Routing-Aware RL for Mobile Relay Navigation in Disconnected Ad Hoc Networks

Can Karacelebi
Department of Computer Engineering
Middle East Technical Univ. Türkiye
can.karacelebi@metu.edu.tr

Ertan Onur
Department of Computer Engineering
Middle East Technical Univ. Türkiye
eronur@metu.edu.tr

Yusuf Şahin
Department of Computer Engineering
Middle East Technical Univ. Türkiye
yusuf.sahin@metu.edu.tr

***Abstract*—Mobile relays can restore end-to-end connectivity in disconnected ad hoc networks, yet classical placement methods rely on global geometry and channel maps that are rarely available in practice. We study routing-aware reinforcement learning (RL) for a single controlled host that moves on a 2D plane to bridge disconnected clusters and enable multi-hop delivery between a source and destination. Built on OMNeT++/INET, our environment exposes only local, router-observable signals and augments them with a lightweight host-discovery memory. We propose a composite reward that couples end-to-end delivery with topology-shaping incentives. We benchmark PPO and QRDQN under domain randomization and a scenario curriculum that rotates layouts every K episodes. Both methods learn to discover hosts, position between clusters, and sustain high forwarding efficiency on unseen topologies; QRDQN converges faster under sparser rewards while PPO yields smoother final policies.**

***Index Terms*—ad hoc networks, mobile relays, reinforcement learning, OMNeT++, PPO, QRDQN, distributional RL, routing-aware observation, domain randomization, curriculum learning**

I. Introduction

Autonomous vehicles are increasingly utilized as opportunistic relays to restore or enhance connectivity in wireless networks, especially ad hoc networks when fixed infrastructure is absent, impaired or in an inoperative state. In such settings, a single autonomous vehicle can dynamically position itself so that disconnected clusters of ground nodes can exchange traffic. Classical approaches assume accurate global knowledge of network geometry and channel conditions; practical deployments rarely afford such information. Targeted deployments include post-disaster and search-and-rescue missions, vehicular/convoy communications on sparsely covered roads, and industrial sites where separated sensor clusters need a temporary bridge to a command station. Our formulation is platform-agnostic (UAV or ground vehicle) and focuses on network-layer placement under such operational constraints.

RL has recently been applied to networking and edge scenarios from 6G service placement to edge-centric orchestration and graph-structured decision problems showing promise for dynamic, communicating systems [1]–[3].

This paper studies the problem of learning movement policies for an intelligent vehicle to act as a relay that bridges spatially disjoint subnetworks of stationary hosts and establishes an end-to-end connection from a designated source host to a destination host. We build a simulation environment on top of OMNeT++/INET via OmnetGym (a C++ based framework built upon OMNeT++ designed for RL research and development on adaptable network simulations) where a controlled host moves in a 2D plane to bridge heterogeneous host/node clusters, which form a disconnected ad hoc network. The environment surfaces a fixed-size observation vector centered on network layer counters and connectivity summaries: IP forwarding/drops/unroutables, MAC queue processed/dropped, radio state, estimated neighbor information, estimated average link quality and edge-computable packet-flow statistics. We also introduce a host-discovery memory for up to M potential hosts in fixed slots: once the controlled host establishes stable connection with a new host its position remain in memory enabling robust spatial reasoning. Together, these signals allow an RL based agent to acquire enhanced reasoning causally improving multi-hop objective routing. This routing-centric partial environmental observation is deliberately designed to avoid privileged global maps while still being sufficient for navigation and placement. We couple this observation with a composite reward that encodes the networking objective directly. Beyond end-to-end delivery rewarding, the reward consists of detection of cluster bridging by rewarding proximity to the line segment between predicted cluster centroids based on host discovery. The reward also encourages coverage of sparse gaps, multi-directional connectivity, forwarding feedback to prevent sparse rewarding and to address the delayed-credit problem arising from packet delivery ratio and penalizes peripheral-occupancy bias and isolation which is a generic measure in a physical RL problem.

We introduce a procedural isomorphic-graph generator that supports topology diversity, positional jitter, and communication-range variation for better generalization across scenarios. We also employed a scenario curriculum that rotates through increasingly challenging topologies over OMNeT++ simulations. The resulting policies succeed to (i) discover hosts efficiently, (ii) position between clusters, (iii) maintain high forwarding efficiency enabling delivery without access to any global topology information. From an algorithmic standpoint, we adopt Proximal Policy Optimization PPO [4] for stable on-policy updates with clipped policy ratios and generalized advantage estimation,

This work is partially supported by METU BAP Project, ADEP-312-2025-11622.

978-3-903176-75-1 ©2025 IFIP

and Quantile Regression DQN (QRDQN) [5] to learn a distribution over returns for improved robustness under sparse and heavy-tailed rewards.

Our routing-aware formulation and evaluation protocol are aligned with recent contributions that apply RL to wireless and ad hoc networking problems, highlighting the promise of learning-based adaptation for routing, link selection, and connectivity maintenance in dynamic environments.
The contributions of this paper are threefold:

- **Routing-aware partial observation design** (Sec. II-C): an observation built from router-observable counters, connectivity summaries, and a host-discovery memory.
- **Delivery- and topology-shaping composite reward** (Sec. II-F): a reward that blends end-to-end delivery with bridging, coverage, multi-directional connectivity, exploration, and early forwarding feedback.
- **Extensive simulation study** (Secs. IV–V): OMNeT++ experiments across four training and three held-out topologies, with curriculum cadence and delivery weight, demonstrating performance and clarifying PPO–QRDQN trade-offs.

All in all, our results suggest that "observe what routers observe"—rather than global state—is a viable recipe for learning effective controlled relay navigation and placement policies in ad hoc networks, aligning with the broader trend of edge-native RL for intelligent systems [1], [2].

II. Mobile Relay Navigation Problem

We study the navigation of a single controlled host to act as a relay in a two-dimensional arena to restore connectivity between otherwise disconnected stationary clusters of hosts including a host destination. We consider a single mobile relay that moves in a rectangular two-dimensional arena populated by stationary hosts and a designated destination D. The relay's task is to navigate and hold a placement that simultaneously remains in range of nodes from different clusters, thereby creating a temporary multi-hop path from source to destination and maximizing end-to-end delivery over the episode horizon.

A. Network Entities and Topology

The environment is implemented in OMNeT++ (v6.2.0)/INET (v4.5.4) as a custom network simulation comprising (i) a set of stationary hosts $H = \{h_0, \ldots, h_{N-1}\}$ (ii) a stationary destination D and (iii) a single controlled host C with a custom module. The simulation area is a $\mathcal{X} = L \times W$ m^2 rectangle.

Scenarios are crafted under two predicates: (i) the communication graph is disconnected in the absence of a relay, (ii) there exists at least two disjoint clusters. Isomorphic scenarios are generated with respect to these conditions and validated after generation. Different cluster shape models are used to generate clusters including circle, h-line/v-line, triangle, square, cross, star and random. For a two island scenario let $H_1, H_2 \subset \mathbb{R}^2$ be host locations and d_{comm} the communication range. After positioning central hosts each island is instantiated by a shape function $S(.)$ that returns host coordinates around its center. Isomorphic scenarios are obtained by a rigid motion $g(x) = R_\theta x + t$ (rotation by θ and translation t) applied to all points including the gap center to preserve the characteristics of a scenario.

B. Node and Network Model

A single application flow is active: a source node h_0 sends 100-byte UDP datagrams every 100 ms, to the destination host D. Nodes run IPv4 with automatic address assignment; ARP resolution is assumed to succeed without loss or delay. Ad hoc On-Demand Distance Vector Routing (AODV) is employed with a short active route timeout of 1s. Healthy multi-hop routes appear only when the communication topology graph becomes connected through the controlled host C. Wireless propagation follows a range based (unit-disk) model. Simple path loss model is employed with no interference, capture and detection-range effects. The receiver does not accumulate or react to concurrent transmissions. The MAC is idealized and contention-free (no collisions) and the PHY rate is fixed at 2 Mbit/s. Communication range R is 199 m for all nodes.

C. Observation Model

The agent receives a fixed-length vector $o_t \in \mathbb{R}^d, o_t = [o^{pos}; o^{route}; o^{conn}; o^{disc}; o^{topo}; o^{flow}]$, normalized and clipped, built as a concatenation of semantically distinct information. Observation is deliberately restricted to quantities that the controlled host could obtain locally or via a plausible control-plane exchange. Preventing an *oracle* topology information to extend the realistic capabilities of a trained agent: where Position $(\hat{x}_t, \hat{y}_t)$: Controlled host's current position is given with affine normalization to [-1,1]. For an area of $L \times W$ m^2 :

$$o^{pos} = \left[\frac{x_t - L/2}{L/2}, \frac{y_t - W/2}{W/2}\right] \in [-1, 1]^2.$$

Routing metrics consist of IP Packets forwarded F, IP-layer drops D_{IP}, unroutable packets U, MAC queue drops $Q_\downarrow$, MAC dequeued/processed $Q_\uparrow$ and radio transmission state $T_s \in \{0, 1, 2, 3\}$. Metrics are divided by 1000 for normalization. All components of o^{route} are measured locally on the controlled host C and require no information from isolated hosts or any central coordinator.

$$o^{route} = \left[\frac{F}{1000}, \frac{D_{IP}}{1000}, \frac{U}{1000}, \frac{Q_\downarrow}{1000}, \frac{Q_\uparrow}{1000}, \frac{T_s}{4}\right].$$

Connectivity Metrics are computed on-the go from discovered neighbors and queues in the stack and consists of normalized neighbor count in range N_r, average distance-based link quality $\mathcal{S}$, link utilization from the MAC queue $L_U = \min(Q_\uparrow/(Q_\downarrow + Q_\uparrow), 1)$, a congestion proxy from drop counts $\mathcal{C} = \min(D_{IP}/100, 1)$ and an estimated number of known routes $N_{route} = 2 \times N_r + \lfloor F/|H_{\max}| \rfloor$.

$$\mathcal{S} = \frac{1}{|H|} \sum_{h \in H} (1 - \frac{||x_t - p(h)||}{R_c}),$$

where $p(h)$: position of host h,

$$o^{conn} = \left[\frac{N_r}{10}, \mathcal{S}, L_U, \mathcal{C}, \frac{N_{route}}{10}\right].$$

Host-discovery memory: For each potential host $i \in \{0, ...|H|-1\}$, a slot in the memory $[z_i, \hat{x}_i, \hat{y}_i, a_i]$ encodes a discovered flag $z_i \in \{0, 1\}$, normalized coordinates $(\hat{x}_i, \hat{y}_i)$ and a normalized activity level a_i based on forwarded and sent packets. We constructed an adjustable discovery system: if the controlled host ever comes within $R_{disc} = \alpha R$ ($\alpha \in \mathbb{R}$) of host i, its slot is filled and persists for the remainder of the episode. Let H_{disc} the set of discovered hosts the observation becomes:

$$o_i^{disc} = \begin{cases} [1, x(h_i)/L, y(h_i)/W, a(h_i)] & \text{if } h_i \in H_{disc} \\ [0, 0, 0, 0] & \text{otherwise} \end{cases}$$

The topology proxies contains topology based heuristics which act as proxies. It consists an estimated graph diameter $\hat{D}_{net}$ based on reachable neighbor hosts providing a discrete number $k \in \{2, 4, 8\}$ inversely proportional to the number of reachable hosts. Estimated number of disconnected network regions $\hat{C}_{disc} \in \{1, 2, 3\}$ also inversely proportional to the neighboring hosts. The path quality $\hat{q}_{t\to d}$ is calculated as delivery rate clipped to 1. Route stability $\hat{\sigma}_{local} \in [0, 1]$ is determined proportional to the number of packets forwarded by the controlled host. Estimated local network density $\hat{\rho}_{local}$ is defined as $\min(N_r/|H|, 1.0)$. Drop-rate based bottleneck severity $\hat{\beta}_{bneck}$ is calculated as $\min(2 \times D_{IP}, 1.0)$. They as a whole form the derived heuristic observations:

$$o^{topo} = [\frac{\hat{D}}{10}, \frac{\hat{C}_{disc}}{5}, \hat{q}_{t\to d}, \hat{\rho}_{local}, \hat{\sigma}_{route}, \hat{\beta}_{bneck}].$$

Packet-flow statistics are derived from both the controlled host and neighboring hosts. S^{src} denotes the number of packets sent from the source host and R^{dest} denotes the number of packets received by the destination host. Featuring relay contribution $C_{rel} = F/\max(1, S^{src})$, relay efficiency $E_{rel} = F/\max(1, F + D_{IP})$, end-to-end success $\mathcal{E} = R^{dest}/\max(1, S^{src})$, an overall activity level A and normalized forwarding and drop counts accumulated from the neighboring hosts. We represent packet-flow statistics as

$$o^{flow} = [C_{rel}, E_{rel}, \mathcal{E}, A, \frac{\Sigma F}{1000}, \frac{\Sigma D_{IP}}{1000}].$$

All scalars are normalized to $[0, 1]$ or $[-1, 1]$.

D. Action Space

We model the control problem as a discounted Partially Observable Markov Decision Process (POMDP) $\mathcal{M} = (\mathcal{X}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \Omega, R, \gamma)$ [6]. In the POMDP setting the agent selects actions according to a policy $\pi(a_t \mid h_t)$ over histories h_t . The action space is a 5-way discrete set. We have deliberately selected a discrete action space to allow a larger set of RL algorithms including both on-policy and off-policy to be trained on the problem:

$$\mathcal{A} = \{STAY, N, S, E, W\}$$

If $u_t \in \mathcal{A}$ denotes the chosen action and $\delta > 0$ the step length in seconds, the kinematics of the controlled host can be formulated with respect to the position p_t, step size δ and speed v. The position $p_t = (x_t, y_t)$ evolves as:

$$p_{t+1} = \Pi_{\mathcal{X}}(p_t + \Delta(u_t)),$$
$$\Delta(N) = (0, v\delta), \Delta(S) = (0, -v\delta),$$
$$\Delta(E) = (v\delta, 0), \Delta(W) = (-v\delta, 0),$$

In our experimental setup default values are determined as $\delta = 3.5s$ and $v = 5m/s$ to comply with realistic experimentation and viable output which is directly connected to the accumulated packet statistics in a given time frame.

E. Domain Randomization

In order to avoid overfitting to a single geometry and to emulate uncertainties might arise from channel and modeling we apply domain randomization (DR) during training. Each episode samples an MDP from a family $\{\mathcal{M}_\phi\}_{\phi\sim\mathcal{D}}$. With probability p^{DR} we generate:

$$x_i^{(0)} \leftarrow x_i^{scenario} + \varepsilon_i, \;\; \varepsilon \sim \mathcal{N}(0, \sigma_{pos}^2 I),$$
$$R \leftarrow R(1 + \zeta), \;\; \zeta \sim Uniform[-\rho, \rho],$$

where $\sigma_{pos} \in \{15, 25, 35\}$ m and $\rho \in \{0.1, 0.2, 0.3\}$ for $\{light, medium, heavy\}$ DR levels respectively. $p^{\text{DR}} \in \{0, 4, 0.5, 0.7\}$. We also change the active training scenario every K episodes, completely changing host counts and cluster layouts.

F. Reward Design

Formal and informal descriptions of the natural environment combined with the problems may cause a sparse rewarding environment which creates a major challenge for an agent to learn.

We built a workaround solution with the help and intuition from the geometric environment so that the agent is exposed to constant reward for its goal forming a continuous guide to its target. As a result we created a composite reward sensitive to both geometric and network signals. We also included small penalties. After introducing the delivery reward as the anchor reward we introduced positive rewards in an empirical style observing the delivery ratio. The reward is formulated as:

$$r_t = \alpha r_t^{deliv} + \beta r_t^{bridge} + \gamma r_t^{coverage} + \delta r_t^{multiDir}$$
$$+\varepsilon r_t^{explore} + \zeta r_t^{fwd} + \eta r_t^{posPen}.$$

where delivery r_t^{deliv} defines our main goal and main rewarding mechanism. α is always larger than any other term coefficient in the composite reward. Several values are experimented as a hyperparameter during training/evaluation. Let $\Delta S_t = S_t^{src} - S_{t-1}^{src}$ and $\Delta R_t = R_t^{dest} - R_{t-1}^{dest}$. Delivery reward is

$$r_t^{deliv} = \begin{cases} 0 & \Delta S_t = 0, \\ \frac{\Delta R_t}{\Delta S_t} & \text{otherwise.} \end{cases}$$

Cluster bridging r_t^{bridge}: The bridging term uses discovery memory to form coarse clusters by single-linkage with a spatial threshold. For each pair of estimated cluster centroids c_1, c_2 we project the position of the controlled host p_t onto the line segment and reward small perpendicular distance $dist_\perp(p_t, [c1, c2])$ if the projection lies between centroids.

$$r_t^{bridge} = \min(1, \sum_{1\le i\le j\le K} [\phi \max(0, 1 - \frac{d_\perp(x_t; c_i, c_j)}{d_{\max}} \mathcal{L}(i,j))])$$

Let $\{C_1, ...C_k\}$ be the connected components, each cluster centroid is defined as $\mu_k = \frac{1}{|C_k|}\sum_{p\in C_k} p$, $\mu_k \in \mathbb{R}^2$. A vector between centroids is $v_{ij} = \mu_j - \mu_i$ with length $L_{ij} = ||v_{ij}||$ and the unit direction is defined as $u_{ij} = v_{ij}/L_{ij}$. The projection of controlled host along the segment $\mu_i \to \mu_j$: $t_{ij} =< x_t - \mu_i, u_{ij} >\in \mathbb{R}$. The " inside-segment " function $\mathcal{L}$ is defined as follows:

$$\mathcal{L}(i,j) = \begin{cases} 1 & \text{if } 0 < t_{ij} < L_{ij}, \\ 0 & \text{otherwise.} \end{cases}$$

Coverage of sparse gaps $r_t^{coverage}$: The gap term encourages being between clusters but not too far from any node: it is positive when the nearest discovered host is $0.5 \times d_{comm} - 1.5 \times d_{comm}$ m away, and slightly negative when it is larger than $1.5 \times d_{comm}$ m. Let $d_{\min} = \min_h ||x_t - p(h)||$

$$r_t^{\text{coverage}} = \begin{cases} 0 & d_{\min} < 0.5 \times d_{comm}, \\ -0.1 & d_{\min} \ge 1.5 \times d_{comm}, \\ 0.2\left(1 - \dfrac{|d_{\min} - d_{comm}|}{0.5 \times d_{comm}}\right) & \text{otherwise.} \end{cases}$$

Multi Directional Local Connectivity $r_t^{multiDir}$: This term bins discovered neighbors within range into eight octants around the controlled host and gives a bonus for opposite-octant coverage conclusively an estimate for forming links to both sides of a gap. N_s denotes number of sectors and N_{opp} denotes the number of opposite pairs.

$$r_t^{multiDir} = 0.1 \times N_s/8 + 0.2 \times N_{opp}$$

Exploration $r_t^{explore}$: The exploration term is a first-visit bonus on a $a \times a$ m cell portioned grid. Granting a bonus on first visit of a cell, with a mild radial term from the start.

$$r_t^{explore} = \begin{cases} 0.1 + 0.05 \times \min(\frac{||x_t - x_0||}{10\times a}, 1) & \text{if new cell,} \\ 0 & \text{otherwise.} \end{cases}$$

Forwarding: Forwarding term rewards increases in number of forwarded packets. with larger weight for the first few forwards forming an early hint for the agent that the placement is enabling routes. Let $\Delta F_t = F_t - F_{t-1}$, then

$$r_t^{fwd} = \begin{cases} 0.2\Delta F_t & \text{if } F_t \le 10, \\ 0.05\Delta F_t & \text{if } F_t > 10, \\ 0 & \Delta F_t \le 0. \end{cases}$$

Positional Penalty r_t^{posPen}: This term applies soft penalties near the arena $p_{edge} \in \{0, 1\}$ (if close to an edge within a meters) to prevent peripheral-occupancy bias and when the controlled host has zero neighbors in range (isolation $p_{iso} \in \{0, 1\}$) ,

$$r_t^{posPen} = -0.05 \times p_{edge} - 0.1 \times p_{iso}.$$

G. Objective and Learning Problem

Given a parameterized policy $\pi_\theta(a_t|o_t)$ the goal is to maximize the discounted return under the episode/scenario distribution:

$$\max_\theta J(\theta) = \mathbb{E}_{\xi\sim\mathcal{D}}\mathbb{E}_{\tau\sim\pi_\theta,\xi}[\sum_{t=0}^{T-1} \gamma^t r_t]$$

The discount factor $\gamma = 0.99$, scenario ξ specifies host layout and communication range drawn via procedural graph generation and further domain randomization during training.

III. Reinforcement Learning Algorithms

A. Proximal Policy Optimization (PPO)

PPO [4] maximizes a clipped surrogate of the advantage to stabilize updates. With old policy $\pi_{\theta_{\text{old}}}$ and ratio:

$$r_t(\theta) = \frac{\pi_\theta(a_t|o_t)}{\pi_{\theta_{\text{old}}}(a_t|o_t)}$$

the objective function is formed as follows:

$$\mathcal{L}_{\text{PPO}}(\theta) = \mathbb{E}_t\Big[\min\big(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\big)\Big]$$

with GAE advantages $\hat{A}_t$, entropy regularization, and a value loss.

B. Quantile Regression DQN (QRDQN)

Distributional RL propagates the full return distribution instead of only its mean [7]. QRDQN [5] models the distribution of returns via N quantile values per action and minimizes the quantile Huber loss [5] between target and current return distributions. Let $\{\tau_i\}_{i=1}^N$ be quantile fractions and $Z_\theta(o, a)$ the quantile outputs; the loss is

$$\mathcal{L}_{\text{QR}} = \mathbb{E}\Big[\frac{1}{N}\sum_{i=1}^{N} \rho_{\tau_i}^\kappa\big(y - Z_\theta(o,a)_i\big)\Big],$$

where $y = r + \gamma \max_{a'} Z_{\bar{\theta}}(o', a')$ uses a target network $\bar{\theta}$ and ρ^κ is the quantile Huber loss. Distributional critics can be more sample-efficient and robust under sparse or heavy-tailed rewards common in networking tasks.

IV. Training and Evaluation Setup

Training is offline, at run time the policy executes on the relay (edge). We use compact MLP policies that are CPU-capable, matching embedded/vehicular platforms. All experiments use the environment described in Section II with a fixed episode horizon of T=1000 steps with light domain randomization. Per-step rewards are clipped to $[-1, 2]$ inside the environment. We train on a bank of four procedurally

TABLE I
TRAINING HYPERPARAMETERS

(a) PPO

Parameter	Setting
Policy / Net	MLP (ReLU), orthogonal init; separate heads with `pi,vf` $= [256, 256, 128]$
γ, GAE λ	0.99, 0.95
Rollout n_{steps}	1024
Batch / Epochs	256 / 8
Clip range ϵ	0.2
Entropy / Value coeff.	0.10 / 0.50
Learning rate	3×10^{-4}
Target KL (soft)	0.02
Total env steps	800,000
Eval freq (steps) / episodes	16384 / $(4 + 3)$
Obs/Reward norm	VecNormalize (obs+reward at train; frozen stats at eval)

(b) QR-DQN

Parameter	Setting
Policy / Net	MLP (ReLU); layers $[512, 512, 256]$; N_q=64 quantiles
Replay size / Start	10^6 / 20k steps
Batch / Train freq / Grad steps	256 / 8 / 8
Discount γ	0.97
Targets	Hard update every 10k; τ=1.0
ε-greedy	frac 0.85, $\varepsilon : 1.0 \rightarrow 0.05$
Learning rate	1×10^{-4}
Frame stack	n_{stack}=4
Obs/Reward norm	Obs-only (no reward norm)
Total env steps	800,000
Eval freq / episodes	8192 / $(4 + 3)$

generated scenarios (disconnected at t=0) and evaluate on seven scenarios, the first four overlapping with training and the remaining three held-out layouts. To avoid overfitting to a fixed level while keeping gradient variance practical, we cycle scenarios during training every K episodes. Such level-resampling curricula are common in RL to promote generalization [8]–[10]. We vary the coefficient α of the delivery term r_t^{deliv} in the composite reward (Sec. II) to study the exploitation/shaping trade-off. Hyperparameters are shown in Table I. At each evaluation checkpoint we run a deterministic episode per scenario using frozen normalization and report: mean/std. dev. reward of episode return.

V. RESULTS

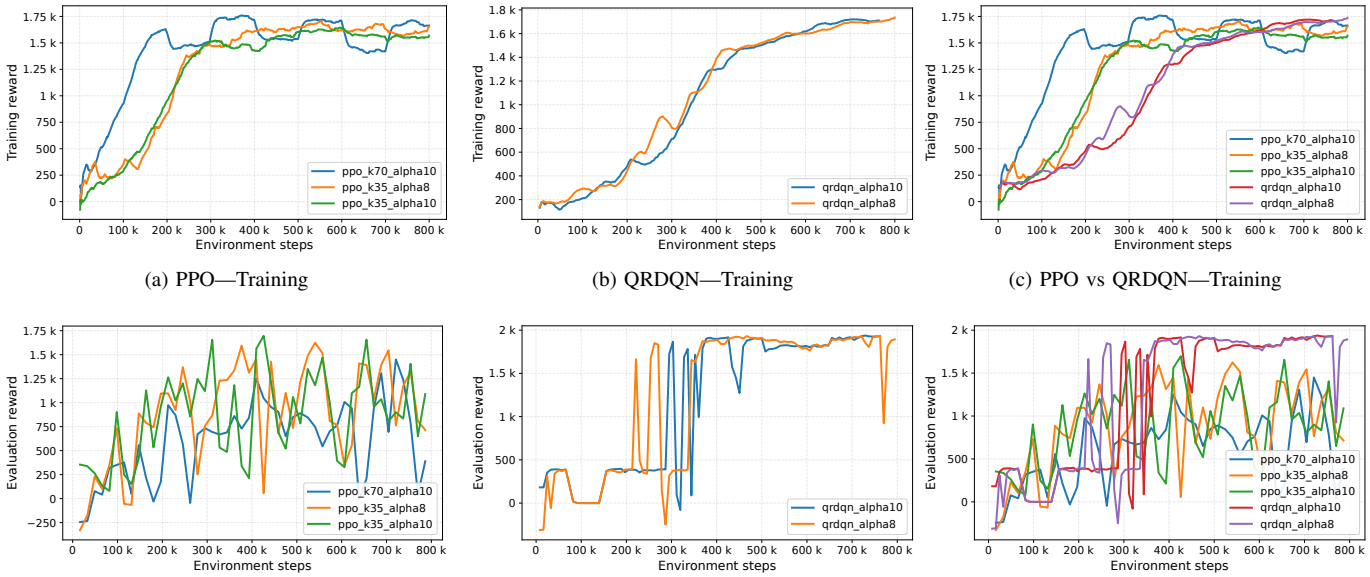
Fig. 1 summarizes training and evaluation curves for PPO and QRDQN under the principal parameters we change: (i) the scenario change frequency $K \in \{35, 50\}$, and (ii) the delivery-weight scale $\alpha \in \{8, 10\}$ used in the composite reward. We report two overlays per algorithm: training reward and evaluation reward on held-out scenarios. With K=35, PPO receives a faster rotation of topologies and thus encounters more frequent distribution shifts early in learning. In our runs, this yields slightly slower initial growth but improved stability in the later plateau, consistent with a higher cadence of experience diversity. K=50 shows quicker early improvements but exhibits occasional regressions when a challenging layout appears late in an epoch; the gap narrows once the policy has seen several full cycles. Increasing the delivery-weight scale from α=8 to α=10 accentuates sparse end-to-end signals relative to the shaping terms. PPO benefits from α=8 with smoother, more monotone curves; α=10 learns faster when delivery signals are reliably discoverable, but can be more variable on difficult maps. QRDQN, by contrast, tolerates α=10 better thanks to distributional targets.

We evaluate on a pool of known and unknown scenarios generated by the same procedural family but with different cluster shapes, gaps, and range jitters. Across both K settings, policies generalize: the evaluation overlays in Fig. 1 track the training curves closely after the first few checkpoints, indicating that policies rely on routing-aware signals rather than memorized coordinates. The bottom row of Fig. 1 also shows evaluation variability. As expected, variability is higher in the mid-training phase when policies first discover bridging placements; the spread narrows once the host-discovery memory is consistently leveraged and the cluster-bridging term is exploited. QRDQN's variability curves are typically flatter after convergence, reflecting its robustness to heavy-tailed rewards.

To interpret behaviors, we visualize trajectories on representative episodes (Fig. 2). In successful cases on unseen layouts, the agent first sweeps to discover at least one cluster, then transitions towards the estimated inter-cluster segment and settles within a corridor that maintains neighbors in opposing octants. Scenario cadence K trades early speed for late stability; K=35 tends to smooth learning as diversity increases. Reward emphasis α=10 accelerates discovery when delivery is attainable; α=8 is safer when reward sparsity is severe. QRDQN is especially effective under sparse/sharp reward spikes, while PPO produces very stable final policies with modest tuning. QRDQN's quantile targets preserve information about the tail of the return distribution, so a few high-value transitions can shape learning, which yields more repeatable evaluation once such transitions are discovered. PPO instead relies on advantage estimates whose variance spikes under sparsity, this produces smooth plateaus but can be less sensitive to rare positive feedback unless schedules are carefully tuned. Trajectory visualizations confirm that the routing-aware observation and cluster-bridging reward shape the intended behavior: discover, align with the gap, and hold a bi-directional relay position. Although the agents may fail under certain unseen scenarios it should be noted that these scenarios generally require a high level of precision and the agent generally can navigate to a close position to the ideal.

VI. CONCLUSION

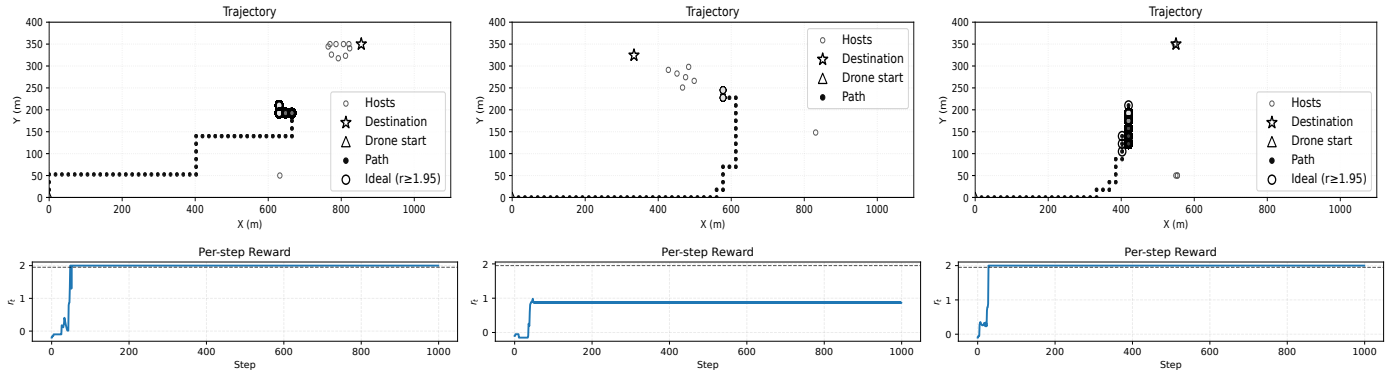
We investigated routing-aware reinforcement learning for mobile relay placement in disconnected ad hoc networks. Our environment exposes router-observable signals augmented by a host-discovery memory and couples them with a composite reward that directly encodes networking intent. Both PPO and QRDQN learned policies that discover clusters, position along inter-cluster corridors, and sustain forwarding on unseen layouts. Results support our claim that robust relay navigation does not require global maps. The generalization we observe



(a) PPO—Training (b) QRDQN—Training (c) PPO vs QRDQN—Training

(d) PPO—Evaluation (e) QRDQN—Evaluation (f) PPO vs QRDQN—Evaluation

Fig. 1. Learning curves, 800k steps. Top: training rewards; bottom: evaluation rewards. Each panel overlays runs within the algorithm.



(a) PPO trained scenario (b) PPO unseen failed scenario (c) QRDQN unseen scenario

Fig. 2. Agent navigation trajectories, each image shows the controlled host's path and other hosts. Controlled host starts from (0,0).

indicates that the agent internalizes routing-relevant structure rather than memorizing coordinates. Overall, the combination of routing-aware observations, topology-shaping rewards, and modest curriculum/domain randomization emerges as a simple, reproducible recipe for learning effective relay behaviors in ad hoc networks. Future work will explore scaling to multi-relay cooperation and integrating more realistic wireless models to further validate in practical deployments.

References

[1] A. F. Ocampo and J. Santos, "Reinforcement learning-driven service placement in 6g networks across the compute continuum," in *2024 20th International Conference on Network and Service Management (CNSM)*, 2024, pp. 1–9.

[2] M. Mounesan, X. Zhang, and S. Debroy, "Edgerl: Reinforcement learning-driven deep learning model inference optimization at edge," in *2024 20th International Conference on Network and Service Management (CNSM)*, 2024, pp. 1–5.

[3] W. Kaili, W. Muqing, and Z. Min, "Finding key nodes in complex networks via deep reinforcement learning and multi attention node connectivity," in *2024 20th International Conference on Network and Service Management (CNSM)*, 2024, pp. 1–5.

[4] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[5] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos, "Distributional reinforcement learning with quantile regression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[6] D. Rengarajan, G. Vaidya, A. Sarvesh, D. Kalathil, and S. Shakkottai, "Reinforcement learning with sparse rewards using guidance from offline demonstration," 2022. [Online]. Available: https://arxiv.org/abs/2202.04628

[7] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," 2017. [Online]. Available: https://arxiv.org/abs/1707.06887

[8] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," ser. ICML '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 41–48. [Online]. Available: https://doi.org/10.1145/1553374.1553380

[9] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman, "Leveraging procedural generation to benchmark reinforcement learning," 2020. [Online]. Available: https://arxiv.org/abs/1912.01588

[10] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters." AAAI, 2018.