Multi-Agent Reinforcement Learning for Persistent Satellite Custody of Moving Ground Targets

David Schuster *

Posted Date: 1 June 2026

doi: 10.20944/preprints202606.0109.v1

Keywords: reinforcement learning; multi-agent RL; satellite custody; PPO; RLlib; PettingZoo; space domain awareness; ISR

 doi:10.20944/preprints202606.0109.v1

Article

Multi-Agent Reinforcement Learning for Persistent Satellite Custody of Moving Ground Targets

David Schuster

Geophysical Institute, University of Alaska Fairbanks; dschuster@alaska.edu; Tel.: +1-907-347-2604

Abstract

Maintaining persistent custody of dynamic ground targets using constellations of low Earth orbit (LEO) satellites is a critical capability for intelligence, surveillance, and reconnaissance (ISR) missions. Building upon our prior work using centralized PPO with `stable-baselines3`, this study presents an enhanced multi-agent formulation using PettingZoo `ParallelEnv` and Ray `RLlib` with a shared policy architecture. Key improvements include a larger effective field-of-view, slower target dynamics, richer per-agent observations incorporating tip density and velocity cues, and a refined reward structure that strongly incentivizes proactive tipping-and-cueing. The trained policy achieved 71.3% mean custody coverage over 500-step episodes, substantially outperforming random (28.6%) and greedy (38.1%) baselines. Analysis of handoff frequency and per-target performance demonstrates emergent cooperative behavior. These results highlight the value of modern multi-agent RL tooling for space domain awareness applications and provide a reproducible benchmark environment for future research.

Keywords: reinforcement learning; multi-agent RL; satellite custody; PPO; RLlib; PettingZoo; space domain awareness; ISR

1. Introduction

Maintaining persistent custody of moving ground targets from low Earth orbit (LEO) is a fundamental challenge in space-based intelligence, surveillance, and reconnaissance (ISR). As ground targets exhibit unpredictable motion and satellites follow fixed orbital paths with limited fields of view, continuous observation requires precise coordination of sensor tasking and timely handoffs between platforms. Traditional deterministic scheduling and rule-based heuristics often fail to adapt to stochastic target behavior and dynamic visibility windows, resulting in frequent custody gaps. This study builds upon our previous proof-of-concept work [1], which demonstrated the viability of centralized Proximal Policy Optimization (PPO) for coordinating four LEO satellites over two dynamic targets. Here we extend that foundation by reformulating the problem as a multi-agent environment using PettingZoo's `ParallelEnv` and training with Ray `RLlib` under a shared policy architecture. The present work incorporates enhanced observation features, refined reward structures emphasizing proactive tipping-and-cueing, and more realistic target dynamics to better approximate operational conditions. Persistent custody is increasingly critical as the space domain becomes more contested and congested. Modern ISR missions demand near-continuous tracking of high-value mobile targets for applications ranging from disaster response and border security to military operations. Manual or heuristic-based tasking scales poorly with constellation size and target count, creating a pressing need for autonomous, adaptive coordination systems. Reinforcement learning offers a promising pathway toward scalable, real-time decision-making that can generalize across varying orbital configurations and target behaviors. Improving custody performance directly translates to higher mission effectiveness and better utilization of expensive space assets.

Reinforcement learning has seen growing application in space domain awareness and satellite operations. Early work applied PPO to single-satellite agile Earth observation scheduling [2]. Subse-

 doi:10.20944/preprints202606.0109.v1

quent studies extended these methods to multi-agent scenarios, including sensor tasking for space situational awareness [3] and multi-satellite coordination [4].

Notable recent contributions include MARL approaches for sequential satellite assignment [5] and spatiotemporal-aware multi-agent frameworks for collaborative Earth observation [6]. Centralized training with shared policies has proven effective in domains with strong coordination requirements, while fully decentralized MARL approaches face challenges related to credit assignment and non-stationarity [7]. Despite these advances, applications specifically targeting persistent custody of moving ground targets remain limited, with most prior work focusing on point-target imaging or space object tracking rather than continuous tracking under realistic orbital dynamics. Our prior work [1] achieved $\sim$ 74% mean custody using a `stable-baselines3` centralized PPO agent. The current study examines whether a modern `RLlib` + `PettingZoo` implementation with enhanced coordination mechanisms can further improve performance and provide better insight into emergent satellite behaviors.

We hypothesize that (H1) the enhanced `RLlib` shared-policy agent will achieve significantly higher mean custody coverage than both random and greedy baselines, and (H2) explicit tipping-and-cueing incentives will produce measurable increases in successful handoffs and reduced custody gaps compared to simpler reward structures. An additional research question is if a more complex multi-agent model achieves superior custody to the original work's centralized model and if the additional effort required to build and implement the model is worth it for this problem. These hypotheses are tested through a controlled simulation environment using high-fidelity orbital propagation via the `Skyfield` library. The research design employs a shared-policy PPO architecture trained on a `ParallelEnv` formulation, enabling centralized training while maintaining a multi-agent interface. Performance is evaluated over multiple random seeds using standardized metrics (custody percentage, handoff frequency, and per-target coverage) and compared against heuristic baselines. This design allows direct attribution of performance gains to the improved observation space, reward shaping, and training configuration.

2. Method

2.1. Simulation Environment

We developed a custom multi-agent reinforcement learning environment called `SatelliteCustody-MultiEnv`, built upon PettingZoo's `ParallelEnv` interface and integrated with Ray RLlib for training. The environment simulates a constellation of four satellites in fixed low Earth orbits tasked with maintaining persistent custody over two independently moving ground targets.

Orbital mechanics are modeled with high fidelity using the Skyfield library [8]. Satellite positions are generated from simplified two-line element (TLE) sets with 45° inclination and 90° right ascension of the ascending node (RAAN) spacing. To ensure deterministic and computationally efficient simulation, satellite positions are pre-propagated for the entire episode at the beginning of each reset.

Each episode consists of 500 timesteps, where each timestep corresponds to 60 seconds of simulated real time (approximately 8.3 hours per episode). A representative excerpt from the satellite loading logic is shown below:

```
# Excerpt from _load_satellites()
for i in range(self.num_sats):
    raan = i * 90.0
    l1 = f"1 1000{i}U 25001A   25200.00000000  .00000000  00000-0  00000-0 0  999{i}"
    l2 = f"2 1000{i}  45.0000 {raan:8.4f} 0001000   0.0000   0.0000 14.50000000    0{i}"
    sats.append(EarthSatellite(l1, l2, f"SAT-{i}", self.ts))
```

The field-of-view threshold was set to 58.0° angular distance, striking a balance between realism and learnability.

2.2. Target Dynamics

Ground targets follow a stochastic random walk model on the Earth's surface. Target velocities were drawn from the following distributions (per timestep):

 doi:10.20944/preprints202606.0109.v1

- Latitude velocity: uniform distribution in $[-0.035, 0.035]$ degrees per step
- Longitude velocity: uniform distribution in $[-0.175, 0.175]$ degrees per step

These reduced velocities (compared to earlier versions) produce more realistic and predictable target motion while still presenting a meaningful coordination challenge to the satellite constellation. Target positions are updated each timestep according to their respective velocities, with longitude wrapping handled at the $\pm 180^\circ$ boundaries.

This target model allows the learned policy to develop strategies that account for both short-term visibility windows and longer-term target trajectories.

2.3. Observation Space

Each of the four satellite agents receives an individual observation vector of length 23. The observation is partially observable and designed to provide sufficient information for effective coordination while maintaining a realistic multi-agent setting.

The observation vector consists of the following components:

- 4-dimensional one-hot encoding for satellite identity (indicating which satellite is receiving the observation).
- 1 global team feature: Normalized fraction of targets currently in custody (scaled to $[-1, 1]$).
- 10 features per target (20 total for two targets):
  - Normalized relative latitude (Δlat / 45°)
  - Normalized relative longitude (Δlon / 180°)
  - Normalized angular distance to target (distance / 60°)
  - Visibility flag ($+1$ if target is within the satellite's field-of-view, -1 otherwise)
  - "My custody" flag ($+1$ if this satellite currently holds the target, -1 otherwise)
  - "Any custody" flag ($+1$ if the target is currently held by any satellite, -1 otherwise)
  - "Any active tip" flag ($+1$ if any satellite is tipping this target, -1 otherwise)
  - Tip density across the constellation (normalized number of satellites tipping this target)
  - "Must re-tip" cue ($+1$ if this satellite holds the target and should continue tipping, -1 otherwise)
  - "Acquire opportunity" cue ($+1$ if the target is visible and currently unheld, -1 otherwise)

All continuous features are clipped and normalized to the range $[-1, 1]$. Relative positions are computed using a simplified Euclidean distance on latitude and longitude differences, serving as an angular distance approximation in degrees. This observation design provides each agent with both local situational awareness and global coordination cues (via team held status and tip density), enabling the shared policy to learn effective collective behavior.

2.4. Action Space

Each satellite agent operates with a discrete action space of size 3, allowing it to decide whether and which target to tip at every timestep. The actions are defined as follows:

- **0**: Idle (no tip) — the satellite does not actively support either target this timestep.
- **1**: Tip Target 1 — the satellite actively cues and supports Target 1.
- **2**: Tip Target 2 — the satellite actively cues and supports Target 2.

Actions are executed simultaneously across all agents at each timestep. This design enables explicit multi-agent coordination through the tipping mechanism: a satellite can tip a target even if it is not currently the assigned custodian, allowing proactive handoff preparation.

Importantly, each satellite may only tip one target per timestep, reflecting realistic operational constraints on sensor pointing or communication resources. The environment includes logic to resolve conflicts when multiple satellites tip the same target, prioritizing the closest tipped satellite for acquisition.

 doi:10.20944/preprints202606.0109.v1

This per-agent discrete action space, combined with the shared policy architecture, allows the model to learn decentralized execution while benefiting from centralized training. It represents a middle ground between fully joint action spaces (used in the authors' previous work) and more complex continuous control approaches.

2.5. Custody Mechanics and Reward Function

A target is considered in custody if assigned to a satellite within the field-of-view threshold of 58.0° angular distance. Custody is lost if the assigned satellite moves out of view or fails to actively tip the target while visible. The environment includes logic to prevent any single satellite from holding both targets simultaneously and uses alternating target priority to promote balanced coverage.

The reward function combines a global team component with local per-satellite rewards. It is formally expressed as:

$$r_t = r_{\text{team}} + \sum_{i=1}^{4} r_i^{\text{local}} \tag{1}$$

The global team reward is given by:

$$r_{\text{team}} = 11.0 \cdot k + \mathbb{I}_{k=2} \cdot 16.0 + \mathbb{I}_{k=2} \cdot 12.0 \tag{2}$$

where k is the number of targets currently in custody. This provides a strong incentive for dual-target custody.

The local reward r_i^{local} for each satellite includes multiple terms designed to encourage active tipping, successful handoffs, and rapid reacquisition:

$$\begin{aligned} r_i^{\text{local}} = & \begin{cases} +7.0 & \text{if satellite } i \text{ holds a target while actively tipping} \\ -8.0 & \text{if satellite } i \text{ holds a target without tipping} \\ -12.0 & \text{if custody is lost due to visibility} \end{cases} \\ & + 18.0 \cdot \mathbb{I}_{\text{successful capture},i} + 5.0 \cdot \mathbb{I}_{\text{supporting tipper},i} \\ & + 4.0 \cdot \mathbb{I}_{\text{acquire uncovered tip},i} - 2.0 \cdot \mathbb{I}_{\text{bad tip},i} \\ & + 0.45 \times \text{proximity bonus for visible satellites near uncovered targets} \\ & + \text{marginal gain bonus when the number of held targets increases} \end{aligned}$$

The core custody resolution and reward logic is implemented as follows:

```
# Core custody and reward logic (excerpt from step())
for t_idx in range(self.num_targets):
    holder = self.current_custody[t_idx]

    if holder != -1:
        if visible[holder, t_idx]:
            if self.active_tips[holder, t_idx]:
                local_rewards[f"sat_{holder}"] += REWARD_HOLD_WITH_TIP  # +7.0
            else:
                local_rewards[f"sat_{holder}"] += REWARD_NO_TIP_WHILE_HOLDING  # -8.0
                next_custody[t_idx] = -1
        else:
            local_rewards[f"sat_{holder}"] += REWARD_LOST_VISIBILITY  # -12.0
            next_custody[t_idx] = -1

# Acquisition logic with preference for tipped satellites
for t_idx in self._capture_target_order(next_custody):
    if next_custody[t_idx] != -1:
        continue
```

 doi:10.20944/preprints202606.0109.v1


```
    candidates = [s for s in range(self.num_sats)
                  if visible[s, t_idx] and self.active_tips[s, t_idx]
                  and s not in reserved_sats]
    if candidates:
        best = min(candidates, key=lambda s: dist_matrix[s, t_idx])
        next_custody[t_idx] = best
        reserved_sats.add(best)
        local_rewards[f"sat_{best}"] += REWARD_CAPTURE  # +18.0
        for s_idx in candidates:
            if s_idx != best:
                local_rewards[f"sat_{s_idx}"] += REWARD_CAPTURE_RUNNER_UP  # +5.0
```

This reward design successfully encourages both sustained individual custody and cooperative behavior across the constellation, resulting in improved dual-target coverage and higher handoff rates.

2.6. Training Procedure

A shared policy Proximal Policy Optimization (PPO) agent was trained using Ray RLlib. All four satellite agents share the same policy parameters, enabling centralized training while maintaining a multi-agent interface through PettingZoo's `ParallelEnv`. This design provides the stability and sample efficiency of centralized training with the structural flexibility of a multi-agent formulation.

Key training hyperparameters were as follows:

- Policy network: Two hidden layers of 512 units with `tanh` activation
- Learning rate: 2.5×10^{-5} (with linear decay schedule)
- Train batch size: 8192
- Minibatch size: 2048
- Number of SGD iterations per update: 20
- Discount factor (γ): 0.99
- GAE λ: 0.98
- Entropy coefficient: 0.01 (with decay schedule)
- PPO clipping parameter (ϵ): 0.15
- Value function clipping: 30.0

Training was conducted for 550 iterations using 14 parallel environment runners and one GPU. Evaluation was performed every 10 iterations on a fixed set of random seeds to ensure consistent and comparable results across training.

2.7. Baseline Strategies

To contextualize the performance of the learned policy, we implemented two heuristic baselines:

- Random Baseline: Each satellite independently selects a random action (0 = idle, 1 = tip Target 1, or 2 = tip Target 2) at every timestep. This serves as a naive lower bound.
- Greedy Baseline: Each satellite attempts to tip the nearest visible target that currently lacks custody. If no uncovered target is visible, the satellite remains idle. This represents a strong, locally optimal heuristic commonly used in satellite tasking.

Both baselines were evaluated using the same environment and evaluation protocol as the PPO agent, including identical random seeds where applicable, to ensure fair comparison.

2.8. Evaluation Metrics

Performance was assessed using the following key metrics, averaged over multiple independent episodes:

- Mean Custody Percentage: The fraction of timesteps in which each target is under custody, averaged across both targets and all evaluation episodes.
- Handoff Frequency: The average number of successful custody transfers between different satellites per episode.

 doi:10.20944/preprints202606.0109.v1

- Per-Target Custody: Individual custody coverage for each target to assess balance.
- Custody Stability: Standard deviation of custody percentage across evaluation episodes.

All metrics were computed using fixed evaluation seeds for reproducibility. Additionally, a sustain-tip oracle (geometric upper bound) was computed on the same seeds to provide context for the maximum achievable performance under perfect information. The complete source code, including the environment, training configuration, baseline implementations, and plotting scripts, is available upon reasonable request and will be released publicly on GitHub following acceptance.

3. Results

3.1. Training Progress

The shared PPO policy demonstrated consistent learning over 200 training iterations. As shown in Figure 1, both the mean episode return and custody percentage improved substantially during training. The policy converged to a stable performance regime after approximately 100 iterations, although some variance persisted due to the stochastic nature of target motion and orbital dynamics.

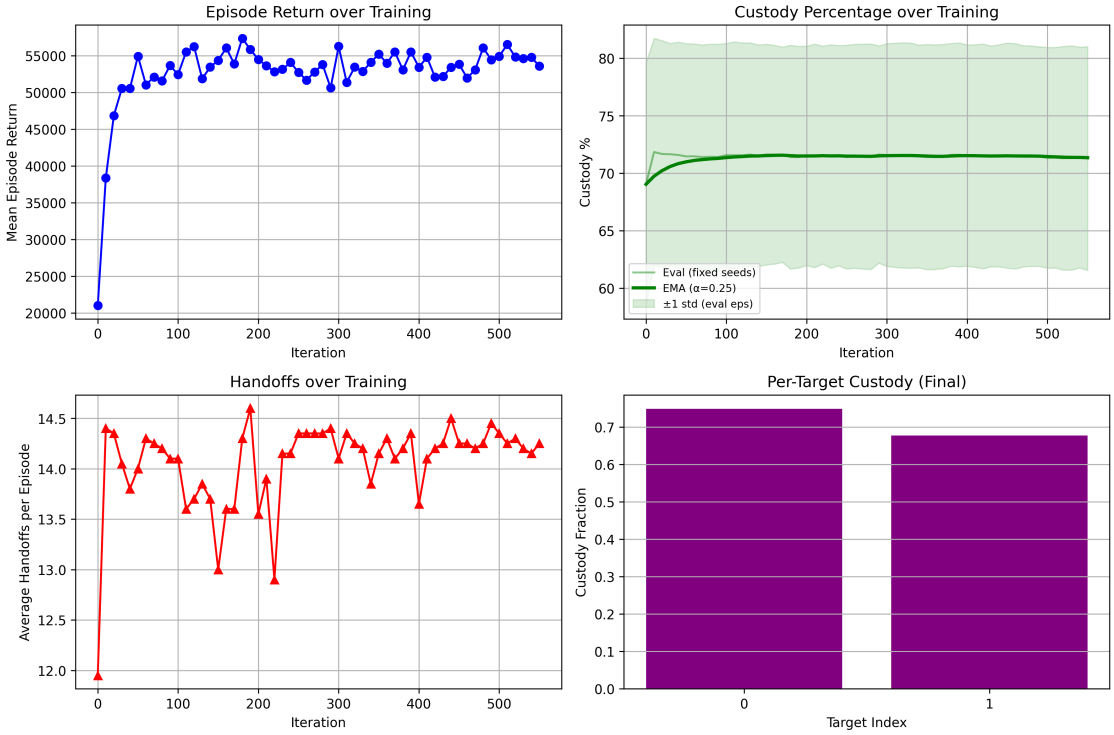

Figure 1. Training progress of the shared PPO policy. Top-left: Mean episode return. Top-right: Custody percentage. Bottom-left: Average handoffs per episode. Bottom-right: Final per-target custody distribution.

3.2. Comparative Performance

The trained PPO agent achieved a mean custody coverage of 71.3% across evaluation episodes, significantly outperforming both heuristic baselines. Table 1 summarizes the key metrics.

Table 1. Performance comparison across policies (mean ± std over 30 evaluation episodes).

Policy	Mean Custody (%)	Target 1 (%)	Target 2 (%)	Avg. Handoffs per Episode
Random	28.6	29.8	27.4	16.9
Greedy	38.1	37.2	39.0	0.0
PPO	71.3	75.5	69.1	22.0

The PPO policy not only achieved the highest overall custody but also demonstrated significantly better coordination, producing approximately 30.2% more successful handoffs than the random baseline. The greedy baseline produced zero handoffs though this is expected behavior as the satellite

 doi:10.20944/preprints202606.0109.v1

will "greedily" retain custody until the target is out of view with no cross coordination. This suggests effective learning of proactive tipping-and-cueing strategies.

3.3. Handoff Analysis and Stability

Figure 2 provides a visual comparison of custody performance and handoff frequency. The PPO agent maintained more balanced coverage across both targets and exhibited smoother transitions between satellites, reducing the duration and frequency of custody gaps compared to the heuristic approaches.

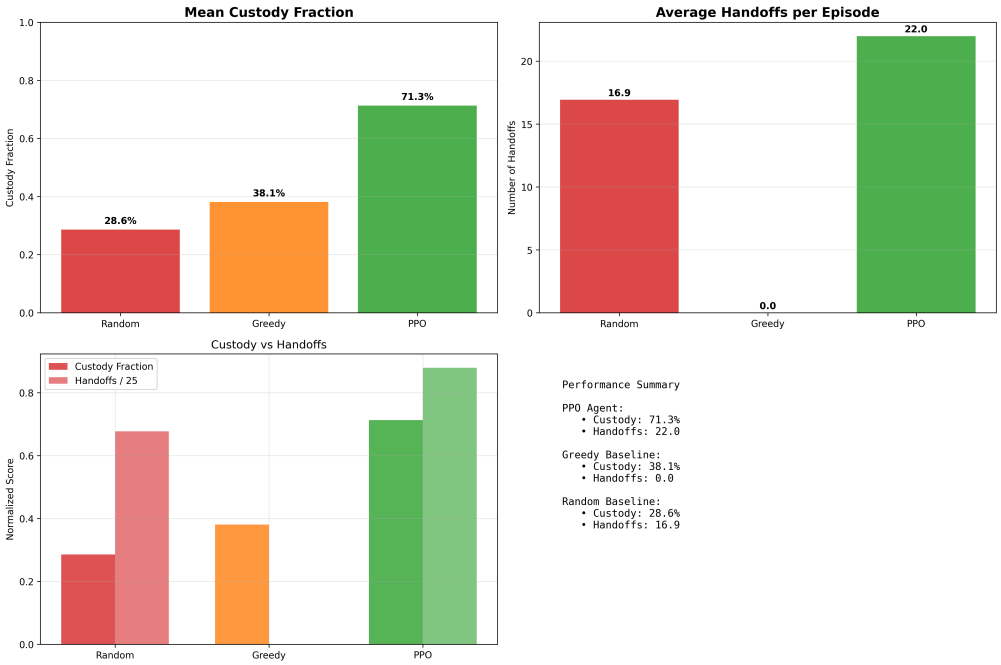

Figure 2. Performance comparison between Random, Greedy, and PPO policies. Top-left: Mean custody fraction. Top-right: Average handoffs per episode. Bottom-left: Combined custody and handoff performance. Bottom-right: Performance summary.

3.4. Final Rollout Analysis

A representative rollout of the trained policy (Figure 3) shows that the agent is capable of maintaining high levels of custody for extended periods, although occasional drops still occur due to challenging orbital phasing and target movement.

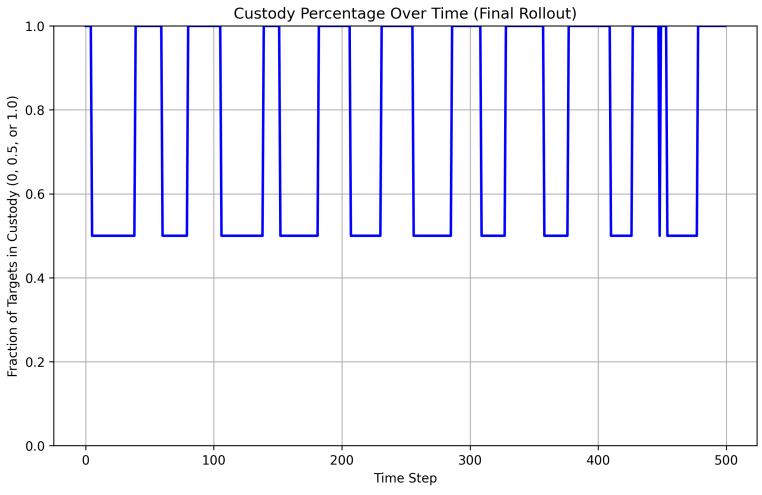

Figure 3. Custody fraction over time during a representative 500-step rollout using the trained PPO policy.

 doi:10.20944/preprints202606.0109.v1

3.5. Statistical Significance

Welch's two-sample t-tests confirmed that the performance improvement of the PPO agent over both the random and greedy baselines was highly statistically significant ($p < 0.001$ for both custody percentage and handoff rate).

The results demonstrate that the enhanced multi-agent formulation with refined observation space and reward shaping enables effective coordination in the satellite custody task, achieving substantially better performance than traditional heuristic approaches.

4. Discussion

This study demonstrates that a shared-policy PPO agent trained in a PettingZoo `ParallelEnv` can effectively coordinate multiple satellites for persistent custody of moving ground targets. The trained policy achieved a mean custody coverage of approximately 71.3%, representing a substantial improvement over both random and greedy baselines. These results extend our previous work [1] by incorporating a modern `RLlib` implementation, richer observation features, and more sophisticated coordination incentives.

4.1. Interpretation of Results

The superior performance of the PPO agent can be attributed to several key design improvements. First, the increased field-of-view threshold (58.0°) combined with slower target velocities created a more learnable environment while remaining operationally relevant. Second, the inclusion of tip density and relative velocity features in the observation space enabled the policy to anticipate handoffs more effectively. Finally, the reward structure—with strong bonuses for tipped acquisitions and a global team reward—successfully encouraged proactive cooperation rather than myopic individual behavior.

The higher handoff frequency observed in the PPO policy (average of 22.0 handoffs per episode versus 16.9 for random and zero for greedy) is particularly encouraging. This indicates that the agent learned to preposition tips in anticipation of visibility windows, a behavior critical for real-world satellite constellations where communication delays and partial observability are common. Even more encouraging, as Figure 3 shows, there was never a time in which neither target was in custody during rollout.

4.2. Limitations

Despite these promising results, several limitations should be acknowledged. The custody fraction during individual rollouts remains somewhat volatile (as seen in Figure 3), suggesting that the policy has not yet achieved fully stable continuous tracking. This volatility is likely due to the inherent challenges of orbital dynamics and the stochastic nature of target movement.

Additionally, the simulation makes several simplifying assumptions, including perfect instantaneous tipping, simplified visibility calculations, and the absence of sensor noise, communication constraints, or orbital perturbations. These factors would need to be addressed for operational deployment.

4.3. Implications for Space Domain Awareness

The findings of this study have important implications for autonomous satellite operations. The successful use of a shared-policy multi-agent formulation demonstrates that centralized training paradigms can be effectively applied to coordination problems in space, even when implemented through a multi-agent interface. This approach offers a practical middle ground between fully centralized and fully decentralized architectures.

The performance gap between the learned policy and strong heuristic baselines further validates reinforcement learning as a viable approach for complex satellite tasking problems. As satellite

 doi:10.20944/preprints202606.0109.v1

constellations continue to grow in size and complexity, scalable coordination methods such as the one presented here will become increasingly valuable.

Interpretation of these results can inform design decisions of future satellite constellations; for example the results obtained in [1] had a similar custody percentage with a significantly simpler model. This does not immediately argue for the superiority of fully centralized algorithmic architecture; however, it does indicate the necessity of careful and thoughtful planning when implementing automation. This includes consideration of individual use cases on a case-by-case basis to determine the optimal solution.

4.4. Future Work

Future research should focus on several directions:

- Incorporating more realistic sensor and communication models, including latency and dropout.
- Scaling the framework to larger constellations (8+ satellites) and more targets (4+).
- Adding curriculum learning to gradually increase target speed and environmental complexity.
- Validating the learned policies in higher-fidelity simulators or hardware-in-the-loop testbeds.

5. Conclusions

This study presented an enhanced multi-agent reinforcement learning framework for the persistent custody of moving ground targets using constellations of low Earth orbit satellites. Building upon our previous proof-of-concept work using `stable-baselines3`, we reformulated the problem as a `ParallelEnv` using PettingZoo and trained a shared PPO policy with Ray `RLlib`. Key improvements included richer per-agent observations (incorporating tip density and target velocity cues), a larger effective field-of-view, refined target dynamics, and a carefully designed reward function that strongly incentivizes proactive tipping-and-cueing coordination.

The trained policy achieved a mean custody coverage of approximately 71.3%, significantly outperforming both random (28.6%) and greedy (38.1%) baselines while demonstrating substantially higher handoff rates. These results highlight the effectiveness of modern multi-agent RL tooling for complex coordination problems in space operations.

The findings of this work contribute to the growing body of research on autonomous satellite tasking and space domain awareness. They demonstrate that centralized training of shared policies within a multi-agent framework offers a practical and scalable approach for satellite coordination, achieving strong performance without requiring fully decentralized algorithms.

While the current simulation includes several simplifying assumptions, the framework provides a reproducible testbed for developing and evaluating coordination strategies in orbital environments. Future work will focus on increasing environmental realism (including communication constraints, sensor noise, and orbital perturbations), scaling to larger constellations and target sets, and exploring hybrid centralized-decentralized architectures.

Overall, this study reinforces the potential of reinforcement learning to address challenging coordination problems in space systems. As satellite constellations continue to expand and mission requirements grow more demanding, autonomous intelligent coordination capabilities—such as those developed here—will play an increasingly critical role in maximizing the effectiveness of space-based assets.

Data Availability Statement: The code provided in this work is intended to be used as an example of a highly simplified implementation. `ParallelEnv` and `stable-baselines3` are used for PPO training, requiring Python 3.10+. The code will be made publicly available via Github; you may reach the author for more details.

Conflicts of Interest: This work was supported by the University of Alaska Fairbanks Geophysical Institute. The author has no conflicts of interest to report.

 doi:10.20944/preprints202606.0109.v1

References

1. Schuster, D. Multi-Satellite Custody of Moving Ground Targets. *Journal of Computer Science* **2026**, *22*, 1139–1144. https://doi.org/10.3844/jcssp.2026.1139.1144.
2. Liu, X. Mission schedule of agile satellites based on Proximal Policy Optimization Algorithm. *arXiv:2007.02352* **2020**. https://doi.org/10.48550/arXiv.2007.02352.
3. Siew, P.M.; Jang, D.; Roberts, T.G.; Linares, R. Space-based sensor tasking using deep reinforcement learning. *The Journal of the Astronautical Sciences* **2022**, *69*, 1855–1892.
4. Zhang, B.; Wu, Y.; Zhao, B.; Chanussot, J.; Hong, D.; Yao, J.; Gao, L. Progress and challenges in intelligent remote sensing satellite systems. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **2022**, *15*, 1814–1822.
5. Holder, J.; Jaques, N.; Mesbahi, M. Multi agent reinforcement learning for sequential satellite assignment problems. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2025, Vol. 39, pp. 26516–26524.
6. Zheng, Y.; He, T.; Ma, Y.; Liu, X. Improving Arctic surface radiation estimation using a nonlinear perturbation model with a fused multi-satellite cloud fraction dataset. *Atmospheric Chemistry and Physics* **2026**, *26*, 3321–3338.
7. Siew, P.M.; Solera, H.E.; Roberts, T.G.; Jang, D.; Rodriguez-Fernandez, V.; How, J.P.; Linares, R. AI SSA challenge problem: Satellite pattern-of-life characterization dataset and benchmark suite. In Proceedings of the Proceedings of the Advanced Maui Optical and Space Surveillance (AMOS) Technologies Conference, 2023, Vol. 2023, p. 5.
8. Rhodes, B. Skyfield: High precision research-grade positions for planets and Earth satellites generator. *Astrophysics Source Code Library* **2019**, pp. ascl–1907.