Autonomous Satellite Inspection in Low Earth Orbit with Optimization-Based Safety Guarantees

Mark Stephenson, Daniel Huterer Prats, and Hanspeter Schaub

Autonomous Vehicle Systems Lab
University of Colorado, Boulder
3775 Discovery Dr., Boulder, CO, 80303
Mark.A.Stephenson@colorado.edu

Abstract

The inspection of objects in low Earth orbit is important to rendezvous and proximity operations, which are of increasing interest to commercial and government organizations. Complex relative motion dynamics in low Earth orbit make the problem of autonomous inspection challenging. Agents must be able to fully inspect an object subject to illumination constraints while avoiding collision. This paper presents a novel approach to autonomous satellite inspection with impulsive maneuvers by combining a semi-Markov decision process formulation of the inspection task with an optimization-based shield for collision avoidance based on the Clohessy-Wiltshire-Hill equations of relative motion. By leveraging natural motion dynamics, the servicer is able to complete the inspection task using 2.6 m/s of Δv in 2.9 orbits, on average. The spacecraft evaluates maneuver actions by evaluating a policy onboard in a closed-loop manner, accounting for actual spacecraft states as opposed to executing a brittle, off-line plan; the policy is trained for the domain of inspection tasks offline in advance of the mission.

Introduction

With the proliferation of satellites in low Earth orbit (LEO), rendezvous and proximity operations (RPO) tasks are becoming increasingly important, whether for servicing and interacting with active spacecraft, deorbiting defunct satellites, or inspecting objects for damage. It is often necessary to inspect an object prior to interacting with it. This inspection task is complex, requiring the agent to control its relative motion with the object to inspect all illuminated surfaces while avoiding collision. Close proximity operations are challenging to operate and require significant ground support to upload open-loop command sequences and monitor the resulting performance. Close relative motion maneuvers must be scheduled in a manner to be fuel efficient, avoid even the potential for collision, and yet still satisfy complex space object imaging requirements and servicer spacecraft safety concerns.

Recent papers (Sabatini, Volpe, and Palmerini 2020; Bernhard et al. 2020; Nakka et al. 2022) demonstrate pipelines for global planning of inspection trajectories for a swarm of satellites, for impulsive and continuous thrust control. These plan a set of stable relative orbits that should provide coverage of the resident space object (RSO), then assign orbits to individual servicers within the swarm, calculating optimal transitions between orbits. Another paper (Lauinger and Ulrich 2025) decomposes a large RSO into primative shapes and projects inspection paths onto them.

More recently, reinforcement learning (RL) has been posed as a method for closed-loop autonomy on the inspection task. RL has the benefits of directly finding a policy to maximize a reward function in an arbitrary environment. In one paper (Aurand et al. 2023), a neural network-based policy is used for high-level planning between predetermined inspection waypoints. Another paper (van Wijk et al. 2023) approaches the problem with a continuous action space for continuous, low-thrust control, considering lighting conditions. In recent work (Lei et al. 2024, 2025), the waypoint-based approach is used for distributed and centralized control of a swarm of servicers. However, a drawback of this class of methods is a lack of safety guarantees. While penalties can be included to disincentivize unsafe actions, there is no guarantee that the agent will not take an unsafe action.

Significant research exists on safe coordinated motion planning in LEO under Clohessy-Wiltshire-Hill (CWH) dynamics. Morgan (Morgan, Chung, and Hadaegh 2014; Morgan et al. 2016) uses sequential convex programming (SCP) to plan collision-free continuous thrust trajectories. Others (Gaias and D'Amico 2015; Koenig and D'Amico 2018) use analytical methods to derive swarm reconfiguration laws with impulsive thrusts in terms of relative orbital elements. RL has been considered for trajectory optimization: a few authors (Guffanti et al. 2024; Takubo et al. 2024) leverage the transformer architecture for RPO, while another (Kuhl et al. 2025) applies Markov decision processes (MDPs) to station keeping with collision avoidance. Most similar to this work, two recent papers (Van Wijk et al. 2024; Dunlap, Hamilton, and Hobbs 2025) ensures safety in a RL-controlled inspection task with continuous control inputs using a control barrier function. Their safety constraints include the keep-out zone present in this work, fixed-horizon passive safety, velocity constraints, a sun pointing constraint, and a keep-in zone.

Shielded RL has been demonstrated as an effective method for various other spacecraft tasking problems, including resource management (Harris et al. 2022), small-body proximity operations (Herrmann and Schaub 2022),

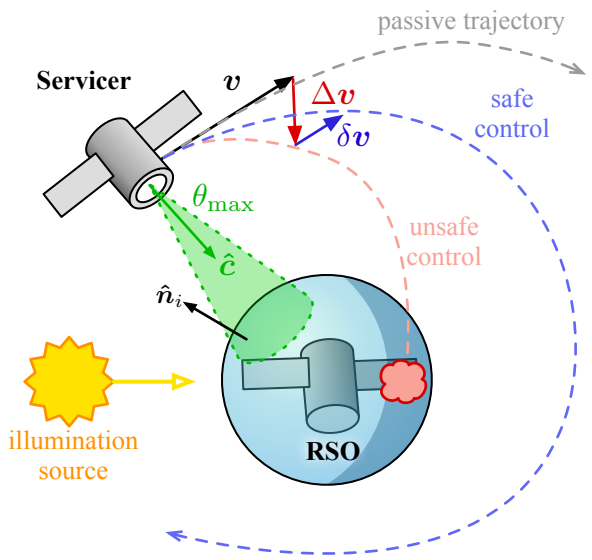

Figure 1: Diagram of inspection task and optimization based shield.

and agile Earth observation under various conditions (Stephenson and Schaub 2024a; Stephenson, Mantovani, and Cheval 2025).

In this work, the RSO inspection problem is considered for a RSO with a Hill-frame-fixed attitude and a single, impulsively thrusting servicer, subject to range, pointing, and lighting constraints for inspection. The problem is formulated as a MDP and solved using deep reinforcement learning (DRL), with a formulation that is specifically designed to reflect the operations of a impulsively maneuvered spacecraft. Unlike other RL-based approaches to the inspection task, a shield (Alshiekh et al. 2018) that uses an optimization-based analysis of the CWH dynamics under impulsive control is employed to ensure that the agent cannot take unsafe actions with respect to the RSO or any other nearby objects in orbit.

Problem Statement

In the inspection task, a servicer spacecraft aims to image all facets of an RSO in LEO, subject to illumination constraints. To achieve this, the servicer must leverage discrete thrusts to control its relative motion with the RSO while avoiding collision.

RSO Model

The RSO is a passive satellite in a circular orbit. In relative motion terms, the RSO is the chief, denoted by subscript C. The RSO has a set of N body-fixed inspection points $p_i \in \mathcal{P}$, each with an associated normal vector $\hat{\boldsymbol{n}}_i$. In this work, the RSO is modelled as a 2-meter radius sphere of $N = 100$ uniformly distributed points with normals in the radial direction.

The RSO's circular, 500 km altitude orbit is perturbed by J2 effects and atmospheric drag. The inclination and right ascension are randomly sampled for each instance of the environment.

It is assumed that the RSO is known to be in a Hill-frame fixed attitude, which could be confirmed via a simultaneous localization and mapping (SLAM) algorithm. This is typical for many spacecraft with nadir-pointing sensors. Further work will extend the problem to the general case of a tumbling RSO.

Servicer Spacecraft Model

The servicer spacecraft is equipped with a body-fixed camera $\hat{\boldsymbol{c}}$ to inspect the RSO. In relative motion terms, the servicer is the deputy, denoted by subscript D. A flight software algorithm controls the attitude of the spacecraft using reaction wheels to point the camera boresight at the RSO. To inspect a point on the RSO, the angle between the normal $\hat{\boldsymbol{n}}_i$ and the camera boresight $\hat{\boldsymbol{c}}$ must satisfy

$$\theta_i = \angle(\hat{\boldsymbol{n}}_i, -\hat{\boldsymbol{c}}) < \theta_{\max} \tag{1}$$

The servicer must also be within $r_{\text{inspect}} = 250$ m of the RSO to inspect a point. When a point is inspected, it is added to the inspected set $\mathcal{I}$.

The servicer's orbit is modelled to high fidelity, including J2 perturbations and atmospheric drag. As a result, the relative motion between the servicer and the RSO is governed by perturbed CWH equations of relative motion. The servicer can control its relative motion with a cluster of impulsive thrusters. These thrusters can produce impulsive thrusts of magnitude up to $\Delta v_{\max}$ in any arbitrary direction. Since the period between thrusts is generally sufficiently long, this could be achieved by a variable-thrust thruster on an actuated platform that reorients between maneuvers. At most, the servicer may use 10 m/s of Δv to complete the task.

If the servicer collides with the RSO, the inspection task is considered a failure. Collision is defined as the distance between the servicer and the RSO being less than the sum of their collision radii:

$$\|\boldsymbol{r}_{DC}\| < R_D + R_C \tag{2}$$

The initial position is of the servicer is randomly set within a 1 km radius of the RSO; the initial relative velocity is near zero.

Illumination Constraints

In addition to the view incidence angle, the inspection points on the RSO must be illuminated by the sun. To be sufficiently illuminated for imaging, the incidence angle between the sun and the point normal must be less than $\phi_{\max}$. The RSO also must not be eclipsed by Earth.

Figure 2 shows the percent of the RSO that is inspectable due to the orbital elements of the RSO for 1000 random cases. Orbits that are closer to a sun-synchronous orbit (SSO) tend to have the smallest fraction of points ever illuminated, as one side of the spacecraft is constantly facing away from the sun.

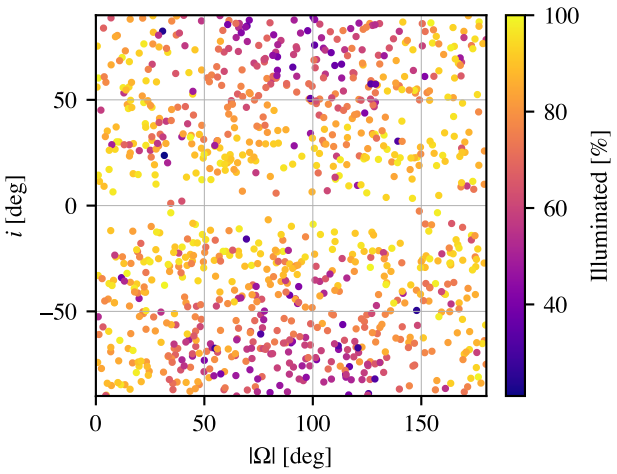

Figure 2: Percent of RSO inspectable due to orbital geometry.

Objective Function

The numerical objective function is to maximize the percentage of the RSO points that are inspected with a weighted penalty

$$\text{maximize} \quad \frac{|\mathcal{I}|}{N} - \alpha \sum \Delta v \tag{3}$$

$$\text{s.t.} \quad \|\boldsymbol{r}_{DC}\| > R_D + R_C \; \forall t \tag{4}$$

subject to a non-collision constraint and the environment dynamics.

Reinforcement Learning

DRL is an effective class of methods for finding policies to maximize a reward function over time for autonomy in robotics problems with continuous state spaces. To be able to leverage DRL, the inspection problem must be formalized as a MDP, which is a framework for sequential decision making. In a MDP, the agent interacts with an environment by taking actions in response to a state, which returns a reward and a new state. The goal of the agent is to find a policy that maximizes the expected sum of rewards over time.

Learning on MDPs

MDPs are a framework for representing sequential optimization problems (Sutton and Barto 2018). In a MDP, the state s evolves with some action a according to a transition function $T(s, a, s')$ and yields a reward $r(s, a, s')$. The goal of reinforcement learning is to find a policy $\pi(a|s)$ that maximizes the expected discounted sum of rewards.

Semi-Markov Decision Processes

While in most control problems, the control input is changed on a fixed interval, this does not reflect the operational realities of impulsive maneuvers. Instead, the servicer should perform an impulsive thrust, then drift for a variable duration of time until the next thrust is to be performed. To represent this problem, the MDP must be extended to allow for variable-duration timesteps.

Element	Dim.	Description
$\boldsymbol{r}_{DC}^{\mathcal{H}}$	3	Hill frame position
$\boldsymbol{v}_{DC}^{\mathcal{H}}$	3	Hill frame velocity
Δv_{avail}	1	available Δv
i	1	orbit inclination
Ω	1	orbit RAAN (rel. sun)
$\hat{\boldsymbol{s}}$	3	sun vector
$[\tau_{\text{ecl}}^{o}, \tau_{\text{ecl}}^{c}]$	2	eclipse start and end times
t	1	time since start of episode
inspection %	M	region inspection status

Table 1: Elements of the observation space.

A MDP with variable-duration timesteps is called a semi-Markov decision process (sMDP). In a sMDP, each step has some Δt associated with it, which represents the time-opportunity cost of the step, and a reward density $\varrho_t(t)$ instead of reward r_t (Bradtke and Duff 1994). The γ-discounted reward for a step is then given by

$$r_t^{(\gamma)} = \int_{t_t}^{t_t+\Delta t_t} e^{\gamma(t-t_t)} \varrho_t(t) \mathrm{d}t \tag{5}$$

In this work, a semi-Markov-modified version of proximal policy optimization (PPO) is used to train a policy for the inspection task (Schulman et al. 2017). Using the RLlib framework, advantage estimation in the base PPO is modified to discount in a timestep-aware manner when learning (Liang et al. 2018).

MDP Formulation

The elements of the partially-observable Markov decision process (POMDP) tuple for the inspection task are as follows:

- **State Space:** The state of the simulator providing the generative model for the MDP. This includes satellite dynamic states, flight software states, and environment states. Terminal states are encountered due to various conditions:
  - $\geq 95\%$ of illuminated points are inspected.
  - The servicer collides with the RSO: $|\boldsymbol{r}_{DC}| < R_C + R_D$
  - The servicer leaves the RSO: $|\boldsymbol{r}_{DC}| > 1$ km. This condition is included to encourage "good" behavior when training, but is not a hard constraint enforced by the shield in evaluation
  - The servicer runs out of available fuel: $\Delta v_{\text{avail}} = 0$
  - The episode times out: $t \geq 10$ orbits
- **Observation Space:** The observation is composed of a subset of the elements of the state space and transformations thereof. The elements of the observation space are given in Table 1. The region inspection status is the percentage of points in $M = 15$ equally spaced regions of the RSO that have been inspected; each region is roughly the same size as the servicer's field of view due to incidence angle requirements.

- **Action Space:** The servicer's action space is $a \in \Delta v_{\max}\mathcal{B}^3 \times [0, \Delta t_{\max}]$. The first three elements specify the direction and magnitude of an impulsive thrust, and the last element specifies the duration to drift before executing the next thrust.
- **Reward Function** The reward density $\varrho(t)$ is the sum of three functions. First, reward is yielded for inspecting points on the RSO.

$$\varrho_{\text{inspection}}(t) = \frac{1}{N}\frac{\mathrm{d}}{\mathrm{d}t}|\mathcal{I}|. \tag{6}$$

A reward is also given for reaching the goal state of $\geq 95\%$ inspection of illuminated points, using the Dirac delta function.

$$\varrho_{\text{success}}(t) = \beta_{\text{success}}\delta(t - t_{\text{success}}) \tag{7}$$

The 95% criteria is used for success, since some points may only be momentarily illuminated due to the discretization of evaluation points. A penalty for fuel use is given at the time of a burn.

$$\varrho_{\text{fuel}}(t) = -\alpha\|\Delta\boldsymbol{v}\|\delta(t - t_{\text{burn}}) \tag{8}$$

Finally, a failure penalty is given for collision, running out of fuel, or leaving the region of space around the RSO. Failure also terminates the episode.

$$\varrho_{\text{failure}}(t) = -\beta_{\text{fail}}\delta(t - t_{\text{failure}}) \tag{9}$$

- **Transition Model:** Instead of a probabilistic transition function, the transition model is implemented as a deterministic generative model. When an action is taken, the environment propagates the simulation forward in time until the next action is to be taken. The sMDP Δt for the step is the duration that the simulator propagated the step for.

The environment is implemented in accordance with the Gymnasium API (Towers et al. 2023) using BSK-RL[1], a package for creating modular, high-fidelity spacecraft tasking RL environments (Stephenson and Schaub 2024b). The underlying spacecraft simulation is Basilisk[2], a high-performance spacecraft simulation package (Kenneally, Piggott, and Schaub 2020). Rigid multi-body dynamics in the perturbed orbital environment and flight-proven flight software algorithms are used to simulate the environment. Basilisk is ideal for offline agent training as its high computation speed enables efficient training with a complex physics-based simulation.

Optimization-Based Shield

Collision with the RSO is of high concern for the servicer spacecraft. Shielding is a method for guaranteeing operational safety of a RL-based agent. For this problem, a shield is developed that prevents the servicer from executing a burn that may lead to eventual collision with the RSO under natural dynamics.

[1]https://avslab.github.io/bsk_rl/

[2]https://avslab.github.io/basilisk/

Shielding for RL

A shield interacts with a policy in order to prevent unsafe actions (i.e. those actions that may cause a transition into $s' \in S_{\text{unsafe}}$ in the next step, or for more robustness, in any future step) (Alshiekh et al. 2018). In general, there are two classes of shields: "preshields", which mask disallowed actions such that the policy may not select them, and "post-shields", which override the policy's action with a safe action. This work takes a hybrid approach, in which the shield overrides the policy's action with the closest safe action.

CWH Dynamics

The CWH equations of relative motion are a set of linear equations that describe the unperturbed relative motion of a deputy spacecraft relative to a chief spacecraft in a circular orbit. In the Hill frame, the CWH equations are given by

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x},\ \boldsymbol{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3n^2 & 0 & 0 & 0 & 2n & 0 \\ 0 & 0 & 0 & -2n & 0 & 0 \\ 0 & 0 & -n^2 & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

where the state vector $\boldsymbol{x} = [\boldsymbol{r}_{DC}^{\mathcal{H}}, \boldsymbol{v}_{DC}^{\mathcal{H}}]$ is composed of the Hill-frame position and velocity of the deputy relative to the chief, and $n = 2\pi/T_{\text{orbit}}$ is the mean motion of the chief.

Optimization Problem

With these dynamics, the shield can be formulated as an optimization problem. In short, the problem is to find the thrust closest to that selected by the policy $\pi(s) = \Delta\boldsymbol{v}$ that does not lead to collision with the RSO under CWH dynamics. Let $\delta\boldsymbol{v}$ be the change in selected thrust required for safety. The optimization problem can be written as

$$\text{min.} \quad \|\delta\boldsymbol{v}\| \tag{11}$$

$$\text{s.t.} \quad \|\Delta\boldsymbol{v} + \delta\boldsymbol{v}\| \leq \Delta v_{\max} \tag{12}$$

$$\boldsymbol{x}_0 = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{v} + \Delta\boldsymbol{v} + \delta\boldsymbol{v} \end{bmatrix}$$

$$\boldsymbol{x}_0^T \exp(\boldsymbol{A}t_i)^T \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \exp(\boldsymbol{A}t_i)\boldsymbol{x}_0$$

$$\geq (R_C + R_D + \epsilon)^2 \ \forall t_i \in [t_0, t_{\max}] \tag{13}$$

$$\pm((R_C + R_D + \epsilon) + 2A_0' \pm y_{\text{off}}')x_{\text{off}}' \leq \frac{3}{2}t_{\max}nx_{\text{off}}'^2 \tag{14}$$

The objective (11) minimizes the change in requested thrust. The first constraint (12) ensures that the thrust does not exceed the maximum possible thrust. The second constraint (13) ensures that the shielded action does not lead to collision with the RSO over some upcoming timespan $[t_0, t_{\max}]$. An additional safety parameter ϵ is added to the collision constraint to account for the effects of perturbations or an additional safety factor beyond what the agent was trained to follow. The final two constraint equations (14) are only necessary if passive safety is desired. With this constraint,

the servicer must either be in a non-colliding periodic orbit or on a secularly moving-away trajectory by $t_{\max}$. This constraint uses the standard definitions of x'_{off}, y'_{off}, and A'_0, where $'$ indicates values calculated using the post-correction state (Schaub and Junkins 2018). See the appendix for the derivation of the final constraint.

The discretization time δt for propagation must be sufficiently small to prevent collisions between timesteps. The condition for this is similar to the one found by Morgan (Morgan, Chung, and Hadaegh 2014). With an upper bound on relative velocity of the servicer estimated as

$$\lceil v \rceil = \sqrt{(A_0 n)^2 + (2A_0 n - \frac{3}{2} n x_{\text{off}})^2 + (B_0 n)^2} + \Delta v_{\max} \tag{15}$$

the maximum undetectable violation of the collision constraint for some δt is given by

$$r_{\text{violation}} \leq \frac{\delta t \lceil v \rceil}{2} \tag{16}$$

It follows that a maximum allowable violation $r_{\text{violation}} < \epsilon$ can be specified and used to select δt.

Results

A hyperparameter search is presented to find the best policy for the inspection task. The selected policy is then evaluated across various unshielded and shielded test cases.

Trained Policies

In total, 24 policies are trained. The varied hyperparameters are the fuel use penalty $\alpha \in [0.0, 0.1, 0.2, 0.3, 0.4, 0.5]$, the learning rate $\in [10^{-5}, 10^{-6}]$, and the discount rate $\gamma \in [0.9995, 0.9999]$. The success and failure rewards $\beta_{\text{success}} = \beta_{\text{fail}} = 1$. The batch size is 1550. Other hyperparameters are set to the RLlib 2.35.0 PPO defaults. Policies were trained for 48 hours each across 32 cores.

Figure 3 shows training curves for all policies: the overall reward, the fuel usage, and the inspection percent (i.e. the number of inspected points divided by the number of illuminated points).

The fuel penalty parameter α has a strong impact on the policy. As expected, lower α values lead to policies that use more fuel and inspect more aggresively, while higher α values lead to policies that use less fuel, sometimes at the expense of satisfactorily completing the task. Policies with a higher α tend to reduce fuel usage by increasing the drift time between burns, while those with a low α reduce fuel usage by completing the task more quickly. This also hints at why the high α polices trained fewer steps in the same amount of time: simulating steps with longer drift times is more computationally expensive.

The selected final policy has learning rate 10^{-5}, discount rate $\gamma = 0.9999$, and fuel penalty $\alpha = 0.2$. This policy was selected for its balance between inspection performance and fuel usage, as shown in Figure 3.

Unshielded Performance

The selected policy is evaluated across various test cases.

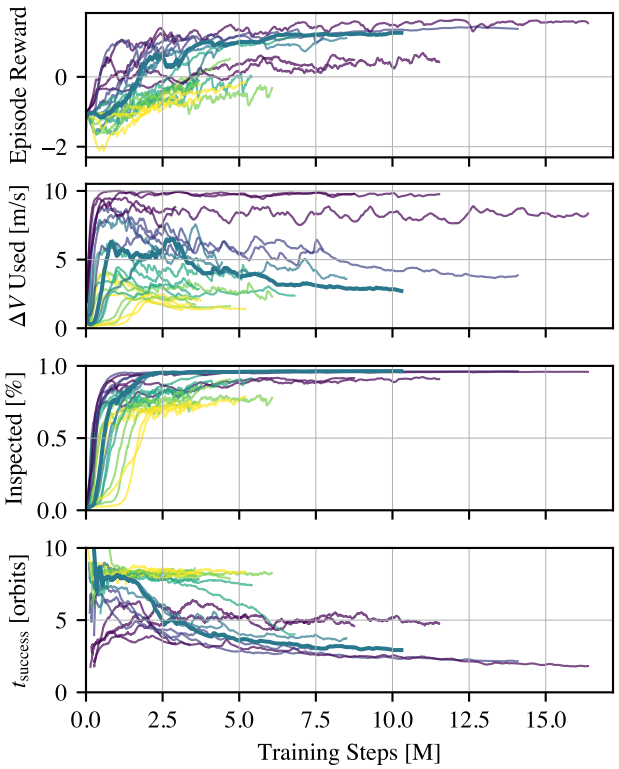

Figure 3: Training curves for all policies. Selected policy highlighted. Purple: $\alpha = 0.0$, yellow: $\alpha = 0.5$.

Unshielded Benchmark To investigate the policy performance, 1000 instances of the environment are sampled (randomizing the RSO orbit and servicer initial relative orbit) and the policy is evaluated on each instance.

Figure 4 demonstrates the overall performance of the unshielded policy. The vast majority (98.0%) of cases are successfully inspected, using on average 2.8 m/s of fuel to complete the task in 3.2 orbits. The most common failure modes are using excessive Δv (1.3% of cases) and using excessive time (0.7 % of cases). Only 2 of the 1000 cases failed due to collision with the RSO. This performance is reflective of the final values of the training curves for the policy.

Example Trajectories The policy's behavior in two cases is inspected more closely: an orbit with a high fraction of illuminated points, and a SSO with a low fraction of illuminated points.

Figure 5 shows results for the highly illuminated case, in which most of the RSO is illuminated at some point of the orbit. Since in this case, the servicer starts relatively far from the RSO, a large portion of the total fuel usage comes from large thrusts at the start of the episode in order to bring the servicer within the inspection radius. Once near the RSO, the policy leverages natural motion dynamics to make occasional adjustments to the trajectory in order to cover the entire inspectable region.

Figure 6 shows results for the less illuminated SSO case, in which only about half of the points are inspectable. The

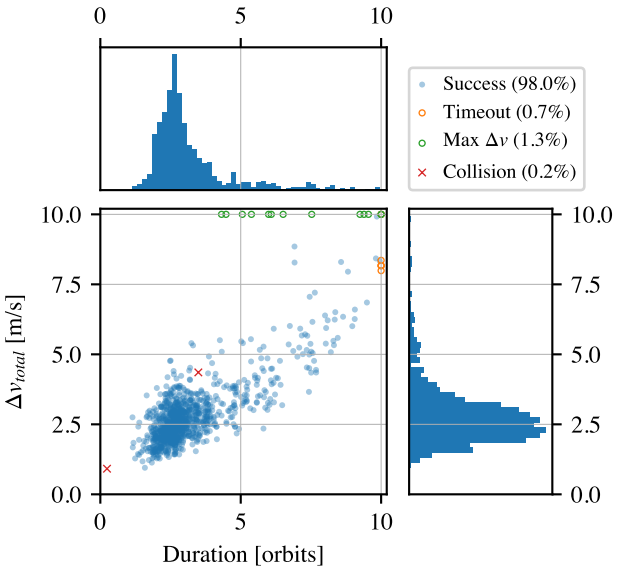

Figure 4: Unshielded performance of the policy.

trajectory reveals that the policy is aware of the limited inspectable hemisphere, even though that information is only implicitly included in the observation. The resulting trajectory tends to stay on the illuminated side of the RSO in order to complete the task more quickly.

Shielded Performance

The same policy is evaluated with the shield to demonstrate safety. First, the shield is used to enforce a narrow constraint close to that used in the training environment. Then, the ability of the shield to ensure safety with a significantly more conservative constraint is demonstrated.

Shielded Benchmark The shielded policy is benchmarked with the same 1000 instances as the unshielded case. The keepout radius is set to $R_C + R_D + \epsilon = 20$ meters and the timestep selected such that the maximum interstep violation is 10 meters, leading to an effective guaranteed safety radius of 10 meters.

Figure 7 shows the aggregate performance of the shield. Most importantly, the shield prevents all collisions with the RSO, as expected. Surprisingly, it improves the overall performance of the policy: The mean fuel usage is reduced to 2.6 m/s (from 2.8, unshielded) and the mean inspection time is reduced to 2.9 orbits (from 3.2, unshielded). The shielded policy likewise reduces the fraction of cases that ran out of time or fuel.

The impact of the shield is quantified in Figure 8. Reflective of the fact that the unshielded policy rarely collided with the RSO, the shield does not often intervene. Even when the shield is activated, it is likely not preventing a violation, but rather enforcing the additional guarantee of passive safety. Shield corrections tend to be very small in magnitude in this case. The shield is relatively expensive to evaluate, averaging about 1.0 second per call compared to the 10 ms per call to the policy. However, this is acceptable since the shield

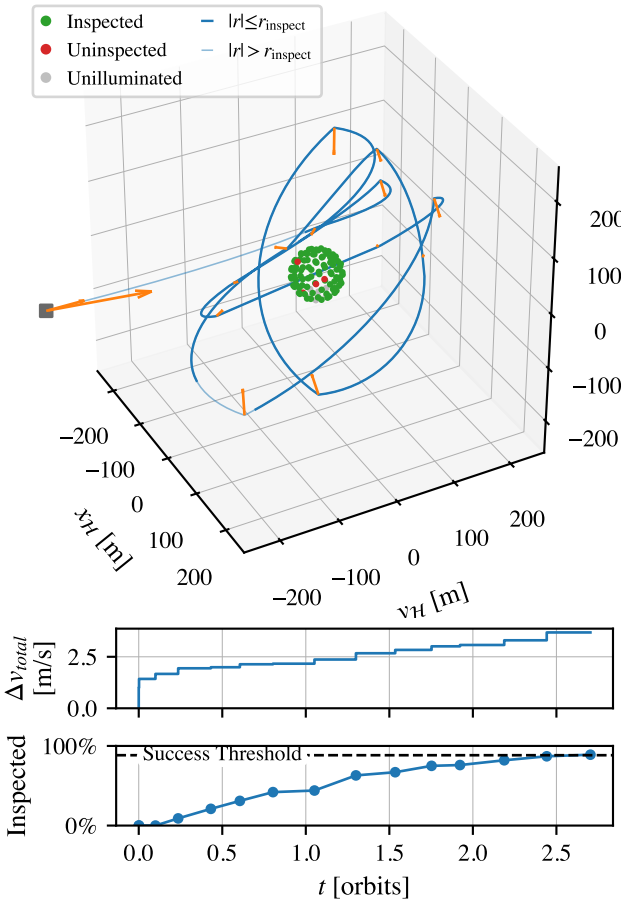

Figure 5: Policy behavior in high illumination case. RSO radius is enlarged.

is only called when the policy is about to execute a thrust, which occurs with a period on the order of hundreds to thousands of seconds.

Comparing to the state of the art (Van Wijk et al. 2024), this shielded policy with impulsive thrusts uses only 10-20% of the Δv required in van Wijk's work (12.86 to 26.00 m/s, depending on experiment configuration) that uses continuous control. Since natural relative motion is leveraged, inspection times are about 2.9 orbits rather than 0.59-0.63 orbits in their continuous control solutions.

Using a More Conservative Shield A primary benefit of the shield formulation is that it allows for a keepout radius to be specified different than that used in training. As a result, the shield can be made more conservative when approaching a RSO with a more uncertain state.

Figure 9 shows the policy executed with a shield that enforces a keepout radius of $R_C + R_D + \epsilon = 100$ meters. The behavior of the policy is similar to the unshielded policy, but the plot of $|\boldsymbol{r}_{DC}|$ over time shows that the keepout radius is never violated. Since the keepout radius is much larger than the collision radius used in training, the shield is required

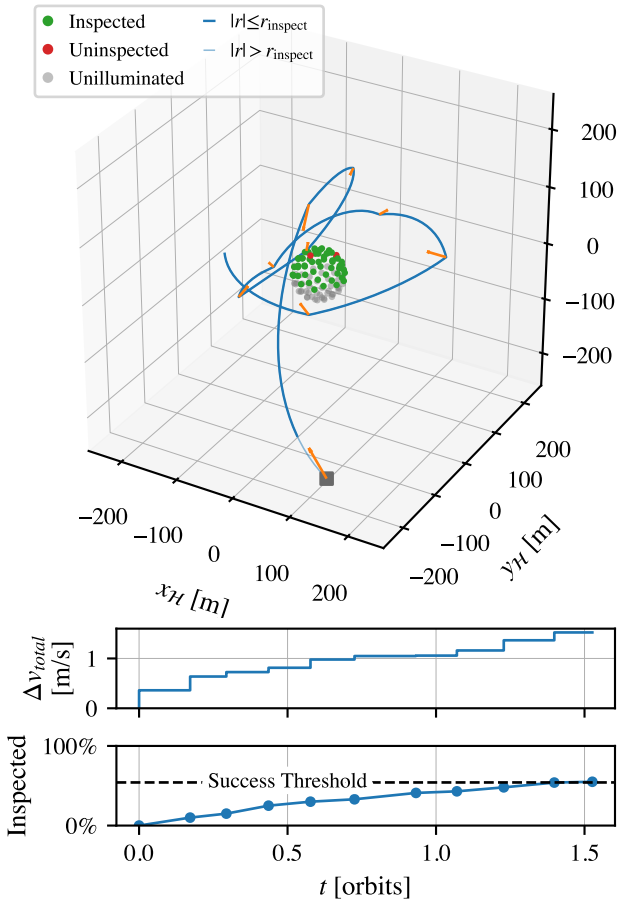

Figure 6: Policy behavior in low illumination case (SSO).

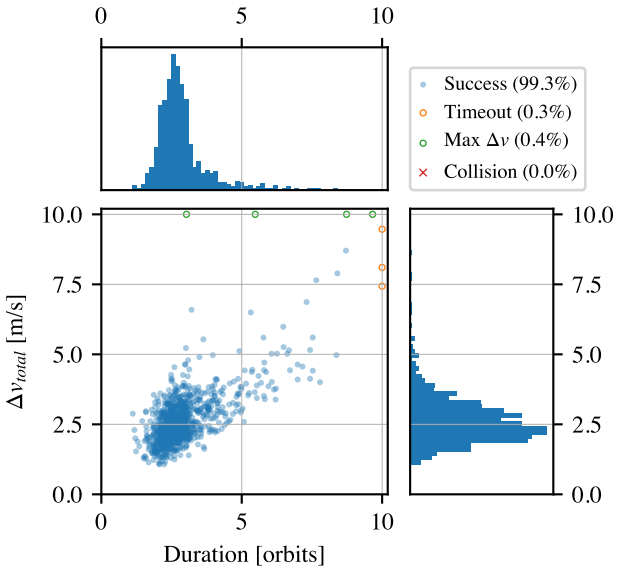

Figure 7: Shielded performance of the policy.

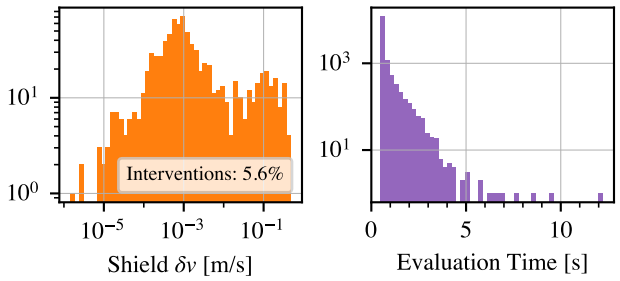

Figure 8: Left: magnitude of nonzero interventions. Right: evaluation time for all shield calls.

to intervene more often to maintain the stricter safety constraint.

Discussion of Future Work

A variety of future work can build on these initial results by increasing the fidelity and complexity of the problem formulation. The properties of the RSO can be generalized to include tumbling and non-passive attitude dynamics. More realistic lighting constraints due to self-shadowing on bodies that cannot be approximated as a sphere or glare on certain surfaces should also be considered. The servicer can also be subject to more challenging constraints, such as a more realistic thrust model with limitations on the thrusts that can be produced or more complex keepout constraints reflective of different CONOPS.

It is also of interest to better investigate the Pareto front between fuel use and inspection time. While the strategy presented of training multiple policies with different penalty factors α is effective, a better solution would be to encode the priority of time versus fuel efficiency as part of the MDP, allowing operators to make situation-dependent calls on which should be prioritized, moving the solution along the Pareto front.

Finally, multiagent inspection tasks present a more complex but interesting problem formulation. The shield can be extended to prevent collisions between multiple servicers, and the agents can be trained to coordinate their inspection tasks to maximize the number of points inspected in a given time. Both centralized and decentralized control strategies can be considered for this problem.

Conclusions

This work presents a novel approach to the problem of autonomous inspection of resident space objects (RSOs) with an impulsively maneuvering servicer spacecraft. The solutions leverage the natural relative motion of satellites in low Earth orbit (LEO) in order to minimize fuel consumption. To this end, a semi-Markov decision process (sMDP) formulation of the problem is introduced to account for variable-duration coasts following impulsive maneuvers. Training an agent on this sMDP results in a policy that implicity learns what regions of the RSO are illuminated in order to minimize the amount of inspection that must be performed.

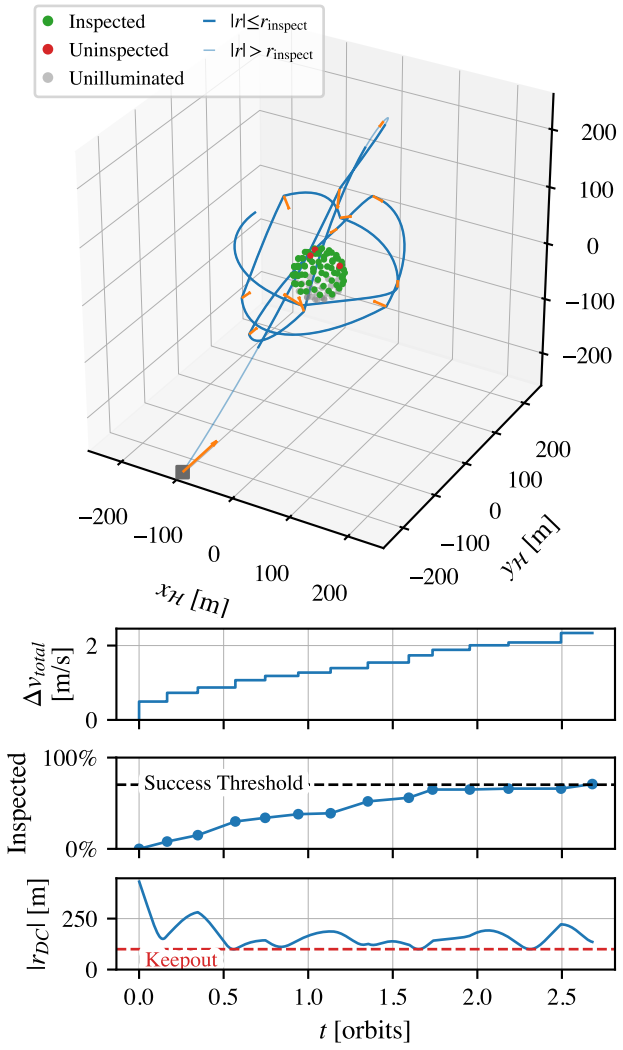

Figure 9: Shielded policy behavior with a 100 m keepout.

To ensure safety, an optimization-based shield is deployed around the policy; this guarantees that the servicer will not enter a keepout region around the RSO under natural dynamics over an infinite uncontrolled horizon. Combined, these methods result in a reliable and efficient method for autonomous inspection of RSOs.

Acknowledgments

This work is supported by a NASA Space Technology Graduate Research Opportunity (NSTGRO) grant, 80NSSC23K1182.

This work utilized the Alpine high-performance computing resource at the University of Colorado Boulder. Alpine is jointly funded by the University of Colorado Boulder, the University of Colorado Anschutz, Colorado State University, and the National Science Foundation (award 2201538).

Appendix: Constraint Derivation

The final constraint in the optimization-based shield, equation 14, guarantees secularly increasing separation or periodic motion between the servicer and the RSO. The general solution to the Clohessy-Wiltshire-Hill (CWH) equations (10) are given by

$$x(t) = A_0 \cos(nt + \alpha) + x_{\text{off}} \tag{17}$$

$$y(t) = -2A_0 \sin(nt + \alpha) - \frac{3}{2} nt x_{\text{off}} + y_{\text{off}} \tag{18}$$

$$z(t) = B_0 \cos(nt + \beta) \tag{19}$$

Note that only the y component (18) has a secular term. Two cases exist for safety:

1. The relative motion is periodic and safe. In this case, the secular term coefficient must be zero to ensure periodicity (up to any perturbations):

$$-\frac{3}{2} n x_{\text{off}} = 0 \implies x_{\text{off}} = 0 \tag{20}$$

The safety of the periodic motion is guaranteed by the collision constraint (13), as long as the period is sufficiently long:

$$t_{\max} - t_0 \geq T_{\text{orbit}} \tag{21}$$

2. Alternatively, the secular term is nonzero and dominates the motion. In this case, the magnitude of the local extrema must be greater than the keepout radius and be growing:

$$\left| -\frac{3}{2} n t_{\max} x_{\text{off}} \pm 2A_0 + y_{\text{off}} \right| \geq R_C + R_D + \epsilon \tag{22}$$

$$\frac{d}{dt} \left| -\frac{3}{2} n t x_{\text{off}} \pm 2A_0 + y_{\text{off}} \right|_{t=t_{\max}} > 0 \tag{23}$$

It is worth noting that this constraint on $y(t)$ is more strict than necessary, as it is possible that the periodic motion of $x(t)$ and $z(t)$ are enough by themselves to prevent collision with arbitrary behavior in $y(t)$.

The constraints 20 $\vee$ (22 $\wedge$ 23) reduce to

$$\frac{3}{2} t_{\max} n x_{\text{off}}^2 \geq -x_{\text{off}} ((R_C + R_D + \epsilon) + 2A_0 + y_{\text{off}}) \tag{24}$$

and

$$\frac{3}{2} t_{\max} n x_{\text{off}}^2 \geq -x_{\text{off}} (-(R_C + R_D + \epsilon) - 2A_0 + y_{\text{off}}) \tag{25}$$

which accounts for all possible combinations of signs.

References

Alshiekh, M.; Bloem, R.; Ehlers, R.; Könighofer, B.; Niekum, S.; and Topcu, U. 2018. Safe Reinforcement Learning via Shielding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Aurand, J.; Cutlip, S.; Lei, H.; Lang, K.; and Phillips, S. 2023. Exposure-Based Multi-Agent Inspection of a Tumbling Target Using Deep Reinforcement Learning. arXiv:2302.14188.

Bernhard, B.; Choi, C.; Rahmani, A.; Chung, S.-J.; and Hadaegh, F. 2020. Coordinated Motion Planning for On-Orbit Satellite Inspection Using a Swarm of Small-Spacecraft. In *2020 IEEE Aerospace Conference*, 1–13.

Bradtke, S.; and Duff, M. 1994. Reinforcement Learning Methods for Continuous-Time Markov Decision Problems. In *Advances in Neural Information Processing Systems*, volume 7. MIT Press.

Dunlap, K.; Hamilton, N.; and Hobbs, K. L. 2025. Deep Reinforcement Learning for Scalable Multiagent Spacecraft Inspection. In *AAS/AIAA Space Flight Mechanics Meeting*.

Gaias, G.; and D'Amico, S. 2015. Impulsive Maneuvers for Formation Reconfiguration Using Relative Orbital Elements. *Journal of Guidance, Control, and Dynamics*, 38(6): 1036–1049.

Guffanti, T.; Gammelli, D.; D'Amico, S.; and Pavone, M. 2024. Transformers for Trajectory Optimization with Application to Spacecraft Rendezvous. In *2024 IEEE Aerospace Conference*, 1–13. Big Sky, MT, USA: IEEE. ISBN 9798350304626.

Harris, A.; Valade, T.; Teil, T.; and Schaub, H. 2022. Generation of Spacecraft Operations Procedures Using Deep Reinforcement Learning. *Journal of Spacecraft and Rockets*, 59(2): 611–626.

Herrmann, A.; and Schaub, H. 2022. Reinforcement Learning for Small Body Science Operations. In *AAS Astrodynamics Specialist Conference*. Charlotte, North Carolina.

Kenneally, P. W.; Piggott, S.; and Schaub, H. 2020. Basilisk: A Flexible, Scalable and Modular Astrodynamics Simulation Framework. *Journal of Aerospace Information Systems*, 17(9): 496–507.

Koenig, A. W.; and D'Amico, S. 2018. Robust and Safe N-Spacecraft Swarming in Perturbed Near-Circular Orbits. *Journal of Guidance, Control, and Dynamics*, 41(8): 1643–1662.

Kuhl, W.; Wang, J.; Eddy, D.; and Kochenderfer, M. 2025. Markov Decision Processes for Satellite Maneuver Planning and Collision Avoidance. arXiv:2501.02667.

Lauinger, T.; and Ulrich, S. 2025. Path Planning for Optimal Coverage of Orbiting Space Structures Using Lissajous Curves. In *AAS/AIAA Space Flight Mechanics Meeting*.

Lei, H.; Lippay, Z. S.; Zaman, A.; Aurand, J.; Maghareh, A.; and Phillips, S. 2025. Stability Analysis of Deep Reinforcement Learning for Multi-Agent Inspection in a Terrestrial Testbed. In *AIAA SCITECH 2025 Forum*. Orlando, FL.

Lei, H. H.; Shubert, M.; Damron, N.; Lang, K.; and Phillips, S. 2024. Deep Reinforcement Learning For Multi-Agent Autonomous Satellite Inspection. In Sandnas, M.; and Spencer, D. B., eds., *Proceedings of the 44th Annual American Astronautical Society Guidance, Navigation, and Control Conference, 2022*, 1391–1412. Cham: Springer International Publishing. ISBN 978-3-031-51928-4.

Liang, E.; Liaw, R.; Moritz, P.; Nishihara, R.; Fox, R.; Goldberg, K.; Gonzalez, J. E.; Jordan, M. I.; and Stoica, I. 2018. RLlib: Abstractions for Distributed Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, 3053–3062.

Morgan, D.; Chung, S.-J.; and Hadaegh, F. Y. 2014. Model Predictive Control of Swarms of Spacecraft Using Sequential Convex Programming. *Journal of Guidance, Control, and Dynamics*, 37(6): 1725–1740.

Morgan, D.; Subramanian, G. P.; Chung, S.-J.; and Hadaegh, F. Y. 2016. Swarm Assignment and Trajectory Optimization Using Variable-Swarm, Distributed Auction Assignment and Sequential Convex Programming. *The International Journal of Robotics Research*, 35(10): 1261–1285.

Nakka, Y. K.; Hönig, W.; Choi, C.; Harvard, A.; Rahmani, A.; and Chung, S.-J. 2022. Information-Based Guidance and Control Architecture for Multi-Spacecraft On-Orbit Inspection. *Journal of Guidance, Control, and Dynamics*, 45(7): 1184–1201.

Sabatini, M.; Volpe, R.; and Palmerini, G. 2020. Centralized Visual Based Navigation and Control of a Swarm of Satellites for On-Orbit Servicing. *Acta Astronautica*, 171: 323–334.

Schaub, H.; and Junkins, J. L. 2018. *Analytical Mechanics of Space Systems*. AIAA Education Series. Reston, VA: American Institute of Aeronautics and Astronautics, Inc, fourth edition edition. ISBN 978-1-62410-521-0.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Stephenson, M.; Mantovani, L.; and Cheval, A. 2025. Quantifying the Optimality of a Distributed Rl-Based Autonomous Earth-Observing Constellation. In *AAS GN&C Conference*. Breckenridge, CO.

Stephenson, M.; and Schaub, H. 2024a. Reinforcement Learning For Earth-Observing Satellite Autonomy With Event-Based Task Intervals. In *AAS Rocky Mountain GN&C Conference*. Breckenridge, CO.

Stephenson, M. A.; and Schaub, H. 2024b. BSK-RL: Modular, High-Fidelity Reinforcement Learning Environments for Spacecraft Tasking. In *75th International Astronautical Congress*. Milan, Italy: IAF.

Sutton, R. S.; and Barto, A. 2018. *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning. Cambridge, Massachusetts London, England: The MIT Press, 2 edition. ISBN 978-0-262-03924-6.

Takubo, Y.; Guffanti, T.; Gammelli, D.; Pavone, M.; and D'Amico, S. 2024. Towards Robust Spacecraft Trajectory Optimization via Transformers. arXiv:2410.05585.

Towers, M.; Terry, J. K.; Kwiatkowski, A.; Balis, J. U.; de Cola, G.; Deleu, T.; Goulão, M.; Kallinteris, A.; KG, A.; Krimmel, M.; Perez-Vicente, R.; Pierré, A.; Schulhoff, S.; Tai, J. J.; Tan, A. J. S.; and Younis, O. G. 2023. Gymnasium.

Van Wijk, D.; Dunlap, K.; Majji, M.; and Hobbs, K. 2024. Safe Spacecraft Inspection via Deep Reinforcement Learning and Discrete Control Barrier Functions. *Journal of Aerospace Information Systems*, 21(12): 996–1013.

van Wijk, D.; Dunlap, K.; Majji, M.; and Hobbs, K. L. 2023. Deep Reinforcement Learning for Autonomous Spacecraft Inspection Using Illumination. arXiv:2308.02743.