LEARNING EFFICIENT COVERAGE UNDER HARD ENERGY CONSTRAINTS: COORDINATING FEASIBILITY ESTIMATION AND POLICY SELECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Minimizing task cost alone is insufficient for robotic missions with hard constraints: a single violation can prevent mission completion. How can robots learn to satisfy these constraints while improving task efficiency? We investigate this question through energy-constrained coverage, where a robot seeks to cover a workspace with minimal travel without exhausting its energy away from a charger. We find that high average feasibility-prediction accuracy alone is insufficient: the task policy can disproportionately select targets incorrectly predicted to be feasible. We propose Return-Aware Coverage PPO (RAC-PPO), which coordinates learned return-cost estimation, coverage-target selection, and recovery. The learned estimates support filtering candidate actions and checking the predicted energy reserve of the target proposed by the policy. When the predicted reserve becomes small, a persistent recovery controller maintains return behavior until recharging. The policy learns to advance coverage while reducing unnecessary travel and unproductive recharge cycles. Experiments on ProcTHOR-derived floorplans show that reward-driven PPO without a feasibility mechanism does not reliably satisfy the energy constraint. RAC-PPO achieves 93.92% mission success and 95.89% coverage, with only 6.08% constraint violations. Its normalized-distance score is approximately 25% lower than that of TSP-Split planning. These findings highlight the importance of coordinating feasibility learning, policy selection, and recovery when learning efficient behavior under hard constraints.

1 INTRODUCTION

How can robots learn to satisfy hard constraints while improving task efficiency? In long-horizon missions, minimizing a weighted combination of task costs and constraint penalties can favor constraint-violating actions when improvements in other objectives outweigh the penalties. A hard-constraint violation, however, can make the remaining mission infeasible regardless of the reward accumulated beforehand. Recent work in safe reinforcement learning addresses this distinction through shielding, feasible-set, and reachability-based formulations that reason about constraint satisfaction throughout execution (Ajdarów et al., 2023; Fan et al., 2025; Ganai et al., 2023). However, overly conservative constraint enforcement can reduce task performance (Ganai et al., 2023), highlighting the need to preserve feasibility without unnecessarily restricting useful task progress.

We study this question through energy-constrained robotic coverage, where a robot seeks to service all required locations with minimal total travel (Tang & Ma, 2024) while retaining enough energy to reach a charger. Visit order and recharge decisions affect both travel cost and subsequent feasibility. Under our energy model, analytically computable return costs provide ground truth for studying how learned feasibility estimates affect mission completion and travel efficiency.

A central challenge is that a policy optimizing task performance can preferentially select actions that a learned feasibility model incorrectly accepts. A return-cost predictor may be accurate over most candidate targets, yet underestimate the return requirement for a target preferred by the coverage policy. If that target leaves little predicted energy reserve, even a small underestimate can leave the robot unable to reach a charger. High average prediction accuracy therefore does not establish

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

constraint satisfaction during execution: the consequences of prediction errors depend on which actions the policy selects and how much reserve those actions leave.

A second challenge is to coordinate coverage and recovery for efficient mission completion. Rejecting risky targets can reduce violations, but stopping leaves coverage unfinished, and repeated returns without progress waste travel. The robot must interrupt coverage while it can still reach a charger and sustain the return until recharging; resuming coverage prematurely can jeopardize the return.

To address these challenges, we propose *Return-Aware Coverage PPO (RAC-PPO)*. A spatial PPO policy selects targets, while a separately trained model predicts the energy required to return from each candidate to a charger. These estimates support candidate filtering and a reserve check on the proposed target. When that reserve is small, or no candidate offers sufficient predicted reserve, the controller initiates a return phase that persists until recharging, after which coverage resumes.

Our main contributions are threefold. **First**, we construct an energy-constrained coverage benchmark from ProcTHOR floorplans for studying hard-constraint learning in long-horizon robotic tasks. The benchmark varies energy budgets, charger placements, and motion models, using analytically computable return costs as ground truth for learning and evaluating feasibility. **Second**, we propose RAC-PPO, which couples a learned return-cost model with a coverage policy trained to reduce travel cost. Learned return estimates support candidate admission, proposal checking, and persistent recovery until recharging. **Third**, we analyze how prediction errors interact with policy selection and use ablation studies to assess how feasibility checking and recovery affect constraint violations and mission completion. These experiments characterize the benefits of this coordination and the remaining failures under tight energy budgets.

2 Related Work

Coverage path planning seeks to reduce unnecessary travel while maintaining high coverage efficiency. Recent work has approached this problem through both combinatorial optimization and reinforcement learning. Local-search methods improve multi-robot coverage makespan by directly refining coverage paths Tang & Ma (2024), while learning-based approaches enable online coverage in unknown environments through compact spatial representations and coverage-oriented rewards Jonnarth et al. (2024). In related navigation tasks, PPO-based frameworks have been evaluated for goal seeking and collision avoidance through transfer from simulation to physical environments Cheng et al. (2024). For battery-powered robots, efficient coverage must also account for recharging requirements. Earlier work addresses coverage with repeated recharging through approximation algorithms on polygonal grids Wei & Isler (2018). More recently, an ergodic search with an energy-aware filter checks whether sufficient energy remains to return to a charging station Naveed et al. (2024).

Constrained reinforcement learning provides several mechanisms for handling such requirements. Action-constrained methods restrict infeasible decisions through sampling or augmented decision processes Hung et al. (2025), while robust constrained optimization addresses violations caused by model mismatch Sun et al. (2024). Complementary formulations study instantaneous hard constraints on states and actions Shi et al. (2023) and anytime budget constraints that must hold throughout execution McMahan & Zhu (2024). Other work focuses on improving constraint estimation, including finite-horizon gradient-based methods Dai et al. (2024) and extreme-value objectives for rare but severe violations Gao et al. (2025).

Research on safety-aware representations and reachability examines how feasibility predictions can guide decisions. Learned safety representations improve constraint estimation Cen et al. (2024); Mani et al. (2025), while reachability and action-masking methods suppress decisions likely to enter unsafe states Yu et al. (2022); Fan et al. (2025). Extensions of ATACOM incorporate learned constraints to address long-term safety and uncertainty Günster et al. (2025). Complementary work on energy-efficient navigation constructs energy-cost maps through aerial–ground collaboration Wei & Isler (2019) and combines aerial terrain information with online ground measurements using Gaussian process regression Wei & Isler (2020). RAC-PPO uses calibrated return-cost estimates to assess both candidate feasibility and the reserve of policy-selected targets.

Once continued coverage is no longer feasible, recovery becomes critical. Shielding and verified-control methods maintain safety through local planning or finite-horizon reachability analysis Banerjee et al. (2024); Wu et al. (2024). Probabilistic shielding further provides formal safety guarantees by

augmenting the decision process and restricting available actions when the safety dynamics are known Belardinelli et al. (2025). In our energy-constrained coverage setting, recovery must maintain the return objective across successive decisions until the robot reaches the charger. RAC-PPO therefore introduces a persistent return mode that remains active until recharging.

3 Problem Formulation

3.1 Coverage Task

We consider a single robot covering a known, static workspace represented by a floorplan with fixed charging locations. We discretize the workspace into grid cells of side length h and construct a grid graph $G = (\mathcal{F}, \mathcal{E}, w)$. Each vertex in $\mathcal{F}$ represents the center of a traversable cell, and each edge in $\mathcal{E}$ represents a permitted transition between neighboring cells, weighted by the distance between their centers. Let $\mathcal{C} \subseteq \mathcal{F}$ denote the charging locations and $d_G(u, v)$ the shortest-path distance between u and v in G.

At decision step t, the robot at $p_t \in \mathcal{F}$ selects an unserviced cell or a charging location as its next target a_t and follows a shortest path in G to reach it. Let $K_t \subseteq \mathcal{F}$ denote the serviced locations. Upon successfully reaching the selected target, $p_{t+1} = a_t$ and $K_{t+1} = K_t \cup \{a_t\}$. Intermediate cells traversed along the path are not automatically marked as serviced.

3.2 Energy Constraint and Objective

We assume that energy consumption is uniform and proportional to travel distance, with the consumption per unit distance normalized to one. Let $B > 0$ denote battery capacity and $E_t \in [0, B]$ the remaining energy. The robot starts at a charging location $p_0 \in \mathcal{C}$ with $E_0 = B$ and $K_0 = \{p_0\}$. After reaching target a_t, the battery is restored to $E_{t+1} = B$ if $a_t \in \mathcal{C}$; otherwise, $E_{t+1} = E_t - d_G(p_t, a_t)$. In this paper, we do not consider the cost with charging time.

To avoid becoming stranded, the robot must retain sufficient energy to reach a charging location after visiting each target. The minimum return energy from location v is

$$R^*(v) = \min_{c \in \mathcal{C}} d_G(v, c), \tag{1}$$

and the corresponding energy reserve for selecting target a is

$$m_t(a) = E_t - d_G(p_t, a) - R^*(a). \tag{2}$$

The objective is to service all traversable cells and return to a charging location with the lowest possible total travel distance:

$$\begin{aligned} \min_{\pi} \quad & \mathbb{E}_\pi \left[\sum_{t=0}^{T-1} d_G(p_t, a_t) \right] \\ \text{subject to} \quad & K_T = \mathcal{F}, \quad p_T \in \mathcal{C}, \quad T < \infty, \\ & m_t(a_t) \geq 0, \quad t = 0, \ldots, T-1. \end{aligned} \tag{3}$$

Here, π is the target-selection policy and T is the mission completion step.

4 Methodology

In this section, we present RAC-PPO. The method coordinates learned return-cost estimation, coverage target selection, and recovery to pursue efficient coverage under the energy constraint. Figure 1 illustrates the overall design. The return-cost model predicts the energy required to reach a charger from each candidate location (Section 4.1). Feasibility checks based on these learned estimates identify the candidates available to the coverage policy, which proposes targets to advance coverage efficiently (Section 4.2). Additional reserve checks determine whether to continue policy-controlled execution or initiate recovery. The recovery controller directs the robot toward charging and retains control until recharging, after which coverage resumes (Section 4.3). Finally, we describe the reward and training procedure used to learn the coverage policy within this decision process (Section 4.4).

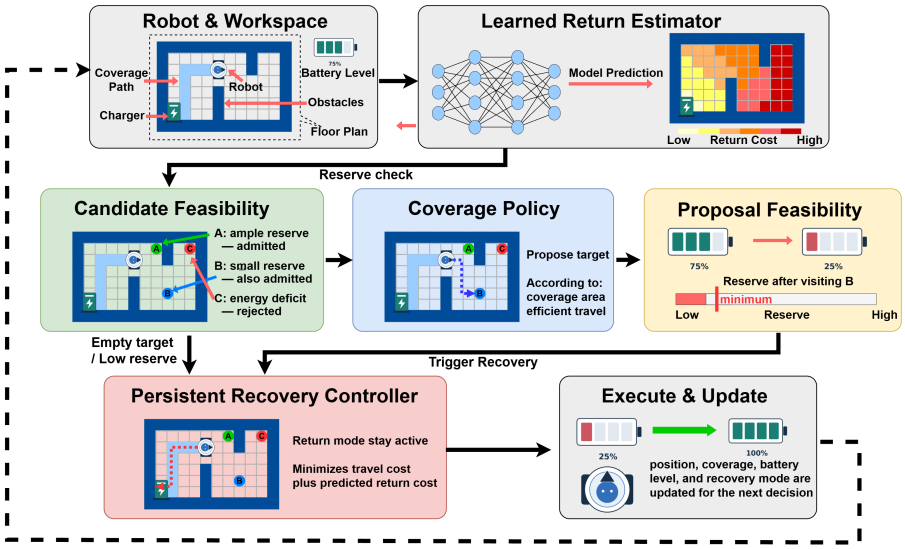


Figure 1: Overview of RAC-PPO. A learned model predicts the energy required to reach a charger from each candidate target. These estimates guide candidate admission for a PPO policy trained to complete coverage with low travel cost. Reserve checks on the admitted candidates and the policy's proposed target determine whether to continue coverage or initiate recovery. The recovery controller directs the robot toward a charger and retains control until recharging, after which coverage resumes.

4.1 Learned Return-Cost Estimation

To assess whether visiting a target leaves enough energy to reach a charger, we learn a model that predicts the return energy required from each traversable cell. These continuous estimates allow the controller to evaluate the energy reserve of coverage targets and compare recovery options.

For training instance j, the input X_j encodes free space, charger locations, and motion connectivity. A convolutional network f_ϕ produces a nonnegative normalized return-cost prediction at each cell. For each traversable cell $v \in \mathcal{F}_j$, the supervision target is $y_{jv} = R_j^*(v)/B_j$, where R_j^* is the exact return cost in equation 1 and B_j is the battery capacity for instance j. We train the model using the mean prediction loss over the traversable cells of each instance:

$$\mathcal{L}_{\text{ret}} = \frac{1}{|\mathcal{F}_j|} \sum_{v \in \mathcal{F}_j} \ell(f_\phi(X_j)_v - y_{jv}) \,, \tag{4}$$

where ℓ is the smooth-L_1 loss.

Underestimating return energy can admit a target that leaves the robot unable to reach a charger. To reduce this risk, we adjust the predictions using residuals from a separate set of training maps that are not used to fit the network. Let q_α denote the empirical α-quantile of the residuals $y_{jv} - f_\phi(X_j)_v$ on these calibration maps, with α chosen near one. The adjusted return-cost estimate is

$$\widehat{R}_j(v) = B_j \left[f_\phi(X_j)_v + q_\alpha \right]_+ , \tag{5}$$

where $[x]_+ = \max\{0, x\}$. This correction accounts for underestimation observed in the calibration data, but it does not guarantee feasible return for every selected target.

After fitting and calibration, we keep the estimator and correction fixed during coverage-policy training and evaluation. For each static scenario, we compute the return estimates once and reuse them throughout the episode. The following subsections explain how these estimates guide target selection and recovery. Network architecture and calibration settings are reported in Appendix B.3.

4.2 Coverage Target Selection

We use the learned return-cost estimates to identify targets that appear feasible under the current energy budget. The coverage policy then selects among these targets with the objective of completing

the mission with the lowest possible total travel cost. Because each visit changes the remaining coverage task, the policy must account for subsequent travel as well as immediate progress.

At decision step t, the candidate set contains unserviced cells and eligible charging locations:

$$\mathcal{A}_t = (\mathcal{F} \setminus K_t) \cup \mathcal{C}_t, \tag{6}$$

where $\mathcal{C}_t \subseteq \mathcal{C}$ contains the charging locations available for selection. A charger at the robot's current position is excluded while coverage remains incomplete.

For each candidate $a \in \mathcal{A}_t$, we combine its learned return requirement with the outbound travel cost and remaining energy. The predicted energy reserve after visiting the target and returning to a charger is

$$\widehat{m}_t(a) = E_t - d_G(p_t, a) - \widehat{R}(a). \tag{7}$$

We admit candidates with nonnegative predicted reserve:

$$\mathcal{S}_t = \{a \in \mathcal{A}_t : \widehat{m}_t(a) \geq 0\}. \tag{8}$$

This feasibility check based on learned return estimates determines the targets available to the coverage policy.

To choose among the admitted targets, the policy receives an observation o_t containing workspace geometry, current coverage, robot and charger locations, candidate descriptors, and global mission state. Candidate descriptors include outbound travel cost and local uncovered structure. Global mission information includes remaining energy, coverage progress, and accumulated travel. The observation excludes exact return costs, exact reserves, and exact-feasibility labels.

A spatial actor combines local and global map features with candidate descriptors and mission-state features to assign each admitted target a score $f_\theta(o_t, a)$. We apply the same scoring network to every target, allowing the policy to accommodate the changing candidate set. For $\mathcal{S}_t \neq \emptyset$, the target-selection distribution is

$$\pi_\theta(a \mid o_t, \mathcal{S}_t) = \frac{\exp f_\theta(o_t, a)}{\sum_{a' \in \mathcal{S}_t} \exp f_\theta(o_t, a')}, \qquad a \in \mathcal{S}_t. \tag{9}$$

When the controller requests a proposal, we sample a target a_t^π from this distribution during training and select the highest-scoring target during evaluation. The reserve checks introduced in Section 4.3 determine when to request a proposal and whether to execute it or initiate recovery. The policy reward and optimization procedure are described in Section 4.4.

4.3 Reserve Checks and Persistent Recovery

A target may pass the learned feasibility check even though visiting it would leave insufficient energy to reach a charger. This can happen when the model underestimates the return requirement, particularly when the predicted reserve is small. We therefore apply additional reserve checks before executing the policy's proposal. When these checks trigger recovery, the controller directs the robot toward a charger and retains control until recharging.

Let $\delta(a) = R^*(a) - \widehat{R}(a)$ denote the return-cost prediction error, with positive values indicating underestimation. The exact reserve is then $m_t(a) = \widehat{m}_t(a) - \delta(a)$, so an admitted target violates the energy constraint when $\delta(a) > \widehat{m}_t(a)$. Thus, the consequence of a prediction error depends on the reserve of the target selected by the policy. We use this relationship to motivate two reserve checks based on the learned estimates.

While recovery is inactive, we first determine whether any admitted target offers sufficient predicted reserve. An empty admitted set $\mathcal{S}_t$ activates recovery immediately. For a nonempty set, recovery is activated when

$$\max_{a \in \mathcal{S}_t} \frac{\widehat{m}_t(a)}{B} < \tau_{\text{set}}, \tag{10}$$

where $\tau_{\text{set}} > 0$ is the set-level reserve threshold.

If this check passes, the coverage policy proposes a target $a_t^\pi \in \mathcal{S}_t$. A high-reserve alternative may exist even when the policy prefers a target with little reserve. We therefore check the proposal itself

Algorithm 1 RAC-PPO decision step

Require: Current state (p_t, E_t, K_t), recovery state b, policy π_θ, and learned return estimates $\widehat{R}$

Construct $\mathcal{A}_t$ using equation 6
if $\mathcal{A}_t = \emptyset$ **then**
    Terminate the episode with no available candidate
end if
Compute $\widehat{m}_t$ and $\mathcal{S}_t$ using equation 7 and equation 8
if $b = 1$, $p_t \in \mathcal{C}$, and $E_t/B \geq \eta_{\text{exit}}$ **then**
    $b \leftarrow 0$
end if
if $b = 0$ **then**
    if $\mathcal{S}_t = \emptyset$ **then**
        $b \leftarrow 1$
    else if the set-level condition equation 10 holds **then**
        $b \leftarrow 1$
    else
        Obtain proposal a_t^π from the coverage policy
        if the proposal condition equation 11 holds **then**
            $b \leftarrow 1$
        end if
    end if
end if
if $b = 1$ **then**
    if $\mathcal{C}_t \neq \emptyset$ **then**
        Select a_t from $\mathcal{C}_t$ with the largest predicted reserve
    else
        Select a_t using equation 12
    end if
else
    $a_t \leftarrow a_t^\pi$
end if
Execute a_t and update position, energy, and coverage
Retain b for the next decision

and activate recovery when

$$\frac{\widehat{m}_t(a_t^\pi)}{B} < \tau_\pi, \tag{11}$$

where $\tau_\pi > 0$ is the proposal-level reserve threshold. Otherwise, the robot executes a_t^π.

To maintain return behavior until recharging, we use a binary recovery state b, initialized to zero. Either reserve check can set $b = 1$. While recovery is active, the controller selects an available charging target with the largest predicted reserve. If no charging target is available, it selects

$$a_t^{\text{ret}} = \arg\min_{a \in \mathcal{A}_t} \left[d_G(p_t, a) + \widehat{R}(a) \right]. \tag{12}$$

This choice minimizes the estimated energy required to visit the target and subsequently reach a charger. Recovery selection uses the full candidate set $\mathcal{A}_t$, allowing the controller to act when $\mathcal{S}_t$ is empty. The selected recovery action is not guaranteed to be feasible.

The recovery state remains active until $p_t \in \mathcal{C}$ and $E_t/B \geq \eta_{\text{exit}}$, where η_{exit} is the recharge-exit threshold. The controller then clears the recovery state and applies the reserve checks before requesting another coverage proposal. Under our full-recharge model, an affordable charger arrival satisfies this exit condition.

Note that both the reserve checks and recovery decisions use learned return estimates. We study this design as a step toward handling more complex constraints for which exact feasibility calculations are unavailable. In the present benchmark, we compute exact reserves after action selection to evaluate constraint violations. Algorithm 1 summarizes the complete decision process. See Appendix B.3 for

implementation details, including threshold values and recovery tie-breaking rules. We next explain how we train the coverage policy within this decision process.

4.4 Training the Coverage Policy

The last component is the training procedure for the coverage policy. We train the policy to complete the mission with the lowest possible travel cost, using the learned feasibility checks and recovery controller described above. The return-cost estimator and controller settings remain fixed during policy optimization.

At decision step t, we use the reward

$$\begin{aligned} r_t = \lambda_{\mathrm{cov}} \frac{\Delta k_t}{N} - \lambda_{\mathrm{dist}} \frac{\ell_t}{Nh} + \lambda_{\mathrm{succ}} \mathbf{1}[\text{mission success}] \\ - \lambda_{\mathrm{fail}} \mathbf{1}[\text{terminal hard failure}] - \lambda_{\mathrm{cycle}} z_t, \end{aligned} \tag{13}$$

where $\Delta k_t = |K_{t+1}| - |K_t|$ is the number of newly serviced cells, $N = |\mathcal{F}|$ is the total number of coverable cells, ℓ_t is the executed travel distance, and h is the grid resolution. The coverage term provides feedback as the mission progresses, and the distance penalty encourages shorter routes. The terminal terms reward successful completion and penalize hard failure. To discourage repeated charging without coverage progress, z_t equals one for a charger revisit with no newly serviced cells since the preceding charger visit, and zero otherwise. The coefficients λ control the contributions of these terms.

During training, the policy samples targets from the distribution in equation 9. The reserve checks determine whether each proposal is executed or replaced by a recovery action. For actor–critic optimization, we retain only transitions in which the policy's proposed action is executed. Diverted proposals and controller-selected recovery actions are excluded from these updates.

Let $t_0, \ldots, t_{J-1}$ index the retained transitions in a collected batch, and let d_{t_j} indicate episode termination at transition t_j. We initialize $\widehat{G}_J = 0$ and compute task-return targets backward as $\widehat{G}_j = r_{t_j} + \gamma(1 - d_{t_j})\widehat{G}_{j+1}$, where γ is the discount factor. This calculation uses the rewards of retained policy transitions and does not include rewards from intervening recovery actions.

The critic $V_\theta(o_t)$ is trained to predict these task-return targets. Using the parameters θ_{old} from data collection, we compute advantages as $\widehat{A}_j = \widehat{G}_j - V_{\theta_{\mathrm{old}}}(o_{t_j})$ and standardize them within each batch. We optimize the actor–critic using PPO's clipped policy objective, a squared-error value loss, and entropy regularization. The policy objective uses the advantages to update target preferences, while entropy regularization encourages exploration among admitted targets. See Appendix B.2 for implementation details.

5 Experiments

In this section, we evaluate constraint satisfaction, mission completion, and travel efficiency in energy-constrained coverage tasks. After describing the experimental setup, we examine whether reward-driven PPO satisfies the energy constraint and whether PPO tends to select targets incorrectly predicted to be feasible. We then compare RAC-PPO with TSP-Split planning to assess travel efficiency. We report the recovery-strategy comparison and additional component ablations in Appendix C, together with results across energy budgets and charger placements.

5.1 Benchmark and Evaluation Scenarios

We construct energy-constrained coverage tasks with different workspace geometries, charger placements, and energy budgets. The ProcTHOR floorplan data Deitke et al. (2022) contain 476 training maps and 98 validation maps after size filtering and duplicate removal. We discretize each floorplan at $h = 0.5\,\mathrm{m}$. We test two motion patterns: four-connected motion, which allows horizontal and vertical moves, and eight-connected motion, which also allows diagonal moves.

To test how energy-budget tightness affects performance, we set $B = 2\beta D$, where $D = \max_{v \in \mathcal{F}} R^*(v)$ and $\beta \in \{1.1, 1.5, 2.0\}$. For a single charger, $2D$ is the round-trip cost of visiting the farthest traversable cell. Larger β provides more energy reserve.

Table 1: Mission performance evaluated by success rate, coverage ratio across all tests (including failed ones), and violations.

Method	Success(%)	Coverage(%)	Violations(%)
Plain PPO	2.04	16.63	91.50
Exact-mask PPO	100.00	100.00	0.00
RAC-PPO (ours)	93.92	95.89	6.08

5.2 Evaluation Metrics

We measure constraint satisfaction and mission completion separately because avoiding energy violations does not necessarily imply successful task execution. *Mission success* requires complete service and final arrival at a charger, whereas a *violation episode* contains at least one negative exact reserve or unaffordable target selection. To capture partial task progress, we also report *mean coverage*, computed by averaging the ratio of serviced cells to all coverable cells in each episode over all evaluation episodes. This includes failed missions, whose unserviced cells remain in the denominator.

We assess coverage efficiency using the *normalized-distance metric*: the total distance traveled across *successful* missions divided by the total number of coverable cells in those missions. Lower values indicate less travel per serviced cell. Restricting this metric to successful missions prevents early termination from being mistaken for efficient coverage.

See Appendix B for implementation settings and additional evaluation definitions.

5.3 Constraint Satisfaction with Reward-Driven PPO

We first report mission success, coverage, and energy-constraint violations for RAC-PPO, together with Plain PPO and Exact-mask PPO as references. These results characterize task completion and constraint satisfaction under reward-driven decisions, feasibility checks based on learned return estimates, and exact return-feasibility checks.

Plain PPO learns coverage and charging decisions from the task reward without a return-feasibility filter, where a large negative reward is assigned when the energy constraint is violated. Exact-mask PPO uses exact shortest-path return costs to exclude targets that would leave insufficient energy to reach a charger.

As shown in Table 1, the low success rate and frequent violations of Plain PPO highlight the difficulty of relying on reward feedback alone to learn behavior that satisfies hard constraints. Plain PPO also has episodes that time out without violating the energy constraint, so its success and violation rates need not sum to 100%. Exact-mask PPO maintains feasibility by excluding unsafe targets before execution. RAC-PPO achieves high mission completion and coverage without an exact return-feasibility oracle at deployment, demonstrating that learned return estimates can support successful task execution when coupled with policy selection and recovery.

5.4 Feasibility Errors on PPO-Selected Targets

Candidate-level accuracy may not reflect the prediction errors encountered during execution. We therefore examine whether PPO preferentially selects targets for which the return-cost model is less reliable, comparing all candidates, admitted candidates, and PPO-selected proposals. A *false-feasible* target passes the learned check but has negative exact reserve.

As shown in Table 2, the false-feasible rate increases from 1.3% among all candidates to 2.1% among admitted candidates and 4.6% among PPO-selected proposals. Selected targets also have smaller predicted reserves and are more often near the feasibility boundary.

These results show that PPO tends to select targets with smaller energy margins, where return-cost underestimation is more consequential. High candidate-level accuracy can therefore conceal the prediction errors most relevant to the policy's decisions.

Table 2: Return-model behavior under different action distributions. All reserve quantities are normalized by battery capacity.

Actions	MAE ↓	Underest. ↓	False-feas. ↓	Mean reserve ↑	Near-bound. ↓
All candidates	0.036	8.7%	1.3%	0.184	6.4%
Admitted candidates	0.038	9.5%	2.1%	0.121	10.8%
PPO-selected proposals	0.041	12.6%	4.6%	0.071	18.9%

Table 3: Efficiency comparison using only successful trials for each method. RAC-PPO achieves a lower normalized distance than the TSP-Split expert policy, outperforming it in coverage efficiency.

Method	Success(%) ↑	Coverage(%) ↑	Violations(%) ↓	Normalized distance ↓
TSP-Split	100.00	100.00	0.00	1.54
RAC-PPO (ours)	93.92	95.89	6.08	1.16

5.5 Coverage Efficiency Compared with TSP-Split Planning

High mission completion does not establish efficient coverage. We therefore compare RAC-PPO's travel cost with a TSP-Split Li et al. (1992); Nagarajan & Ravi (2012); Wei & Isler (2018) based method, which has a competitive approximation bounds, to test whether our learned policy produces coverage routes comparable in efficiency to those of an expert planner.

As shown in Table 3, RAC-PPO achieves high mission completion and coverage while recording a lower normalized-distance score than TSP-Split. These results indicate that our learned policy achieves coverage comparable to the expert planner.

RAC-PPO's normalized-distance score, averaged over two policy evaluations, is approximately 25% lower than TSP-Split's (Table 3). TSP-Split, however, completes every mission without violations using exact return costs. The distance metric summarizes each method's successful missions.

Additional experiments. We compare conservative stopping, one-step recovery, and persistent recovery in Appendix C.1. Persistent recovery achieves higher mission completion than conservative stopping and fewer observed violations than one-step recovery. We examine the effects of proposal checking and recharge-cycle penalization in Appendix C.2. We also evaluate RAC-PPO across energy budgets and charger placements in Appendix C.3, where the remaining violations occur only at the tightest tested budget, $\beta = 1.1$.

6 Conclusion and Future Work

In this paper, we studied learning efficient behavior under hard constraints through energy-constrained coverage. RAC-PPO coordinates learned return-cost estimation, coverage-target selection, and persistent recovery, achieving approximately 94% mission success and 96% coverage with a lower normalized-distance score than TSP-Split.

Our current method addresses a single resource constraint using hand-designed checks on learned costs. We will extend the study to interacting constraints, such as energy limits and task deadlines, and investigate learning action-feasibility checks directly to reduce reliance on problem-specific rules and test generalization across planning problems.

References

Michal Ajdarów, Šimon Brlej, and Petr Novotný. Shielding in resource-constrained goal pomdps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 14674–14682, 2023. doi: 10.1609/aaai.v37i12.26715.

Arko Banerjee, Kia Rahmani, Joydeep Biswas, and Isil Dillig. Dynamic model predictive shielding for provably safe reinforcement learning. In *Advances in Neural Information Process-*

ing Systems, volume 37, pp. 100131–100159. Curran Associates, Inc., 2024. doi: 10.52202/079017-3177. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/b589d92785e39486e978fa273d0dc343-Paper-Conference.pdf.

Francesco Belardinelli, Alexander W Goodall, et al. Probabilistic shielding for safe reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 16091–16099, 2025.

Zhepeng Cen, Yihang Yao, Zuxin Liu, and Ding Zhao. Feasibility consistent representation learning for safe reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 6002–6019. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/cen24b.html.

Wen-Chung Cheng, Zhen Ni, Xiangnan Zhong, and Minghan Wei. Autonomous robot goal seeking and collision avoidance in the physical world: An automated learning and evaluation framework based on the ppo method. *Applied Sciences*, 14(23), 2024. ISSN 2076-3417. doi: 10.3390/app142311020. URL https://www.mdpi.com/2076-3417/14/23/11020.

Juntao Dai, Yaodong Yang, Qian Zheng, and Gang Pan. Safe reinforcement learning using finite-horizon gradient-based estimation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 9872–9903. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/dai24d.html.

Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in neural information processing systems*, 35:5982–5994, 2022.

Ke Fan, Jinpeng Zhang, Xuefeng Zhang, Yunze Wu, Jingyu Cao, Yuan Zhou, and Jianzhu Ma. Safety-polarized and prioritized reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 15862–15886. PMLR, 2025.

Milan Ganai, Zheng Gong, Chenning Yu, Sylvia Herbert, and Sicun Gao. Iterative reachability estimation for safe reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 69764–69797, 2023.

Shiqing Gao, Yihang Zhou, Shuai Shao, Haoyu Luo, Yiheng Bing, Jiaxin Ding, Luoyi Fu, and Xinbing Wang. Extreme value policy optimization for safe reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 18772–18793. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/gao25v.html.

Jonas Günster, Puze Liu, Jan Peters, and Davide Tateo. Handling long-term safety and uncertainty in safe reinforcement learning. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4670–4697. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/gunster25a.html.

Wei Hung, Shao-Hua Sun, and Ping-Chun Hsieh. Efficient action-constrained reinforcement learning via acceptance-rejection method and augmented MDPs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=AgMpK7z4bz.

Arvi Jonnarth, Jie Zhao, and Michael Felsberg. Learning coverage paths in unknown environments with deep reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22491–22508. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/jonnarth24a.html.

Chung-Lun Li, David Simchi-Levi, and Martin Desrochers. On the distance constrained vehicle routing problem. *Operations research*, 40(4):790–799, 1992.

Kaustubh Mani, Vincent Mai, Charlie Gauthier, Annie S Chen, Samer B. Nashed, and Liam Paull. Safety representations for safer policy learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=gJG4IPwg6l.

Jeremy McMahan and Xiaojin Zhu. Anytime-constrained reinforcement learning. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 4321–4329. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/mcmahan24a.html.

Viswanath Nagarajan and R Ravi. Approximation algorithms for distance constrained vehicle routing problems. *Networks*, 59(2):209–214, 2012.

Kaleb Ben Naveed, Devansh Agrawal, Christopher Vermillion, and Dimitra Panagou. Eclares: Energy-aware clarity-driven ergodic search. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14326–14332. IEEE, 2024.

Ming Shi, Yingbin Liang, and Ness Shroff. A near-optimal algorithm for safe reinforcement learning under instantaneous hard constraints. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 31243–31268. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/shi23c.html.

Zhongchang Sun, Sihong He, Fei Miao, and Shaofeng Zou. Constrained reinforcement learning under model mismatch. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 47017–47032. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/sun24d.html.

Jingtao Tang and Hang Ma. Large-scale multi-robot coverage path planning via local search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17567–17574, 2024. doi: 10.1609/aaai.v38i16.29707.

Minghan Wei and Volkan Isler. Coverage path planning under the energy constraint. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 368–373. IEEE, 2018.

Minghan Wei and Volkan Isler. Air to ground collaboration for energy-efficient path planning for ground robots. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1949–1954, 2019. doi: 10.1109/IROS40897.2019.8967963.

Minghan Wei and Volkan Isler. Energy-efficient path planning for ground robots by and combining air and ground measurements. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 766–775. PMLR, 30 Oct–01 Nov 2020. URL https://proceedings.mlr.press/v100/wei20a.html.

Junlin Wu, Huan Zhang, and Yevgeniy Vorobeychik. Verified safe reinforcement learning for neural network dynamic models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 117762–117783. Curran Associates, Inc., 2024. doi: 10.52202/079017-3740. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/d5a25f990ffa1cb759cc88916d7fbe96-Paper-Conference.pdf.

Dongjie Yu, Haitong Ma, Shengbo Li, and Jianyu Chen. Reachability constrained reinforcement learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 25636–25655. PMLR, 2022.

APPENDIX

A AI USE STATEMENT

In this work, AI models were used to assist with spelling and grammar checking, as well as minor text polishing. We also used them to assist the implementation and debug of the presented methods. All AI-assisted edits were reviewed and verified by the authors, who take full responsibility for the content, accuracy, and conclusions presented in this paper.

B IMPLEMENTATION AND EVALUATION DETAILS

We provide the workspace-construction, policy-training, return-estimation, and evaluation details omitted from the main experimental section.

B.1 WORKSPACE CONSTRUCTION AND EPISODE TERMINATION

We discretize each floorplan at a resolution of $h = 0.5\,\mathrm{m}$ and use a robot radius of $0.2\,\mathrm{m}$ in traversability preprocessing. We test four-connected and eight-connected motion. Horizontal and vertical moves cost h, while diagonal moves cost $\sqrt{2}h$. To prevent corner cutting, we allow a diagonal move only when the neighboring cells in both its horizontal and vertical directions are free. We compute shortest-path distances using Dijkstra's algorithm and reuse them within each static scenario.

The central charger layout uses the map's designated central charger. The peripheral layout places the charger at the traversable cell farthest from the central charger under the specified motion model. Training also includes two-charger layouts. We recompute the return-distance scale D and battery capacity B whenever charger placement or motion connectivity changes.

The robot starts fully charged at a charger, which is counted as serviced at initialization. Only successfully reached targets are marked as serviced; intermediate cells traversed along a path are not automatically serviced. Candidate targets are ordered deterministically. The charger at the robot's current location is excluded from the charger candidates while coverage remains incomplete. We do not filter candidates using exact return feasibility before applying the learned feasibility check.

Each episode ends upon mission completion, terminal failure, or a limit of $3N$ target-selection decisions, where $N = |\mathcal{F}|$ is the number of traversable cells. An episode that reaches this limit without completing the mission is counted as a timeout. Selecting a target with outbound cost $d_G(p_t, a_t) > E_t$ terminates the episode without moving the robot. For an affordable action, a negative exact reserve is recorded as a return-feasibility violation, but the action is not retroactively canceled.

B.2 POLICY ARCHITECTURE AND TRAINING CONFIGURATION

We hold the policy architecture, return estimator, and PPO optimization settings fixed across the RAC-PPO component comparisons to assess changes to proposal checking and recharge-cycle penalization. Table 4 summarizes the policy inputs.

The actor and critic share a spatial encoder. For each candidate, the actor combines its local spatial embedding, the globally pooled map embedding, its candidate embedding, and the global-state embedding. The critic combines the pooled map embedding, mean-pooled admitted-candidate embeddings, and the global-state embedding. Table 5 specifies the network structure.

We optimize the actor–critic parameters using AdamW, an adaptive gradient-based optimizer. Table 6 gives the optimization and reward settings. The final policy-training stage uses $25{,}000$ executed decisions, including recovery actions. This budget excludes policy initialization and return-estimator training.

During training, we sample policy proposals from the categorical distribution over admitted targets. During evaluation, we select the highest-scoring admitted target. Only policy-proposed actions that are actually executed contribute transitions to actor–critic updates. Diverted proposals and controller-selected recovery actions are excluded from these updates.

Table 4: Policy observations. Exact return costs, exact reserves, and exact-feasibility labels are excluded from the deployed policy input.

Input	Dimension	Contents
Spatial rasters	9 channels	Free space, obstacles, covered cells, uncovered cells, robot location, charger locations, visible candidate support, robot-distance field, and a zero-filled return-oracle channel.
Candidate descriptors	26 features per candidate	Coordinates and relative displacement, outbound travel cost, energy and coverage state, charger indicator, deterministic priority and map-size features, local uncovered density, uncovered-component size, graph degree, topology indicators, and charger-proximity features; four oracle-derived entries are fixed to zero.
Global mission state	9 scalars	Normalized remaining energy, coverage fraction, accumulated travel, recharge count, and decision count; four unused history-related entries are fixed to zero.

Table 5: Coverage-policy architecture.

Network component	Architecture or operation
Spatial encoder	Four 3×3 convolutions, with channels $9 \to 32 \to 64 \to 64 \to 128$.
Spatial activation	GELU after each convolution.
Candidate encoder	Two-layer MLP, $26 \to 128 \to 128$.
Global-state encoder	Maps 9 scalars to a 64-dimensional embedding.
Actor input	128 local spatial + 128 pooled spatial + 128 candidate + 64 global = 448 dimensions.
Actor head	$448 \to 192 \to 1$, producing one logit per candidate.
Critic input	128 pooled spatial + 128 pooled candidate + 64 global = 320 dimensions.
Critic head	$320 \to 192 \to 1$.
Candidate aggregation	Mean pooling over admitted candidates.
Action distribution	Categorical distribution normalized over admitted targets.

Table 6: PPO optimization and reward settings. The recharge-cycle penalty is enabled only in Cycle-Penalized PPO and RAC-PPO.

Parameter	Value
Optimizer	AdamW
Learning rate	3×10^{-4}
Discount factor γ	0.99
PPO clipping coefficient	0.20
Value-loss coefficient	0.50
Entropy coefficient	0.01
Gradient-norm clipping	0.50
Optimization passes per batch	4
Retained policy transitions per batch	256
Final-stage interaction budget	25,000 executed decisions
Coverage coefficient λ_{cov}	1.00
Travel coefficient λ_{dist}	0.02
Success coefficient λ_{succ}	1.00
Failure coefficient λ_{fail}	1.00
Recharge-cycle coefficient λ_{cycle}	0.05

Table 7: Return-estimator and controller configuration.

Component or parameter	Specification
Input	3 spatial channels: free space, chargers, and motion connectivity.
First convolution	$3 \to 32$ channels, 3×3 kernel.
Second convolution	$32 \to 32$ channels, 3×3 kernel.
Hidden activation	ReLU.
Output convolution	$32 \to 1$ channel, 1×1 kernel.
Output activation	Softplus.
Supervision target	Exact return cost divided by battery capacity, R^*/B.
Training loss	Smooth-L_1, evaluated on valid free cells.
Training protocol	Separate fitting, then frozen during PPO.
Calibration data	Training maps disjoint from field-fitting maps.
Upper residual quantile	0.995.
Calibration offset	Approximately 0.2442 in normalized return-cost units.
Set-level reserve threshold τ_{set}	0.03.
Proposal-level reserve threshold τ_π	0.01 when proposal checking is enabled.
Recharge-exit threshold η_{exit}	0.75.
Charging model	Instantaneous full recharge upon affordable charger arrival.

B.3 Return Estimation and Controller Configuration

We train the return-cost estimator separately from the coverage policy and keep it fixed during policy training and evaluation. Its input contains free-space, charger-location, and motion-connectivity channels. Two convolutional layers extract spatial features, and a final output layer produces a nonnegative normalized return-cost estimate at each cell. Table 7 gives the estimator and controller settings.

We fit the estimator using exact return costs from training maps. For one-sided calibration, we use the empirical 0.995 quantile of the underprediction residuals on training maps disjoint from those used for fitting. The resulting correction is approximately 0.2442 in normalized return-cost units. We freeze both the estimator and the correction during policy training and evaluation.

For the proposal-threshold study, we use separate sets of 54 fitting maps and 54 calibration maps from the training split. The fitting subset contains seven unsafe policy proposals. No candidate threshold satisfies the prespecified acceptance criterion, so $\tau_\pi = 0.01$ is retained as an exploratory setting rather than a calibrated safety bound.

We compute the calibrated return estimates once for each static scenario and reuse them throughout the episode. For recovery selection, the controller ranks candidates using

$$u_t(a) = \sigma\left(-\frac{10\widehat{m}_t(a)}{B}\right), \tag{14}$$

where σ is the logistic function. This score ranks predicted reserves; it is not a calibrated failure probability. Among charger candidates, the controller minimizes u_t, breaking ties by actor logit and then canonical candidate index. If no charger candidate is available, it minimizes outbound plus predicted return cost, with u_t, actor logit, and candidate index as successive tie-breakers.

B.4 Additional Evaluation Definitions

We distinguish prediction errors from executed constraint violations. A false-safe candidate, reported as false-feasible in Table 2, passes the learned feasibility check but has negative exact reserve. The prediction analysis considers all generated candidates, admitted candidates, and PPO-proposed targets separately. A proposal diverted by the controller can contribute to proposal-level prediction statistics, but it does not count as an executed violation. An infeasible recovery action contributes to the violation metric even when it was not admitted by the learned feasibility check. We evaluate exact reserve with a numerical tolerance of 10^{-9}.

Table 8: Comparison of recovery strategies on the same 98 validation scenarios. Success and violations show episode counts, with percentages in parentheses. Coverage is the mean terminal coverage.

Method	Success	Coverage (%)	Violations
Admission-and-Stop PPO	12/98 (12.24)	27.74	1/98 (1.02)
Reactive-Return PPO	29/98 (29.59)	50.11	52/98 (53.06)
Persistent-Return PPO	62/98 (63.27)	82.66	7/98 (7.14)

For the return-model comparison, normalized mean absolute error measures $|R^*(a) - \widehat{R}(a)|/B$. The underestimation rate is the fraction of targets for which $\widehat{R}(a) < R^*(a)$. Mean normalized reserve measures $\widehat{m}_t(a)/B$, and the near-boundary rate is the fraction satisfying $0 \leq \widehat{m}_t(a)/B < \tau_\pi$.

The PPO action fraction is the number of executed policy-selected actions divided by all executed actions. We record conservative stops separately from timeouts. An empty learned-admission set does not itself constitute a stop when the controller can initiate recovery.

C Additional Experimental Results

We report component ablations and performance across energy budgets and charger placements to examine how the method's design and task conditions affect mission completion and constraint violations.

C.1 Comparison of Recovery Strategies

We examine how different recovery strategies affect constraint satisfaction and mission completion when task decisions rely on learned return-cost estimates. We compare three PPO variants on the same 98 validation scenarios. *Admission-and-Stop PPO* terminates the episode when the learned feasibility check admits no task action. *Reactive-Return PPO* instead executes a one-step recovery action before returning control to PPO. *Persistent-Return PPO* maintains a recovery mode across successive decisions until the robot reaches a charger and recharges.

Table 8 shows that conservative stopping limits violations but leaves most missions incomplete. Admission-and-Stop PPO records only one violating episode, yet completes just 12 missions and reaches 27.74% mean coverage. Thus, a low violation rate alone does not establish successful execution of the constrained task. One-step recovery increases the number of completed missions to 29, but records violations in 52 episodes.

Persistent-Return PPO completes 62 missions and reaches 82.66% mean coverage, while reducing violating episodes from 52 to seven relative to Reactive-Return PPO. It therefore completes more missions than either alternative and records fewer violations than one-step recovery, although its violation count remains higher than that of conservative stopping. These results support maintaining recovery until recharge as a way to improve mission completion while limiting violations relative to one-step fallback. We examine the additional effects of proposal checking and recharge-cycle penalization separately in Appendix C.2.

C.2 Effects of Proposal Checking and Recharge-Cycle Penalization

We examine how checking the policy's proposed target and penalizing charger revisits without coverage progress affect mission completion and energy-constraint violations. We evaluate four variants on the same 98 scenarios, keeping the return-cost estimator, policy architecture, and PPO optimization settings fixed.

Early-Return PPO uses learned candidate admission and persistent recovery, with a set-level reserve threshold of $\tau_{\text{set}} = 0.03$. Proposal-Aware PPO additionally checks the reserve of the proposed target using $\tau_\pi = 0.01$. Cycle-Penalized PPO instead adds the recharge-cycle penalty with weight $\lambda_{\text{cycle}} = 0.05$, without proposal checking. RAC-PPO combines both additions.

Table 9: Effects of proposal checking and recharge-cycle penalization on the same 98 evaluation scenarios. Success and violations show episode counts, with percentages in parentheses. Coverage is the mean terminal coverage. PPO actions is the percentage of executed actions selected by the coverage policy.

Method	Success	Coverage (%)	Violations	PPO actions (%)
Early-Return PPO	64/98 (65.31)	92.78	5/98 (5.10)	95.31
Proposal-Aware PPO	64/98 (65.31)	96.29	4/98 (4.08)	87.72
Cycle-Penalized PPO	70/98 (71.43)	92.47	5/98 (5.10)	95.05
RAC-PPO	67/98 (68.37)	93.80	4/98 (4.08)	91.84

Table 10: RAC-PPO performance across energy budgets. All performance values are percentages averaged over the two trained policies. Each budget includes 96 scenarios per policy, spanning both charger placements and both motion patterns. The overall row includes all 288 scenarios per policy. Violations is the percentage of episodes containing at least one energy-constraint violation.

Battery multiplier β	Success	Coverage	Violations
1.1	81.77	87.67	18.23
1.5	100.00	100.00	0.00
2.0	100.00	100.00	0.00
Overall	93.92	95.89	6.08

Proposal checking. We compare Early-Return PPO with Proposal-Aware PPO to test whether checking the selected target provides a benefit beyond checking the available candidate set. Table 9 shows that proposal checking increases coverage and produces one fewer violating episode, while leaving the number of completed missions unchanged. The existence of an admitted target with sufficient reserve does not ensure that the policy selects it; checking the proposal addresses this distinction.

Recharge-cycle penalization. We compare Early-Return PPO with Cycle-Penalized PPO to test whether discouraging recharge without coverage progress improves mission completion. Adding the penalty increases the number of completed missions while leaving the violation count unchanged. This improvement is obtained through the policy-training reward, without changing the return-cost estimator or the recovery trigger.

Combined method. We evaluate RAC-PPO to examine the effect of using both additions together. RAC-PPO retains the lower violation count of Proposal-Aware PPO while completing more missions. Cycle-Penalized PPO achieves the highest mission success in this comparison, but records one additional violating episode relative to RAC-PPO.

C.3 Mission Performance Across Energy Budgets and Charger Placements

We vary battery capacity and charger placement to examine when RAC-PPO completes missions and when energy-constraint violations occur. For each of 24 validation maps, we test central and peripheral charger placements, battery multipliers $\beta \in \{1.1, 1.5, 2.0\}$, and four-connected and eight-connected motion. These combinations yield $24 \times 2 \times 3 \times 2 = 288$ scenarios for each evaluated policy. Both charger layouts contain a single charger. The 24 maps were used during method development, so this evaluation examines scenario variations on reused validation geometry.

We evaluate two trained RAC-PPO policies on the same scenarios and average their results. Table 10 reports performance at each energy budget, combining both charger placements and both motion patterns. Figure 2 shows the two policy evaluations separately.

The energy-budget breakdown shows that the remaining failures occur at the tightest setting, $\beta = 1.1$. At $\beta = 1.5$ and $\beta = 2.0$, both policies complete every evaluated mission with full coverage and no observed violations. With less energy available, return-cost underestimation can admit a target that

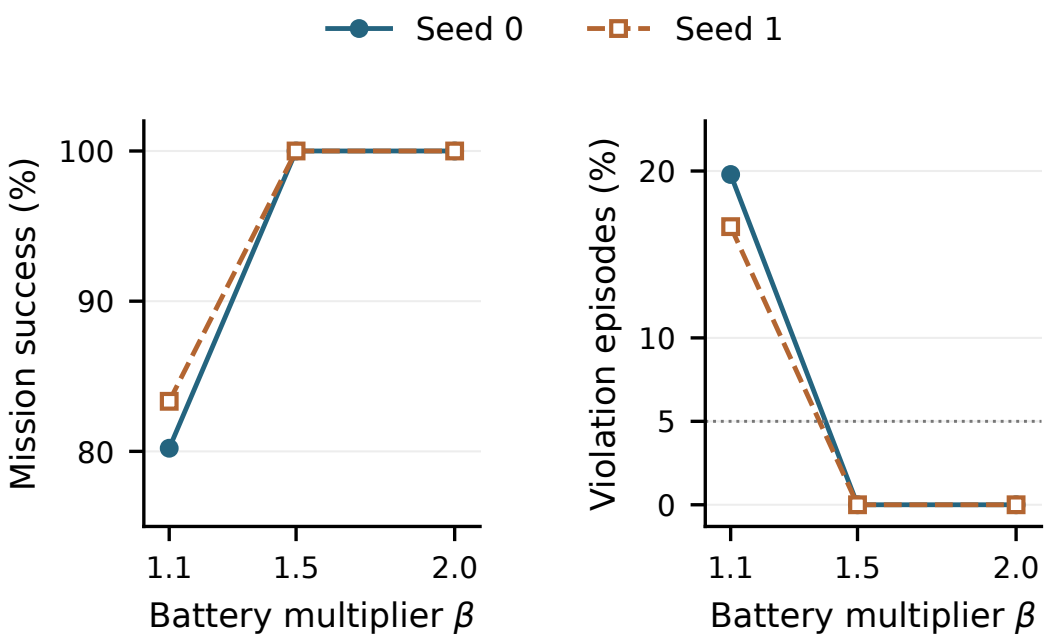


Figure 2: Mission success and energy-constraint violations across battery budgets. The curves show the two evaluated policies separately. Each point includes 96 scenarios covering both charger placements and both motion patterns.

leaves insufficient energy to reach a charger. The results therefore identify the tightest energy budget as the principal failure condition in this evaluation.