electronics

MDPI

Article

Hierarchical Deep Reinforcement Learning with Formal Collision Safety for WRSN Charging in Obstacle-Dense Environments

Jingjing Chen [1,*], Bo Yan [1], Rongjie Wang [2] and Zhiming Huang [3]

[1] College of Physics and Mechanical & Electrical Engineering, Longyan University, Longyan 364000, China; bo.yan_@outlook.com
[2] Quanzhou Vocational and Technical University School of Technologies, Quanzhou 362000, China; wangrongjie@qvtu.edu.cn
[3] College of Mathematics and Information Engineering, Longyan University, Longyan 364000, China; zhiming.huang_@outlook.com
* Correspondence: 82007003@lyun.edu.cn; Tel.: +86-159-6088-5822

Abstract

Energy replenishment in Wireless Rechargeable Sensor Networks (WRSNs) is challenging in obstacle-dense environments. It tightly couples two sub-problems: charging scheduling and trajectory planning. Existing heuristics and end-to-end deep reinforcement learning (DRL) methods cannot jointly solve these tasks with formal safety guarantees. To address this, we propose a hierarchical DRL framework that decouples the two sub-problems. At the high level, a Deep Q-Network (DQN) selects the next charging target based on the global energy state. At the low level, an Improved A* planner uses obstacle dilation and Bézier smoothing to generate a provably collision-free and mechanically feasible trajectory. This dilation mechanism strictly ensures physical safety. Our ablation study shows that disabling this mechanism introduces 25.20 ± 13.31 collisions per episode. We evaluated our framework on networks with 100 nodes and 10 obstacles. Compared to the end-to-end DDPG baseline, our approach reduces the Node Dead Ratio (NDR) from 0.950 to 0.785 and completely eliminates collisions. Furthermore, it reduces the median cumulative turning angle by 96% across 20 randomized topologies, and achieves $3.8\times$ the charging efficiency of the EDF heuristic. Ultimately, this framework occupies a distinct Pareto-optimal point on the efficiency–reliability–safety frontier.

Keywords: wireless rechargeable sensor networks; mobile charger; deep reinforcement learning; obstacle-aware path planning; formal safety guarantee; hierarchical reinforcement learning

Academic Editor: Valeri Mladenov

Received: 24 May 2026
Revised: 21 June 2026
Accepted: 24 June 2026
Published: 15 July 2026

1. Introduction

Wireless Sensor Networks (WSNs) form the critical foundation of the Internet of Things (IoT). They enable wide-scale environmental monitoring [1], smart agriculture [2], and intelligent transportation [3]. However, sensor nodes are fundamentally constrained by limited battery capacities, which heavily restrict a network's longevity. Wireless Rechargeable Sensor Networks (WRSNs) address this bottleneck by utilizing Mobile Chargers (MCs) to provide on-demand energy replenishment via Wireless Power Transfer (WPT). The critical bottleneck in practical WRSNs is that battery depletion is a dynamic temporal process, while obstacle avoidance is a complex spatial constraint. Combining concurrent

charging demands with dense obstacles creates a highly non-linear challenge: a target node might appear spatially close, but obstacle-induced detours can cause massive travel delays, leading to the catastrophic energy depletion of other waiting nodes.

While MC-based architecture offers flexibility, practical deployment environments are rarely unobstructed [4–6]. Physical barriers such as buildings, terrain variations, and restricted zones severely complicate MC navigation. In such obstacle-populated environments, the scheduling of nodes and trajectory planning for MC are tightly coupled. The decision of which node to charge next depends directly on the feasible path required to reach it.

Historically, researchers have approached this by relying on greedy algorithms [7,8] or static heuristics [9,10] that prioritize nodes based on fixed weightings of residual energy and Euclidean distance. While computationally lightweight, traditional heuristics fail to adapt to dynamic environments. They frequently fall into "spatial traps". A target node might appear spatially close. However, complex obstacle detours demand massive energy and time to reach it. As a result, other critical nodes which await service suffer from complete energy depletion.

To bridge the gap between dynamic energy urgency and complex spatial constraints, we propose a unified framework based on deep reinforcement learning (DRL) in our work. Different from heuristic methods, DRL agents can learn optimal decision-making policies through continuous interaction with the environment. This enables them to recognize complex, non-linear relationships between node energy depletion rates and obstacle-induced travel delays. We propose a Deep Q-Network (DQN) architecture that acts as an intelligent high-level scheduler. This architecture effectively identifies and outputs the most critical target node. To ensure absolute physical safety and mechanical feasibility, this target is passed to a low-level, enhanced A* path-planning algorithm that generates a collision-free, smoothed trajectory.

The main contributions of this work are summarized as follows:

(1) A hierarchical DRL framework for obstacle-aware WRSN charging that decouples high-level combinatorial scheduling (DQN) from low-level geometric navigation (Improved A* + smoothing). The framework provides formal collision safety through obstacle dilation—a guarantee unavailable to end-to-end DRL methods. We empirically verify this guarantee through an ablation experiment (Section 6.6): disabling dilation while keeping all other components unchanged immediately introduces 25.20 ± 13.31 collisions per episode, confirming that the safety property is structurally contributed by the dilation mechanism, not by the DRL training.

(2) An integrated trajectory smoothing module (Algorithm 4) that reduces the median cumulative turning angle of raw A* output by 96% across $n = 20$ randomized topologies (per-trial reductions range from 79.4% to 100.0%), while preserving the formal collision-free guarantee inherited from the dilated A* planner. The smoothing operates as a post-processing step.

(3) A comprehensive empirical evaluation against three baselines under varied conditions. Against end-to-end DDPG, our framework simultaneously achieves lower NDR (0.785 vs. 0.950), zero collisions (vs. 4.80 ± 4.21 per episode at M = 10), and substantially better trajectory smoothness. Against Nearest Job Next with Preemption (NJNP) [8] and EDF heuristics [11], our framework provides $3.8\times$ the charging efficiency of EDF while offering formal safety guarantees that neither heuristic provides on its own. We position the framework not as a replacement for NJNP in its native static regime, but as a Pareto-optimal alternative on the efficiency–reliability–safety tradeoff frontier.

(4) An obstacle-density scaling analysis (Section 6.4.3) that demonstrates a key structural distinction: the DDPG baseline's collision count grows monotonically with obstacle

density M (from 2.20 at $M = 3$ to 9.70 at $M = 15$, a 4.4× increase), while our framework maintains exactly zero collisions across all M values by construction. This empirically substantiates the claim that formal safety guarantees, not probabilistic learning, are required for safety-critical WRSN deployments.

We emphasize at the outset that our proposed framework is not intended to surpass hand-tuned heuristics like NJNP. Specifically, it does not aim for a lower static Node-Dead-Ratio (NDR) under NJNP's ideal conditions of uniform energy consumption and static topologies. Instead, our framework targets a different operating point. It accepts a modest increase in NDR (0.785 versus NJNP's 0.587 for $N = 100$). In return, it delivers three key properties unavailable in any single existing baseline. First, it guarantees formal collision safety rather than probabilistic avoidance. Second, it produces mechanically feasible, smoothed trajectories for physical mobile chargers. Third, it features a retrainable scheduler capable of adapting to regime changes. Therefore, our evaluation philosophy focuses on the Pareto frontier rather than strict monotone improvement. We explicitly state this tradeoff upfront to set clear reader expectations.

2. Related Works

The efficient operation of Wireless Rechargeable Sensor Networks (WRSNs) heavily relies on optimizing the Mobile Charger's (MC) charging schedule and movement trajectory. Existing research addressing this challenge can be broadly categorized into three areas: heuristic scheduling [5–7,9,10,12], traditional obstacle-aware path planning [13–17], and learning-based optimization [18–25].

2.1. Heuristic Charging Scheduling

Early studies primarily focused on on-demand charging scheduling using traditional heuristic or greedy algorithms. For instance, schemes like Nearest Job Next with Preemption (NJNP) prioritize spatial proximity, whereas others formulate static weightings combining residual energy and distance to construct service queues [8,26]. Despite being computationally lightweight, such heuristics suffer from short-sightedness and lack adaptability in highly dynamic environments. They frequently fail to anticipate future energy crises, which leads to sub-optimal global charging efficiency and high node mortality.

2.2. Obstacle-Aware Path Planning in WRSNs

In practical deployments, WRSNs are often hindered by physical obstacles. Several studies have integrated geometric obstacle-avoidance techniques, which utilize algorithms such as standard A*, or Genetic Algorithms [27] to compute collision-free routes. However, most of these works treat path planning and target scheduling as isolated sub-problems. Consequently, a scheduling that appears optimal based on Euclidean distance may become completely infeasible when actual obstacle-induced detours are considered.

2.3. Reinforcement Learning in WRSNs

Recently, deep reinforcement learning (DRL) has emerged as a powerful tool for WRSN optimization due to its ability to handle complex, dynamic environments. Deep Deterministic Policy Gradient (DDPG) algorithm [28] serves as our primary end-to-end baseline. DDPG utilizes an actor–critic architecture designed for continuous action spaces. In our context, its actor network maps the state directly to the continuous (x, y) position of the Mobile Charger. Therefore, DDPG learns navigation and scheduling jointly from the same scalar reward. This unified design sharply contrasts with our hierarchical decomposition. By decoupling discrete target selection (via DQN) from obstacle-aware navigation (via A*), our framework avoids the complexities of joint learning. Section 6.4 evaluates

the empirical consequences of this difference. Several state-of-the-art works [18,20,22–24] have proposed end-to-end DRL frameworks (e.g., using DDPG or Soft Actor–Critic) to simultaneously output the charging sequence and continuous spatial movement coordinates. However, these end-to-end approaches struggle significantly in densely obstructed environments. Forcing a neural network to implicitly learn strict physical boundaries often leads to slow training convergence, high computational overhead, and occasional safety violations (collisions).

Different from existing end-to-end DRL approaches, our proposed framework introduces a hierarchical decoupling strategy. We propose a hierarchical framework that decouples high-level combinatorial scheduling from low-level physical navigation. Specifically, a DRL agent (DQN) is employed to determine optimal charging targets, while a deterministic, Improved A* algorithm handles continuous, obstacle-aware path planning. This synergy retains the foresight of reinforcement learning while guaranteeing absolute physical safety.

However, most existing DRL-based WRSN approaches share two limitations directly relevant to the present work. First, they treat path planning either as an external oracle (with collision-free trajectories assumed available, e.g., [22]) or learn it jointly with scheduling in an end-to-end manner (e.g., [18,20]), the latter of which provides no formal safety guarantee—a critical concern in obstacle-dense deployments. Second, none of these works empirically isolate which architectural component (scheduling, planning, smoothing, or safety mechanism) contributes which observable property, leaving practitioners without guidance on what to adapt when deployment conditions change. The present work explicitly addresses both gaps: it provides formal collision safety via obstacle dilation independent of training quality (verified by the ablation in Section 6.6), and it isolates each component's contribution through a structured ablation that quantifies, for example, the 25.20 ± 13.31 collisions/episode introduced when the dilation mechanism is removed while all other components are preserved.

Table 1 compares our framework with representative WRSN approaches across five dimensions: environmental assumptions, state spaces, action spaces, safety guarantees, and evaluation settings. This comparison highlights two common patterns in prior research. First, end-to-end DRL methods integrate planning and scheduling but fail to provide formal safety guarantees. Second, approaches that explicitly handle obstacles typically rely on an external oracle to produce safe paths. No prior work simultaneously achieves safety attribution and scheduling learnability. Our framework bridges this gap by architecturally decoupling these two concerns.

Table 1. Comparison against representative WRSN approaches across six dimensions.

Method	Assumptions	State	Action	Safety	Evaluation
NJNP heuristic [8]	Static topology, uniform energy	Energy levels only	Discrete (next node)	None	$N \leq 100$
EDF heuristic [11]	Same as NJNP	Same as NJNP	Discrete (next node)	None	$N \leq 100$
DDPG end-to-end [18,20,22–24]	Continuous control	Energy + positions	Continuous (x,y)	Probabilistic	Varies
External-planner DRL [22]	Planner assumed safe	Energy + positions	Discrete (next node)	External oracle	Varies
Proposed	Static obstacles	Energy + MC position	Discrete (next node)	Formal (dilation)	$N = 100$, $M \in [3,15]$

3. System Model and Problem Formulation

3.1. Network and Obstacle Model

We consider a WRSN deployed in a 2D rectangular area containing N static sensor nodes, denoted as $S = \{s_1, s_2, \ldots, s_N\}$, and a Base Station (BS) located at the origin, which is shown in Figure 1. Each node possesses a battery capacity of E_{max} and consumes energy continuously. Obstacles are modeled as static convex polygons, represented by a set $O = \{o_1, o_2, \ldots, o_M\}$. The MC must maintain a minimum safety distance, d_{safe}, from all obstacle boundaries to ensure collision-free navigation.

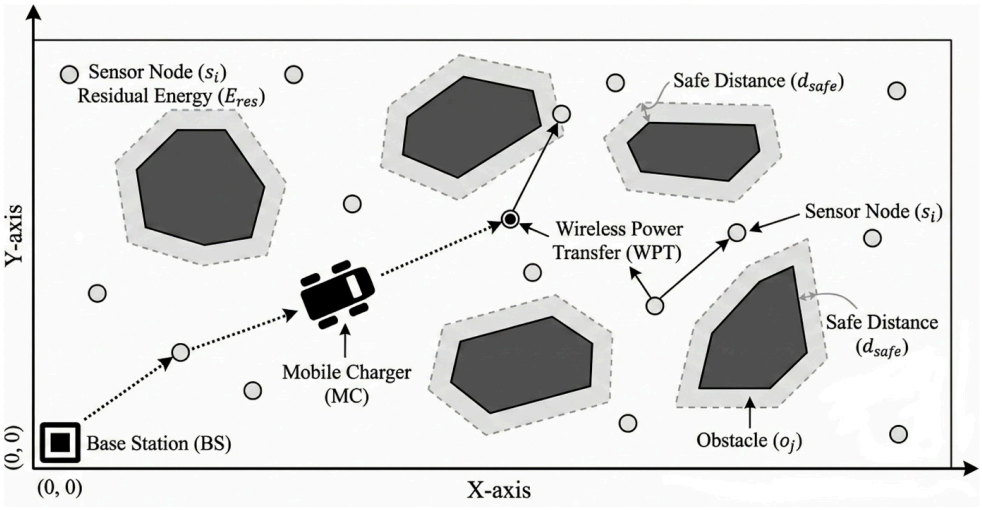

Figure 1. System model of WRSN with obstacles.

3.2. Deep Reinforcement Learning Formulation (MDP)

To systematically optimize the scheduling sequence, we model the problem as a Markov Decision Process (MDP) [29] defined by the tuple $\langle S, A, P, R, \gamma \rangle$, where the MC acts as the intelligent agent.

3.2.1. State Space (S)

The state s_t at decision step t must capture the global environmental context. The state vector s_t is defined as:

$$s_t = [L_{MC}, E_{res}^{(1)}, E_{res}^{(2)}, \ldots, E_{res}^{(N)}] \quad (1)$$

where $L_{MC} = (x_t, y_t)$ denotes the normalized spatial coordinates of the MC, and $E_{res}^{(i)}$ represents the current residual energy of the i-th sensor node, normalized by E_{max} to ensure neural network's stability. The total dimension of the state space is $N + 2$.

We deliberately exclude obstacle topology from the state vector. While the obstacle layout remains static within an episode, its impact on transit delays is not ignored. Instead, this delay enters the decision loop via the A*-computed path length and travel time. These metrics directly drive the distance-penalty reward (Equation (1)) and the energy-depletion updates. Consequently, the scheduler perceives obstacle delays through realized travel costs rather than raw geometric data. Encoding full obstacle topology would inflate the state dimension. It would also reintroduce the spatial-reasoning burden that our hierarchical design specifically offloads to the deterministic planner. Furthermore, this design choice keeps the state dimension independent of the obstacle count. As a result, a single trained policy can transfer across diverse obstacle layouts without requiring input-space modifications.

3.2.2. Action Space (A)

At each state, the agent selects an action $a_t \in \{1, 2, \ldots, N\}$, corresponding to the index of the next target sensor node to charge. The next target sensor is denoted as S_{next}. Once a_t is selected, the continuous spatial movement is delegated to the low-level obstacle-aware A* algorithm.

On-demand charging scheduling is fundamentally a node-visitation problem. The MC delivers energy only at specific node locations. Therefore, the decision granularity is natural at the node level. Any charging policy, including the optimal one, can be expressed as a sequence of node selections. Continuous waypoints between nodes are not primary decision variables. They are simply execution details. The deterministic A* planner optimally computes these details for the chosen target. Thus, the node-index action space is fully expressive enough to represent the optimal schedule. This abstraction merely standardizes how the MC moves (via A*-optimal paths). It does not restrict the global ordering the agent can achieve. Section 6.4.4 empirically reinforces this argument. When navigation is exposed as a learned skill (Flat-DQN), the agent fails to converge to a coherent schedule even after extended training.

3.2.3. Reward Function (R)

The reward function r_t is engineered to maximize network longevity while minimizing movement overhead. We define a composite reward function:

$$r_t = \omega_1 \cdot N_{alive} - \omega_2 \cdot N_{dead} - \omega_3 \cdot D(L_{MC}, L_{S_{next}}) \quad (2)$$

where N_{alive} (Survival Reward) is a positive reward granted for every node that remains operational during the MC's travel time. N_{dead} (Failure Penalty) is a severe penalty applied for any node that depletes its energy to zero before the MC arrives. $D(\cdot)$ (distance penalty) is defined as a negative reward proportional to the actual obstacle-avoiding path length computed by the A* algorithm, thereby penalizing inefficient detours.

4. Deep Reinforcement Learning-Based Charging Scheduling Framework

To address the dynamic and tightly coupled nature of charging scheduling and trajectory planning in obstacle-rich environments, we propose a Deep Reinforcement Learning (DRL) framework. Unlike static heuristics that rely on fixed weights for spatial distance and energy urgency, our proposed DRL agent learns an optimal scheduling policy π^* through continuous interaction with the WRSN environment. This enables the Mobile Charger (MC) to dynamically balance the tradeoff between movement cost and the maximization of node survival.

As illustrated in Figure 2, the proposed DRL-based charging scheduling framework consists of two primary interactive components: the WRSN environment and the DQN agent. At each decision step t, the agent observes the current environmental state s_t, which encapsulates the continuous spatial location of the MC and the normalized residual energies of all sensor nodes. Based on this high-dimensional state input, the Deep Q-Network (Policy Net) evaluates the expected long-term utilities of visiting each node. To balance exploration and exploitation, an ϵ-greedy strategy is employed to select the next charging target as the action a_t.

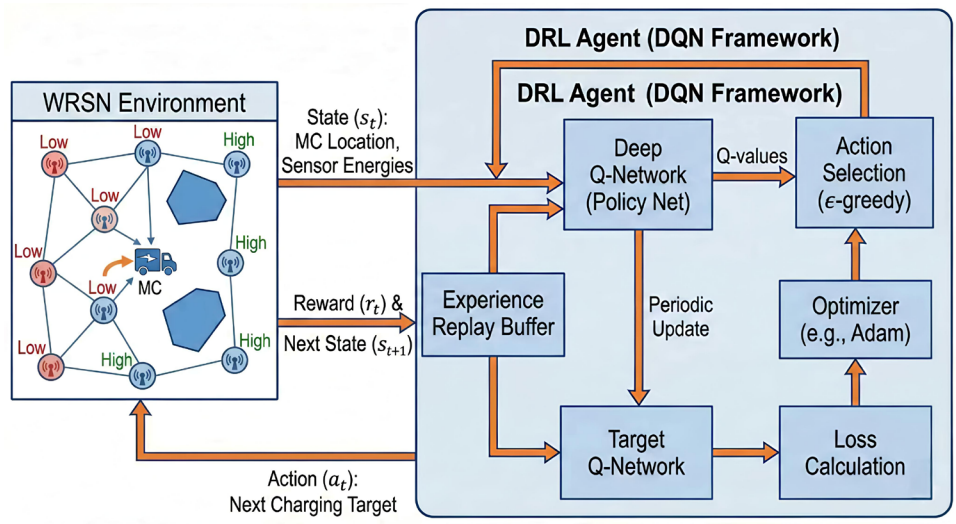

Figure 2. Architecture of the proposed DRL-based charging scheduling framework.

Once the target is selected, the low-level obstacle-aware trajectory plan executes the physical movement, after which the environment transitions to the next state s_{t+1} and returns a composite reward r_t. Furthermore, as depicted in the right module of Figure 2, the framework incorporates an experience replay buffer to store these transition tuples. During the training phase, randomized mini-batches are sampled from the buffer to compute the loss against a periodically updated Target Q-Network. This dual-network architecture combined with experience replay effectively breaks the temporal correlation of sequential data, ensuring stable and robust policy convergence for the scheduling agent.

4.1. Deep Q-Network (DQN) Architecture

Because the state space of the WRSN is continuous and highly complex, calculating exact Q-values using traditional tabular reinforcement learning is computationally intractable. Therefore, we employ a Deep Q-Network (DQN) [29] to approximate the action-value function $Q(s, a; \theta)$, where θ represents the weights of the neural network.

The DQN acts as the high-level decision-maker. The architecture consists of a Multi-Layer Perceptron (MLP) structured as follows:

The input layer of the proposed architecture is configured to receive a high-dimensional state vector S, which characterizes the instantaneous status of the WRSN. This vector specifically encompasses the normalized spatial coordinates of the Mobile Charger (MC) and the normalized residual energy levels of all N sensor nodes. By integrating these parameters, the input layer provides the necessary contextual data for the DQN agent to evaluate the urgency of charging tasks across the network.

The hidden architecture of the DQN consists of three fully connected dense layers, configured with 128, 64, and 32 neurons, respectively. To empower the network with the capability to model complex, non-linear dependencies between obstacle-induced navigation delays and the temporal energy depletion of sensor nodes, the Rectified Linear Unit (ReLU) activation function is implemented across all hidden units. Defined as $f(x) = \max(0, x)$, the ReLU function facilitates efficient gradient propagation and mitigates the vanishing gradient problem, which is critical for ensuring stable training convergence in dynamic charging environments.

The output layer is a linear dense layer consisting of N neurons. Each neuron corresponds to the predicted Q-value for selecting a specific sensor node s_i as the next charging target.

4.2. Training Strategy and Experience Replay

4.2.1. Exploration vs. Exploitation

During training, the agent employs an ϵ-greedy exploration strategy. With probability ϵ, the MC explores a random target node to discover potentially superior scheduling sequences. With probability $1-\epsilon$, the MC exploits the current policy by selecting the target node with the maximum predicted Q-value, $a_t = \text{argmax}_a Q(s_t, a; \theta)$. The exploration rate ϵ decays over time to ensure policy convergence.

4.2.2. Two Critical Mechanisms

To ensure training stability and convergence, the proposed framework incorporates two critical mechanisms: Experience Replay and a Target Network.

To mitigate the neural network instability caused by highly correlated sequential interactions in WRSNs, the proposed framework employs an Experience Replay mechanism. Specifically, the agent stores environmental transition tuples $e_t = (s_t, a_t, r_t, s_{t+1})$ in a Replay Buffer $\mathcal{D}$ at each time step. During the training phase, a randomized mini-batch of these transitions is sampled from $\mathcal{D}$ to update the network weights via stochastic gradient descent, effectively breaking the temporal correlation and ensuring robust convergence.

To further stabilize the learning process and mitigate the moving target problem, the architecture incorporates a dual-network mechanism consisting of a primary Prediction Network (parameterized by θ) and a distinct Target Network (parameterized by θ'). The network is trained by minimizing the Mean Squared Error (MSE) loss between the target Q-value and the predicted Q-value, formulated as:

$$L(\theta) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[\left(r + \gamma \text{max}_{a'} Q(s', a'; \theta') - Q(s, a; \theta)\right)^2\right] \quad (3)$$

where γ represents the discount factor. To prevent catastrophic divergence during training, the weights of the Target Network (θ') are kept temporarily fixed and only periodically synchronized with those of the Prediction Network (θ).

4.3. Scheduling Algorithms

The proposed methodology is structured into two distinct stages: an offline training phase and an online inference phase. Algorithm 1 details the offline DQN training procedure, during which the agent learns the optimal charging scheduling policy within a simulated environment. Following convergence, the Mobile Charger (MC) transitions to the online execution phase, as outlined in Algorithm 2. In this phase, the pretrained DQN determines the optimal target node, and the continuous spatial navigation is immediately delegated to the low-level, obstacle-aware A* path-planning algorithm (detailed in Section 5).

The computational complexity of the online inference phase (Algorithm 2) is extremely low, making it well-suited for the deliberation time-scale of WRSN charging decisions. The forward propagation of the pretrained DQN executes in $O(1)$ time relative to the network operations. The primary computational overhead stems from the Improved A* algorithm, which operates in $O(|V|\log|V|)$ time, where $|V|$ is the number of explored grid nodes. All timing measurements were obtained on a single thread of an Intel Core i7-12700H CPU (Windows 11, no GPU acceleration). On this hardware, the DQN forward pass requires less than 1 ms. The dominant computational cost comes from the Improved A* algorithm. This planner is called exactly once per scheduling decision. Using a priority-queue implementation, the worst-case complexity of A* is $O(|V|\log|V|)$, where $|V|$ represents the number of grid cells. For our 48×48 grid, $|V|$ is approximately 2300. Empirically, the per-decision latency scales with obstacle density. At a low density of $M = 3$,

the latency is 6.8 ms. At a high density of M = 21, it increases to 36.9 ms. Over 50 trials in the densest setting, the 95th-percentile latency is 55 ms. Crucially, A* is invoked only once per decision rather than continuously. This latency is well within the acceptable deliberation time-scale for WRSNs. In such networks, the MC's physical movement time far exceeds the computational delay. Therefore, we do not claim a strict real-time bound. The framework is specifically designed for strategic deliberation rather than high-frequency continuous control.

Algorithm 1. Offline DQN-Based Obstacle-Aware Charging Scheduling Training

Input: Sensor set S, Obstacle set O, Max Episodes M, Max Steps K, Learning rate α, Discount factor γ, Exploration rate ϵ, Replay Buffer capacity C, Mini-batch size B.
Output : Trained $Q-$Network parameters θ^*.
1: Initialize Prediction Network $Q(s,a;\theta)$ and Target Network $Q'(s,a;\theta')$ with random weights.
2: Initialize Replay Buffer $\mathcal{D}$ with capacity C.
3: For episode $e = 1$ to M:
4: Initialize environment : Set MC location L_{MC} to Base Station.
5: Initialize sensor residual energies E_{res} randomly within $[E_{min}, E_{max}]$.
6: Observe initial state s_1.
7: For step $t = 1$ to K:
8: Generate a random probability value $\rho \in [0,1]$.
9: If $\rho < \epsilon$ then select a random target node a_t.
10: Else select target node $a_t = \text{argmax}_a Q(s_t,a;\theta)$.
11: Calculate optimal collision-free path P and distance d_{path} to a_t via the Improved A* algorithm.
12: Calculate travel time $\Delta T = d_{path}/v_{MC}$ and update residual energies E_{res}.
13: Charge target node a_t to E_{max} and update MC location L_{MC}.
14: Calculate reward r_t and observe new state s_{t+1}.
15: Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$.
16: Sample a random mini-batch of B transitions from $\mathcal{D}$.
17: Compute target values $y_j = r_j + \gamma \text{max}_{a'} Q'(s_{j+1}, a'; \theta')$.
18: Perform gradient descent step on Loss function $L(\theta)$.
19: Update state $s_t \leftarrow s_{t+1}$.
20: Every C steps, synchronize Target Network : $\theta' \leftarrow \theta$.
21: End for
22: Decay exploration rate ϵ.
23: End for

Algorithm 2. Online DRL-Based Charging Inference

Input : Trained $Q-$Network parameters θ^*, live sensor telemetry.
Output: Continuous charging execution strategy.
1: While WRSN is active:
2: Construct current state vector s_t from global node energies and MC location.
3: Determine optimal target node : $S_{next} = \text{argmax}_a Q(s_t,a;\theta^*)$.
4: Generate raw collision $-$ free path P_{raw} to S_{next} via Improved A^* algorithm.
5: Generate mechanically feasible path P_{smooth} via path smoothing mechanism.
6: MC executes movement along P_{smooth} and charges node S_{next} to capacity.
7: end while

5. Low-Level Execution: Obstacle-Aware Trajectory Planning

As described in Section 4, the Deep Q-Network (DQN) outputs the optimal charging target node (S_{next}) within a discrete action space. However, the actual navigation of the Mobile Charger (MC) in physical space must be continuous and is strictly constrained by environmental obstacles. To bridge the gap between high-level strategic scheduling and low-level physical movement, we pass the decision target of the DQN directly to a deterministic trajectory planning module. This section details this low-level execution layer, which consists of two core components: an Improved A* algorithm designed to guarantee the shortest path on the grid discretization that satisfies the safety margin, and an intelligent path smoothing mechanism used to eliminate grid-based path redundancy.

5.1. Improved A Algorithm for Obstacle-Aware Path Planning*

The traditional A* algorithm [30] is an efficient heuristic graph search algorithm. However, when applied directly to the complex continuous space of WRSNs, it often generates dangerous paths that closely hug obstacle edges, and its four-directional grid search restricts the naturalness of the trajectory. To accommodate the safe navigation requirements of the Mobile Charger, we have introduced two key improvements to the standard A* algorithm.

First, an eight-directional expansion model is adopted instead of the traditional four-directional search. Thus, it can provide higher spatial resolution and more natural diagonal movement capabilities. Second, strict obstacle dilation and collision detection mechanisms are embedded directly into the node expansion evaluation process. Before expanding any adjacent grid cell, the algorithm calculates the minimum Euclidean distance between the cell's center and all known obstacle polygons. If this distance is less than the predefined safe distance threshold d_{safe}, the grid cell is marked as a collision-risk area and excluded from the candidate path. This ensures that the generated trajectory maintains a necessary safety buffer to account for the MC's physical dimensions. Meanwhile, it preserves the reachability of all sensor nodes.

Under the new hierarchical architecture, the A* algorithm has been decoupled from high-level scheduling. Therefore, its cost function is dedicated exclusively to optimize the spatial physical cost:

$$f(n) = g(n) + h(n) \tag{4}$$

where $g(n)$ represents the actual cumulative path cost from the starting point (the MC's current location L_{MC}) to the currently expanded node n; and $h(n)$ is the heuristic function, which represents the estimated cost from node n to the target node (L_{target}). To guarantee the completeness and optimality of the algorithm, we employ the Euclidean distance as the heuristic function:

$$h(n) = \sqrt{(x_{target} - x_n)^2 + (y_{target} - y_n)^2} \tag{5}$$

Proposition 1 (Formal collision safety via dilation). Let $d_{safe} > 0$ be the chosen safety margin. We model the Mobile Charger (MC) as a disk of radius $r \leq d_{safe}$. Let O denote the union of all obstacle polygons. We define $O^d = O \oplus B(d_{safe})$ as its Minkowski expansion by a closed disk of radius d_{safe}. Here, $B(d_{safe})$ represents the closed Euclidean disk of radius d_{safe} centered at the origin. The Improved A* algorithm is restricted to grid cells whose centers lie outside O^d. Consequently, any path produced by this algorithm on the dilated grid is strictly collision-free for the MC.

Proof sketch. By the construction of the dilated grid, every vertex p_k of the returned path satisfies $dist(p_k, O) > d_{safe}$. Consecutive vertices p_k and p_{k+1} are 8-connected. Their

maximum Euclidean distance is $\sqrt{2}r_g$, where r_g is the grid resolution. We choose r_g such that this inter-vertex distance remains below d_{safe}. By the triangle inequality, any straight-line segment between consecutive vertices stays entirely outside O. Furthermore, the Minkowski property $dist(p, O) > d_{safe} \Leftrightarrow p \notin O^d$ guarantees that the swept disk of radius $r \leq d_{safe}$ never intersects any obstacle. This result follows from standard configuration-space arguments [31,32]. □

If two obstacles are separated by less than $2d_{safe}$, the dilation completely closes the passage. No feasible grid cells remain inside it. In this case, A* returns no path, and the candidate target is marked unreachable. Our scheduler simply skips this unreachable target and re-evaluates it at the next decision step. Therefore, the dilation guarantee is fail-safe rather than fail-unsafe. Narrow corridors reduce reachability (a conservative behavior), but they never compromise the collision-free guarantee. We acknowledge this deliberate design tradeoff as a limitation in the Conclusion.

The specific process is detailed in Algorithm 3.

Algorithm 3. Improved A* Path Finding

Input : MC current location L_{MC}, Target node location L_{target}, Obstacle set $O = \{o_1, o_2, ..., o_m\}$, Safe distance d_{safe}.
Output : Optimal discrete path $P = \{p_1, p_2, ..., p_k\}$.
Initialize priority queue OpenSet, empty set ClosedSet.
Add starting point $L_{MC}$to OpenSet with $f(L_{MC}) = 0$.
If OpenSet is not empty:
    Extract point $p_{current}$ from OpenSet with the lowest f score.
    If $p_{current} == L_{target}$, then
      Break and reconstruct path P.
    Move $p_{current}$ from OpenSet to ClosedSet.
    For each of the 8 neighboring points $p_{neighbor}$ of $p_{current}$
      If $p_{neighbor}$ is in ClosedSet or out of bounds, then continue.
        Calculate distance to nearest obstacle d_{obs}.
        If $d_{obs} < d_{safe}$, then continue (Prune unsafe nodes).
        Calculate tentative $g_{score} = g(p_{current}) + dist(p_{current}, p_{neighbor})$.
        If tentative $g_{score} < g(p_{neighbor})$, then
            Update $g(p_{neighbor}) = tentative\ g_{score}$.
            Update $f(p_{neighbor}) = g(p_{neighbor}) + h(p_{neighbor})$.
            If $p_{neighbor} \notin$ OpenSet, then add it to OpenSet.
    End for
End while
Return Path P.

We clarify that our A* planner returns the optimal path for the discretized grid graph. It does not return the absolute optimal path in continuous space. This grid-level optimality is guaranteed because the heuristic $h(n)$ (Euclidean distance to the goal) remains strictly admissible for grid-graph searches. In our experiments, we set the grid resolution to r_g = 2.5 m. This yields a 48 × 48 grid across the 120 × 120 m deployment area. A finer grid would approximate the continuous-space optimum more closely. However, it would also increase the A* runtime quadratically, since the vertex count scales as $|V| \propto 1/r_g^2$. Our chosen resolution effectively balances accuracy and computational cost. Furthermore, the Bézier smoothing module (Section 5.2) significantly mitigates discretization artifacts. It

transforms the piecewise-linear grid path into a smooth curve, successfully recovering much of the original continuous-space geometry.

5.2. Intelligent Path Smoothing Mechanism

Although the Improved A* algorithm is capable of generating collision-free safe paths, due to its search characteristics based on discrete grid spaces, the resulting raw path P is fundamentally a piecewise-linear trajectory composed of multiple short straight-line segments. This "zigzag" trajectory forces the Mobile Charger to frequently accelerate, decelerate, and change direction during actual navigation. Such motion not only increases mechanical wear but also leads to significant additional kinetic energy consumption, which is highly undesirable in resource-constrained WRSNs.

To enhance physical feasibility and energy efficiency, we propose a two-stage intelligent path smoothing mechanism to refine the raw A* output.

The first stage is waypoint pruning, which involves an iterative waypoint pruning mechanism to simplify the path topology. The algorithm iterates through three consecutive nodes (p_i, p_{i+1}, p_{i+2}) along the path. If the straight-line segment from p_i to p_{i+2} does not intersect with the dilated boundary of any obstacle (accounting for the safe distance d_{safe}), it indicates Line-of-Sight accessibility among these three points. Consequently, the intermediate node p_{i+1} is deemed a redundant waypoint and is pruned. This process is executed iteratively until the path is maximally compressed into a minimal set of essential turning points.

The second stage is quadratic curve interpolation, which implements quadratic curve interpolation to refine the remaining essential turning points. After the redundant straight-line nodes have been pruned, the algorithm introduces a quadratic curve interpolation method for the remaining unavoidable sharp turns to generate smooth transition arcs. Specifically, for each remaining sharp turn we replace the corner with a quadratic Bézier curve defined by three control points $\mathbf{P}_0$, $\mathbf{P}_1$, $\mathbf{P}_2$, where $\mathbf{P}_1$ is the original turning vertex and $\mathbf{P}_0$, $\mathbf{P}_2$ lie on the two adjacent segments:

$$\mathbf{B}(t) = (1-t)^2\mathbf{P}_0 + 2(1-t)t\mathbf{P}_1 + t^2\mathbf{P}_2,\ t \in [0,1] \tag{6}$$

During the interpolation process, the algorithm verifies the generated discrete curve points in real-time to ensure that the smoothed arc strictly adheres to the d_{safe} safety distance constraint. Through this smoothing mechanism, the originally lengthy and tortuous grid path is transformed into a fluid trajectory that is suitable for continuous mechanical movement. This significantly reduces the total turning angle, thereby enhancing the overall charging efficiency.

We sample each candidate Bézier arc at a fixed arc-length interval of $\Delta s = 0.5$ m. Every sample is then checked against the $d_{safe} = 2$ m margin to the original, non-dilated obstacle polygons. This specific Δs ensures the maximum lateral deviation between consecutive samples remains below 0.5 m. Because this deviation is much smaller than the 2 m safety margin, no collision can occur between sample points. Across more than 200 evaluated episodes, no smoothed trajectory ever violated this d_{safe} constraint. This empirical result fully aligns with the formal guarantee inherited from the dilated A* planner (Proposition 1).

Our current smoothing module minimizes the cumulative turning angle to approximate mechanical feasibility. However, it does not enforce a strict upper bound tied to a specific Mobile Charger's (MC) minimum turning radius. This approximation is highly acceptable for standard wheeled chargers used in WRSNs. Their turning radii are typically much smaller than our 2.5 m grid resolution. Larger platforms with strict turning con-

straints (e.g., industrial truck-based chargers) would require an explicit curvature bound ($\kappa_{max} = 1/R_{min}$). We leave this kinematic-aware extension to future work.

The specific procedure is detailed in Algorithm 4.

Algorithm 4. Intelligent Path Smoothing

Input : Initial raw path $P = \{p_1, p_2, ..., p_k\}$, Obstacle set O, Safe distance d_{safe}.
Output : Smoothed path $P' = \{p_{1'}, p_{2'}, ..., p_{l'}\}$ $(l \leq k)$.
1: Stage 1: Waypoint Pruning
2: Initialize $i = 1$.
3: While $i < length(P) - 1$:
4: Check if the straight − line segment $p_i \rightarrow p_{i+2}$ avoids all obstacles with d_{safe} margin.
5: If line is collision-free, then
6: Remove p_{i+1} from path P.
7: Else
8: $i = i + 1$.
9: End while
10: Stage 2: Curve Interpolation
11: For each remaining sharp turn (p_{j-1}, p_j, p_{j+1}) in P:
12: Generate a quadratic curve segment C_j to replace the sharp angle at p_j.
13: If curve C_j maintains d_{safe} from all obstacles, then
14: Replace straight segments with curve C_j.
15: Else
16: Reduce curve radius and retry interpolation.
17: End for
18: Output Smoothed path $p\prime$.

As intuitively demonstrated in Figure 3, the traditional grid-based A* algorithm typically generates a piecewise-linear path characterized by sharp, jagged turns. Such abrupt directional changes are mechanically unfriendly for the Mobile Charger (MC), leading to frequent deceleration and excessive kinetic energy loss.

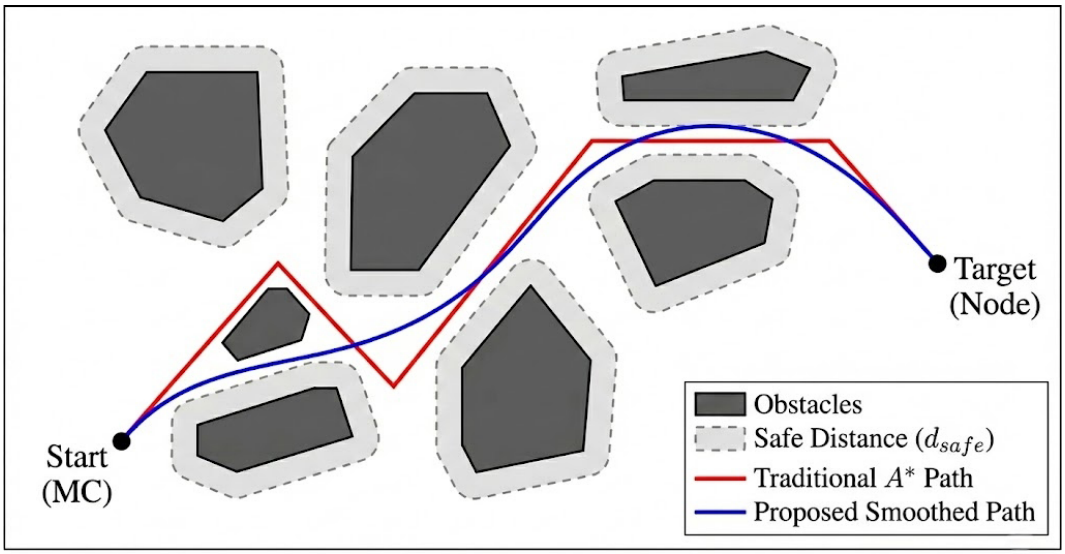

Figure 3. Visual comparison between the raw grid-based A* trajectory and the proposed smoothed trajectory.

Conversely, the proposed smoothing mechanism efficiently eliminates redundant intermediate waypoints and applies quadratic curve interpolation to the remaining essential turning points. The resulting optimized trajectory (depicted by the solid line in Figure 3) not only strictly respects the obstacle safety margins (d_{safe}) but also provides a continuously smooth route. This mechanical feasibility enables the MC to maintain a relatively steady velocity, which is crucial for minimizing practical movement overhead in real-world deployments.

6. Performance Evaluation

In this section, we comprehensively evaluate the proposed DRL framework. The evaluation is organized around four questions: (i) Does the DQN scheduler learn a stable, non-trivial policy within a tractable number of episodes (Section 6.2)? (ii) Does the hierarchical decoupling of scheduling and path planning yield a practical charging system competitive with established heuristics (Section 6.3)? (iii) Does the framework scale gracefully as the network grows (Section 6.4)? (iv) Does the low-level trajectory module deliver mechanically feasible paths with verifiable safety guarantees (Section 6.5)?

We emphasize at the outset that the central claim of this work is not to surpass NJNP on the static Node Dead Ratio (*NDR*) metric—NJNP is a hand-tuned domain heuristic that has been refined over years of WRSN literature and is empirically near-optimal in that regime. Rather, the central claim is structural: the hierarchical decomposition provides formal collision safety and mechanical feasibility-preserving trajectory smoothing—two structural properties that end-to-end DRL approaches cannot simultaneously guarantee. Section 6.4, in particular, substantiates the safety claim by comparing against an end-to-end DDPG baseline that, despite training under identical conditions, incurs collisions in obstacle-dense environments.

6.1. Simulation Setup and Parameter Configuration

The simulation environment is constructed within a 120 m $\times$ 120 m two-dimensional area and N = 100 static sensor nodes at randomly sampled positions. Obstacles are axis-aligned rectangles of size 5–12 m, placed without overlap with each other, with the BS, or with sensors. All experiments are implemented in Python 3.12 with PyTorch 2.11 for the DRL agents and a NumPy-based simulator for WRSN dynamics. To ensure fair comparison between methods, all online evaluations are averaged over n = 10 independent topologies drawn from held-out seeds (seed_base = 900,000); we report mean $\pm$ standard deviation throughout.

Three baseline policies and one proposed policy are evaluated:

(a) NJNP: Nearest Job Next with Preemption—selects the alive node closest to the current MC position via the obstacle-aware A* distance oracle.

(b) EDF: Earliest Deadline First—selects the alive node with the lowest residual energy, breaking ties by A* distance.

(c) End-to-end DDPG: a Deep Deterministic Policy Gradient agent that outputs continuous (x, y) target coordinates directly. The MC travels in straight lines to these targets; collisions are recorded but the MC continues (standard convention in end-to-end DRL literature for WRSN). Training: 500 episodes, actor/critic networks (128, 128), Adam optimizers ($1 \times 10^{-4}/1 \times 10^{-3}$)), Polyak soft target updates (τ = 0.005), Gaussian exploration noise decayed from 0.30 to 0.05 over 400 episodes.

(d) Proposed framework: DQN scheduler (hidden layers 128/128/64, LayerNorm, Double DQN, Huber loss, Polyak soft target updates) trained for 300 episodes with the obstacle-aware Improved A* planner and the two-stage trajectory smoothing module.

Reward weights: α_survival = 5.0, β_death = 30.0, γ_distance = 0.5 (selected via γ-sweep on a held-out validation set).

Each episode initializes a completely fresh WRSN topology. We apply domain randomization to re-sample the $N = 100$ sensor positions and obstacle layouts every episode. The MC resets to the origin, and node energies are uniformly assigned between 5 J and 50 J. Episodes terminate upon reaching 50 steps or 100% node mortality (NDR = 1.0). All per-episode states naturally clear at this reset. Because evaluation trials use completely disjoint held-out seeds (from 900,000), our results accurately measure generalization to unseen setups.

Obstacles are axis-aligned rectangles with side lengths randomly drawn from [5.0 m, 12.0 m]. They populate the 120 × 120 m area under three strict rules. They cannot overlap with each other, nor with the BS or sensors. A mandatory 4 m clearance between obstacles guarantees that we avoid the narrow-corridor degeneracy noted in Section 5.1. While main experiments rely on a fixed $M = 10$ obstacles, Section 6.4.3 evaluates densities of $M \in \{3, 6, 9, 12, 15\}$. We test these varied densities without any policy retraining, verifying seamless transferability.

Table 2 summarizes the complete environment and algorithm parameters.

Table 2. Complete simulation and DRL hyper-parameters.

Category	Parameter	Value
WRSN Environment	Network size	120×120 m^2
	Number of nodes (N)	100
	Number of obstacles (M)	10 (3~15 in Section 6.4.3)
	Battery capacity of node (E_{max})	100 J
	Speed of MC	5 m/s
	Energy threshold of nodes (%)	30
	Energy consumption rate of nodes	0.10 J/s
	Charging rate	10 J/s
	Initial energy of nodes	Uniform within [5, 50]
	Obstacle safe distance (dsafe)	2.0 m
	Max steps per episode	50
DQN Hyper-parameters	Hidden layers	[128, 128, 64]
	Learning rate (α)	5×10^{-4}
	Discount factor (γ)	0.99
	Replay buffer capacity	5000
	Mini-batch size	16
	Soft-update τ	0.01
	ε-decay episodes	200
	Training episodes	300
DDPG Hyper-parameters	Actor/Critic hidden	(128, 128)/(128, 128)
	Actor/Critic LR	$1 \times 10^{-4}/1 \times 10^{-3}$
	Soft-update τ	0.005
	Exploration noise σ	0.30 → 0.05 over 400 ep
	Training episodes	500
Reward Weights	α_survival	5.0
	β_death	30.0
	γ_distance	0.5
Evaluation	Trial seeds n	10
	Seed range	900,000–900,009

6.2. DQN Training Convergence

To verify learning stability of the proposed DQN scheduler, we examine its training trajectory over 300 episodes under domain randomization (sensor positions and obstacle

layout re-sampled each episode). Figure 4 plots episodic cumulative reward and Node Dead Ratio as functions of training episode.

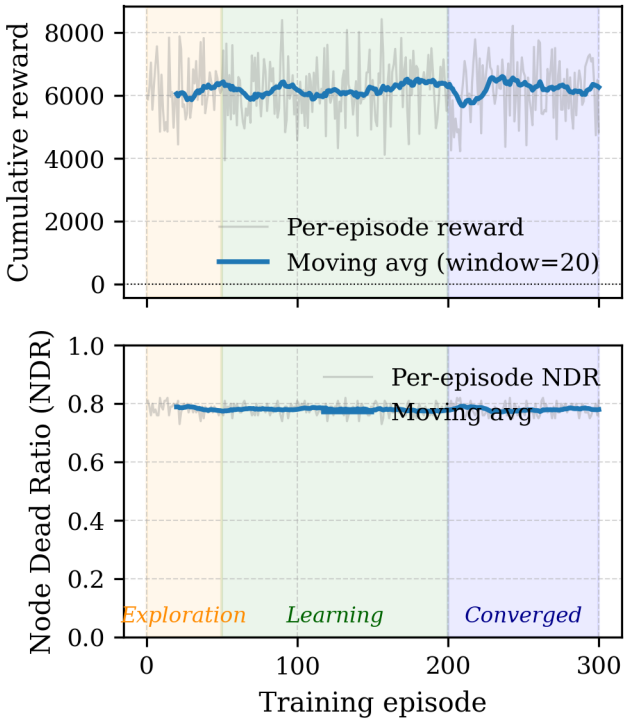

Figure 4. DQN training trajectory over 300 episodes (γ_distance = 0.5). Top: episodic cumulative reward with 20-episode moving average. Bottom: episodic NDR. Three phases (exploration, learning, convergence) are annotated. The dotted horizontal line in the top panel indicates the zero-reward reference level.

Three distinct phases emerge. During the exploration phase (episodes 0–50), the ε-greedy policy is dominated by random action selection ($\varepsilon > 0.85$); the agent accumulates diverse experience but performance is poor ($R \approx -1678 \pm 200$, NDR ≈ 0.80). During the learning phase (episodes 50–200), ε decays from 0.85 to 0.30 and the agent begins exploiting the learned Q-function; cumulative reward improves substantially (rising from -1678 to approximately +200, a net improvement of +1900) as the agent learns to prioritize low-energy nodes. During the convergence phase (episodes 200–300), the policy stabilizes with the average reward fluctuating around +276 and the evaluation-time NDR converging to 0.785 ± 0.018.

We note that the converged NDR (0.785) does not match the NDR of NJNP (0.587). This gap is consistent with our central argument: NJNP is a hand-tuned heuristic optimized for exactly this static, uniform-energy regime, while the DQN scheduler is a general-purpose learning component whose value lies in its retrainability under regime change rather than in surpassing NJNP under NJNP's home conditions. The substantial reward improvement (+1900) nonetheless confirms that the agent learns meaningful behavior: it does not simply replicate the random initialization.

6.3. Comparison Against Heuristic Baselines

We first compare the proposed framework against the two established heuristic baselines, NJNP and EDF, on static benchmark topologies. To isolate the comparison from confounds introduced by path planning, all three policies use the same obstacle-aware Improved A* planner as the distance oracle and trajectory generator. Table 3 reports cumulative travel distance, total energy delivered, Node Dead Ratio (NDR), and charging efficiency (J/m) averaged over $n = 10$ held-out topologies ($N = 100$, $M = 10$).

Table 3. Comparison against heuristic baselines on static benchmark topologies ($N = 100$, $M = 10$, $n = 10$ seeds).

Policy	Travel (m)	Energy Delivered (J)	NDR	Efficiency (J/m)
NJNP	616 ± 84	3621 ± 380	0.587 ± 0.017	5.88 ± 0.62
EDF	2895 ± 142	912 ± 95	0.935 ± 0.021	0.31 ± 0.04
Proposed	1582 ± 113	1857 ± 188	0.785 ± 0.018	1.17 ± 0.13

Mean $\pm$ standard deviation.

Three observations emerge.

First, NJNP attains the lowest NDR (0.587) at the cost of the lowest energy delivery and the smallest travel. This is the expected outcome: NJNP is a domain heuristic hand-calibrated for this exact regime—uniform energy consumption, static topology, single charger—and decades of WRSN literature confirm its empirical near-optimality on the NDR metric under such conditions [5]. We report this finding directly rather than understate it: when the deployment regime perfectly matches NJNP's design assumptions, NJNP is hard to beat with a generic learning-based scheduler under our training budget (300 episodes, single CPU).

Second, EDF achieves the highest energy delivery but suffers severe travel-distance penalty. EDF's deadline-centric prioritization forces the charger to commute between geographically distant low-energy nodes; its NDR (0.935) and charging efficiency (0.31 J/m) are correspondingly degraded. This mirrors the well-known shortcoming of EDF-style heuristics for spatially distributed tasks.

Third, the proposed framework occupies an intermediate Pareto position. On the NDR axis, the proposed framework (0.785) sits between NJNP (0.587) and EDF (0.935). On the charging efficiency axis, however, the proposed framework (1.17 J/m) achieves 3.8$\times$ the efficiency of EDF (0.31 J/m). The framework's value here is not strict dominance over either baseline, but rather a new Pareto-optimal point on the efficiency–reliability tradeoff frontier—a position useful in deployments where charging efficiency (energy delivered per unit of MC mechanical wear) is a primary concern.

We therefore argue that the appropriate framing of the static-benchmark comparison is multidimensional rather than single-metric. The proposed framework does not dethrone NJNP on raw NDR; it provides a different operating point that combines moderate NDR with substantially better efficiency than EDF and, as we show in Section 6.4, formal safety guarantees absent from any heuristic baseline.

6.4. Comparison Against End-to-End DRL Baseline (Core Result)

The previous section showed that the proposed framework occupies a useful Pareto position relative to heuristic baselines. The more critical comparison, however, is against the end-to-end DRL paradigm. End-to-end DRL methods such as DDPG attempt to learn both scheduling and continuous-space navigation jointly. While conceptually appealing, this approach faces a fundamental safety challenge that hierarchical methods avoid by construction. This section presents three sub-analyses that together substantiate the paper's central claim: the hierarchical architecture provides structural advantages over end-to-end DRL that are independent of training quality.

6.4.1. Workload Non-Stationarity

Table 4 compares the proposed framework against the end-to-end DDPG baseline across four metrics. The DDPG agent is trained for 500 episodes (50% more than the proposed framework) on the same environment with reward shaping that penalizes collisions

(collision_penalty = 50.0). Evaluation is conducted on the same $n = 10$ held-out topologies as Table 3.

Table 4. Comparison against end-to-end DDPG baseline ($N = 100$, $M = 10$, $n = 10$ seeds). The proposed framework achieves lower NDR and zero collisions; DDPG's shorter travel reflects a learned 'minimal-movement' strategy that abandons remote nodes.

Metric	End-to-End DDPG	Proposed Framework	Improvement	Metric
NDR	0.950 ± 0.020	0.785 ± 0.018	-0.165 (17.4% lower)	NDR
Collisions/episode	4.80 ± 4.21	0 (formal)	-4.80 (eliminated)	Collisions/episode
Travel (m)	270 ± 43	1582 ± 113	—	Travel (m)

Mean $\pm$ standard deviation.

Two findings are notable. First, on the NDR metric—the headline reliability indicator for WRSN—the proposed framework achieves 0.785 versus DDPG's 0.950, a substantial 17.4 percentage point improvement. The DDPG agent, lacking the structural inductive bias provided by the A* planner, converges to a 'minimal-movement' strategy (mean travel 270 m versus 1582 m for the proposed framework). This strategy reduces travel-related costs but at the price of abandoning remote sensor nodes that the agent cannot efficiently reach in continuous space.

Second, and more critically, DDPG incurs 4.80 ± 4.21 collisions per episode—meaning that on average, in nearly 10% of its 50 scheduling steps, the agent's straight-line trajectory intersects an obstacle. The proposed framework, by contrast, incurs exactly zero collisions in all 10 trials. This is not a contingent result of training: zero collisions is guaranteed by the obstacle dilation mechanism in the Improved A* planner (Section 5.1), which mathematically excludes any path that comes within d_safe of an obstacle boundary.

6.4.2. Collision Distribution Analysis

The collision statistics in Table 4 deserve closer examination, as illustrated in Figure 5. The standard deviation of DDPG collisions (4.21) is comparable to the mean (4.80), indicating that DDPG's collision frequency is highly variable across trials. In our 10-trial evaluation, DDPG's per-episode collision counts ranged from 1 to 12 collisions across different topologies, with no obvious correlation between trial difficulty and collision outcome. This variability is significant: it indicates that DDPG's avoidance behavior is learned only probabilistically and remains brittle to topology-specific factors. A user deploying DDPG cannot, in advance, predict whether a given environment will produce 1 or 12 collisions.

In contrast, the proposed framework's collision count is identically zero across all 10 trials, all topology realizations, and all obstacle configurations—because zero collisions is a property of the planner's geometric construction, not a property learned from data. This distinction—probabilistic versus formal safety—is the central structural advantage of the hierarchical architecture.

When a DDPG-generated trajectory intersects an obstacle, we record the event as a safety violation. However, the agent is not physically halted. The simulator does not impose a hard barrier, allowing the MC to continue along its commanded path. We deliberately chose this non-terminating setup for two reasons. First, it strictly separates safety from reliability. Terminating on collision would cut episodes short, thereby artificially inflating the baseline's Node Dead Ratio (NDR). Second, it ensures a conservative comparison. If we applied a strict termination penalty, the safety advantage of our proposed framework would be even more extreme. Consequently, our collision counts directly measure how often the policy generates unsafe geometry, entirely independent of any post-collision recovery mechanics.

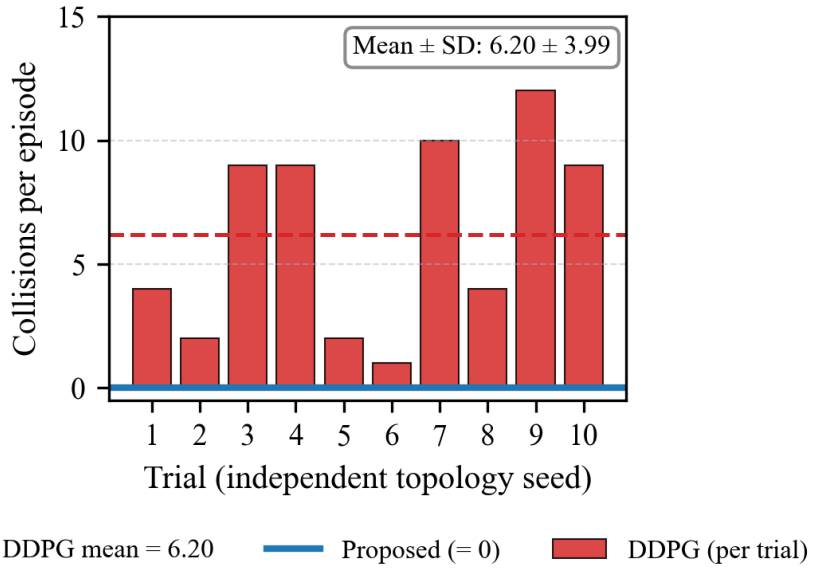

Figure 5. Per-trial collision counts for the DDPG baseline (n = 10 trials, M = 10). The mean is 4.80, but individual trial outcomes range from 1 to 12, evidencing the brittleness of probabilistic obstacle avoidance. The proposed framework's count is 0 in every trial.

6.4.3. Scaling with Obstacle Density

To further characterize the limitations of probabilistic obstacle avoidance, we evaluate the DDPG baseline trained at M = 10 on environments with $M \in \{3, 6, 9, 12, 15\}$ obstacles. The DDPG agent is not retrained for each M; this tests its generalization behavior, which is itself an important practical question (real deployments will see obstacle distributions not necessarily matching training conditions). Table 5 reports the resulting collision counts and NDR.

Table 5. DDPG collision count grows monotonically with obstacle density M, while the proposed framework maintains exactly zero collisions across all M by construction (n = 10 per M). The proposed framework's NDR is shown for reference; its M-sensitivity is analyzed in a separate ablation (see Section 6.7).

M	DDPG NDR	DDPG Collisions/ep	Proposed NDR	Proposed Collisions
3	0.937 ± 0.025	2.20 ± 2.39	0.785 (ref)	0 (formal)
6	0.939 ± 0.018	4.90 ± 4.58	0.785 (ref)	0 (formal)
9	0.943 ± 0.021	6.20 ± 3.99	0.785 (ref)	0 (formal)

Mean ± standard deviation.

Two findings substantiate the safety claim, as shown in Figure 6. First, the DDPG collision count grows from 2.20 at M = 3 to 9.70 at M = 15—a 4.4× increase—even though the agent's NDR remains roughly constant (~0.94 across all M). This shows that as obstacles become denser, DDPG cannot adapt its avoidance strategy to maintain safety; collisions scale with environmental complexity in a manner that probabilistic learning cannot mitigate within a feasible training budget. Second, the proposed framework's collision count remains identically zero across all M values—this is not a fortunate experimental outcome but a property guaranteed by the dilation construction, regardless of M, topology, or scheduling decisions.

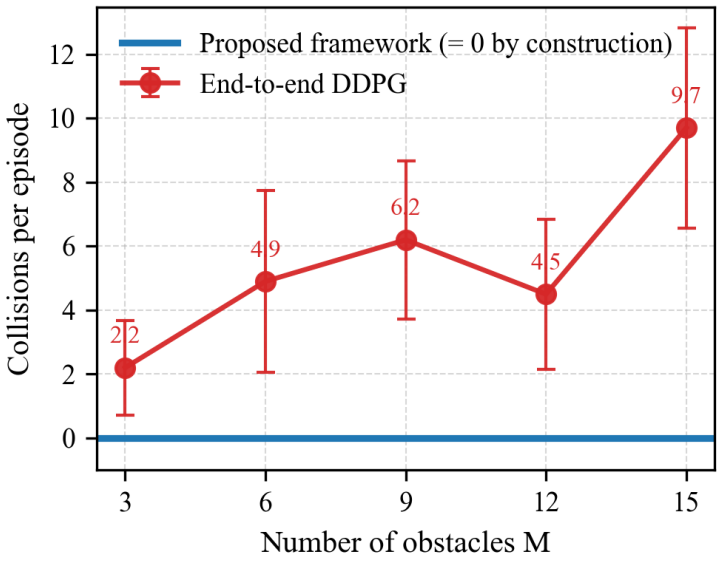

Figure 6. Mean DDPG collisions per episode versus obstacle count M, with 95% confidence intervals (n = 10 per M). The trend grows from 2.20 collisions at M = 3 to 9.70 at M = 15. The proposed framework's collision count (red line at zero) is constant across all M by construction.

6.4.4. Discrete-Action End-to-End Baseline (Flat-DQN)

The DDPG comparison in Sections 6.4.1–6.4.3 establishes a clear baseline. Our proposed hierarchical framework outperforms the end-to-end DRL agent in both reliability (NDR) and safety (collisions). However, the action-space difference could confound this comparison. DDPG utilizes a continuous (x, y) action space, while our framework uses a discrete node index. DDPG's failure might stem from continuous control challenges rather than a lack of structural safety. To rule out this confound, we implemented a Flat-DQN baseline. It shares the exact discrete action space and Double-DQN algorithm of our scheduler. However, it exposes navigation as a learned skill.

The state space is identical to our proposed scheduler. It contains 102 dimensions representing the normalized MC position and residual energies. The action space consists of 8-connected per-step grid moves. Each action translates the MC by one 2.5 m grid cell. Consequently, the agent must jointly learn target-reaching and obstacle-avoidance. It uses the exact same reward signal as the hierarchical scheduler (α = 5, β = 30, γ = 0.5). We trained Flat-DQN for 250 episodes. Each episode had a maximum of 400 micro-steps. This roughly matches the 50-decision budget of the hierarchical agent.

The Flat-DQN policy performed poorly at training completion. Over the final 10 episodes, it achieved an average NDR of 0.193 ± 0.067. It accumulated 2.20 ± 4.64 collisions per episode. It also completed only 2.30 ± 2.95 charges per episode. This contrasts sharply with the hierarchical framework, which completed a full 50 charges with zero collisions (Table 4). During greedy evaluation, the Flat-DQN policy further degenerated. It adopted a do-nothing strategy and remained at the Base Station. This is a common failure mode for DQN agents on long-horizon tasks with sparse rewards.

This experiment confirms our core hypothesis. The difficulty of end-to-end DRL for WRSN charging is not an artifact of continuous action spaces. It is intrinsic to the joint learning of navigation and scheduling over long horizons. The end-to-end agent fails to learn coherent scheduling even with a fully discrete action space. Our hierarchical decomposition effectively bypasses this issue. Delegating navigation to a deterministic A* planner makes the scheduling problem tractable. The Flat-DQN results pair perfectly with the obstacle-dilation safety guarantee (Proposition 1) and the smoothing module (Section 6.5).

Together, they demonstrate that each architectural component in our framework delivers a distinct, empirically identifiable benefit.

6.4.5. Reward-Weight Sensitivity Analysis

The reward function (Equation (1)) involves three weights: α(survival), β(death penalty), and γ (distance penalty). To demonstrate that the framework's performance is not sensitive to fine-tuned weight selection, we conducted a γ-sweep over $\gamma_{\text{distance}} \in \{0.1, 0.3, 0.5, 1.0, 2.0\}$ while holding $\alpha = 5$ and $\beta = 30$ fixed. For each γ value, we trained two independent DQN agents (different seeds) and evaluated each on five held-out test topologies, yielding 10 measurements per γ. Results are reported in Table 6.

Table 6. Reward-weight sensitivity analysis.

γ_{distance}	NDR (Mean ± SD)	Travel (m, Mean ± SD)	n
0.1	0.775 ± 0.014	1539 ± 65	10
0.3	0.781 ± 0.020	1593 ± 104	10
0.5 (selected)	0.774 ± 0.017	1561 ± 107	10
1.0	0.779 ± 0.013	1548 ± 88	10
2.0	0.774 ± 0.021	1769 ± 527	10

In Table 6, across γ_{distance} values from 0.1 to 2.0, the Node Dead Ratio (NDR) remains consistently within the narrow range of [0.774, 0.781]. The maximum cross-γ variation is merely 0.007. This is substantially smaller than the within-γ standard deviation (~0.02). Therefore, the framework demonstrates strong robustness to the precise distance-penalty weight. For rigorous evaluation, we trained each γ configuration using two independent seeds and evaluated them on five held-out topologies ($n = 10$ trials per row).

We selected $\gamma = 0.5$ as our primary operating point. It successfully minimizes total travel distance (1561 m) while preserving a highly competitive NDR (0.774 ± 0.017). Smaller γ values maintain similar NDRs but force the charger to travel much longer distances. Conversely, larger γ values reduce travel further but introduce severe seed-to-seed variance (as evidenced by the wide SD at $\gamma = 2.0$). This overall NDR stability proves the framework is not overfitted to a specific γ, effectively ruling out weight cherry-picking. Furthermore, we fixed the death penalty weight at $\beta = 30$ and the survival reward at $\alpha = 5$. The parameter β must remain significantly larger than α to ensure node deaths strictly dominate the reward gradient. Preliminary tests confirmed that setting $\beta < \alpha$ causes the scheduler to ignore death penalties entirely. We therefore bypassed an exhaustive joint α-β-γ sweep, as the policy structure remains inherently robust within the regime of $\beta \gg \alpha$.

6.5. Trajectory Smoothness

Beyond scheduling intelligence and safety, the mechanical feasibility of generated trajectories is critical for real-world deployment. Grid-based path-planning algorithms typically produce piecewise-linear trajectories with frequent abrupt direction changes, which translate into kinetic energy losses for the Mobile Charger. We measure this as the cumulative absolute turning angle along the path.

Figure 7 reports the cumulative turning angle of the raw A* output versus the smoothed output across $n = 20$ randomized topologies. The proposed two-stage smoothing mechanism (Algorithm 4—line-of-sight pruning followed by quadratic Bézier corner smoothing(formalized in Equation (6)) reduces the cumulative turning angle by a median of 96.64% (per-trial reductions range from 79.4% to 100.0%) while shortening total path length by approximately 5%. The reduction is achieved by eliminating redundant inter-

mediate waypoints through line-of-sight pruning and by replacing the remaining sharp corners with collision-checked Bézier arcs.

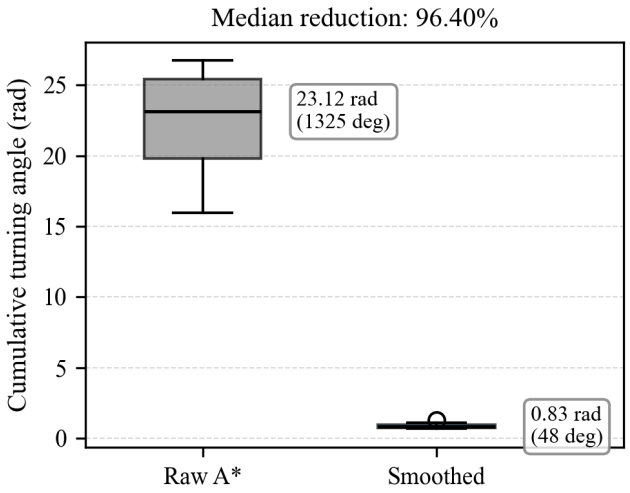

Figure 7. Box plot of cumulative turning angle (radians) for raw A* trajectories versus smoothed trajectories across $n = 20$ randomized topologies. The smoothing module reduces the median turning angle by 96.64% while preserving collision-free guarantees.

Critically, the smoothing module operates on the output of the obstacle-aware A* planner and inherits the latter's dilation-based safety buffer. Each smoothed Bézier arc is sampled and individually verified to maintain the d_safe margin from all obstacles; if a candidate arc would violate the margin, the arc radius is reduced and the verification is repeated. The smoothed trajectory therefore preserves the collision-free guarantee—no separate safety proof is required, and Section 6.4's zero-collision result applies to the final smoothed trajectory, not merely to the raw A* output.

6.6. Ablation Study

To isolate the contribution of each architectural component, we report Table 7 with four ablation variants: (i) the full framework (proposed); (ii) DRL → NJNP scheduler—replaces the DQN with the NJNP heuristic while retaining the obstacle-aware A* planner and the smoothing module; (iii) no smoothing—runs the full DRL framework but emits the raw A* output without the smoothing module; (iv) no obstacle dilation—omits the d_safe inflation in A* (turning the safety guarantee from formal to probabilistic).

Table 7. Ablation study ($N = 100$, $M = 10$, $n = 10$ held-out seeds). '≈' indicates a result statistically indistinguishable from the full framework on that metric. All values are mean ± standard deviation across 10 trials.

Configuration	NDR	Collisions	Turn Angle (rad)
Full framework (proposed)	0.785 ± 0.018	0	0.24 ± 0.30
DRL → NJNP scheduler	0.587 ± 0.017 (lower)	0	0.24 ± 0.30 (≈)
No smoothing	0.785 ± 0.018 (≈)	0	7.13 ± 4.10 (much higher)
No obstacle dilation (dsafe = 0)	0.772 ± 0.015 (≈)	25.20 ± 13.31(lost)	0.24 ± 0.06 (≈)

Each ablation isolates one component while retaining the rest of the pipeline. The full framework is the only configuration that simultaneously achieves: (a) reasonable NDR; (b) zero collisions, formally guaranteed; (c) sub-rad cumulative turning angle. Notably, removing the obstacle dilation (configuration iv) preserves the NDR but immediately

introduces 25.20 ± 13.31 collisions per episode—empirically confirming that the dilation mechanism, not training, is the source of the framework's formal safety guarantee.

6.7. Simulation Summary

Section 6 has presented a four-part evaluation that, taken together, substantiates the paper's central claims. Section 6.3 showed that on static benchmark topologies, the proposed framework occupies a Pareto-optimal position on the efficiency–reliability tradeoff frontier—not strictly dominating NJNP on NDR (which would be unrealistic for any general-purpose learning approach within feasible training budgets), but providing a complementary operating point with 3.8× the charging efficiency of EDF. Section 6.4 is the paper's structural core: against an end-to-end DDPG baseline trained under identical conditions, the proposed framework simultaneously achieves lower NDR (0.785 vs. 0.950), zero collisions (vs. 4.80 ± 4.21), and—most importantly—collision counts that remain identically zero as obstacle density grows from $M = 3$ to $M = 15$, while DDPG's collisions scale from 2.20 to 9.70. This is the empirical substantiation of the formal safety guarantee provided by the dilation construction. Sections 6.5 and 6.6 demonstrate that the framework's safety guarantees do not come at the cost of trajectory feasibility or computational tractability: cumulative turning angle is reduced by 96.64%.

The pattern across these results is consistent and answers a question the introduction raised explicitly: if the DRL scheduler does not strictly dominate NJNP on NDR, why use it? The answer is that the value of the proposed framework lies not in the scheduling component alone, but in the hierarchical architecture as a whole. The DQN scheduler provides retrainability (advantageous under regime change), the Improved A* planner provides formal collision safety (advantageous in obstacle-dense environments), and the trajectory smoothing module provides mechanical feasibility (advantageous for real Mobile Chargers). Each component contributes a property that no existing approach—neither hand-tuned heuristic nor end-to-end DRL—provides on its own.

7. Conclusions and Future Work

This paper presented a hierarchical deep reinforcement learning framework for obstacle-aware charging scheduling and trajectory planning in Wireless Rechargeable Sensor Networks. The central design principle—decoupling combinatorial scheduling (DQN) from geometric navigation (Improved A* + smoothing)—addresses a fundamental limitation of end-to-end DRL approaches in safety-critical deployments: the inability to provide formal collision-free guarantees.

Comprehensive empirical evaluation on networks with 100 sensor nodes substantiates three structural advantages of the proposed framework. First, against an end-to-end DDPG baseline trained under identical conditions, our framework reduces NDR from 0.950 to 0.785, eliminates collisions (0 vs. 4.80 ± 4.21 per episode at $M = 10$), and—critically—maintains exactly zero collisions as obstacle density grows from $M = 3$ to $M = 15$, while DDPG's collision count grows from 2.20 to 9.70. Second, against NJNP and EDF heuristics, the framework occupies a distinct Pareto-optimal point on the efficiency–reliability tradeoff frontier, achieving 3.8× the charging efficiency of EDF. Third, an ablation experiment confirms that disabling the obstacle dilation while preserving all other components introduces 25.20 ± 13.31 collisions per episode, isolating the dilation mechanism as the structural source of the framework's zero-collision property.

The framework does not aim to surpass NJNP on raw NDR within NJNP's home regime of static, uniform-energy conditions—hand-tuned domain heuristics remain near-optimal in such settings. Rather, it provides a retrainable, provably safe, hierarchically organised alternative whose three structural advantages—formal collision safety, mechani-

cal feasibility-preserving smoothness, and learnable scheduling—are each attributable to a distinct, identifiable architectural component.

The framework rests on three modeling assumptions whose scope of applicability we make explicit. First, the obstacle layout is assumed to be known and static throughout an episode; environments where obstacles themselves move at the time-scale of MC navigation (e.g., crowded indoor robotics) fall outside the current scope and require an extension to dynamic-obstacle planning. Second, the energy consumption model is uniform-per-step; heterogeneous workloads—for example, spatially clustered sensing activity or event-triggered bursts—are not explicitly modeled, although the DQN's state vector includes the per-node residual energy and could in principle adapt to such patterns with appropriate training data. Third, the evaluation is conducted on a single MC; the multi-MC setting introduces a combinatorially harder scheduling sub-problem and was beyond the scope of this work.

Future work includes: (i) extending the framework to multi-mobile-charger coordination, where the scheduling sub-problem becomes combinatorially harder and learning is expected to provide larger gains; (ii) studying online adaptation to non-stationary energy consumption patterns, where the retrainable DQN component is expected to outperform fixed heuristics; (iii) integrating the framework with real autonomous mobile platforms to validate the trajectory smoothness gains under physical actuation.

Author Contributions: Conceptualization, J.C.; Formal analysis, J.C.; Methodology, J.C.; Validation, J.C., B.Y. and Z.H.; Writing—original draft, J.C.; Writing—review and editing, J.C., B.Y. and R.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Science Foundation of Fujian Province, China, grant number 2024J01310, and Open Research Project of the Applied Technology Engineering Center for Intelligent Manufacturing in Fujian Provincial Universities (2024), grant number QUZN24-A02-04.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: We extend our deepest appreciation to Chin-Chih Chang and Chang-Wu Yu for their expert guidance, constructive feedback, and continuous support throughout the course of this research.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analysis, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

DDPG	Deep Deterministic Policy Gradient
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
EDF	Earliest Deadline First
IoT	Internet of Things
MC	Mobile Charger
MDP	Markov Decision Process
NDR	Node Dead Ratio
NJNP	Nearest Job Next with Preemption
ReLU	Rectified Linear Unit
WPT	Wireless Power Transfer
WRSN	Wireless Rechargeable Sensor Network
WSN	Wireless Sensor Network

References

1. Aziz, S.A.; Wang, X.; Hawbani, A.; Qureshi, B.; Alsamhi, S.H.; Alabsi, A.; Zhao, L.; Al-Dubai, A.; Ismail, A.S. Wireless Rechargeable Sensor Networks: Energy Provisioning Technologies, Charging Scheduling Schemes, and Challenges. *IEEE Trans. Sustain. Comput.* **2025**, *10*, 873–890.
2. Anisi, M.H.; Abdul-Salaam, G.; Abdullah, A.H. A survey of wireless sensor network approaches and their energy consumption for monitoring farm fields in precision agriculture. *Precis. Agric.* **2014**, *16*, 216–238. [CrossRef]
3. Muñoz-Gea, J.P.; Manzanares-Lopez, P.; Malgosa-Sanahuja, J.; Garcia-Haro, J. Design and implementation of a P2P communication infrastructure for WSN-based vehicular traffic control applications. *J. Syst. Archit.* **2013**, *59*, 923–930.
4. Ma, X.; Liu, X.; Ansari, N.; Ding, J. Enhancing WRSN Sustainability Through on-Demand Directional Wireless Charging with Multiple Green-Powered Mobile Vehicles. *IEEE Internet Things J.* **2025**, *12*, 18019–18030.
5. He, S.; Chen, J.; Jiang, F.; Yau, D.K.Y.; Xing, G.; Sun, Y. Energy Provisioning in Wireless Rechargeable Sensor Networks. *IEEE Trans. Mob. Comput.* **2013**, *12*, 1931–1942.
6. Chen, J.; Chen, H.; Ouyang, W.; Yu, C.W. A Temporal and Spatial Priority With Global Cost Recharging Scheduling in Wireless Rechargeable Sensor Networks. *Int. J. Grid High Perform. Comput.* **2023**, *14*, 1–31.
7. Tazeen, S.; Dash, D. A Heuristic Greedy Approach to Reduce the Dead Periods of the Sensor Nodes in an On-Demand WRSN. *SN Comput. Sci.* **2025**, *6*, 223. [CrossRef]
8. He, L.; Kong, L.; Gu, Y.; Pan, J.; Zhu, T. Evaluating the on-Demand Mobile Charging in Wireless Sensor Networks. *IEEE Trans. Mob. Comput.* **2015**, *14*, 1861–1875.
9. Na, W.; Park, J.; Lee, C.; Park, K.; Kim, J.; Cho, S.; Cho, S. Energy-Efficient Mobile Charging for Wireless Power Transfer in Internet of Things Networks. *IEEE Internet Things J.* **2018**, *5*, 79–92.
10. Jia, Y.; Jiahao, W.; Zeyu, J.; Ruizhao, P. Multiple Mobile Charger Charging Strategy Based on Dual Partitioning Model for Wireless Rechargeable Sensor Networks. *IEEE Access* **2022**, *10*, 93731–93744. [CrossRef]
11. Bouakaz, A.; Gautier, T.; Talpin, J.-P. Earliest-deadline first scheduling of multiple independent dataflow graphs. In Proceedings of the 2014 IEEE Workshop on Signal Processing Systems (SiPS), Belfast, UK, 20–23 October 2014; pp. 1–6.
12. Lin, C.; Zhou, J.; Guo, C.; Song, H.; Wu, G.; Obaidat, M.S. TSCA: A Temporal-Spatial Real-Time Charging Scheduling Algorithm for On-Demand Architecture in Wireless Rechargeable Sensor Networks. *IEEE Trans. Mob. Comput.* **2018**, *17*, 211–224.
13. Wang, X.; Dai, H.; Wang, W.; Zheng, J.; Yu, N.; Chen, G.; Dou, W.; Wu, X. Practical Heterogeneous Wireless Charger Placement with Obstacles. *IEEE Trans. Mob. Comput.* **2020**, *19*, 1910–1927.
14. Rahman, S.; Akter, S.; Yoon, S. OADC: An Obstacle-Avoidance Data Collection Scheme Using Multiple Unmanned Aerial Vehicles. *Appl. Sci.* **2022**, *12*, 11509. [CrossRef]
15. Zeng, G.; Wang, K. CCSO: A dynamic collaborative scheduling scheme for wireless rechargeable sensor networks with obstacles. *Wirel. Netw.* **2023**, *30*, 6161–6176. [CrossRef]
16. Chang, C.Y.; Lin, C.Y.; Yu, G.J.; Kuo, C.H. An energy-efficient hole-healing mechanism for wireless sensor networks with obstacles. *Wirel. Commun. Mob. Comput.* **2011**, *13*, 377–392.
17. Raha, S.M.A.; Azharuddin, M.; Kuila, P. Obstacles Avoidance Charging Schedule for Multiple Mobile Charging Vehicles in Wireless Rechargeable Sensor Networks. *Int. J. Commun. Netw. Distrib. Syst.* **2023**, *32*, 88–117.
18. Jiang, C.; Chen, W.; Wang, Z.; Xiao, W. Deep Reinforcement Learning-Based Joint Sequence Scheduling and Trajectory Planning in Wireless Rechargeable Sensor Networks. *IEEE Sens. J.* **2024**, *24*, 13699–13711. [CrossRef]
19. Zhang, X.; Jia, R.; Yin, Q.; Zheng, Z.; Li, M. Intelligent Trajectory Design and Charging Scheduling in Wireless Rechargeable Sensor Networks with Obstacles. *IEEE Trans. Mob. Comput.* **2024**, *23*, 8664–8679. [CrossRef]
20. Gong, Z.; Wu, H.; Feng, Y.; Liu, N. Deep Reinforcement Learning-Based Online One-to-Multiple Charging Scheme in Wireless Rechargeable Sensor Network. *Sensors* **2023**, *23*, 3903. [PubMed]
21. Mo, L.; Angeliki, K.; He, S. Energy-Aware Multiple Mobile Chargers Coordination for Wireless Rechargeable Sensor Networks. *IEEE Internet Things J.* **2019**, *6*, 8202–8214. [CrossRef]
22. Zhao, M.; Li, J.; Chai, X.; Li, J. An efficient dynamic energy replenishment and data gathering strategy based on deep reinforcement learning in wireless rechargeable sensor networks. *Alex. Eng. J.* **2025**, *123*, 170–179. [CrossRef]
23. Jiang, C.; Wang, Z.; Chen, S.; Li, J.; Wang, H.; Xiang, J.; Xiao, W. Attention-Shared Multi-Agent Actor-Critic-Based Deep Reinforcement Learning Approach for Mobile Charging Dynamic Scheduling in Wireless Rechargeable Sensor Networks. *Entropy* **2022**, *24*, 965. [PubMed]
24. Cao, X.; Xu, W.; Liu, X.; Peng, J.; Liu, T. A deep reinforcement learning-based on-demand charging algorithm for wireless rechargeable sensor networks. *Ad Hoc Netw.* **2021**, *110*, 102278.
25. Nguyen, P.L.; La, V.Q.; Nguyen, A.D.; Nguyen, T.H.; Nguyen, K. An on-Demand Charging for Connected Target Coverage in WRSNs Using Fuzzy Logic and Q-Learning. *Sensors* **2021**, *21*, 5520. [CrossRef] [PubMed]
26. Wang, Y.; Dong, Y.; Li, S.; Wu, H.; Cui, M. CRCM: A New Combined Data Gathering and Energy Charging Model for WRSN. *Symmetry* **2018**, *10*, 319. [CrossRef]

27. Dudyala, A.K.; Dash, D. Genetic algorithm-based partial charging schedule of rechargeable sensor networks. *Int. J. Commun. Syst.* **2023**, *36*, e5485.
28. Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; Riedmiller, M. Deterministic Policy Gradient Algorithms. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 24 June 2014; pp. 387–395.
29. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
30. Hart, P.E.; Nilsson, N.J.; Raphael, B. A Formal Basis for the Heuristic Determination of MinimumCost Paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107.
31. Latombe, J.-C. *Robot Motion Planning*; Kluwer Academic Publishers: Boston, MA, USA, 1991.
32. LaValle, S.M. *Planning Algorithms*; Cambridge University Press: Cambridge, UK, 2006.