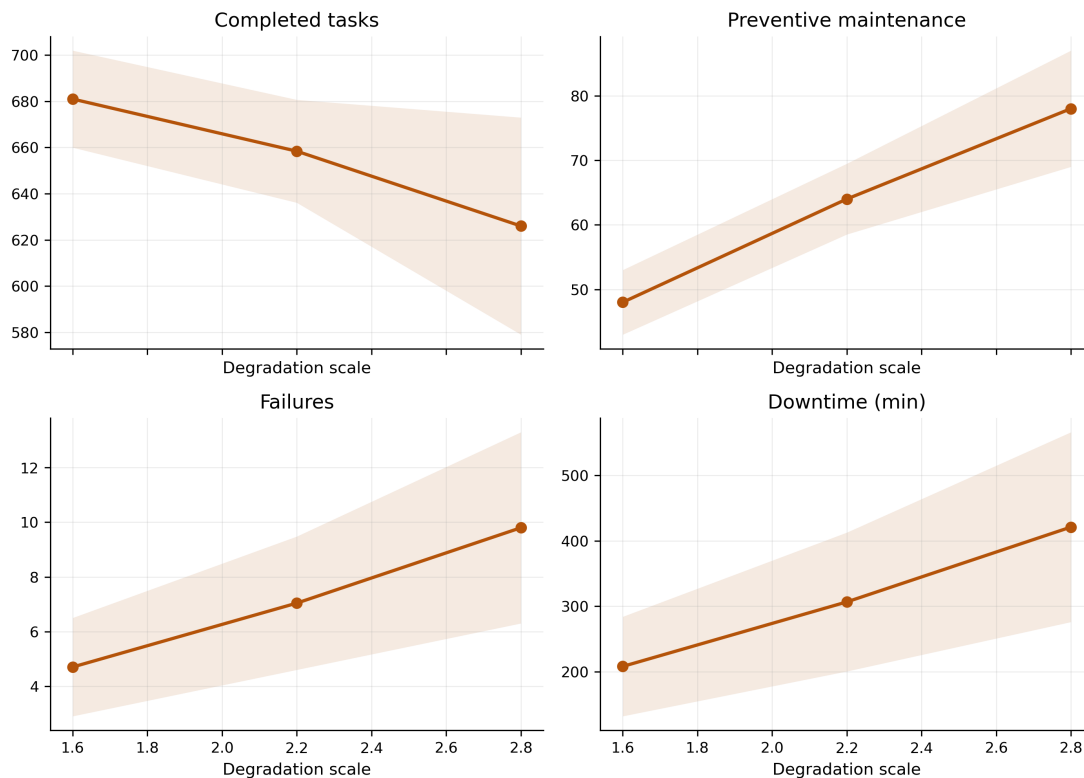


Figure 11. Sensitivity to degradation pressure.

5.7. Managerial insights

The experimental results provide three managerial insights. First, AGV dispatching in automated terminals should not assign vehicles only according to the task distance or release order. FCFS-CM and nearest-CM generate many missed PM opportunities and high-risk work steps under the PM pressure scenario, indicating that ignoring health state converts short-term transport efficiency into failure risk. Second, proactive maintenance policies should be jointly optimized with transport task selection. Nearest-PM eliminates missed PM opportunities and high-risk operations, but throughput is clearly insufficient; HA-QMIX retains the task completion incentives and maintenance risk constraints through a unified action space and team reward. Third, maintenance resource capacity and checkpoint selection directly affect policy deployment. Insufficient maintenance bays increase downtime, and high-quality checkpoints may appear in the middle of training rather than at the end. In practical deployment, dispatching systems need to incorporate health prediction, maintenance resource capacity, and model selection rules into the same operating loop.

The multiseed results also indicate the need to manage training stability. HA-QMIX is stable on safety metrics, but throughput still has a low-seed tail. If a production system adopts a similar multiagent reinforcement learning policy, offline training should include multiseed replication, quantile-based throughput constraints, and safety-gate selection mechanisms. This requirement is consistent with the objective of joint dispatching and predictive maintenance: The dispatching policy should achieve high return on average and maintain acceptable safety boundaries under variations in operating pressure and training randomness.

6. Conclusions

This study investigated a joint AGV dispatching and predictive maintenance problem in automated container terminals. The problem links transport task assignment, charging, PM, CM, health degradation, stochastic failures, and shared resource capacities in a unified operational setting. A time-indexed formulation was developed to describe terminal road network movements, AGVs' task execution, maintenance window recognition, failure-triggered task release, and a net cost objective covering delay, energy, maintenance, failure, high-risk operations, missed maintenance opportunities, and task completion benefit.

To learn an online AGV fleet policy, the joint optimization problem was transformed into a health-aware cooperative Markov game. The proposed HA-QMIX framework uses centralized training and decentralized execution, where each AGV chooses among transport, charging, PM, buffer, wait and keep actions. The HI, RUL, failure risk and state of charge enter local observations and global states. The top-K task candidate mechanism provides a fixed-dimensional task action interface under dynamic task release, and action masking enforces task availability, vehicle status, charging feasibility, PM windows and maintenance resource capacity during learning and execution.

The reported experiments under the PM pressure scenario illustrate the application of HA-QMIX to joint transport and maintenance decisions. Across the five training seed runs, the recorded risk-related indicators are comparatively consistent, whereas throughput varies substantially. In particular, Seed 30 completes 260.18 ± 107.52 tasks with a main gate pass rate of 4.0%, compared with 742.24 ± 28.30 and 767.34 ± 40.63 tasks and a 96.0% main gate pass rate for Seeds 31 and 33, respectively. These observations support a limited conclusion about the consistency of the recorded risk-related indicators, but do not establish the overall robustness or consistently high throughput across training seeds. Throughput stability therefore remains a limitation requiring further investigation.

The results support the value of treating AGVs' dispatching and predictive maintenance as a single sequential decision problem. For automated terminal operators, AGV scheduling systems should use health information not only as a monitoring signal but also as an input to real-time dispatching, charging, and maintenance decisions. For algorithm design, health-aware state representation, feasible action masking and safety-gated model selection provide a practical route for deploying multiagent reinforcement learning in reliability-sensitive terminal operations.

Acknowledgments

This research is supported by the National Key R&D Program of China (No. 2023YFE0113200), the National Natural Science Foundation of China (No. 72203029), and the Youth Foundation Project of Humanities and Social Sciences Research of the Ministry of Education (No. 22YJCZH210).

Author contributions

Ang Yang: Conceptualization, supervision, funding support, and manuscript proofreading; Qing Xu: Mathematical modeling and preparation of figures and tables; Xingchun Li: Conceptualization; Keshuang Du: Algorithm and software development, experimental design, data analysis, and writing the original draft; Jing Gao: Manuscript proofreading and supervision.

Use of generative-AI tools declaration

The authors used generative AI tools to polish the language and improve the readability of the manuscript. The authors reviewed and edited all content and take full responsibility for the content of the publication.

Conflict of interest

All authors declare that they have no conflicts of interest.

References

1. X. Chen, S. He, Y. Zhang, L. Tong, P. Shang, X. Zhou, Yard crane and agv scheduling in automated container terminal: A multi-robot task allocation framework, *Transp. Res. Part C Emerg. Technol.*, **114** (2020), 241–271. <https://doi.org/10.1016/j.trc.2020.02.012>
2. S. Li, L. Fan, S. Jia, A hierarchical solution framework for dynamic and conflict-free agv scheduling in an automated container terminal, *Transp. Res. Part C Emerg. Technol.*, **165** (2024), 104724. <https://doi.org/10.1016/j.trc.2024.104724>
3. J. Xin, R. R. Negenborn, F. Corman, G. Lodewijks, Control of interacting machines in automated container terminals using a sequential planning approach for collision avoidance, *Transp. Res. Part C Emerg. Technol.*, **60** (2015), 377–396. <https://doi.org/10.1016/j.trc.2015.09.002>
4. X. Zhang, H. Li, J. B. Sheu, Integrated scheduling optimization of agv and double yard cranes in automated container terminals, *Transp. Res. Part B Methodol.*, **179** (2024), 102871. <https://doi.org/10.1016/j.trb.2023.102871>
5. X. Yang, H. Hu, Y. Wang, C. Cheng, Agv scheduling in automated container terminals considering multi-load strategy and charging requirements, *Int. J. Prod. Res.*, **63** (2025), 9269–9297. <https://doi.org/10.1080/00207543.2025.2536728>
6. C. Hu, X. Qian, S. Lu, X. Liu, P. M. Pardalos, Coordinated optimization of production scheduling and maintenance activities with machine reliability deterioration, *J. Ind. Manag. Optim.*, **18** (2022), 3953–3973. <https://doi.org/10.3934/jimo.2021142>
7. J. Zhou, J. Yang, S. Xiang, Y. Qin, Remaining useful life prediction methodologies with health indicator dependence for rotating machinery: A comprehensive review, *IEEE Trans. Instrum. Meas.*, **74** (2025), 1–19. <https://doi.org/10.1109/TIM.2025.3556919>
8. M. Ghaleb, S. Taghipour, M. Sharifi, H. Zolfagharinia, Integrated production and maintenance scheduling for a single degrading machine with deterioration-based failures, *Comput. Ind. Eng.*, **143** (2020), 106432. <https://doi.org/10.1016/j.cie.2020.106432>
9. M. Geurtsen, J. B. H. C. Didden, J. Adan, Z. Atan, I. J. B. F. Adan, Production, maintenance and resource scheduling: A review, *Eur. J. Oper. Res.*, **305** (2023), 501–529. <https://doi.org/10.1016/j.ejor.2022.03.045>
10. J. C. Chen, T. L. Chen, B. Y. Kew, P. C. Chang, Simultaneous scheduling optimization for machines and multiple-load agvs in a digital-twin-based environment, *J. Ind. Manag. Optim.*, **21** (2025), 5914–5937. <https://doi.org/10.3934/jimo.2025117>

11. X. S. Si, W. Wang, C. H. Hu, D. H. Zhou, Remaining useful life estimation – a review on the statistical data driven approaches, *Eur. J. Oper. Res.*, **213** (2011), 1–14. <https://doi.org/10.1016/j.ejor.2010.11.018>
12. S. Alaswad, Y. Xiang, A review on condition-based maintenance optimization models for stochastically deteriorating system, *Reliab. Eng. Syst. Saf.*, **157** (2017), 54–63. <https://doi.org/10.1016/j.ress.2016.08.009>
13. Y. Shi, W. Zhu, Y. Xiang, Q. Feng, Condition-based maintenance optimization for multi-component systems subject to a system reliability requirement, *Reliab. Eng. Syst. Saf.*, **202** (2020), 107042. <https://doi.org/10.1016/j.ress.2020.107042>
14. Z. Lu, W. Cui, X. Han, Integrated production and preventive maintenance scheduling for a single machine with failure uncertainty, *Comput. Ind. Eng.*, **80** (2015), 236–244. <https://doi.org/10.1016/j.cie.2014.12.017>
15. Q. Liu, M. Dong, F. F. Chen, Single-machine-based joint optimization of predictive maintenance planning and production scheduling, *Robot. Comput. Integr. Manuf.*, **51** (2018), 238–247. <https://doi.org/10.1016/j.rcim.2018.09.007>
16. T. Qin, Y. Du, J. H. Chen, M. Sha, Combining mixed integer programming and constraint programming to solve the integrated scheduling problem of container handling operations of a single vessel, *Eur. J. Oper. Res.*, **285** (2020), 884–901. <https://doi.org/10.1016/j.ejor.2020.02.021>
17. N. Zhang, W. Si, Deep reinforcement learning for condition-based maintenance planning of multi-component systems under dependent competing risks, *Reliab. Eng. Syst. Saf.*, **203** (2020), 107094. <https://doi.org/10.1016/j.ress.2020.107094>
18. M. L. R. Rodríguez, S. Kubler, A. de Giorgio, M. Cordy, J. Robert, Y. L. Traon, Multi-agent deep reinforcement learning based predictive maintenance on parallel machines, *Robot. Comput. Integr. Manuf.*, **78** (2022), 102406. <https://doi.org/10.1016/j.rcim.2022.102406>
19. H. Li, J. Peng, X. Wang, J. Wan, Integrated resource assignment and scheduling optimization with limited critical equipment constraints at an automated container terminal, *IEEE Trans. Intell. Transp. Syst.*, **22** (2021), 7607–7618. <https://doi.org/10.1109/TITS.2020.3005854>
20. C. Yang, Y. Zhang, J. Wang, L. He, H. Wu, A deep reinforcement learning based multi-agent simulation optimization approach for igv bidirectional task allocation and charging joint scheduling in automated container terminals, *Comput. Oper. Res.*, **183** (2025), 107189. <https://doi.org/10.1016/j.cor.2025.107189>
21. J. C. Chen, T. L. Chen, Y. C. Teng, Meta-model based simulation optimization for automated guided vehicle system under different charging mechanisms, *Simul. Model. Pract. Theory*, **106** (2021), 102208. <https://doi.org/10.1016/j.simpat.2020.102208>
22. T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. Foerster, S. Whiteson, Monotonic value function factorisation for deep multi-agent reinforcement learning, *J. Mach. Learn. Res.*, **21** (2020), 1–51.



AIMS Press

©2026 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (<https://creativecommons.org/licenses/by/4.0>)