Secure Offloading And Data Privacy Protection Strategies For Edge Computing In Multi-Robot Systems

Duanjiao Li[1], Yongchao Liang[1], Wenxing Sun[1], Zibin Zhu[2], and Jianguo Zhang[3*]

[1] Guangdong Power Grid Co., Ltd, Guangzhou GuangDong, 510030, China
[2] Zhaoqing Power Supply Bureau, Guangdong Power Grid, Zhaoqing GuangDong, 526060, China
[3] Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen Guangdong, 518129, China
[*] Corresponding author. E-mail: zhangjianguo_2@126.com

Received: Jul. 21, 2026; Accepted: Aug. 16, 2026

Multi-robot systems in intelligent warehousing, industrial inspection, disaster relief, and UAV collaboration face complex tasks including target detection, map fusion, path planning, status diagnosis, and log uploading. Limited onboard computing, heterogeneous edge resources, and fluctuating links expose trajectories, maps, images, and status data during offloading, and traditional latency- or energy-only optimization fails to address scalability, security, and privacy together. This study proposes SPP-ESO, a security- and privacy-aware edge offloading strategy based on a three-layer robot-edge-cloud architecture, integrating task profiling, privacy level identification, dynamic trust assessment, candidate node filtering, multi-objective reinforcement learning, differential privacy perturbation, data minimization, and edge-failure migration into a unified model optimizing latency, energy, load balancing, security risk, privacy risk, and failure penalty together. With 200 robots, SPP-ESO achieves 218 ms average latency, 4.73 J energy consumption, 96.4% task completion, 3.6% SLA violation, 3.1% attack success, and 0.22 privacy risk, outperforming local-only, cloud-only, greedy-latency, DRL-offloading, and trust-aware baselines. At 300 robots under high concurrency, completion rate remains ∼94%.

Keywords: multi-robot system, scalable information system, edge computing, secure offloading, privacy protection, trust assessment, differential privacy, multi-agent reinforcement learning

http://dx.doi.org/10.6180/jase.202612_35.013

1. Introduction

Multi-robot systems are moving from labs to industrial, urban, and public safety scenarios, requiring path coordination, real-time anomaly identification, and collaborative perception under restricted communication. Modern multi-robot systems are complex information systems spanning terminals, networks, edge servers, cloud, and security modules, not just single-robot motion control, requiring edge or cloud resources given limited onboard capacity [1, 2].

Task offloading is not a simple local-versus-cloud choice: SLAM, recognition, path planning, and diagnosis tasks differ in data volume, load, deadlines, and privacy sensitivity [3, 4], requiring dynamic balance between task granularity, network state, edge resources, and safety.

Notable gaps remain in multi-robot edge computing research: many studies accelerate individual functions (SLAM, navigation, detection) without modeling the whole system as a scalable distributed information system [5]; offloading research spans mathematical programming, heuristics, and learning [6, 7] but rarely incorporates trust, attacks, or privacy into a unified goal; and security/privacy are often add-on modules given edge nodes' susceptibility to eavesdropping and inference [8]. This framework instead treats scalability, trust, privacy, and recovery as coupled design dimensions.

Robot-uploaded data typically includes location trajec-

tories, local maps, visual images, point cloud fragments, and collaborative task relationships, whose inference could expose warehouse routes, equipment layouts, or rescue strategies. Edge privacy protection must be designed with service scheduling and data transmission together [9], with multi-dimensional risks spanning identity, location, task content, and task-association privacy during MEC offloading [10]. Existing location-privacy-aware and privacy-entropy offloading studies [11, 12] are mostly based on phones, vehicles, or IoT terminals, without adequately characterizing joint "map-trajectory-image-task collaboration" privacy in multi-robot scenarios.

This study proposes a secure edge offloading and privacy protection strategy for scalable multi-robot systems, viewed as a three-layer robot-edge-cloud architecture. Each task gets a profile identifying data volume, computational needs, deadline, privacy level, and sensitivity; edge nodes undergo dynamic trust assessment with untrusted ones filtered; multi-objective optimization jointly considers latency, energy, edge load, security risk, privacy risk, and failure penalty; and the system selects among original offloading, feature-level uploading, DP perturbation, local execution, or trusted edge execution by privacy level. Simulations cover convergence, latency, energy, completion rate, SLA violation, load balancing, privacy budget, attacks, edge failure, ablation, and visualization — coupling offloading, trust, privacy, and recovery in one loop so security decisions precede resource allocation.

Contributions: a scalable robot-edge-cloud secure collaborative architecture integrating robot tasks, edge computing, cloud management, and security/privacy control into a unified system; a joint optimization model spanning task latency, energy, edge load imbalance, security risk, privacy risk, and failure penalty; the SPP-ESO dynamic offloading algorithm integrating privacy level identification, trust-aware candidate filtering, multi-objective decision-making, and task migration; and extensive simulation and visualization experiments verifying scalability, security robustness, and privacy protection.

Edge computing migrates tasks the robot cannot efficiently process to nearby edge servers. Multi-agent cloud robot research identifies challenges in resource pooling and task scheduling given limited robot-side computing [1]; edge robot research shows low-latency support for real-time perception and navigation [2]; Huang et al. accelerate multi-robot map construction via edge-deployed backend optimization [3]; Li et al. propose edge-accelerated collaborative motion planning [4].

These studies show edge computing improves task processing but often lack overall multi-robot system scalability modeling; Groshev et al. note edge robot deployment remains limited by network jitter and service orchestration [5]. This study therefore models the multi-robot system as a scalable information system handling heterogeneous tasks (detection, SLAM, path planning, diagnosis, log uploading) in one framework.

Task offloading is a core MEC issue: Islam et al. show offloading must consider resources, link quality, and task dependencies [6]; Saeik et al. review optimization, AI, and control-theory trade-offs [7].

Intelligent offloading has developed rapidly: attention-based DRL [13] and GNN-DRL dependency-task offloading [14] — but most focus on cost/latency with limited consideration of malicious nodes, privacy budgets, or multi-type robot data sensitivity.

Edge security issues arise from end-edge-cloud collaboration: MEC reviews show vulnerability to authentication, isolation, and privacy inference issues [8]; and security should be designed with scheduling and resource management, not just encryption [9]. For multi-robot systems, a compromised edge node may return incorrect results, causing path deviations or task failures.

Privacy-preserving offloading research shows data content, attributes, and location may be exposed during offloading [10]; location-privacy-aware offloading couples node selection with location protection [11]; and privacy-entropy-based offloading quantifies risk [12].

Federated learning, differential privacy, and edge-sensing clouds provide further support: federated learning reduces raw-data uploading in IoT [15].

Existing research provides an important foundation but has three shortcomings: multi-robot edge computing research is mostly single-task oriented (SLAM, navigation, UAV monitoring) without a unified security offloading model; intelligent offloading research treats edge nodes as trusted, without incorporating trust scoring, malicious nodes, or privacy budgets into a unified state space; and privacy-preserving offloading focuses on mobile/IoT location privacy without joint map-trajectory-image-task modeling for robots. SPP-ESO addresses these gaps, with the privacy level of each task directly affecting candidate-node filtering, reward evaluation, and migration decisions.

2. Methods

2.1. Scalable Multi-Robot Edge Cloud Architecture

The system has a robot terminal layer (mobile robots, drones, AGVs, inspection robots with sensors, communication, local computing, and task-cache), an edge computing layer (servers on gateways, industrial servers, or warehouse nodes handling detection, map fusion, path coordi-

nation, and scheduling), a cloud center layer (global model training, archiving, policy updates), and a security/privacy control layer spanning all three (authentication, trust assessment, privacy budget management, access control, edge failure migration) — shown in Fig. 1.

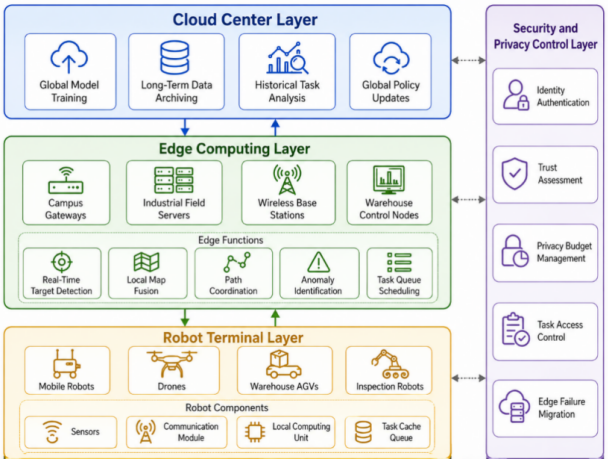

Fig. 1. Scalable robot-edge-cloud architecture for secure multi-robot information systems

The architecture's key is enabling task offloading, data protection, and trust assessment to work together: robots send task profiles and encrypted metadata to edge nodes, which return inference results and offloading decisions, while the cloud periodically updates global models and security policies. The security layer dynamically determines offloading eligibility, target node, DP perturbation need, and migration triggers based on privacy level and node trust, recording updates for later audit.

2.2. Task Model

Assume that there are N robots and M edge nodes in the system. Robot r_i generates the k task in time slice t, denoted by Eq. (1).

$$\mathcal{T}_{i,k}(t) = \{d_{i,k}, c_{i,k}, \tau_{i,k}, p_{i,k}, s_{i,k}, a_{i,k}, \rho_{i,k}\} \tag{1}$$

Local features include the task's input data amount, required CPU cycles, deadline, privacy level, security sensitivity, task type, and decomposability, classified into five task categories (Table 1) distinguishing indivisible real-time tasks from partitionable ones. $d_{i,k}c_{i,k}\tau_{i,k}p_{i,k}s_{i,k}a_{i,k}\rho_{i,k}$

2.3. Communication and Computation Model

Tasks can be executed locally, at the edge, or in the cloud. The offloading decision variable $x_{i,k}(t) \in \{0, 1, 2, \ldots, M, M+1\}$ is defined. Here, $x_{i,k} = 0$ indicates local execution, $x_{i,k} = m$ indicates offloading to edge node e_m, $x_{i,k} = M+1$ indicates uploading to the cloud. The wireless transmission rate between robot r_i and edge node e_m is defined as Eq. (2).

$$R_{i,m}(t) = B_{i,m}(t) \log_2 \left(1 + \frac{P_i(t) h_{i,m}(t)}{\sigma^2 + I_{i,m}(t)}\right) \tag{2}$$

The upload delay, execution delay, and total delay depend on the allocated bandwidth, transmit power, channel gain, noise, and interference, evaluated at the task level and averaged over completed tasks in simulation results. $B_{i,m}P_i h_{i,m}\sigma^2 I_{i,m} T^{\text{up}}_{i,k,m}(t) = \frac{\phi(p_{i,k})d_{i,k}}{R_{i,m}(t)} T^{\text{exe}}_{i,k,m}(t) = \frac{c_{i,k}}{f_m(t)\mu_{i,k,m}(t)}$

$$T^{\text{edge}}_{i,k,m}(t) = T^{\text{up}}_{i,k,m}(t) + T^{\text{queue}}_{m}(t) + T^{\text{exe}}_{i,k,m}(t) + T^{\text{down}}_{i,k,m}(t) \tag{3}$$

where $\phi(p_{i,k})$ denotes the data retention ratio after privacy processing; for high-privacy tasks, $\phi < 1$ can be achieved through feature-level uploading or local summary uploading; f_m denotes the computation frequency of edge nodes; and $\mu_{i,k,m}$ denotes the proportion of computational resources allocated to the task. The local execution latency and energy consumption of the robot are as follows:

$$T^{\text{local}}_{i,k}(t) = \frac{c_{i,k}}{f^{\text{local}}_i(t)}, \quad E^{\text{local}}_{i,k}(t) = \kappa_i c_{i,k} \left(f^{\text{local}}_i(t)\right)^2 \tag{4}$$

The offloading power consumption is mainly determined by the wireless transmission power and upload latency.

$$E^{\text{off}}_{i,k,m}(t) = P^{\text{tx}}_i(t) T^{\text{up}}_{i,k,m}(t) + P^{\text{idle}}_i(t) \left(T^{\text{queue}}_m(t) + T^{\text{exe}}_{i,k,m}(t)\right) \tag{5}$$

2.4. Privacy Model

Robot data privacy has four levels: Level 1 (ordinary status — battery, speed, temperature); Level 2 (task logs, diagnostics); Level 3 (location trajectories, collaborative paths, map indexes); and Level 4 (images, videos, point clouds, task-sensitive data), with higher levels less suitable for direct raw upload. This rule-based scheme can be replaced by application-specific classifiers when labeled privacy data are available.

$$\begin{aligned} R^{\text{priv}}_{i,k}(t) =& \alpha_1 P^{\text{loc}}_{i,k} + \alpha_2 P^{\text{assoc}}_{i,k} + \alpha_3 P^{\text{content}}_{i,k} + \alpha_4 P^{\text{map}}_{i,k} \\ &+ \alpha_5 P^{\text{identity}}_{i,k} \end{aligned} \tag{6}$$

where P^{loc} denotes the risk of location leakage, P^{assoc} denotes the risk of task association leakage, P^{content} denotes the risk of image or video content leakage, P^{map} denotes the risk of map structure leakage, and P^{identity} denotes the risk of robot identity or task identity leakage. For trajectories and map summaries that require differential privacy protection, the perturbation mechanism $\mathcal{M}$ satisfies Eq. (7).

$$\Pr[\mathcal{M}(D) = y] \leq e^{\epsilon} \Pr[\mathcal{M}(D') = y] \tag{7}$$

where D and D' are neighboring datasets, and ϵ is the privacy budget. A smaller ϵ provides stronger privacy protection, but may negatively impact task accuracy and latency costs. The privacy leakage score is normalized to the interval [0,1], where larger values indicate higher inferred exposure under the same task type.

Table 1. Task types and privacy levels in multi-robot systems

Task type	Main data type	Real-time requirement	Privacy	Typical processing
Obstacle avoidance	Sensor stream	High	Low	Local or nearby edge
Object detection	Image/video	High	Medium-high	Feature upload and edge inference
SLAM update	Point cloud/map	Medium-high	High	Trusted edge fusion
Path planning	Trajectory/constraints	High	High	Local constraints and edge coordination
Status diagnosis	Logs/status	Medium-low	Medium	Edge batch processing or cloud archive

2.5. Trust and Threat Model

Each edge node e_m has a dynamic trust value $Q_m(t)$. The trust value is determined by the historical task completion rate, anomaly return rate, response stability, load stability, access behavior, and security event records. The current behavior score can be expressed as follows:

$$\begin{aligned} S_m(t) =& \beta_1 C_m(t) + \beta_2(1 - A_m(t)) + \beta_3 R_m(t) + \beta_4 L_m(t) \\ &+ \beta_5 B_m(t) + \beta_6 H_m(t) \end{aligned} \tag{8}$$

The trust update process is as follows.

$$Q_m(t+1) = \lambda Q_m(t) + (1-\lambda) S_m(t) - \chi_m(t)\Delta_m(t) \tag{9}$$

where C_m is the task completion rate, A_m is the abnormal return rate, R_m is the response stability, L_m is the load stability, B_m is the access behavior compliance, H_m is the historical security record, $\chi_m(t)$ is the attack detection trigger variable, and $\Delta_m(t)$ is the trust penalty. High-privacy tasks must meet the trust threshold constraint:

$$\begin{aligned} Q_m(t) &\geq Q_{\min}(p_{i,k}), \\ Q_{\min}(p=4) &> Q_{\min}(p=3) \\ &> Q_{\min}(p=2) > Q_{\min}(p=1) \end{aligned} \tag{10}$$

The model considers five threats: eavesdropping (transmission phase), malicious edge nodes (computation phase), forged results (task reliability), privacy inference (location/trajectory/map data), and edge failure (fault tolerance). A proportion of semi-honest or malicious nodes is assumed to exist; collusion among multiple compromised nodes is left for future work.

2.6. SPP-ESO Security- and Privacy-Aware Offloading Method

2.6.1. *Overall Method Flow*

SPP-ESO's workflow (Fig. 2): read tasks from the queue and generate a task profile; identify privacy level from task type and data content; calculate edge-node trust from historical behavior and real-time feedback; filter unqualified nodes via trust threshold to form a candidate set; and make multi-objective offloading decisions within the candidate set, selecting original upload, feature upload, DP perturbation, local execution, or trusted edge execution by privacy level.

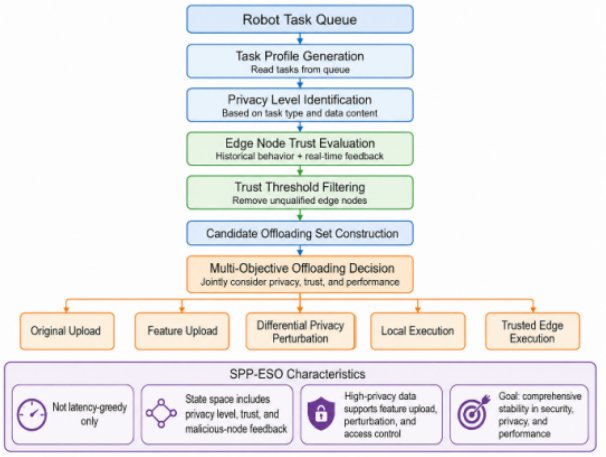

Fig. 2. Workflow of the proposed SPP-ESO algorithm

Unlike latency-greedy algorithms, SPP-ESO doesn't offload to the fastest-responding node; unlike standard DRL, it incorporates trust, privacy level, and malicious-node feedback into its state space; unlike trust-only algorithms, it performs feature-level uploading and perturbation for high-privacy data — maintaining comprehensive stability in security, privacy, and performance as the system scales, separating security filtering from reward optimization.

2.6.2. *Multi-Objective Optimization Model*

This study constructs the offloading decision as a multi-objective constrained optimization problem. The overall objective function is defined as:

$$\min_{x,\mu,\epsilon} J = \omega_T \bar{T} + \omega_E \bar{E} + \omega_L \mathcal{L}_{\text{imb}} + \omega_S \mathcal{R}_{\text{sec}} + \omega_P \mathcal{R}_{\text{priv}} + \omega_F P_{\text{fail}} \tag{11}$$

where $\bar{T}$ is the average task latency, $\bar{E}$ is the average robot energy consumption, $\mathcal{L}_{\text{imb}}$ is the edge load imbalance, $\mathcal{R}_{\text{sec}}$ is the security risk, $\mathcal{R}_{\text{priv}}$ is the privacy leakage risk, and P_{fail} is the task failure penalty. The edge load imbalance is defined as Before aggregation, each objective term is normalized so that latency,

energy consumption, load imbalance, security risk, privacy risk, and failure penalties can be compared on a consistent scale.

$$\mathcal{L}_{\text{imb}} = \frac{1}{M}\sum_{m=1}^{M}\left(u_m - \frac{1}{M}\sum_{j=1}^{M}u_j\right)^2 \tag{12}$$

where u_m denotes the CPU utilization of edge node e_m. Security risk is defined as the following:

$$\mathcal{R}_{\text{sec}} = \frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K}\mathbb{I}(x_{i,k}=m)(1-Q_m(t))s_{i,k} \tag{13}$$

The penalty for task failure is defined as:

$$P_{\text{fail}} = \frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K}[\mathbb{I}(T_{i,k}>\tau_{i,k}) + \mathbb{I}(\hat{y}_{i,k}\neq y_{i,k})] \tag{14}$$

The constraints of the optimization problem include task deadline, edge computing resource, wireless bandwidth, privacy budget, and trust threshold constraints.

$$\begin{gathered} T_{i,k}(t) \le \tau_{i,k}, \\ \sum_{i=1}^{N}\sum_{k=1}^{K}\mathbb{I}(x_{i,k}=m)c_{i,k} \le C_m(t), \\ \sum_{i=1}^{N} B_{i,m}(t) \le B_m^{\max}, \\ 0 < \epsilon_{i,k} \le \epsilon_{\max}(p_{i,k}), \\ Q_m(t) \ge Q_{\min}(p_{i,k}). \end{gathered} \tag{15}$$

2.6.3. Dynamic Offloading Decision

The SPP-ESO models the multi-robot edge offloading process as a Markov decision process. The state space is:

$$S_t = (\mathbf{D}_t, \mathbf{C}_t, \boldsymbol{\tau}_t, \mathbf{P}_t, \mathbf{Q}_t, \mathbf{U}_t, \mathbf{R}_t, \mathbf{A}_t) \tag{16}$$

where $\mathbf{D}_t$ is the task data volume matrix, $\mathbf{C}_t$ is the computation requirement matrix, $\boldsymbol{\tau}_t$ is the task deadline matrix, $\mathbf{P}_t$ is the privacy level matrix, $\mathbf{Q}_t$ is the edge node trust matrix, $\mathbf{U}_t$ is the edge load status, $\mathbf{R}_t$ is the link rate status, and $\mathbf{A}_t$ is the attack detection history. The action space includes execution-location selection, edge-node selection, privacy-processing level selection, and task-migration selection. The reward function is defined as

$$G_t = -J_t + \eta_1\Gamma_t^{\text{succ}} - \eta_2\Gamma_t^{\text{attack}} - \eta_3\Gamma_t^{\text{leak}} - \eta_4\Gamma_t^{\text{sla}} \tag{17}$$

where Γ^{succ} is the reward for task success, Γ^{attack} is the penalty for attack, Γ^{leak} is the penalty for privacy leakage, and Γ^{sla} is the penalty for SLA violation. To reduce the action space complexity in large-scale scenarios, the algorithm performs candidate node filtering before making reinforcement learning decisions. The reward weights are kept fixed within each scenario, which makes the comparison across different robot scales and malicious-node ratios more stable.

$$\begin{aligned} \mathcal{E}_{i,k}^{\text{cand}}(t) = \{e_m \mid\ & Q_m(t) \ge Q_{\min}(p_{i,k}),\ C_m(t) \\ & > c_{i,k},\ R_{i,m}(t) > R_{\min}\} \end{aligned} \tag{18}$$

Only edge nodes meeting trust, resource, and link conditions enter the candidate set, preventing high-privacy tasks from reaching low-trust nodes, reducing invalid actions, and improving convergence in scenarios above 100 robots, since constraints are checked before the reinforcement-learning action is selected.

2.6.4. Privacy Protection Mechanism

SPP-ESO's privacy mechanism has three layers: transmission security (authenticated, lightweight-encrypted metadata and results); data minimization (uploading intermediate features and summaries rather than full images or trajectories); and differential privacy perturbation with access control (noise added to trajectories and map summaries, restricting low-trust node access to high-privacy tasks).

For object detection, robots upload only candidate regions and intermediate features; for SLAM, only keyframes and map summaries rather than the complete map; for path planning, perturbed trajectory constraints and safe-region boundaries; and diagnosis data is batch-processed at the edge. This hierarchical approach avoids uniform privacy treatment — low-privacy tasks avoid excessive perturbation while high-privacy tasks get feature uploading or trusted processing.

2.6.5. Algorithm Complexity and Scalability Analysis

With N robots, M edge nodes, and average T tasks per robot, exhaustive offloading complexity is unsuitable at scale. SPP-ESO filters candidates by privacy level, trust, resource, and link status, reducing candidates per task substantially, with per-round complexity from task profiling, candidate filtering, and policy reasoning combined. $NMK\mathcal{O}\left((M+2)^{NK}\right) M\bar{M}\bar{M} < M\mathcal{O}(NK)\mathcal{O}(NKM)\mathcal{O}(NK\bar{M})$

$$\mathcal{O}_{\text{SPP–ESO}} = \mathcal{O}(NKM) + \mathcal{O}(NK\bar{M}) + \mathcal{O}(M) \tag{19}$$

With a small candidate ratio and trust-constrained filtering, complexity grows approximately linearly with tasks. This candidate filtering and regional scheduling prevents centralized bottlenecks in large-scale systems, shifting cost from exhaustive enumeration to policy inference over a smaller trusted set. $M\bar{M}$

2.7. Simulation Setup

2.7.1. Experimental Environment and Parameters

Simulation used Python 3.9/PyTorch 1.13 with ROS/Gazebo-style visualization, configured with 20-300 robots and 3-10 edge nodes. Each robot generates Poisson-process tasks (obstacle avoidance, detection, SLAM, path planning, diagnosis) with 0.5-50MB data size and 0.5-30 Gcycles demand. Privacy budgets and malicious-node ratios (0-30%) were varied (Table 2), with each configuration repeated across random seeds and averaged. $\epsilon = 0.5, 1, 2, 4, 8$

2.7.2. Comparison Algorithms and Evaluation Metrics

Seven baselines are compared (Table 3): Local-only (all local); Cloud-only (all cloud); Random Offloading (random location); Greedy-Latency (lowest estimated latency); DRL-Offloading (standard RL, no trust/privacy); Trust-aware Only (trust without privacy budget); and SPP-ESO — covering local, centralized, random, latency-greedy, learning-based, and trust-aware scheduling.

Evaluation metrics include average latency, energy consumption, completion rate, SLA violation rate, throughput, edge load balancing, migration success rate, attack success rate, privacy leakage risk, communication overhead, and convergence speed, with privacy risk computed from location, association, content, and map re-identification inference. Lower latency, energy, SLA violation, attack success, and privacy risk indicate better performance; higher completion and recovery rates indicate better service quality.

Table 2. Simulation parameters

Parameter	Setting
Number of robots	20, 50, 100, 150, 200, 300
Number of edge nodes	3, 5, 8, 10
Cloud server	1
Task types	Detection, SLAM, planning, diagnosis, log uploading
Task data size	0.5–50 MB
Computational demand	0.5–30 Gcycles
Robot CPU frequency	0.5–2.0 GHz
Edge CPU frequency	10–80 GHz
Wireless bandwidth	5, 10, 20, 40 MHz
Privacy budget	0.5, 1, 2, 4, 8, No-DP
Malicious edge ratio	0%, 10%, 20%, 30%
Edge failure ratio	0%, 5%, 10%, 20%
Training episodes	1000
Learning rate	0.001

Table 3. Baseline methods

Method	Description
Local-only	All tasks are executed locally on robots
Cloud-only	All tasks are uploaded to the cloud
Random Offloading	Execution location is selected randomly
Greedy-Latency	The position with the lowest estimated latency is selected
DRL-Offloading	Deep reinforcement learning without trust and privacy constraints
Trust-aware Only	Trust is considered, but privacy budget is not modeled
SPP-ESO	Proposed method considering latency, energy, load, trust, and privacy

3. Results and discussion

3.1. Algorithm Convergence

Fig. 3 shows reward convergence across 50-300 robots: larger scale requires more episodes (50 robots: ∼300; 100: ∼420; 200: ∼580; 300: ∼750, with slower but still stable convergence), showing candidate filtering, trust constraints, and multi-objective reward design mitigate policy oscillation at scale.

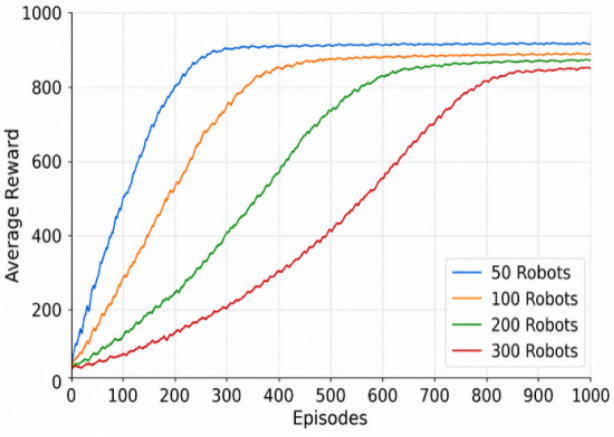

Fig. 3. Convergence curves under different robot scales

3.2. Average Latency Comparison

Fig. 4 shows average latency by robot count: Local-only rises steepest from all-local computation; Cloud-only suffers uplink/round-trip latency; Random Offloading fluctuates from ignoring link/resource status; Greedy-Latency performs well at low load but congests at high load; DRL-Offloading learns dynamic strategies but may choose low-trust nodes. SPP-ESO achieves 218 ms average latency at 200 robots, lower than Trust-aware Only, DRL-Offloading, and Greedy-Latency — showing security/privacy constraints don't impose large latency penalties but reduce overall latency via load balancing and data minimization.

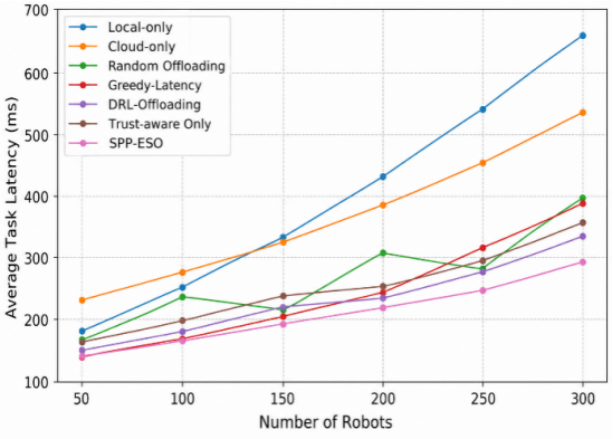

Fig. 4. Average task latency under different robot numbers

3.3. Energy Consumption Comparison

Fig. 5 shows energy consumption: Local-only consumes most from extended local computation; Cloud-only reduces local computing energy but full-data upload increases communication energy; Greedy-Latency's high-bandwidth link preference destabilizes energy use; SPP-ESO reduces raw upload volume via feature-level and summary uploading while avoiding long compute-intensive local tasks, maintaining low energy across scales — showing privacy protection combined with data minimization can reduce transmission overhead rather than increase it.

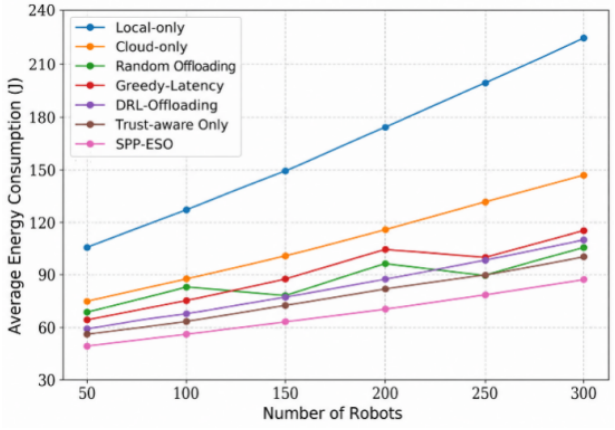

Fig. 5. Average energy consumption under different robot numbers

3.4. Task Completion Rate and SLA Satisfaction

Fig. 6 shows completion rate by robot count: differences are small at low counts, but above 150 robots, edge competition and congestion widen the gap. Cloud-only and Random Offloading decline rapidly; Greedy-Latency declines after 200 robots from node overload; SPP-ESO maintains ∼94% completion at 300 robots, showing stable performance at scale.

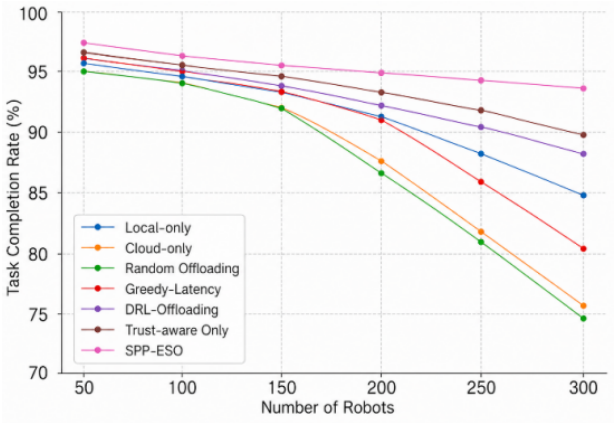

Fig. 6. Task completion rate under different robot numbers

Table 4 shows 200-robot results: SPP-ESO balances latency, energy, completion rate, SLA violation, attack success, and privacy risk. Local-only has 0% attack success (no offloading) but excessive latency/energy; Greedy-Latency has lower latency but highest attack success and privacy risk, showing latency-only node selection creates security risk. SPP-ESO's advantage is filtering untrusted nodes via trust threshold, then optimizing within the trusted set — improving efficiency and security/privacy together.

3.5. Edge Load Balancing Analysis

Fig. 7 shows edge CPU utilization at 200 robots: Greedy-Latency concentrates tasks on best-link nodes (some exceeding 85% utilization while others are low); Random Offloading distributes randomly but with unstable completion; DRL-Offloading improves distribution but may offload sensitive tasks to low-trust nodes without privacy filtering; SPP-ESO achieves more balanced utilization by scheduling within trusted nodes, reducing both average latency and overload/failure risk.

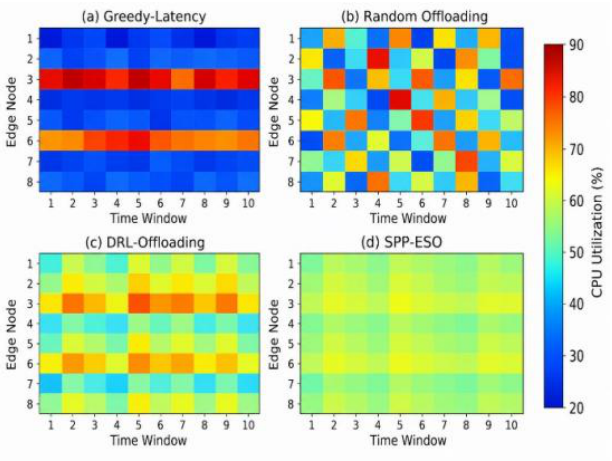

Fig. 7. Edge load balancing heatmap under different algorithms

3.6. Security Robustness Under Malicious Edge Nodes

Fig. 8 shows attack success as malicious-node ratio rises: at 0%, all algorithms are unaffected; at 10-30%, Random Offloading, Greedy-Latency, and DRL-Offloading rise significantly from offloading to untrusted low-latency nodes; Trust-aware Only reduces success but lacks privacy-level control for high-privacy tasks; SPP-ESO's attack success stays low even at 30% malicious nodes, showing trust scoring, privacy level, and candidate filtering work synergistically.

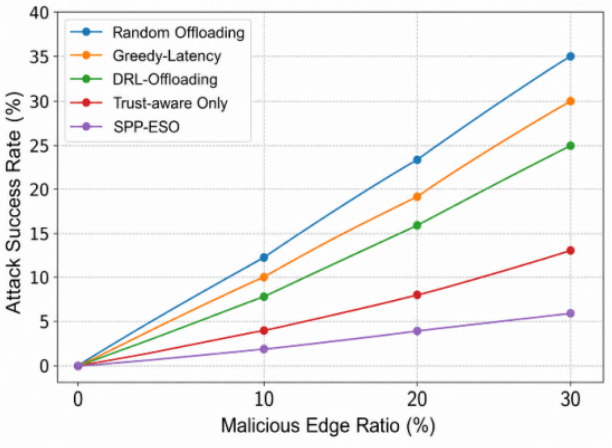

Fig. 8. Attack success rate under different malicious edge ratios

Fig. 9 shows trust-value evolution: normal nodes gradually stabilize at high trust from consistent task completion; slightly anomalous nodes fluctuate; malicious nodes decline rapidly after abnormal returns and violations, falling below the high-privacy threshold — showing the trust mechanism is dynamically updated from operational feedback, not a static whitelist.

Table 4. Representative quantitative results under 200 robots

Method	Latency/ms	Energy/J	Completion/%	SLA/%	Attack/%	Privacy risk
Local-only	412	8.91	82.4	17.6	0.0	0.18
Cloud-only	386	6.72	85.1	14.9	8.7	0.41
Random	338	6.28	87.9	12.1	14.8	0.46
Greedy-L	279	5.84	90.6	9.4	18.5	0.51
DRL	246	5.27	92.8	7.2	13.6	0.43
Trust-only	261	5.51	93.5	6.5	6.2	0.36
SPP-ESO	218	4.73	96.4	3.6	3.1	0.22

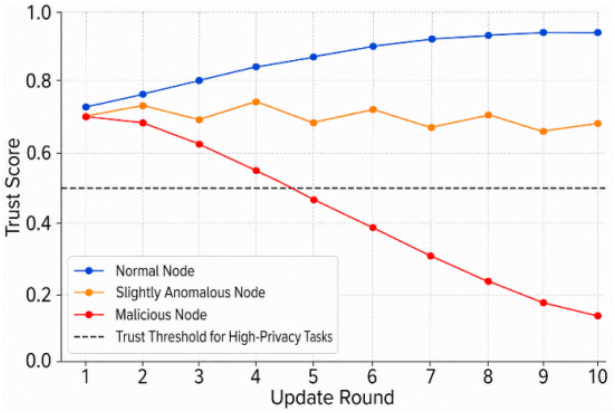

Fig. 9. Trust score evolution of normal and malicious edge nodes

3.7. Edge Failure Recovery Capability

Fig. 10 shows completion-rate recovery after a high-load edge node fails at time slice 500: Greedy-Latency drops significantly from concentrated tasks; DRL-Offloading takes long to relearn distribution; Trust-aware Only improves recovery speed but lacks privacy-task migration priority; SPP-ESO recovers quickly via candidate reconstruction, task migration, and high-privacy task priority protection — demonstrating fault tolerance in dynamic edge environments.

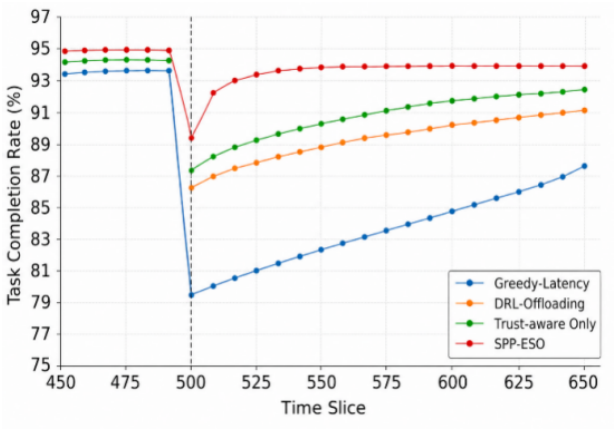

Fig. 10. System recovery after edge node failure

3.8. Ablation Experiments

Table 5's ablation shows: removing the trust module significantly raises attack success; removing the privacy module significantly raises privacy risk; removing load balancing raises SLA violation rate; removing migration slows failure recovery. Full SPP-ESO has the lowest overall cost, showing each module contributes independently.

3.9. Comparison of Real-Scene Visualizations under Different Algorithms

Fig. 11 compares object detection and privacy masking on the same scene: Cloud-only has high confidence but large latency; Greedy-Latency has low latency but fails to mask sensitive areas; DRL-Offloading balances detection and latency but may expose device numbers or personnel; Trust-aware Only avoids some low-trust nodes but insufficient privacy processing; SPP-ESO maintains low latency while masking sensitive areas and high detection confidence — showing privacy-aware offloading affects actual visual outputs, not just abstract metrics.

Fig. 11. Real-scene visual comparison of object detection and privacy masking under different offloading algorithms

Fig. 12 compares map fusion and trajectory privacy processing: Cloud-only gives a complete map but high latency; Greedy-Latency preserves the original trajectory, exposing the operational path; DRL-Offloading improves efficiency but lacks perturbation; Trust-aware Only restricts some areas but handles trajectory weakly; SPP-ESO preserves key map structure while perturbing sensitive areas and precise trajectories — showing map completeness alone is insufficient without considering trajectory exposure.

Table 5. Ablation and robustness results

Variant	Latency/ms	Energy/J	SLA/%	Attack/%	Privacy risk	Recovery slots
SPP-ESO full	218	4.73	3.6	3.1	0.22	80
w/o trust	229	4.86	4.3	11.8	0.31	105
w/o privacy	214	4.62	3.9	6.4	0.49	92
w/o load	246	5.18	8.7	4.6	0.25	130
w/o migration	232	4.95	5.9	4.0	0.24	210

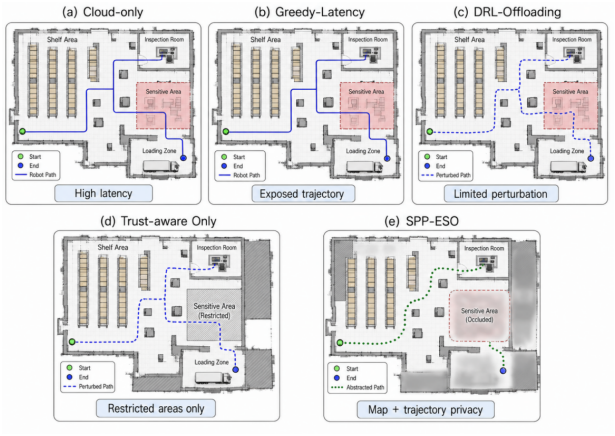

Fig. 12. Real-scene visual comparison of local map fusion and trajectory privacy under different algorithms

3.10. Discussion

This methodology aligns with scalable information systems: the research object is a distributed system of many robots, edge nodes, a cloud center, and security modules, not a single robot controller; multi-robot tasks are processed in parallel across local, edge, and cloud; and experiments span 20-300 robots, varying edge nodes and attack ratios to verify scaling behavior across latency, energy, completion rate, load, and privacy risk.

SPP-ESO suits three deployment scenarios: intelligent warehousing (AGVs handling, avoiding obstacles, reporting status, with edge nodes in warehouse control areas allocating resources by urgency); industrial inspection (processing equipment images while preventing leakage of equipment numbers and layouts); and disaster relief (unstable communication and partial edge failure, where task migration improves robustness) — all requiring edge nodes to maintain low-latency inference and auditable privacy-control records.

Limitations: experiments are mainly simulation-based, requiring further validation against real hardware, sensor errors, and physical constraints; trust scoring relies on historical feedback, risking detection delays for short-term intense attacks; differential privacy perturbations can affect map quality and path accuracy, motivating future semantic-level privacy protection and trusted execution environments; and privacy risk indicators focus on location, association, content, and map leakage, with attacker background knowledge and coordinated attacks left for future work alongside hardware-in-the-loop validation and adaptive privacy budgets.

4. Conclusions

This study addresses secure offloading and data privacy protection for multi-robot edge computing via SPP-ESO, integrating task profiling, trust assessment, candidate filtering, multi-objective offloading, differential privacy, and edge failure migration into a unified framework. At 200 robots, it achieves 218 ms latency, 4.73 J energy, 96.4% completion, 3.6% SLA violation, 3.1% attack success, and 0.22 privacy risk, maintaining ∼94% completion at 300 robots. Limitations include simulation-based validation, historical-feedback trust scoring's potential lag, and DP perturbation's effect on map quality; future work includes hardware-in-the-loop testing, semantic-level privacy protection, trusted execution environments, and federated reinforcement learning.

Acknowledgments

This paper is supported by the Guangdong Power Grid Company Limited (Grant No. GDKJXM20240403).

References

[1] M. Afrin, J. Jin, A. Rahman, A. Rahman, J. Wan, and E. Hossain, (2021) *"Resource Allocation and Service Provisioning in Multi-Agent Cloud Robotics: A Comprehensive Survey"* **IEEE Communications Surveys & Tutorials** ***23***(2): 842–870. DOI: 10.1109/COMST.2021.3061435.

[2] N. Tahir and R. Parasuraman, (2025) *"Edge Computing and Its Application in Robotics: A Survey"* **Journal of Sensor and Actuator Networks** ***14***(4): 65. DOI: 10.3390/jsan14040065.

[3] P. Huang, L. Zeng, X. Chen, K. Luo, Z. Zhou, and S. Yu, (2022) *"Edge Robotics: Edge-Computing-Accelerated Multirobot Simultaneous Localization and Mapping"* **IEEE Internet of Things Journal** ***9***(15): 14087–14102. DOI: 10.1109/JIOT.2022.3146461.

[4] G. Li, R. Han, S. Wang, F. Gao, Y. C. Eldar, and C. Xu, (2025) *"Edge Accelerated Robot Navigation With Collaborative Motion Planning"* **IEEE/ASME Transactions on Mechatronics** ***30***(2): 1166–1178. DOI: 10.1109/TMECH.2024.3419436.

[5] M. Groshev, G. Baldoni, L. Cominardi, A. de la Oliva, and R. Gazda, (2023) *"Edge Robotics: Are We Ready? An Experimental Evaluation of Current Vision and Future Directions"* **Digital Communications and Networks** ***9***(1): 166–174. DOI: 10.1016/j.dcan.2022.04.032.

[6] A. Islam, A. Debnath, M. Ghose, and S. Chakraborty, (2021) *"A Survey on Task Offloading in Multi-Access Edge Computing"* **Journal of Systems Architecture** ***118***: 102225. DOI: 10.1016/j.sysarc.2021.102225.

[7] F. Saeik, M. Avgeris, D. Spatharakis, N. Santi, D. Dechouniotis, J. Violos, A. Leivadeas, N. Athanasopoulos, N. Mitton, and S. Papavassiliou, (2021) *"Task Offloading in Edge and Cloud Computing: A Survey on Mathematical, Artificial Intelligence and Control Theory Solutions"* **Computer Networks** ***195***: 108177. DOI: 10.1016/j.comnet.2021.108177.

[8] P. Ranaweera, A. D. Jurcut, and M. Liyanage, (2021) *"Survey on Multi-Access Edge Computing Security and Privacy"* **IEEE Communications Surveys & Tutorials** ***23***(2): 1078–1124. DOI: 10.1109/COMST.2021.3062546.

[9] M. Yahuza, M. Y. I. B. Idris, A. W. B. Abdul Wahab, A. T. S. Ho, S. Khan, S. N. B. Musa, and A. Z. B. Taha, (2020) *"Systematic Review on Security and Privacy Requirements in Edge Computing: State of the Art and Future Research Opportunities"* **IEEE Access** ***8***: 76541–76567. DOI: 10.1109/ACCESS.2020.2989456.

[10] T. Li, X. He, S. Jiang, and J. Liu, (2022) *"A Survey of Privacy-Preserving Offloading Methods in Mobile-Edge Computing"* **Journal of Network and Computer Applications** ***203***: 103395. DOI: 10.1016/j.jnca.2022.103395.

[11] Z. Wang, Y. Sun, D. Liu, J. Hu, X. Pang, Y. Hu, and K. Ren, (2024) *"Location Privacy-Aware Task Offloading in Mobile Edge Computing"* **IEEE Transactions on Mobile Computing** ***23***(3): 2269–2283. DOI: 10.1109/TMC.2023.3254553.

[12] D.-G. Zhang, H.-Z. An, J. Zhang, T. Zhang, W.-M. Dong, and X.-R. Jiang, (2024) *"Novel Privacy Awareness Task Offloading Approach Based on Privacy Entropy"* **IEEE Transactions on Network and Service Management** ***21***(3): 3598–3608. DOI: 10.1109/TNSM.2024.3355967.

[13] K. Li, X. Wang, Q. He, M. Yang, M. Huang, and S. Dustdar, (2023) *"Task Computation Offloading for Multi-Access Edge Computing via Attention Communication Deep Reinforcement Learning"* **IEEE Transactions on Services Computing** ***16***(4): 2985–2999. DOI: 10.1109/TSC.2022.3225473.

[14] Z. Cao, X. Deng, S. Yue, P. Jiang, J. Ren, and J. Gui, (2024) *"Dependent Task Offloading in Edge Computing Using GNN and Deep Reinforcement Learning"* **IEEE Internet of Things Journal** ***11***(12): 21632–21646. DOI: 10.1109/JIOT.2024.3374969.

[15] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, (2021) *"Federated Learning for Internet of Things: A Comprehensive Survey"* **IEEE Communications Surveys & Tutorials** ***23***(3): 1622–1658. DOI: 10.1109/COMST.2021.3075439.