



Learning-based framework for multistage space debris collision warning and avoidance

Liwen Yang^a, Jikai Wang^a, Ming Xu^{a,b}, Xue Bai^{a,b,*}

^a School of Astronautics, Beihang University, Beijing 100191, China

^b Key Laboratory of Spacecraft Design Optimization and Dynamic Simulation Technology, Ministry of Education, Beijing, China

ARTICLE INFO

Communicated by: Cummings Russell

Keywords:

Multistage debris collisions
Collision warning and avoidance
Reinforcement learning

ABSTRACT

The rapid increase in human space activities has resulted in a growing accumulation of space debris in low Earth orbit, which poses a serious threat to spacecraft safety. The large amount of space debris has expanded the collision problem from single-stage encounters to multistage ones, where a spacecraft may face new debris after avoiding an initial collision. To address this multistage collision problem, this paper proposes a learning-based framework for autonomous early warning and multistage collision avoidance of space debris. Four types of collision scenarios are constructed, including single-debris collision, strict multistage collision, probability-based multistage collision, and complex multistage collision. Based on the characteristics of orbital dynamics, a complete two-stage screening mechanism for multistage collision warning is developed, which filters out high-risk debris through static orbital altitude screening and dynamic time-of-crossing-node analysis. On this basis, a multi-objective optimization framework is designed to jointly minimize collision probability, fuel consumption, and orbital deviation. Three learning-based algorithms—Monte Carlo Tree Search (MCTS), Cross-Entropy Method (CEM), and Proximal Policy Optimization (PPO)—are implemented and compared against the numerical baseline Powell method across scenarios of varying complexity, under both the two-body and J_2 -perturbed dynamical models. The proposed framework successfully achieves autonomous collision avoidance in complex multistage collision scenarios with over 1,00 debris objects from real and probabilistic sources. Among the tested methods, MCTS and CEM achieve over 99.9% risk reduction and superior fuel efficiency, while PPO ensures near-instantaneous inference under 25 s, balancing optimal safety with real-time responsiveness in perturbed environments.

1. Introduction

In recent years, the rapid proliferation of commercial satellite launches has led to explosive growth in the population of space debris in low Earth orbit (LEO) [1]. This escalating density poses a severe threat to the safety and sustainability of spacecraft operations, potentially triggering chain reactions that further degrade the space environment [2]. However, most recent collision avoidance research treats encounters as isolated events, ignoring that in reality avoidance of one object can increase risk with others. Therefore, the need for early warning and avoidance of multistage debris collision is increasingly urgent. This task can generally be divided into two components: collision warning and collision avoidance. In this section, collision warning is first introduced, followed by a discussion of early avoidance methods for single-debris encounters and recent advancements in multi-debris collision

avoidance.

Collision warning is a prerequisite for collision avoidance, mainly based on accurate orbit prediction and reasonable collision risk assessment. Orbit prediction relies on analytical and numerical propagation methods formulated in appropriate orbital reference frames. Analytical models such as SGP4/SDP4 provide computational efficiency [3,4], whereas high-fidelity numerical propagators like GEODYN and HPOP achieve greater accuracy by modeling complex perturbations [5]. Appropriate prediction therefore depends on balancing computational cost and uncertainty modeling [6]. Once orbit prediction has been performed, early warning of collision risk becomes a critical step. Early collision warning often relied on the "box region" approach, where warning zones are defined around a spacecraft [7]. This method is straightforward, but it tends to be overly conservative, leading to unnecessary maneuvers. To mitigate computational overhead, Hoots et al.

* Corresponding author at: School of Astronautics, Beihang University, Beijing 100191, China.

E-mail address: baixue1002@buaa.edu.cn (X. Bai).

<https://doi.org/10.1016/j.ast.2026.112305>

Received 25 October 2025; Received in revised form 27 March 2026; Accepted 5 April 2026

Available online 6 April 2026

1270-9638/© 2026 Elsevier Masson SAS. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

developed an efficient analytical method to determine future close approaches and use a series of prefilters to eliminate many unthreatening cases from further consideration [8]. Subsequent studies have largely expanded on Hoots' approach, and some scholars have conducted research on the method for determining the closest distance of the confocal Keplerian orbits [9–11]. More recently, Casanova et al. extended the triple-filter method to continuous-thrust collision avoidance [12]. Operational collision warning services have also been developed by space agencies to support mission operations. For example, the ESA Space Debris Office provides a dedicated collision avoidance service that monitors conjunction events, computes collision probabilities, and supports maneuver decision-making for operational missions [13]. To conduct collision warnings more accurately, probabilistic collision probability evaluation methods have gained prominence. In the late 20th century, Chan proposed an analytical approximation of collision probability [14], while Foster developed a two-dimensional polar coordinate integration method to simplify numerical calculations [15]. Patera later introduced a decoupled integration method for the velocity vector, improving computational efficiency and accuracy [6], and Alfano proposed a numerical method for collision probability computation for spherical objects under uncertainty [16]. Chobotov et al. employed Monte Carlo simulations combined with adjusted orbital models considering Earth's oblateness to evaluate the collision probability [17]. More recent contributions include Bai's proposal of an enhanced algorithm for estimating orbit prediction errors and their effects on collision probability [18]. Meanwhile, Sun et al. compared the box region and probabilistic methods and demonstrated that the latter was more accurate and efficient under equivalent constraints [19]. Their findings showed that probabilistic methods are better suited to onboard systems with limited fuel and computational resources, enabling smarter maneuver decisions. However, the reliability of these methods is strongly affected by the uncertainty of tracking data. To improve the accuracy of risk assessment, recent studies have suggested equipping satellites with onboard sensors to autonomously observe and track potentially hazardous objects shortly before conjunction, enabling refined state estimation and reducing false alarm rates [20,21].

If a predicted conjunction is assessed as high risk, a collision avoidance maneuver must be initiated to adjust the spacecraft trajectory. Early studies on collision avoidance focused on single-debris encounters and primarily relied on optimization methods. Alfano proposed an instantaneous maneuver method for quick collision mitigation [22]. Mueller framed the avoidance planning as a nonlinear optimal control problem and solved it using discretization-based nonlinear programming, enhancing robustness and increasing post-maneuver safe time [23,24]. Kelly proposed an optimal strategy based on minimum safe distance constraints using nonlinear optimization techniques [25], while Lewis explored the Pareto frontier for fuel-optimal maneuvering and derived closed-form expressions for delta- v under impulsive action [26]. Bombardelli and Hernando-Ayuso investigated optimal analytical impulsive avoidance for LEO satellites [27,28], while recent study by Gonzalo et al. proposed an analytical framework for designing collision avoidance maneuvers to optimize spacecraft displacement and collision probability [29]. Armellin have further explored convex optimization formulations for efficient multi-impulse collision avoidance maneuver design [30]. Palermo analyzed efficient numerical techniques for both impulsive and low-thrust maneuver planning, providing valuable insights for practical mission design [31]. These methods have achieved remarkable results in single-target obstacle avoidance. However, in high-density continuous encounter scenarios, they encounter difficulties in autonomously deciding the timing of maneuvers, may face computational bottlenecks, and cannot consider complex multi-coupled constraints during a single maneuver.

In order to effectively deal with the multi threat posed by space debris, spacecraft need to have more autonomous and efficient decision-making capabilities. Recent advancements include multi-pulse

maneuver avoidance based on differential algebra and polynomial programming [32], as well as improved search strategies with adaptive depth and learning priors [33]. Seong and Kim compared the performance of genetic algorithms, particle swarm optimization, differential evolution, and simulated annealing for multi-threat avoidance scenarios, with differential evolution showing superior fuel-efficiency under both single- and multi-objective formulations [34,35]. However, these methods typically require a priori modeling and may offer limited adaptability to changing environments. In contrast, reinforcement learning (RL), has emerged as powerful tools for addressing these challenges. Yuan applied a Markov Decision Process (MDP) framework to study RL-based spacecraft rendezvous and collision avoidance under complex constraints [36]. Qu proposed a collision control strategy using a deep deterministic policy gradient (DDPG) algorithm with adaptive policy gradients [37]. Liu proposed a space debris avoidance strategy based on reinforcement learning for proximal policy optimization (PPO) algorithm [38]. Mu et al. further introduced a safe reinforcement learning framework for autonomous satellite avoidance, formulating the problem as a Constrained MDP and solving it with a penalty-based PPO algorithm. The use of Long Short-Term Memory (LSTM) networks enabled the model to handle variable numbers of debris, significantly improving decision robustness and avoidance performance under fuel and collision constraints [39]. These efforts validate the effectiveness of multi-agent RL in adversarial and highly dynamic space environments. Solomon and Paduraru compared PPO and Deep Q-Network (DQN) algorithms for low-thrust trajectory planning, offering practical insights into algorithm selection for different mission types [40]. Insights from other domains, such as maritime navigation, have also enriched research in space debris avoidance. For instance, Woo and Kim proposed a DQN-based obstacle avoidance method for unmanned surface vessels. Their method used a grid-based environmental representation combined with a semi-Markov decision process to enable rule-constrained dynamic navigation [41]. These methods demonstrate strong capabilities in handling multi-debris environments. However, in chain-type multi-debris scenarios, avoiding one collision may increase the likelihood of subsequent hazardous encounters. Existing approaches primarily address the growing number of objects but often overlook the spatio-temporal coupling and cascading risks inherent in sequential encounters.

In summary, most existing research still exhibits limitations and room for improvement, mainly in the following two aspects: 1) Spatio-temporal coupling and cascading collision risk in multi-debris scenarios is rarely modeled in a unified manner. 2) Emerging multi-object avoidance methods lack an integrated early-warning and prescreening framework, resulting in a less complete operational pipeline compared with mature single-debris systems.

To address these two issues, this paper constructs four types of distinct space debris collision scenarios, presents a complete framework for autonomous early warning and avoidance of multistage space collision, including a multistage collision warning framework with a two-stage screening architecture and a learning-based multistage collision avoidance system. During the warning and avoidance process, debris are first filtered based on orbital altitude and intersection time. Collision probabilities are then computed accurately using orbital propagation errors. Finally, learning-based methods are employed for optimization in collision avoidance. The effectiveness of three approaches, Monte Carlo Tree Search (MCTS), Cross-Entropy Method (CEM), and Proximal Policy Optimization (PPO), is evaluated and compared. The novelty of this work lies in a unified framework that combines multistage scenario design, a scaling-oriented comparison and a multistage-tailored screening process. The main contributions of this study are as follows:

- 1) Regarding the scenario construction issue, a progressive multistage collision scenario framework is proposed. Unlike traditional single-stage settings, this framework systematically constructs multistage

collision scenarios by considering the coupling relationships among debris objects, enabling the simulation of situations where a spacecraft encounters new debris after avoiding an initial collision.

- 2) Under the proposed multistage scenarios, we conduct a unified scaling-oriented comparison among heterogeneous decision-making approaches, including Powell, MCTS, CEM, and PPO, focusing on their scalability and performance under different kinds of scenarios and providing useful insights for the community.
- 3) Building on existing collision warning and avoidance paradigms, we incorporate a two-stage early warning screening mechanism into a learning-based multistage collision avoidance framework. Compared with direct learning-based avoidance, this design reduces problem scale while preserving global collision information, improving computational efficiency and scalability in dense debris environments.

The rest of this paper is organized as follows: Section 2 introduces the orbital dynamics model and builds the system framework; Section 3 demonstrates the existence of multistage collisions and constructs four representative multistage collision scenarios; Sections 4 and 5 respectively elaborate on the early warning and autonomous avoidance methods for multistage collisions; Section 6 conducts a comparative analysis of the experimental results using a multi-dimensional unified framework; Finally, Section 7 summarizes the research conclusions and looks forward to future work directions.

2. Dynamics and system framework

2.1. Orbital dynamics and covariance propagation

To simulate multistage encounters and provide a robust basis for the proposed reinforcement learning framework, we primarily employ a two-body dynamical model for large-scale training efficiency, while concurrently conducting feasibility validation against higher-fidelity models that incorporate Earth's perturbations. The state of a space object is defined as $\mathbf{x} = [\mathbf{r}^T, \mathbf{v}^T]^T$ in the Earth-Centered Inertial (ECI) J2000 frame. The total acceleration acting on a space object is formulated as:

$$\ddot{\mathbf{r}} = \mathbf{a}_{2b} + \mathbf{a}_{J2} + \mathbf{a}_{drag} \quad (1)$$

The fundamental motion is governed by Keplerian dynamics, denoted as $\mathbf{a}_{2b} = -(\mu/r^3)\mathbf{r}$. For long-term screening efficiency, an analytical propagator utilizing a critical position-dependent term, $\partial \mathbf{a}_D / \partial \mathbf{r}$ is implemented to solve Kepler's equation, thereby avoiding the accumulation of numerical integration errors in unperturbed scenarios. To further account for the Earth's oblateness, which significantly affects nodal precession and relative phasing in LEO, the J_2 zonal harmonic acceleration is incorporated as follows:

$$\mathbf{a}_{J2} = \frac{3J_2\mu R_e^2}{2r^5} \begin{bmatrix} x \left(\frac{5z^2}{r^2} - 1 \right) \\ y \left(\frac{5z^2}{r^2} - 1 \right) \\ z \left(\frac{5z^2}{r^2} - 3 \right) \end{bmatrix} \quad (2)$$

Furthermore, for objects residing in lower LEO regimes, atmospheric drag represents the primary non-conservative force causing timing desynchronization. In this study, an exponential density model $\rho = \rho_0 \exp(-(h-h_0)/H)$ is adopted. The resulting acceleration is calculated as:

$$\mathbf{a}_{drag} = -\frac{1}{2} \rho B_c \mathbf{v}_{rel} \mathbf{v}_{rel} \quad (3)$$

where $B_c = C_d A / m$ denotes the ballistic coefficient and $\mathbf{v}_{rel}$ represents the velocity relative to the rotating atmosphere. These combined effects

ensure that the proposed framework maintains a balance between computational throughput and physical fidelity.

To quantify the collision risk throughout the multistage encounter process, the evolution of state uncertainty must be rigorously modeled alongside the nominal trajectory. The state covariance $\mathbf{P}(t)$ is propagated from an initial epoch t_0 to the Time of Closest Approach (TCA), denoted as t_c , utilizing the State Transition Matrix (STM) $\Phi(t, t_0)$:

$$\Phi(t, t_0) = \begin{bmatrix} \Phi_{rr} & \Phi_{rv} \\ \Phi_{vr} & \Phi_{vv} \end{bmatrix} \quad (4)$$

where $\Phi_{rv}(t, t_0)$ specifically maps the velocity impulse at t_0 to the position deviation at t . This matrix represents the sensitivity of the state at time t to small perturbations at t_0 and is governed by the variational differential equations:

$$\dot{\Phi}(t, t_0) = \mathbf{A}(t)\Phi(t, t_0), \quad \Phi(t_0, t_0) = \mathbf{I}_{6 \times 6} \quad (5)$$

where $\mathbf{A}(t) = \partial \dot{\mathbf{x}} / \partial \mathbf{x}$ is the plant matrix, partitioned into 3×3 blocks as:

$$\mathbf{A}(t) = \begin{bmatrix} \mathbf{0}_{3 \times 3} & \mathbf{I}_{3 \times 3} \\ \frac{\partial \mathbf{a}}{\partial \mathbf{r}} & \frac{\partial \mathbf{a}}{\partial \mathbf{v}} \end{bmatrix} \quad (6)$$

In this framework, the acceleration gradients $\partial \mathbf{a} / \partial \mathbf{r}$ and $\partial \mathbf{a} / \partial \mathbf{v}$ are derived via a high-fidelity analytical Jacobian to ensure numerical stability and suppress the accumulation of linearization errors. Specifically, the gravity gradient contribution from point-mass dynamics is formulated as:

$$\partial \mathbf{a}_{2b} / \partial \mathbf{r} = -(\mu / r^3) \mathbf{I} + (3\mu / r^5) \mathbf{r} \mathbf{r}^T \quad (7)$$

For perturbed scenarios, the gravity gradient is augmented by the Hessian of the J_2 potential to account for non-spherical effects like nodal precession. Non-conservative forces are similarly incorporated through a high-fidelity atmospheric drag gradient. This gradient consists of two primary components: a critical position-dependent term $\partial \mathbf{a}_D / \partial \mathbf{r}$, and a velocity-dependent term $\partial \mathbf{a}_D / \partial \mathbf{v}$. Finally, by integrating these analytical derivatives into the variational equations, the state covariance is mapped to the Time of Closest Approach (TCA) using the linear transformation:

$$\mathbf{P}(t_c) = \Phi(t_c, t_0) \mathbf{P}(t_0) \Phi^T(t_c, t_0) \quad (8)$$

2.2. Multistage collision warning and avoidance framework

To address the increasingly complex challenges posed by multistage space debris collisions in LEO, we propose a systematic multistage collision early warning and autonomous avoidance framework. This framework integrates scenario construction, high-risk target screening, and optimization-based maneuver planning into a unified architecture. Fig. 1 shows the overall system framework.

As shown in the upper left part of Fig. 1, the system begins with the construction of representative multistage collision scenarios, forming the basis for evaluating avoidance strategies. The simplest case, single debris collision, models an isolated encounter and serves as a baseline. The strict multistage collision scenario introduces a new threat after the initial avoidance, simulating deterministic sequential risks. The probability-based multistage collision increases complexity by generating large numbers of debris based on probabilistic distributions, amplifying the likelihood of secondary collisions. The complex multistage collision combines the above three with real orbital data and scales the debris population to over 100 objects. This high-fidelity environment presents realistic and computationally intensive challenges. Together, these four scenarios provide a scalable test-bed for developing and benchmarking autonomous collision avoidance algorithms under progressively more demanding space conditions.

As shown in the lower left part of Fig. 1, to efficiently manage the computational cost associated with high-dimensional avoidance

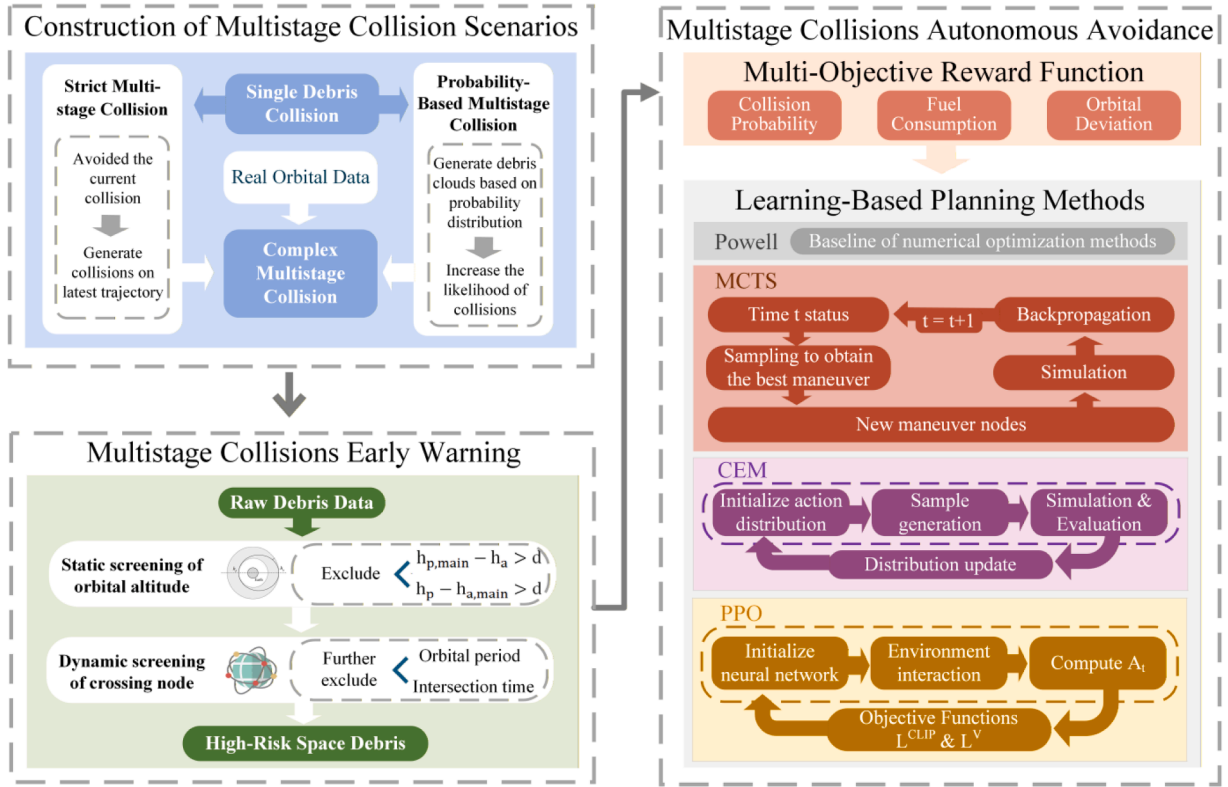


Fig. 1. Overall framework for autonomous multistage debris collision warning and avoidance, comprising three core stages: (i) Scenario construction (Section 3), (ii) Two-stage screening early warning (Section 4), and (iii) Multi-objective optimization and learning-based maneuver planning (Section 5).

problems, a two-stage early warning and screening mechanism is incorporated. This module acts as a preprocessing filter, selectively excluding low-risk debris before optimization begins. The first stage, static screening, evaluates the geometric relationship between the satellite and surrounding debris based on orbital altitude differences. If the radial and vertical separation thresholds are satisfied, corresponding debris objects are categorized as safe and removed from further consideration. In the second stage, a dynamic screening method is applied, which analyzes time-of-crossing-node characteristics using orbit propagation. This method assesses the timing, spatial proximity, and intersection windows between the target satellite and potential threats to determine dynamic risk levels. Debris objects with imminent intersection windows and short encounter distances are labeled as high-risk targets, while others are excluded. Next, precise collision probability calculations are carried out for the selected candidate space individuals. Among the candidate individuals, if the collision probability of any candidate exceeds the preset safety threshold, the subsequent avoidance maneuver planning process will be triggered. All the candidate individuals obtained by the screening mechanism will be passed to the downstream optimization module. This hierarchical screening reduces the number of debris involved in downstream optimization, leading to substantial savings in computational resources and enabling faster convergence of maneuver planning algorithms.

As shown in the right part of Fig. 1, the core of the autonomous avoidance strategy is defined by a multi-objective optimization formulation. The optimization framework jointly considers three critical factors: collision probability, fuel consumption, and orbital deviation. Collision probability reflects the primary objective of ensuring satellite safety and is evaluated using probabilistic proximity models. Fuel consumption corresponds to the total Δv required for the maneuver, which impacts satellite lifetime and mission cost. Orbital deviation quantifies the impact of the maneuver on the satellite's trajectory and mission continuity. These objectives are combined into a weighted reward

function in learning setup. Within the proposed framework, an avoidance maneuver is suggested by solving a multi-objective decision problem in which the primary objective is to reduce collision probability below a prescribed safety threshold, while secondary objectives include minimizing fuel consumption and limiting deviation from the nominal orbit. An avoidance maneuver is considered successful when the post-maneuver collision probabilities of all remaining threatening objects fall below the safety threshold, while fuel consumption and orbital deviation remain within acceptable limits. To solve the optimization problem under various conditions, three learning-based methods are implemented: MCTS, CEM, and PPO. Each method is embedded within the system to autonomously generate avoidance maneuvers tailored to its respective decision-making paradigm. MCTS is a tree-based planning algorithm that incrementally constructs a search tree by simulating potential maneuvers, evaluating outcomes, and backpropagating value estimates. CEM, by contrast, is a sampling-based algorithm that generates maneuver candidates from a probabilistic action distribution, evaluates them through simulation, and updates the distribution based on high-performing samples. PPO utilizes a deep neural network to learn an optimal policy through gradient ascent on a clipped surrogate objective.

The complete system architecture possesses strong modularity and adaptability. Scenario construction ensures diversity of training and evaluation environments, early warning mechanisms reduce computational complexity, and learning-based algorithms offer scalable solutions for multistage collision avoidance. Together, these components form a cohesive and intelligent system capable of autonomously managing multistage collision risks. The integration of physical modeling with learning-based control further enables this framework to serve as a foundational tool for the future deployment of autonomous collision avoidance in intelligent satellite constellations and large-scale space infrastructure.

3. Existence of multistage collisions and scenario construction

To evaluate the proposed collision avoidance framework, it is crucial to first establish the physical existence and operational relevance of multistage collision threats. While historical collision avoidance primarily focused on isolated events, the increasing density of objects in LEO—particularly following massive fragmentation events—has significantly raised the probability of sequential or “cascading” encounters. Based on the existence of these threats, we construct multistage collision scenarios where a spacecraft may encounter subsequent threats from other debris after avoiding an initial collision. These scenarios reflect the dynamic and high-risk environment of LEO, where space debris is densely distributed and collision threats can evolve rapidly. The scenarios involve the following 4 types: single debris collision, strict multistage collision, probability-based multistage collision, and complex multistage collision. Single debris collision is the simplest collision scenario, which is able to serve as effective pre-training environments for more complex cases. In order to build complex “multistage” collision that satellite may collide with new debris after avoiding the initial one, strict multistage collision scenario is constructed. Here, “strict” denotes a deterministic encounter where a secondary threat is precisely positioned to intersect the spacecraft’s post-maneuver trajectory which could be seen in Section 3.3 for the detailed formulation. Given the complexity of constructing strict multistage collision scenarios, we simplify the construction method and design a probability-based multistage collision scenario which enables rapid setup. Finally, the above two multistage collision scenarios are fused with real orbit data to construct a complex multistage collision scenario which is the most challenging. This progressive design not only improves training efficiency for reinforcement

learning but also enhances the generalization ability of the avoidance policy in realistic and dynamic orbital environments.

3.1. Empirical verification of multistage collision existence

To address concerns regarding the realism of multistage threats, we conducted a case study using historical data from the Iridium 33/ Cosmos 2251 collision event [42]. This event generated a vast debris cloud, creating a high-risk environment where a single maneuver can inadvertently lead to secondary hazards.

As shown in Fig. 2, We simulated a hypothetical satellite positioned in a representative LEO orbit with the initial state in Table 1. By propagating this orbit against the TLE data of fragments from the Iridium-Cosmos event, we identified a multistage threat involving multiple fragments within a 3.5-h window.

As illustrated in Fig. 3, the dotted line represents the collision probability before the avoidance maneuver, while the solid line represents the collision probability after the avoidance maneuver. The satellite first encounters a primary threat from Iridium 33 Fragment 2, with a collision probability exceeding the 10^{-4} threshold [43]. SQP algorithm is used to get a numerical optimization-based maneuver to minimize the risk from Fragment 2. If the maneuver $\Delta v = [-2.3661, 0.8142, 5.6570] \text{ m/s}$ is executed at 600.19 s, the satellite’s post-maneuver trajectory significantly shifts. Critically, while this “optimal” maneuver successfully mitigates the primary threat, it causes a sharp increase in collision probabilities for Iridium 33 Fragment 3 and Cosmos 2251 Fragment 1, which is indicated by the bold lines in Fig. 3. This empirical evidence demonstrates that multistage collisions are not merely theoretical coincidences but a tangible risk in dense orbital environments where localized maneuvers

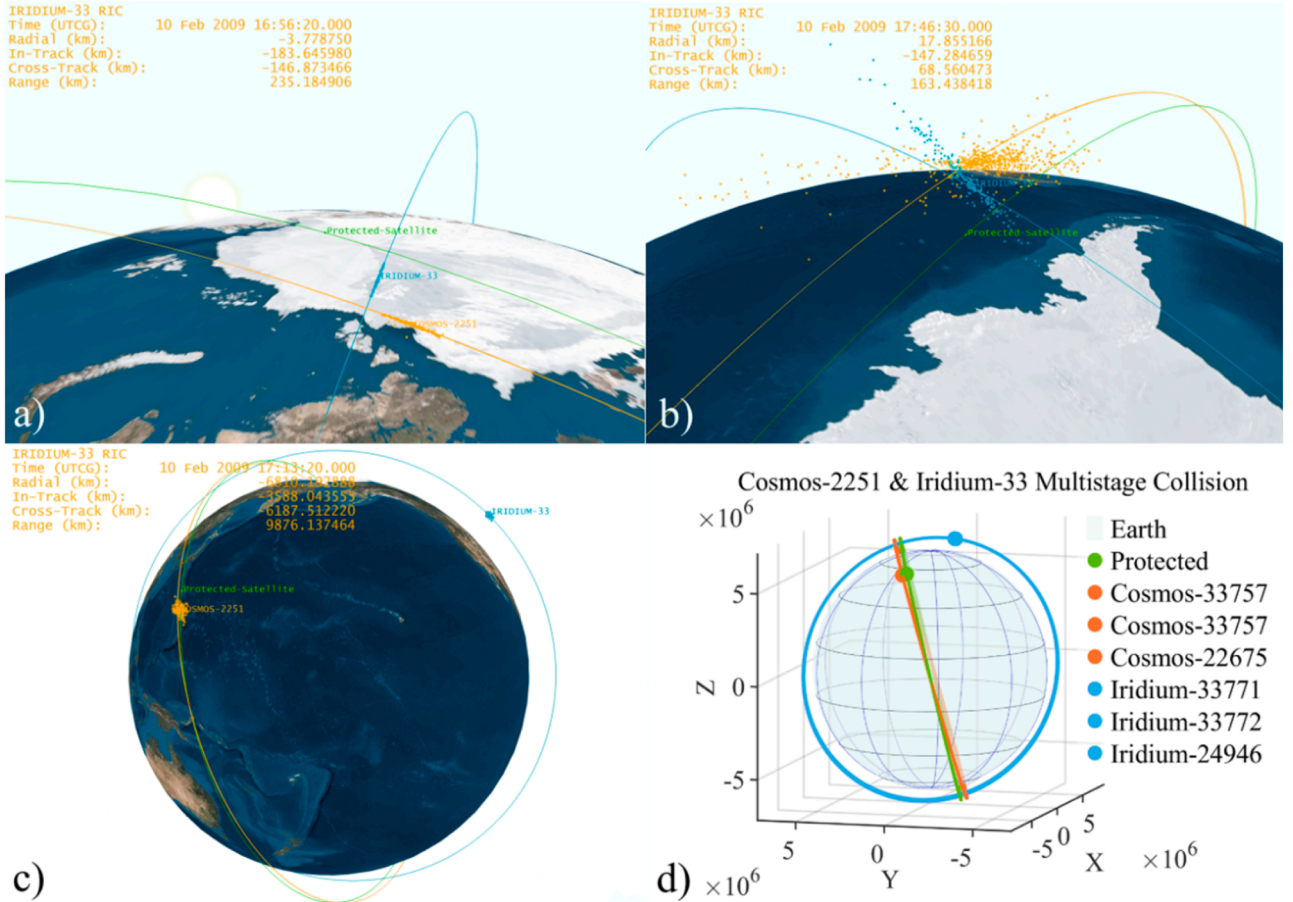


Fig. 2. Diagram of the multistage collision scenario for the Iridium 33/ Cosmos 2251 collision event: a) Primary collision at event start b) Secondary collision after half orbital period c) Orbital overview d) Selected debris for simulation.

Table 1
Orbital elements of the protected spacecraft orbit.

t (J2000)	a (m)	e	i (rad)	Ω (rad)	ω (rad)	$M(t)$ (rad)
3328.2116018519	7158504.927957	0.0008096739	1.3369714084	0.2968032745	1.4977189465	0.6818807705

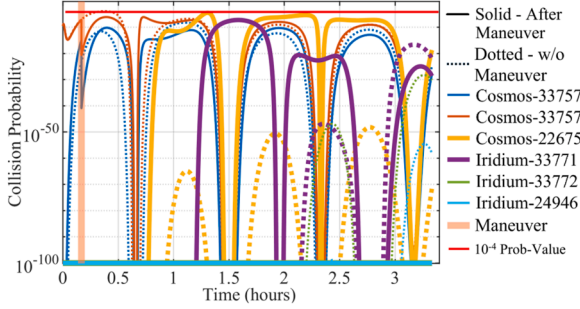


Fig. 3. Comparison of collision probabilities across multiple encounter stages in the multistage collision scenario for the Iridium 33/ Cosmos 2251.

may exacerbate secondary risks.

3.2. Single debris collision

This scenario serves as the most fundamental test environment, involving a single piece of space debris on a trajectory that brings it into close proximity with the spacecraft. The purpose of this case is to validate the agent's basic collision avoidance capability under controlled and low-complexity conditions. The debris orbit is initialized such that it intersects the spacecraft's path within a predictable time window, and the spacecraft must execute a single impulsive maneuver to avoid the collision. This setup provides a benchmark for comparing more complex scenarios and serves as the initial stage of training for the reinforcement learning model. Fig. 4 shows the effect of a single debris collision scenario.

3.3. Strict multistage collision

In the strict multistage collision scenario, the spacecraft, after successfully avoiding an initial collision, is immediately confronted with a new threat from a different debris object. Unlike the Monte Carlo method used in current research for randomly injecting collisions, this scenario is constructed as a strictly coupled sequential conjunction chain, where each subsequent threat is deterministically injected based on the spacecraft's optimal response to previous threats. As shown in the

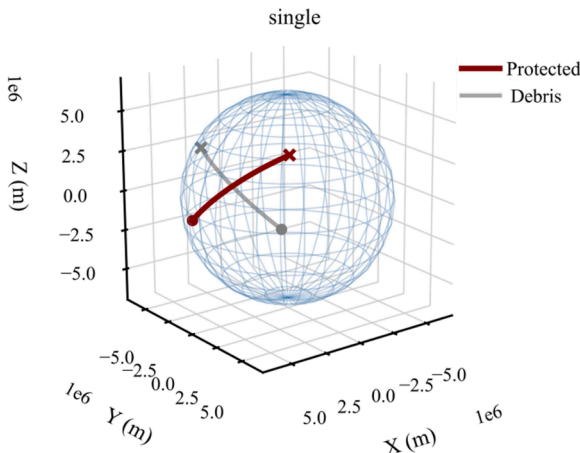


Fig. 4. Diagram of single debris collision.

pseudo-code in Fig. 5, the generation follows an iterative process. For each stage n , the environment calculates a reference avoidance trajectory using an analytical impulsive maneuver framework [27–29]. Specifically, let $\mathbf{x}_p(t_0)$ be the initial state of the protected satellite. For a collision at $t_{c,n}$ and a maneuver executed at $t_{m,n}$, the sensitivity of the satellite's position at $t_{c,n}$ relative to an impulsive velocity change $\Delta \mathbf{v}$ at $t_{m,n}$ is linearized using the Cartesian STM $\Phi(t_{c,n}, t_{m,n})$:

$$\delta \mathbf{r}_p(t_{c,n}) \approx \Phi_{rv}(t_{c,n}, t_{m,n}) \Delta \mathbf{v}(t_{m,n}) = \mathbf{Z}_{3d} \Delta \mathbf{v} \quad (9)$$

where $\mathbf{Z}_{3d} = \Phi_{rv} \in \mathbb{R}^{3 \times 3}$ is the upper-right block of the STM. To optimize the maneuver, the encounter geometry is projected onto the 2D encounter B-plane (plane). The projection matrix $\mathbf{M}_b \in \mathbb{R}^{2 \times 3}$ is defined by the relative velocity vector $\mathbf{v}_{rel}$ at $t_{c,n}$. The miss distance vector $\mathbf{b}$ in the B-plane is expressed as:

$$\mathbf{b} = \mathbf{M}_b (\mathbf{r}_p(t_{c,n}) - \mathbf{r}_{dn}(t_{c,n})) \quad (10)$$

The optimization of the maneuver direction follows the analytical framework established by Gonzalo et al. [29]. To maximize the miss distance in the encounter B-plane for a given impulsive $\Delta \mathbf{v}$ magnitude, we define the task-specific objective matrix $\mathbf{Q}$ as:

$$\mathbf{Q} = \mathbf{Z}_{3d}^T \mathbf{M}_b^T \mathbf{M}_b \mathbf{Z}_{3d} \quad (11)$$

The optimal maneuver direction $\hat{\mathbf{u}}$ is then identified as the eigenvector corresponding to the maximum eigenvalue of $\mathbf{Q}$. This direction represents the most "efficient" orientation to shift the satellite's position at the time of closest approach $t_{c,n}$ using minimum fuel. This linearized approach provides high fidelity for short-term conjunctions while maintaining computational efficiency [28,29]. Then we manually introduce the $(n+1)$ -th debris into the environment after obtaining the optimized trajectory of n debris. Specifically, the $(n+1)$ -th debris is initialized by backtracking from a conjunction point located on this newly optimized trajectory at $t_{c,n+1}$. This ensures that the threat is not random but is strategically placed to intercept the spacecraft's most likely avoidance path. Notably, this iterative injection is a methodology for scenario synthesis; during actual algorithm evaluation, the agent must re-calculate and physically update the entire maneuver sequence to account for all threats. This raises the demand for spacecraft to have a global perspective on collision information.

Algorithm 1 Strict Multistage Collision Generation

```

1: Input: Initial state  $\mathbf{x}_p(0)$ , Total stages  $N$ , Global seed  $S$ 
2: Initialize: SeedRandomGenerator( $S$ ) {Ensures deterministic reproducibility}
3:  $\mathcal{M} \leftarrow \emptyset$  {List of reference maneuvers for injection}
4:  $\mathcal{D} \leftarrow \emptyset$  {Set of generated debris objects}
5: for  $n = 1$  to  $N$  do
6:   Determine  $t_{c,n}$  and  $t_{m,n}$  for the  $n$ -th stage
7:   // Propagate satellite along the path modified by all previous maneuvers
8:    $\mathbf{r}_p(t_{c,n}) \leftarrow \text{PropagateWithManeuvers}(\mathbf{x}_p(0), t_{c,n}, \mathcal{M})$ 
9:   // Step 1: Inject targeted debris based on current satellite position
10:   $\mathbf{v}_{d,n} \leftarrow \text{SampleValidVelocity}(\mathbf{r}_p(t_{c,n}))$  {Ensures perigee > 120 km}
11:   $\mathbf{x}_{d,n}(0) \leftarrow \text{Backtrack}(\mathbf{r}_p(t_{c,n}), \mathbf{v}_{d,n}, t_{c,n})$ 
12:   $\mathcal{D} \leftarrow \mathcal{D} \cup \{\mathbf{x}_{d,n}(0)\}$ 
13:  // Step 2: Analytical maneuver calculation for next stage (Refs [1–3])
14:   $\mathbf{Z}_{3d} \leftarrow \text{LinearSensitivity}(t_{c,n}, t_{m,n})$  {Extract  $\Phi_{rv}$  from STM}
15:   $\hat{\mathbf{u}} \leftarrow \text{PrincipalEigenvector}(\mathbf{Z}_{3d}^T \mathbf{M}_b^T \mathbf{M}_b \mathbf{Z}_{3d})$ 
16:   $\Delta v_n \leftarrow \text{SearchMagnitude}(\hat{\mathbf{u}}, P_c < 10^{-7})$ 
17:   $\mathcal{M} \leftarrow \mathcal{M} \cup \{(t_{m,n}, \Delta v_n \hat{\mathbf{u}})\}$ 
18: end for
19: Output: Final reproducible scenario  $\mathcal{D}$ 

```

Fig. 5. Pseudo-code for strict multistage collision scenario construction.

Finally, we create a realistic and challenging scenario that tests the agent's ability to adapt to emergent risks and highlights the need for real-time, robust collision avoidance strategies. Fig. 6 shows the effect of a strict multistage collision scenario.

3.4. Probability-based multistage collision

This scenario introduces uncertainty by modeling the emergence of collision threats as a stochastic process. In this method, we simulate the formation of a debris cloud resulting from a space collision, which will subsequently have a dangerous encounter with the protected spacecraft, resulting in a probability-based multistage collisions. As shown in the pseudo-code in Fig. 7, to ensure the scenario is both challenging and physically reproducible, we implement a deterministic backtracking generation procedure. First, a virtual "parent body" is initialized to intersect the spacecraft's trajectory at a randomly sampled collision time $t_{c,prob}$ within the simulation duration. To simulate a cloud, we define a breakup lead-time $\Delta t_{breakup}$ as 300 s, such that the fragmentation occurs at $t_{breakup} = t_{c,prob} - \Delta t_{breakup}$. The state of the parent body is back-propagated from $t_{c,prob}$ to $t_{breakup}$. The initial state of each fragment is determined by applying a relative ejection velocity increment δv to the state vector of the parent body. Each fragment i receives a velocity vector $\mathbf{v}_{f,i} = \mathbf{v}_{parent} + \delta \mathbf{v}_i$, where $\delta \mathbf{v}_i$ is sampled using a deterministic random number generator fixed by a scenario-specific seed. While the velocity distribution in real-world breakups is typically non-Gaussian, we characterize the magnitude of this increment $\|\delta v\|$ using a truncated normal distribution to ensure computational tractability and scenario controllability during reinforcement learning training. Based on the empirical analysis of hypervelocity impact events, such as the 2009 Iridium-Cosmos collision [44,45], the distribution parameters are set with a mean $\mu_{\delta v} = 100\text{m/s}$ and a standard deviation $\sigma_{\delta v} = 80\text{m/s}$, with values truncated within the $[0, 200]\text{m/s}$ range. Crucially, each generated fragment is subjected to a physical validity check: any fragment with a perigee altitude below 120 km or a hyperbolic trajectory is discarded and re-sampled, ensuring the cloud remains within a realistic orbital regime.

The key idea is to increase the likelihood that the spacecraft, while attempting to avoid one piece of debris, may enter the risk zone of another, thereby triggering cascading collision threats. Although this method does not strictly ensure that each individual object contributes directly to a multistage collision, it significantly increases the probability of generating complex multistage collision scenarios. This setup represents a local high-density conjunction event, providing high scenario diversity and supporting robust policy training for reinforcement learning agents under uncertain and dynamic environments. Fig. 8 shows the effect of a probability-based multistage collision scenario.

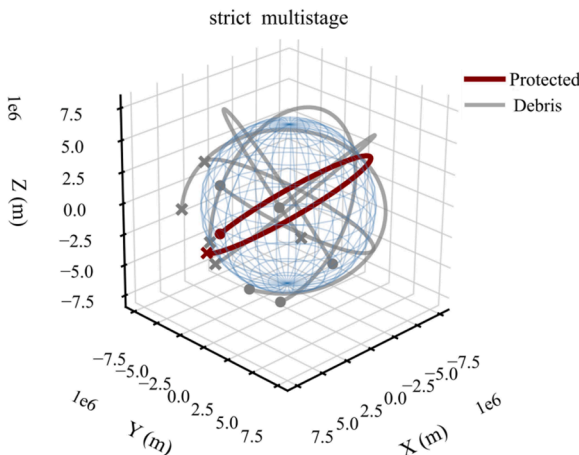


Fig. 6. Diagram of strict multistage collision.

Algorithm 2 Probability-Based Multistage Collision Generation

```

1: Input: Spacecraft state  $\mathbf{x}_p(0)$ , fragmentation lead-time  $\Delta t_{breakup}$ , fragment count  $N_{frag}$ , Seed  $S$ 
2: Initialize: SeedRandomGenerator( $S$ )
3: Sample  $t_{c,prob} \in [0.2, 0.8] \times \text{Duration}$ 
4:  $\mathbf{x}_p(t_c) \leftarrow \text{Propagate}(\mathbf{x}_p(0), t_c)$ 
5:  $\mathbf{v}_{parent} \leftarrow \text{GenerateValidVelocity}(\mathbf{r}_p(t_c), \text{RNG})$ 
6:  $\mathbf{x}_{parent}(t_{breakup}) \leftarrow \text{Backtrack}(\mathbf{r}_p(t_c), \mathbf{v}_{parent}, \Delta t_{breakup})$ 
7:  $\mathcal{D} \leftarrow \emptyset$ 
8: for  $i = 1$  to  $N_{frag}$  do
9:   repeat
10:     $\delta \mathbf{v} \sim \mathcal{N}_{truncated}(\mu, \sigma)$  {Relative ejection velocity}
11:     $\mathbf{x}_{f,i}(t_{breakup}) \leftarrow [\mathbf{r}_{parent}(t_{breakup}), \mathbf{v}_{parent} + \delta \mathbf{v}]$ 
12:    until  $\text{IsOrbitValid}(\mathbf{x}_{f,i}(t_{breakup}))$  is True
13:    // Propagate back to simulation start for environment setup
14:     $\mathbf{x}_{f,i}(0) \leftarrow \text{Backtrack}(\mathbf{x}_{f,i}(t_{breakup}), t_{breakup})$ 
15:     $\mathcal{D} \leftarrow \mathcal{D} \cup \{\mathbf{x}_{f,i}(0)\}$ 
16:   end for
17: Output: Debris cloud  $\mathcal{D}$ 

```

Fig. 7. Pseudo-code for probability-based multistage collision scenario construction.

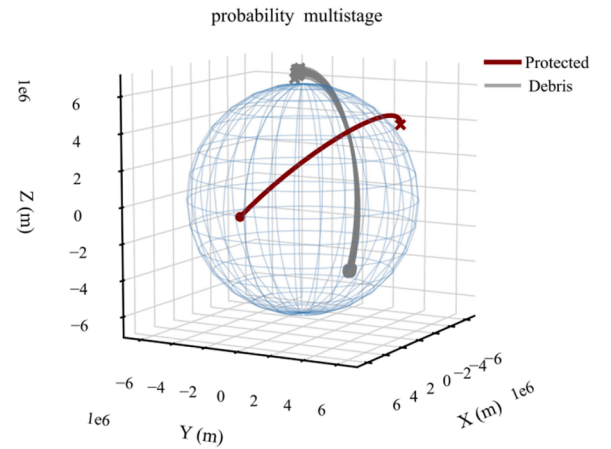


Fig. 8. Diagram of probability-based multistage collision.

3.5. Complex multistage collision

In the method shown, we design a complex multistage collision scenario generation approach to evaluate the effectiveness of the entire warning and avoidance workflow in our model. This scenario is set for an adversarial stress test. As shown in the pseudo-code in Fig. 9, it integrates the deterministic sequential conjunctions from strict multistage collision in Section 3.3, the localized stochastic debris clouds in

Algorithm 3 Complex Multistage Collision Generation

```

1: Input: Initial state  $\mathbf{x}_p(0)$ , TLE Groups  $\mathcal{G}$ , = {FY-1C, Iridium-33, Cosmos-2251}, Seed  $S$ 
2: Initialize: SeedRandomGenerator( $S$ )
3: // Layer 1: Targeted threats following Algorithm 1
4:  $\{\mathcal{D}_{strict}, \mathcal{M}_{ref}\} \leftarrow \text{StrictMultistageGen}(\mathbf{x}_p(0), N_s, S)$ 
5: // Layer 2: Stochastic cloud following Algorithm 2
6:  $\mathcal{D}_{cloud} \leftarrow \text{ProbabilityBasedGen}(\mathbf{x}_p(0), N_f, S)$ 
7: // Layer 3: Authentic Fragment Catalog Injection
8:  $\mathcal{D}_{tle} \leftarrow \emptyset$ 
9: for each  $tle\_data \in \mathcal{G}$  do
10:   $\mathbf{x}_{obj}(0) \leftarrow \text{SGP4\_Propagate}(tle\_data, t = 0)$ 
11:  if  $\text{IsOrbitValid}(\mathbf{x}_{obj}(0))$  is True then
12:     $\mathcal{D}_{tle} \leftarrow \mathcal{D}_{tle} \cup \{\mathbf{x}_{obj}(0)\}$ 
13:  end if
14: end for
15:  $\mathcal{D}_{total} \leftarrow \mathcal{D}_{strict} \cup \mathcal{D}_{cloud} \cup \mathcal{D}_{tle}$ 
16: Output:  $\mathcal{D}_{total}$  (Adversarial Stress Test Environment)

```

Fig. 9. Pseudo-code for complex multistage collision scenario construction.

probability multistage collision from Section 3.4, and a large-scale background of real on-orbit objects. To ensure the scenario reflects a realistic and high-density orbital environment, we incorporate real space objects sourced from the CelesTrak TLE catalog. Specifically, we select objects belonging to three historically significant fragmentation events: the Fengyun-1C debris, the Iridium-33 debris, and the Cosmos-2251 debris [42,46]. These datasets represent the most cluttered orbital regimes in LEO. The "relevant orbital regime" is thus defined by the orbital shells of these major breakup events. For each object, the state at $t = 0$ is derived via the SGP4 propagator. To maintain physical consistency, all imported objects undergo a validity check: orbits with perigee altitudes below 120 km or those exhibiting numerical anomalies during propagation are excluded.

Ultimately, this forms a training scenario with a massive number of potential collision bodies. By mixing targeted "adversarial" threats with authentic catalog traffic from major debris clouds, we prevent the agent from over-fitting to simple geometries and force it to maintain a global perspective. All complex scenarios are generated using a fixed random seed S to ensure full reproducibility. Fig. 10 shows the effect of a complex multistage collision scenario.

4. Early warning of multistage collisions

Multistage collisions refer to situations in which an in-orbit spacecraft, after performing a maneuver to avoid an initial potential collision, subsequently encounters new debris and faces additional collision risks. With the growing deployment of space constellations and the increasing number of space debris, this phenomenon has become increasingly prominent. It is practically infeasible to rely on ground-based manual collision warning for every individual spacecraft in real time. Therefore, to ensure the operational safety of spacecraft, it is essential that they possess the capability to autonomously predict and respond to multistage collision threats. The current collision warning methods usually rely on defining warning areas and calculating the closest approach distance and collision probability. However, these methods often neglect the strong coupling characteristics of multistage collisions and have limitations when dealing with the calculation of massive space debris. Building upon these concepts, this study proposes an approach that enables autonomous warning of multistage collisions by performing a multistage-tailored screening and collision probability calculation. This approach is divided into two parts: the static screening of orbital altitude and the dynamic screening of time of crossing node. The effectiveness of this two-stage screening approach is illustrated in Fig. 11, which demonstrates the reduction of potential collision targets through successive filtering. The specific implementation details of the static and dynamic screening processes are described in Sections 4.1 and

4.2, respectively.

4.1. Static screening of orbital altitude

First, the orbital trajectories of detected space objects are predicted using the aforementioned dynamical model. A static screening of potential collisions is then performed based on orbital altitude characteristics, using the perigee-apogee filtering method. The perigee and apogee represent the lowest and highest orbital altitudes relative to Earth, respectively. The perigee radius h_p and apogee radius h_a of a space object can be computed as follows:

$$\begin{cases} h_p = a(1 - e) \\ h_a = a(1 + e) \end{cases} \quad (12)$$

where a and e are the semi-major axis and eccentricity of the object's Keplerian elliptical orbit. Let $h_{p,main}$ and $h_{a,main}$ denote the perigee and apogee radius of the primary spacecraft, respectively. Objects that satisfy the following condition is considered incapable of colliding with the spacecraft:

$$h_{p,main} - h_a > d \quad \text{or} \quad h_p - h_{a,main} > d \quad (13)$$

Here, d is a non-negative distance threshold. Space objects satisfying this condition are excluded, and the remaining orbital region is retained, as illustrated in the Fig. 12.

To ensure the screening is conservative under realistic uncertainty growth and non-Keplerian perturbations, the safety distance threshold d can be estimated based on the position covariance in the corresponding direction. The radial uncertainty σ_r is obtained by propagating the initial ECI covariance $\mathbf{P}_0$ to $\mathbf{P}(t)$ at the potential conjunction epoch using the STM $\Phi(t, t_0)$. The predicted radial standard deviation σ_r is then extracted by projecting the positional covariance onto the radial unit vector e_r in the Radial-Along-track-Cross-track (RSW) frame, a spacecraft-centered local orbital coordinate system:

$$\sigma_r = \sqrt{e_r^T \mathbf{P}(t) e_r} \quad (14)$$

Subsequently, the dynamic threshold d_{dyn} is defined as:

$$d_{dyn} = \max(k \cdot (\sigma_{r,main} + \sigma_{r,deb}) + d_{pad}, d_{min}) \quad (15)$$

where k is the confidence scaling factor, which is set to 6.0 in the sensitivity analysis experiments in Section 4.3 to align with the 10^{-6} risk threshold. d_{pad} is the geometric padding for J_2 -induced radial oscillations, which is set to 5 km. And d_{min} is the safety floor of the distance threshold. This formulation ensures zero missed detections even for highly eccentric adversarial orbits.

4.2. Dynamic screening of time of crossing node

Subsequently, a further screening of potential collisions is performed based on the time difference at which the space object and the primary spacecraft pass through their orbital intersection points. This process excludes objects that are temporally incompatible with a collision scenario.

Assuming the line of intersection between the two orbital planes is as illustrated in the Fig. 13, let τ_1 and τ_2 be the normal vectors of the orbital planes of the primary spacecraft and the space object, respectively. In the J2000 coordinate system, these vectors can be expressed as:

$$\begin{cases} \tau_1 = [\sin\Omega_1 \sin i_1, -\cos\Omega_1 \sin i_1, \cos i_1] \\ \tau_2 = [\sin\Omega_2 \sin i_2, -\cos\Omega_2 \sin i_2, \cos i_2] \end{cases} \quad (16)$$

where Ω_1, i_1 are the right ascension of the ascending node and inclination of the primary spacecraft's orbit, and Ω_2, i_2 are the corresponding parameters of the space object. The direction vectors of the intersection line, denoted $\mathbf{n}_1$ and $\mathbf{n}_2$, can then be computed using the cross product of

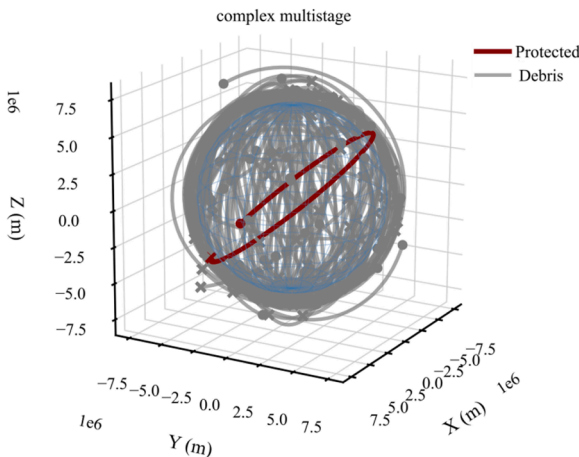


Fig. 10. Diagram of complex multistage collision.

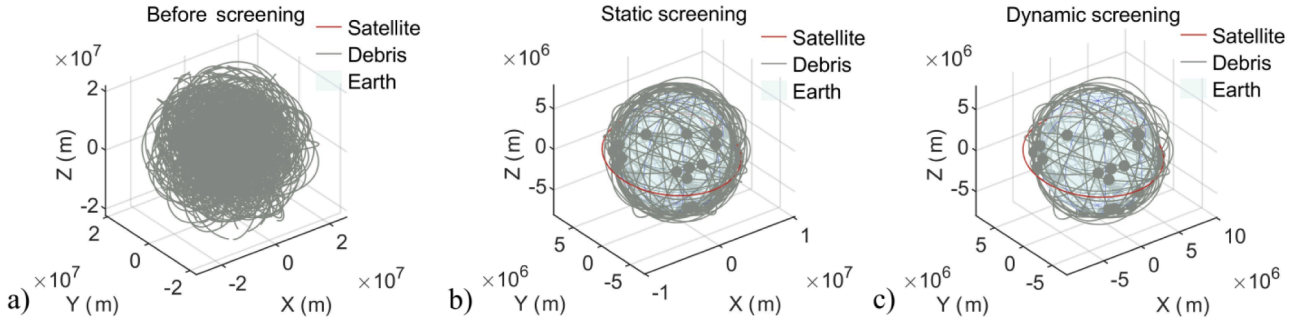


Fig. 11. Effect comparison: a) before screening; b) static screening; c) dynamic screening.

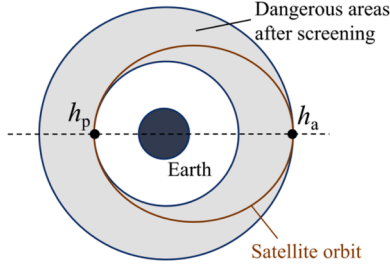


Fig. 12. Diagram of static screening of orbital altitude.

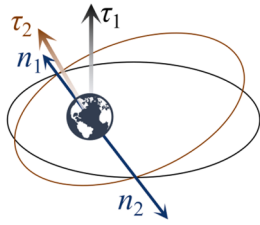


Fig. 13. Diagram of orbital intersection point.

the respective normal vectors:

$$\begin{cases} \mathbf{n}_1 = \boldsymbol{\tau}_1 \times \boldsymbol{\tau}_2 \\ \mathbf{n}_2 = \boldsymbol{\tau}_2 \times \boldsymbol{\tau}_1 \end{cases} \quad (17)$$

The vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ define the line of intersection between the two orbital planes. It should be noted that for two non-coplanar Keplerian orbits, they generally do not intersect at a single spatial point but rather possess two potential encounter regions along this line. Specifically, each orbit intersects the line $(\mathbf{n}_1, \mathbf{n}_2)$ at two distinct points (the ascending and descending nodes relative to the other orbital plane), resulting in four intersection points in total. These points form two 'conjunction pairs' based on their radial proximity. For each pair, given the standard gravitational parameter μ and the semi-major axis a , the orbital period T_s of the primary spacecraft and the space object can be calculated. Using the time t_1 at which either body first passes through the intersection point, the time of the i th passage t_i can be estimated as:

$$T_s = 2\pi\sqrt{\frac{a^3}{\mu}} \quad (18)$$

$$t_i = t_1 + (i-1)T_s \quad (19)$$

Finally, any space object for which the time difference between passages satisfies $|t_j - t_k| > t_c$ is excluded, where t_j is the time of the j -th intersection passage of the primary spacecraft, t_k is the corresponding time for the space object, and t_c is the temporal threshold used for filtering. The original space debris data has undergone the above-

mentioned Static screening of orbital altitude and dynamic screening of time of crossing node. Invalid debris with low collision probability has been excluded, and the remaining debris are the high-risk space debris with high collision probability.

4.3. Probabilistic risk evaluation for final multistage collision warning

Following the dual-stage geometric screening process, the remaining candidate objects are subjected to probabilistic risk evaluation to determine whether a collision warning should be issued. While the preceding screening steps significantly reduce the computational burden by eliminating evidently safe conjunctions, they do not provide a quantitative assessment of collision likelihood. Therefore, a probabilistic collision model is employed as the final decision layer of the proposed multistage warning framework.

Different collision probability evaluation methods are used in the training and evaluation phases. During the training and optimization phase, collision probability is computed using an analytical Gaussian approximation in the encounter plane, following the classical formulation presented by Chan [14]. This analytical formulation provides a closed-form expression that is computationally efficient and numerically smooth, making it well suited for large-scale sampling and iterative policy updates in the RL process. Assuming that the relative position uncertainty at TCA follows a Gaussian distribution, the joint position covariance of the spacecraft and debris is projected onto the plane normal to the relative velocity vector. Let $\mathbf{d} \in \mathbb{R}^2$ denote the projected relative position vector in the encounter plane, $\mathbf{P}_{2D}$ the corresponding 2×2 covariance matrix. R is the combined hard-body radius of the two objects, and is set to $R = 20$ m throughout the whole study. The collision probability is approximated by:

$$p = \frac{R^2}{2\sqrt{\det(\mathbf{P}_{2D})}} \exp\left(-\frac{1}{2}\mathbf{d}^T \mathbf{P}_{2D}^{-1} \mathbf{d}\right) \quad (20)$$

To enhance the reliability of the proposed multistage collision warning framework under realistic operational conditions, a high-fidelity numerical integration method is adopted during the final evaluation in Section 6, following the numerical implementation strategy proposed by Alfano [16]. In this step, the two-dimensional Gaussian probability density function in the encounter plane is numerically integrated over the circular collision domain of radius R . Specifically, the collision probability is computed as

$$p = \int \int_{\mathcal{D}} \frac{1}{2\pi\sqrt{\det(\mathbf{P}_{2D})}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{d})^T \mathbf{P}_{2D}^{-1} (\mathbf{x} - \mathbf{d})\right) d\mathbf{x} \quad (21)$$

where $\mathcal{D}$ denotes the disk defined by $|\mathbf{x}| \leq R$. The integral is evaluated using adaptive numerical quadrature to ensure convergence and stability. By separating the computationally efficient analytical model used in training from the high-precision numerical integration employed in the final evaluation, learning-based maneuver decisions are generated efficiently while the ultimate warning decision is supported by a

rigorous probabilistic assessment.

Within the proposed multistage collision warning and avoidance framework, a warning is issued and the avoidance maneuver planning phase is initiated whenever the numerically evaluated collision probability exceeds a predefined risk threshold P_{th} . After a maneuver is executed, the spacecraft state and covariance are updated, and the entire screening–evaluation cycle is repeated. This recursive process enables autonomous identification of subsequent collision threats that may arise from maneuver-induced orbital changes, thereby supporting continuous multistage risk mitigation.

4.4. Sensitivity analysis and threshold optimization of the dual-stage screening

In the Dual-Stage Screening mentioned in Section 4.1 and 4.2, the calculation of the dynamic distance threshold d_{dyn} and the temporal threshold t_c is critical for balancing computational efficiency with collision avoidance safety. To rigorously justify these settings, we conducted a grid-search-based sensitivity analysis using a high-density orbital dataset. Considering that sparse debris data may not be voluminous enough to trigger sufficient "missed detection" events, we utilized the complete Starlink constellation dataset which includes 14,128 satellites from CelesTrak as the screening pool. This provides a high-fidelity, high-density environment to test the limits of our screening approach. The protected spacecraft's orbit was configured as Table 2. Following the NASA's operational standards for collision risk management [43], we have adopted a relatively conservative approach and defined the situation where the probability of erroneous exclusion of a collision risk from the screening process exceeds the threshold of 10^{-6} as a missed detection.

The initial state uncertainty of the protected spacecraft is modeled in the J2000 frame as a block-diagonal covariance matrix:

$$P_0 = \begin{bmatrix} \sigma_r^2 I_3 & 0 \\ 0 & \sigma_v^2 I_3 \end{bmatrix} \quad (22)$$

where $\sigma_r = 1000\text{m}$ and $\sigma_v = 1\text{m/s}$ denote the standard deviations of position and velocity along each J2000 axis. The choice of a kilometer-level position uncertainty is supported by independent analyses comparing TLE-derived positions with high-accuracy GPS measurements [47]. The covariance is then propagated to the time of closest approach using two-body orbital dynamics, after which it is transformed into the RSW frame of the protected spacecraft.

Figs. 14 show the sensitivity test results of dual-stage screening in the two-body environment, J_2 environment, and J_2 environment with atmospheric drag, respectively. Fig. 14a shows that in the two-body environment zero missed detection rate is achieved with $d = 110\text{ km}$ and $t_c = 160\text{ s}$. As shown in Fig. 14b and 14c, when perturbations are introduced, the required thresholds increase significantly. In J_2 environment, the thresholds increase to $d = 130\text{ km}$ and $t_c = 290\text{ s}$. And in J_2 environment with atmospheric drag, they increase to $d = 132\text{ km}$ and $t_c = 289\text{ s}$. This discrepancy arises because perturbations induce secular drifts and short-period oscillations in orbital elements, which expand the spatial-temporal volume in which a conjunction might occur. Consequently, a more conservative safety floor is required for operational robustness. However, while larger thresholds guarantee safety, they also increase the number of objects passed to the complex probability calculation stage, thereby raising the computational burden. Fig. 14d shows the results of screening thresholds selection and computational load, where the heatmap illustrates the number of remaining high-risk

debris candidates. As can be observed, as the distance threshold d increases from 0 to 200 km and the time threshold t_c increases from 0 to 600 s, the number of remaining high-risk debris candidates rises monotonically from 0 to 2400. Based on these findings, we set the minimum radial safety threshold $d_{min} = 150\text{ km}$ and the minimum temporal threshold $t_c = 400\text{ s}$, which maintains sufficient margin for the zero-missed-detection threshold while reducing the computational burden.

Building upon these calibrated benchmarks, in the environment considering J_2 and atmospheric drag, we validated the static screening method in Section 4.1 based on covariance uncertainty estimation. Fig. 15 presents the performance of the improved screening approach. The histogram shown in Fig. 15a shows the distribution of the dynamically calculated thresholds d_{dyn} . Leveraging the 6σ confidence interval and the d_{min} safety floor, the thresholds adaptively scale between 150 km and 632.9 km. This ensures that objects with large uncertainties are captured by a wider "radial net," while well-tracked objects are filtered using a tighter, yet safe, boundary. The dual-axis sensitivity plot shown in Fig. 15b demonstrates the safety-efficiency trade-off of the method. Using the dynamic screening method mentioned in Section 4.2 on the remaining debris set after static screening, the dangerous debris set is further obtained. The overall missed detection rate drops to absolute zero at $t_c = 290\text{ s}$, while maintaining a high filtering ratio of over 92%. This confirms the validity of the dynamic distance threshold d_{dyn} used in this framework and the sufficiency of the time threshold $t_c = 400\text{ s}$. The proposed method significantly alleviates the computational load of high-fidelity collision probability calculations without compromising safety in perturbed orbital environments.

Indeed, this two-stage screening mechanism also has its own shortcomings. It mainly focuses on short-term orbital geometric compatibility and time alignment, and may not be able to fully capture the long-term risks caused by long-period perturbations in the orbit evolution. During the monitoring process, it needs to be re-evaluated periodically, or combined with a long-term propagation strategy to ensure stability in long-duration missions and dense orbit areas. However, for time-critical collision avoidance scenarios, the considered warning horizon is sufficient, while longer-term orbital adjustments fall beyond the scope of collision avoidance and relate more to space environment management.

5. Learning-based collision avoidance

After the collision warning system identifies high-risk debris, the selected debris states are passed to a learning-based decision module that outputs maneuver commands for the spacecraft. To formalize the collision avoidance problem with impulsive maneuvers within a learning-based framework, we define the state space $\mathcal{S}$ and action space $\mathcal{A}$ as follows. The state vector $\mathbf{s}_t = [\mathbf{X}_{sat}, \mathbf{X}_{rel,1}, \dots, \mathbf{X}_{rel,n}, \tilde{t}, N_d] \in \mathcal{S}$ at each decision epoch provides a comprehensive representation of the orbital environment. Here, $\tilde{t} = t/T_{max}$ denotes the normalized mission time, and N_d represents the number of the dangerous debris. The satellite feature vector $\mathbf{X}_{sat} = [\tilde{a}, \tilde{e}, \tilde{i}, \sin\Omega, \cos\Omega, \sin\omega, \cos\omega, \sin M, \cos M]$ is constructed from the classical orbital elements of the protected spacecraft, where the semi-major axis a , eccentricity e , and inclination i are normalized, and the angular elements, right ascension of the ascending node Ω , argument of perigee ω , and mean anomaly M , are encoded using sine–cosine pairs to avoid discontinuities. For each of the high-risk debris identified by the screening module, the normalized TCA $\tilde{t}_{tca,i}$ and the distance $\tilde{d}_{min,i}$ at the TCA is calculated first. Then the debris objects whose $\tilde{t}_{tca,i} \geq 0$ are ranked according to the $\tilde{d}_{min,i}$ and the closest 20 objects are retained. The debris-related observation is defined as $\mathbf{X}_{rel,i} = [\Delta \mathbf{r}_i^{RSW}, \Delta \mathbf{v}_i^{RSW}, \tilde{t}_{tca,i}, \tilde{d}_{min,i}, \lambda_i, m_i]$, where $\Delta \mathbf{r}_i^{RSW}$ and $\Delta \mathbf{v}_i^{RSW}$ are the relative position and velocity in the spacecraft-centered RSW frame, λ_i represents the eigenvalue-based uncertainty descriptor of the relative position covariance matrix, and m_i is a validity mask to apply Zero-padding when fewer debris objects are present. The action space $\mathcal{A}$ is

Table 2

Orbital elements of the protected spacecraft orbit.

t (J2000)	a (m)	e	i (rad)	Ω (rad)	ω (rad)	$M(t)$ (rad)
8674.5	6800,000	0.001	$\pi/3$	$\pi/18$	$\pi/9$	0

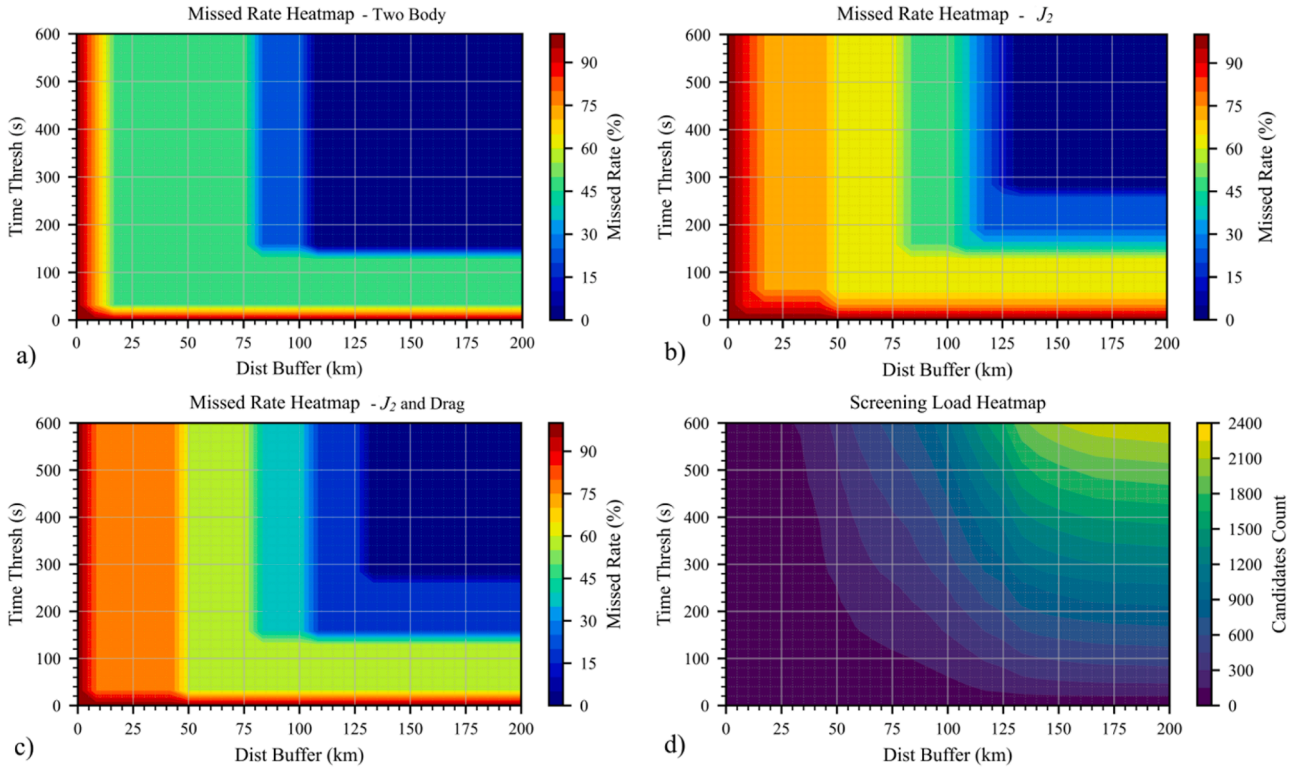


Fig. 14. Diagram of threshold selection for static and dynamic screening: a) missed rate heat map for two-body dynamic; b) missed rate heat map considered J_2 perturbation; c) missed rate heat map considered J_2 perturbation and atmospheric drag; d) Screening load heat map of the remaining candidate targets.

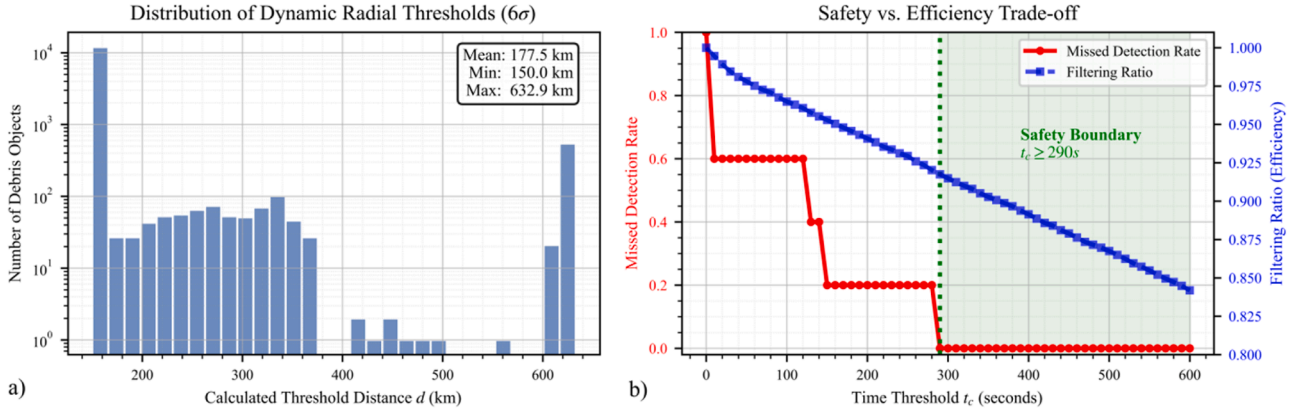


Fig. 15. Effects of the screening method based on covariance uncertainty estimation: a) distribution of the dynamic distance threshold d_{dyn} in the sensitivity test; b) the safety and efficiency trade-off during dynamic screening after static screening using dynamic distance threshold d_{dyn} .

defined in a continuous domain as an impulsive maneuver vector $\mathbf{a}_t = [\Delta v_R, \Delta v_S, \Delta v_W, \gamma]$, $\Delta v_R, \Delta v_S, \Delta v_W$ represent the velocity increments applied after coasting duration in the RSW coordinate system, and the coasting ratio γ is defined as the ratio between the coasting duration before the maneuver and the total episode duration. This structured representation allows the agent to perceive both its own orbital dynamics and the evolving encounter geometry of multiple potential threats, enabling the optimization of multistage avoidance trajectories.

In this section the design of the multi-objective loss function is detailed, and three learning-based frameworks are introduced: MCTS, CEM, and PPO. Each method leverages different mechanisms for balancing exploration and exploitation, and the combination with the well-formulated optimization objective provides a robust foundation for safe and efficient multistage collision avoidance in complex debris scenarios.

5.1. Design of multi-objective reward function

During collision avoidance, parameters such as collision probability and fuel consumption are directly related to the spacecraft's safety and operational cost, serving as core metrics for evaluating the effectiveness of avoidance strategies. To train the autonomous agent and evaluate avoidance strategies, we formulate a multi-objective cumulative reward function $\mathcal{R}$ to be maximized during the learning process. Unlike static optimization, this function jointly accounts for the time-integral of collision risk, fuel consumption, and orbital deviations over the entire mission duration. The total reward is defined as the sum of step-wise rewards r_t where the components represent the weighted objectives for safety and efficiency:

$$\mathcal{R} = \sum_{t=0}^T r_t = \mathcal{R}_{risk} + \mathcal{R}_{fuel} + \mathcal{R}_{dev} + \mathcal{R}_{sparse} \quad (23)$$

The risk term $\mathcal{R}_{risk}$ is calculated as the time-integral of the log-scale collision probability density along the trajectory, which is numerically approximated via the trapezoidal rule over the discrete time intervals throughout the simulation:

$$\mathcal{R}_{risk} = k_1 \cdot \int_0^T (\log_{10} p_{max}(t) - \log_{10} p_{threshold}) dt \quad (24)$$

where k_1 is a negative weight coefficient and the max collision probability $p_{max}(t)$ among the remaining target at time t is calculated in the object-centered RSW coordinate system using the method described in Section 4.4 and $p_{threshold}$ is set to 10^{-12} during the rewards calculation.

Fuel consumption is calculated as the cumulative sum of the magnitudes of all impulsive maneuvers executed during the avoidance sequence:

$$f = \sum_{i=1}^n \|\Delta V_i\| \quad (25)$$

Then the penalty for total propellant usage is calculated as $\mathcal{R}_{fuel} = k_2 f$, where k_2 is a negative weight coefficient for fuel efficiency.

To ensure the spacecraft maintains its mission orbit, the orbital deviation d is defined as the Euclidean norm of the normalized differences in five Keplerian orbital elements (excluding mean anomaly M , as it changes rapidly with time). To account for the different units and scales of these elements, each is scaled by its characteristic magnitude:

$$d = \sqrt{\sum_{j \in \{a, e, i, \Omega, \omega\}} \left(\frac{\alpha_j - \alpha_{j,nominal}}{\sigma_j} \right)^2} \quad (26)$$

where α_j and $\alpha_{j,nominal}$ represent the current and nominal orbital elements, respectively, and σ_j is the normalization factor for the j -th element. Then the penalty for deviations is defined as $\mathcal{R}_{dev} = k_3 d$, where k_3 is a negative weight coefficient for orbital maintenance.

At last, to guide the agent toward safe and successful mission completion, we implement a sparse reward $\mathcal{R}_{sparse}$ that triggers upon episode termination. Significant penalties are applied for catastrophic failures, specifically -2000 for collision probability beyond threshold and -1500 for entering an invalid orbit. Conversely, a base positive reward of 500 is granted upon successful completion. This composite multi-objective optimization function guides the learning agent to generate maneuvers that are both safe and efficient.

The weighting coefficients k_1, k_2 and k_3 balance the importance of each objective, their selection is critical to the agent's decision-making behavior. In our framework, the "Balanced" configuration (Case 1: $k_1=-100, k_2=-10, k_3=-1$) is chosen as the baseline for the following safety priority and magnitude alignment reasons. As for safety priority, in orbital operations, collision avoidance is often treated as a "hard constraint." Since the collision probability p is typically in the order of 10^{-4} to 10^{-6} , a large k_1 ensures that the risk component dominates the $\mathcal{R}$ function, compelling the agent to prioritize safety. As for magnitude alignment, the fuel consumption f typically range between 10^{-1} and 10^2 , orbital deviation r typically range between 0.1 and 10 . Setting $k_2=-10$, $k_3=-1$ keeps these terms at a comparable magnitude, allowing the agent to optimize for efficiency once the safety threshold is met.

To verify the influence of these weights, we performed a grid-search analysis across seven configurations in 5-stage strict multistage collision scenario based on numerical Powell method. The results are summarized in Table 3 and visualized in the Pareto frontier in Fig. 16.

As shown in Table 3 and Fig. 16, increasing the magnitude of k_1 in Cases 2–3 leads to a significant reduction in collision probability at the cost of higher fuel usage. Conversely, increasing the magnitude of k_2 in Cases 4–5 minimizes fuel consumption but results in higher residual risk.

Table 3

Performance comparison under different weighting configurations.

ID	Weight Config ($k_1; k_2; k_3$)	Probability of Collision p	Fuel Consumption f	Orbit Deviation d	Decision Characteristic
Case 1	(−100, −10, −1)	1.37E-06	5.4392	9.48E-01	Balanced
Case 2	(−500, −10, −1)	2.29E-07	7.7676	9.86E-01	Slightly Conservative
Case 3	(−1000, −10, −1)	1.23E-07	8.5369	9.96E-01	Conservative
Case 4	(−100, −50, −1)	4.24E-06	4.0705	9.79E-01	Fuel-Efficient Focus
Case 5	(−100, −100, −1)	1.75E-05	1.5362	3.77E-01	Fuel-Optimal
Case 6	(−100, −10, −5)	1.81E-06	6.617	3.40E-01	Slight Station-Keeping
Case 7	(−100, −10, −10)	2.57E-06	7.658	9.90E-02	Station-Keeping Focus

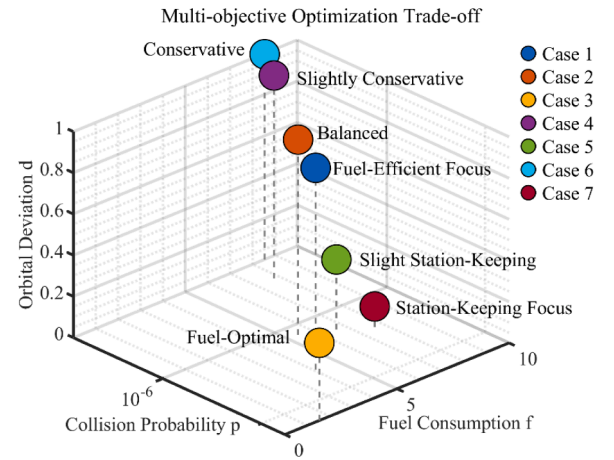


Fig. 16. Diagram of the optimization results under different weighting coefficients.

Higher magnitude of k_3 values in Cases 6–7 effectively constrain the satellite to its nominal orbit. These results demonstrate that the proposed multi-objective function allows for flexible adjustment of the avoidance policy based on specific mission priorities. The Balanced configuration provides a robust trade-off, ensuring that p remains well below the 10^{-4} safety threshold while maintaining reasonable operational costs.

5.2. Learning-based optimization methods

In the domain of spacecraft autonomous collision avoidance, learning-based methods offer a powerful framework for sequential decision-making under uncertainty. Unlike traditional rule-based methods, they enables agents to learn optimal control strategies through interaction with the environment, adapting to dynamic conditions such as evolving debris fields. These approaches are unified under the learning-based paradigm as they all iterate through a cycle of environment interaction, reward-based evaluation, and systematic policy improvement to maximize cumulative returns. This section focuses on three representative learning-based approaches suitable for continuous action spaces - MCTS, CEM, and PPO. These methods differ in their core principles: MCTS builds a search tree through simulations and backpropagation to perform online policy lookahead, CEM iteratively refines action distributions based on elite samples as a derivative-free policy search, while PPO leverages deep neural networks to learn

policies using gradient-based updates within a trust region. These methods cover most categories of search-based and generative learning-based optimization, enabling our comparative experiments to span different computational paradigms, performance characteristics, and application scenarios, thereby providing a comprehensive and objective reference for method selection in multistage space debris collision avoidance tasks. Each method provides distinct advantages in balancing exploration and exploitation, handling long-horizon planning, and generating fuel-efficient avoidance maneuvers. The following subsections detail their algorithmic frameworks:

5.2.1. Monte Carlo tree search method

MCTS is a powerful planning algorithm that builds a decision tree through stochastic simulations and backpropagation of estimated rewards. It is widely applied in decision-making problems under uncertainty. In spacecraft autonomous collision avoidance tasks, each decision involves choosing a continuous-valued control action, such as a velocity increment in 3D space. This continuous nature of the action space makes traditional discrete-action MCTS inapplicable without modifications. To address this, we adopt a continuous-action MCTS framework, where actions are sampled from a parametric probability distribution, and the search tree is incrementally built by evaluating and propagating the outcomes of sampled trajectories. A single iteration of the algorithm consists of the following four steps:

- 1) Selection: Starting from the root node representing the spacecraft state at time t , the algorithm traverses the tree by recursively selecting actions. For each node, actions are sampled from a probability distribution, and a criterion is used to balance exploitation and exploration. Crucially, “not performing a maneuver” is treated as an explicit candidate action. This allows the algorithm to evaluate whether delaying a maneuver might yield a better long-term outcome than acting immediately. The modified selection criterion can be written as:

$$a = \operatorname{argmax}_a \left[Q(s, a) + c \cdot \frac{P(a|s)}{1 + N(s, a)} \right] \quad (27)$$

where $Q(s, a)$ is the estimated return, $P(a|s)$ is the prior from a learned or hand-crafted policy, and $N(s, a)$ is the visit count.

- 2) Expansion: If a selected action a leads to a state s' not yet in the tree, the node is expanded. In continuous domains, the number of child nodes is limited, and actions are chosen either by sampling from a focused distribution or by optimizing the action locally. The option to expand a “no-maneuver” child node is always preserved, and is selected with a probability of 30% during expansion to encourage exploration of waiting strategies.
- 3) Simulation: From the new state node, a Monte Carlo simulation is performed to estimate the long-term return of the current strategy. The simulation continues until reaching a terminal condition such as mission success, collision, or time horizon.
- 4) Backpropagation: After the simulation, the resulting cumulative reward is backpropagated along the path traversed in the tree. This updates the visit counts and value estimates $Q(s, a)$ for each node, guiding future selections.

By explicitly considering “whether to maneuver” as part of the action decision, this MCTS framework naturally integrates timing decisions into the action selection process. That is, at every time step, the agent not only searches for the best maneuver, but also evaluates the possibility of postponing action, which is often critical in environments with evolving risks such as dynamic space debris fields. This results in more strategic, long-horizon planning and enables effective decision-making

under uncertainty in continuous control domains.

5.2.2. Cross entropy method for continuous action space

CEM is a Monte Carlo method for optimization and importance sampling, different from tree search method. Similar to evolutionary algorithms, CEM scatter points in space according to certain rules, obtain the error of each point, and then decide the rule of the next round of scattering points based on this error information. The cross-entropy method is so named because the goal is to minimize the cross-entropy between the distribution of the data obtained from random scattered points and the actual distribution of the data, so that the sample distribution is as distributed as possible. In the early warning and avoidance scenario of space debris, CEM algorithm finds the optimal strategy by iteratively improving the probability distribution of action, which is especially suitable for solving reinforcement learning problems with continuous action space. The algorithm process is as follows:

- 1) Initialization: Define an initial probability distribution $p(x; \theta)$ for variables to be optimized, typically a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, whose mean μ_0 and variance σ_0^2 are based on prior knowledge of the problem or random initialization. In this study, each maneuvering time point was uniformly initialized based on the number of maneuvers, and then further adjusted during the iterative optimization process to achieve efficient time avoidance actions.
- 2) Sample collection: In each iteration i , a large number of samples $\mathbf{X} = \{x_1, x_2, \dots, x_{N_{pop}}\}$ are drawn from the current distribution $p(x; \theta_i)$, where N_{pop} represents the population size.
- 3) Performance evaluation: The performance of each sample is evaluated by simulation or actual execution, usually measured by calculating cumulative rewards.
- 4) Sample selection: A subset of “elite” samples is identified by selecting the top performers based on their rewards. The number of elite samples is defined as $N_e = \lceil \rho_e N_{pop} \rceil$, where ρ_e is the elite fraction. These samples represent the current approximation of the optimal solution manifold.
- 5) Update parameters: The distribution parameters are updated using the elite set. To ensure stable convergence, learning rates α_μ and α_σ are applied to update the mean and variance:

$$\begin{aligned} \mu_{i+1} &= (1 - \alpha_\mu) \mu_i + \alpha_\mu \mu_{elite} \\ \sigma_{i+1} &= (1 - \alpha_\sigma) \sigma_i + \alpha_\sigma \sigma_{elite} \end{aligned} \quad (28)$$

where μ_{elite} and σ_{elite} is the mean and standard deviation of the selected elite subset. Additionally, a variance decay factor γ_{decay} is often incorporated to progressively narrow the search focus and prevent premature convergence.

5.2.3. Proximal policy optimization method

PPO is a deep reinforcement learning algorithm based on Actor-Critic framework. In this framework, two neural networks, Actor and Critic, are set up. In the space debris early warning avoidance scenario, the Actor network is used to determine the best avoidance action generated in the given state. The Critic network is used to estimate the value of the current policy action. The PPO algorithm is based on the strategy gradient method, its working principal block diagram is shown as Fig. 17.

The algorithm is implemented in the following five steps:

- 1) Initialize the network: Define two neural networks, one as Actor and the other as Critic. The Actor network outputs the probability of taking various possible actions in a particular state, while the Critic network outputs an estimate of the value of the state under the current policy.

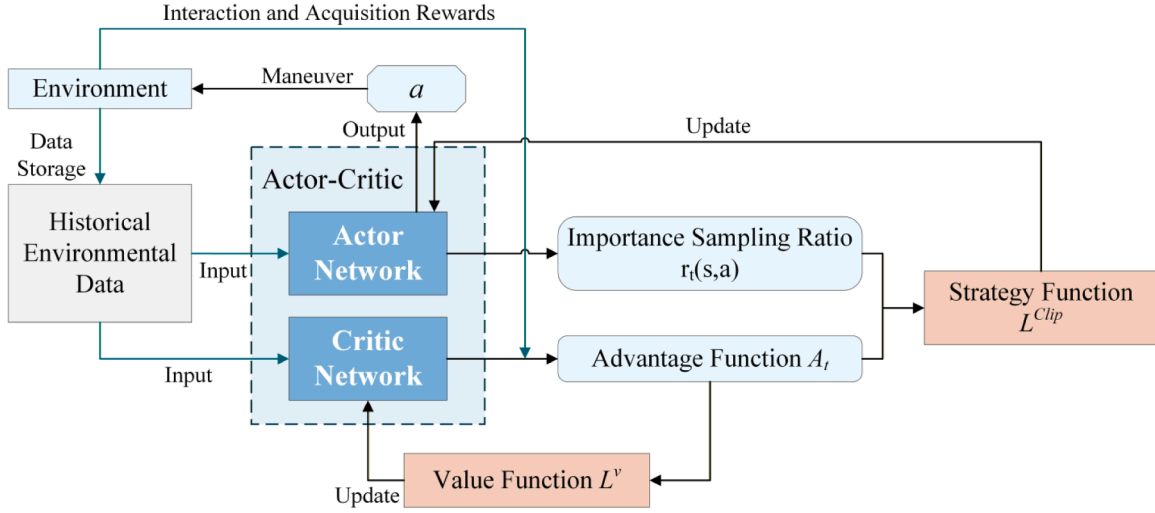


Fig. 17. PPO algorithm training schematic.

- 2) Interaction and Acquisition: Interact with the environment by performing actions generated by the Actor network to collect a series of experiential data, including states, actions, rewards, and the next state. This collection process is conducted over a fixed horizon of N_{steps} per iteration.
- 3) Compute the advantage function A_t : Use the Critic network to compute the advantage function A_t , which measures the additional value of taking an action in a particular state compared to the average strategy. In PPO, the advantage function is a key component in calculating the gradient of the strategy and is used to guide the updating of the strategy so that it is optimized in the direction of improving cumulative returns. In the calculation of A_t , the discount factor γ is utilized to balance the importance of immediate and future rewards.
- 4) Calculation of objective function: PPO uses importance sampling ratio $r_t(s,a)$ and Clip clipping methods to assist calculation of objective strategy function L^{Clip} and objective value function L^v to optimize update strategy and value network. Let $V(s_t)$ be the value function and A_t is the advantage function mentioned above, the objective value function L^v is:

$$L^v = (V(s_t) - A_t)^2 \quad (29)$$

Let $\pi_\theta(a|s)$ represent the probability of performing action a in state s under the new policy, and $\pi_{\theta_{old}}(a|s)$ represents the probability of performing action a in state s under the old policy, then the importance sampling ratio $r_t(s,a)$ is:

$$r_t(s,a) = \frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)} \quad (30)$$

Then, the objective strategy function L^{Clip} is obtained using a clipping method, where ϵ is the clipping parameter used to constrain the policy update magnitude:

$$L^{Clip} = \mathbb{E}_t \{ \min[r_t(s,a)A_t, \text{clip}(r_t(s,a), 1-\epsilon, 1+\epsilon)A_t] \} \quad (31)$$

To further encourage exploration and prevent premature convergence, an entropy bonus $c_{ent}H(\pi)$ is often added to the total loss function, where c_{ent} is the entropy coefficient and $H(\pi)$ represents the entropy of the policy, which measures the stochasticity of the action selection to prevent the policy from becoming prematurely deterministic.

- 5) Optimize the network: compute ∇L^{Clip} and ∇L^v to optimize the target policy function L^{Clip} and the target value function L^v , and update

the parameters of the Actor network and the Critic network by the gradient ascent method. The optimization is performed using a batch size B over N_{epochs} iterations for each data buffer, with the step size controlled by the learning rate α .

6. Experimental validation and comparison

To verify the effectiveness and robustness of the proposed collision avoidance framework, this chapter conducts a series of numerical simulations across diverse orbital environments. The experiments are categorized into four parts: a case study for 5-stage strict multistage collision avoidance (Section 6.1), a multi-scenario performance comparison of four algorithms (Section 6.2), an analysis of algorithmic sensitivity to J_2 perturbations (Section 6.3), and an evaluation of time complexity under problem scale expansion (Section 6.4).

In the simulations, the environmental time step was set to 10 s, all numerical simulations and training process are conducted on a computing platform equipped with an 11th Gen Intel(R) Core(TM) i7-11800H CPU @ 2.30 GHz, 32.0 GB of RAM, and an NVIDIA GeForce RTX 3060 Laptop GPU. To simulate a realistic orbital environment, the initial state uncertainties are characterized by 1-sigma standard deviations in the RSW local frame. The combined hard-body radius is set to $R = 20$ m, and the initial position and velocity uncertainties for protected spacecraft are configured as $\sigma_r = [20, 100, 20]^T$ m and $\sigma_v = [0.01, 0.05, 0.01]^T$ m/s. And for debris they are configured as $\sigma_r = [200, 1000, 200]^T$ m and $\sigma_v = [0.1, 0.5, 0.1]^T$ m/s. The initial state uncertainty of the protected spacecraft is modeled as a block-diagonal covariance matrix P_0 as described in Eq. (22). The difference in initial errors between protected spacecraft and space debris stems from considerations of orbit determination precision. Artificial spacecraft are equipped with more sophisticated monitoring systems, thus achieving higher precision in orbit determination compared to space debris. The along-track uncertainty is intentionally set an order of magnitude higher than the radial and cross-track components. This configuration accounts for the typical characteristics of ground-based tracking and the fact that along-track errors accumulate most rapidly due to atmospheric drag modeling inaccuracies and energy variations. As for algorithm parameters, the MCTS search tree is configured with a maximum depth of $D = 7$ and a branching factor of $C = 50$ candidates per node. The CEM algorithm is configured with a population size $N_{pop} = 100$, and elite fraction $\rho_e = 0.2$. To balance exploration and exploitation, the learning rates of CEM are specified as $\alpha_\mu = 0.3$ and $\alpha_\sigma = 0.5$. Additionally, a sigma decay factor $\gamma_{decay} = 0.95$ is applied to progressively narrow the search space, complemented by an early stopping criterion with a

standard deviation threshold of 0.1. The hyperparameters used in the PPO training are shown in Table 4.

6.1. Case study: 5-stage strict multistage collision optimization

To visually demonstrate the optimization effects of various methods, we conduct a case study using a strict 5-stage multistage collision scenario. This scenario is constructed using the method described in Section 3.3, with parameters set as follows: collision stages $N = 5$ and global random seed $S = 1$. We initialized the protected spacecraft state $\mathbf{x}_p(t_0)$ with the parameters shown in Table 5. Each collision time $t_{c,n}$ is separated by 1000s. The total environment duration is 5000 s, and the experimental simulation consists of 500 steps. The time for analytical avoidance maneuver is set as $t_{m,n} = t_{c,n} - 1000$. Below, we evaluate collision avoidance optimization for this scenario using MCTS, CEM, and PPO methods.

First, we evaluate the performance of the MCTS method with the planner random seed set to 1. The optimization process is executed for 200 iterations to ensure sufficient exploration of the maneuver space. As the training process shown in Fig. 18, the algorithm completed the optimization in 1204.73 s, achieving a cumulative final reward of 377.82. The peak collision probability is suppressed to 9.99×10^{-5} , and the total fuel consumption is 14.4686 m/s. Furthermore, the final terminal state deviation is maintained at 1.0475.

The final maneuver vectors after the optimization are listed in Table 6, and the spatial positions and directions of the impulsive maneuvers are illustrated in Fig. 19a. Fig. 19b shows the changes in cumulative reward, fuel consumption, and orbital deviation during the full-scenario simulation. As the maneuvers proceed, the cumulative reward grows steadily. Both the maneuver timing and fuel consumption are concentrated at the initial stage of the scenario, accompanied by an increasing accumulation of the final orbital deviation.

Fig. 20 illustrates the propagation of position uncertainty for both the protected spacecraft and the five space debris objects throughout the collision avoidance scenario. As demonstrated in the four subplots of Fig. 20, the Total Position Error (RSS) of the debris exhibits a significant secular growth over the 5000 s duration, primarily driven by the expansion of the along-track component. In contrast, the uncertainty of the protected spacecraft remains relatively constrained due to its higher initial state knowledge accuracy. The radial and cross-track components exhibit characteristic periodic oscillations, consistent with the linearized Clohessy-Wiltshire (CW) dynamics. This evolving uncertainty serves as the basis for the instantaneous collision probability calculation in our optimization framework.

Fig. 21 illustrates the collision probability ablation study results for the final maneuver vectors listed in Table 6. As shown, the blue line represents the baseline case with no maneuver applied; the red line represents the full plan case where all maneuvers have been implemented; and the dotted lines represent the ablation cases for the first 1 maneuver and first 2 maneuvers. It can be observed that the initial peak collision probability at around 13 min exceeds 10^{-4} . After the first avoidance maneuver, the probability decreases below the threshold, but it rises again at 40 and 60 min, indicating that the residual probability stems from secondary conjunctions caused by prior trajectory changes,

Table 4

Hyperparameters used in PPO training.

Parameter	Value
Learning Rate α	3×10^{-4}
Discount Factor γ	0.995
Entropy Coefficient c_{ent}	0.02
Number of Steps N_{steps}	1024
Batch Size B	256
Optimization Epochs N_{epochs}	10
Hidden Layers	[256, 256]

Table 5

Orbital elements of the protected spacecraft orbit.

a (m)	e	i (deg)	Ω (rad)	ω (rad)	$M(t)$ (rad)
6878,137	0.001	45	0	0	0



Fig. 18. Diagram of the learning curve of the MCTS method in 5-stage strict multistage collision scenario.

Table 6

Maneuver vector table of MCTS.

t (s)	dV_x (m/s)	dV_y (m/s)	dV_z (m/s)	Magnitude (m/s)
2.6421	2.604213	4.14033	-1.624781	5.154044
130.9635	1.570059	-3.580374	-4.125142	5.683394
189.6933	2.045572	-2.421217	-1.771681	3.631185

which reflects the multistage nature of the scenario. After all maneuvers have been implemented, the risk of secondary collisions is reduced to some extent, and the peak collision probability is decreased to 9.99×10^{-5} . These results confirm the need for sequential optimization and that collision avoidance in congested orbits is a progressive risk-reduction process rather than a one-shot solution.

Then, we evaluate the performance of the CEM method with the planner random seed set to 1. The optimization process is executed for 80 iterations to facilitate the convergence of the sampling distribution towards the high-reward regions of the maneuver space. As the training process shown in Fig. 22, the algorithm completed the optimization in 974 s, achieving a cumulative final reward of 667.1256. The peak collision probability is suppressed to 4.71×10^{-5} , and the total fuel consumption is 10.0119 m/s. Furthermore, the final terminal state deviation is maintained at 1.250436, demonstrating that the CEM method provides a relatively efficient and fuel-optimized solution compared to the baseline.

The final maneuver vectors after the optimization are listed in Table 7, and the spatial positions and directions of the impulsive maneuvers are illustrated in Fig. 23a. Fig. 23b illustrates the temporal evolution of the cumulative reward, fuel consumption, and orbital deviation during the evaluation. It can be observed that the cumulative reward does not grow linearly; instead, it remains relatively stable during the propagation and exhibits a sharp increase at the terminal stage, reflecting the successful reward of the final state constraints. Consistent with the maneuver magnitudes in the table, the fuel consumption is primarily concentrated at $t = 0$ and approximately $t = 30$ min. Notably, the orbital deviation is significantly reduced following these primary control actions, demonstrating the effectiveness of the optimized maneuver sequence in both avoiding collisions and maintaining trajectory accuracy.

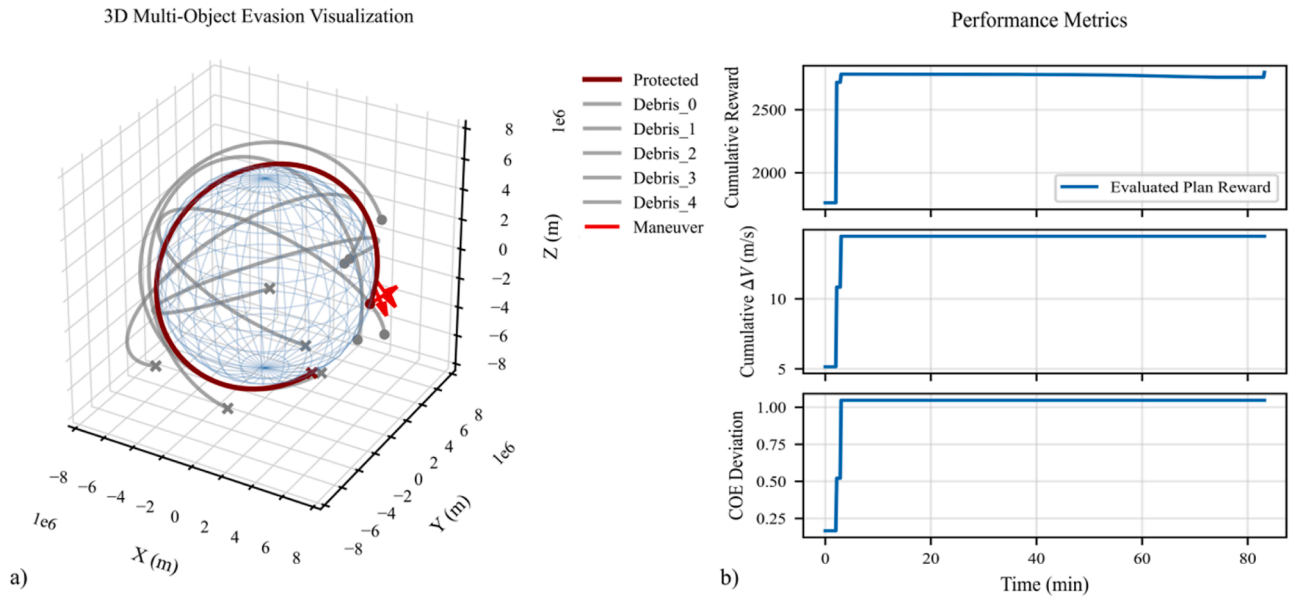


Fig. 19. Diagram of: a) the implementation positions and directions of each maneuver of the MCTS method; b) Cumulative rewards, ΔV and orbital deviation during the full-scenario simulation.

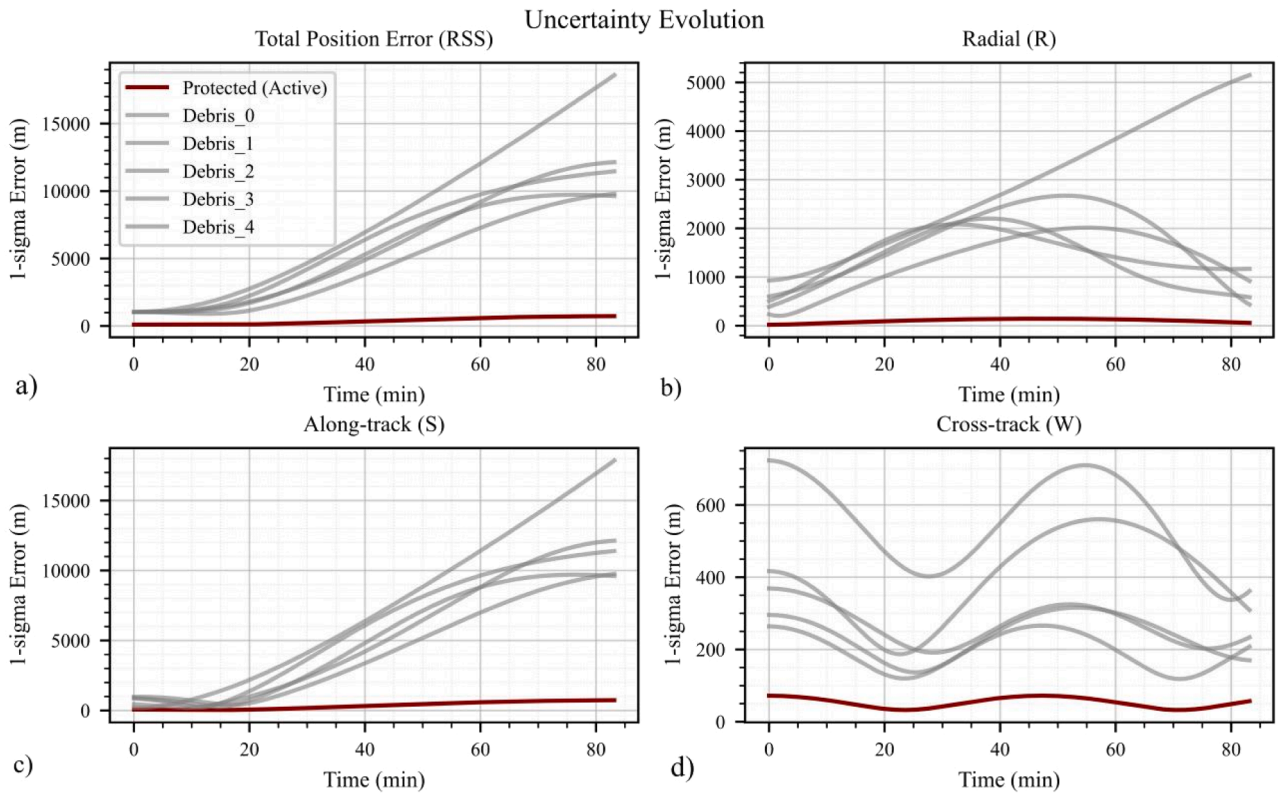


Fig. 20. Evolution of estimation errors for each space object: a) total position error over time; b) radial (R) error component over time; c) along-track (S) error component over time; d) cross-track (W) error component over time.

Fig. 24 illustrates the propagation of position uncertainty for both the protected spacecraft and the five space debris objects throughout the collision avoidance scenario. The initial setting of the covariance is exactly the same as that in the MCTS method. It can be observed that the final propagation results and trend also largely consistent with those of MCTS.

Fig. 25 illustrates the collision probability ablation study results for the final maneuver vectors listed in Table 7. As shown, the blue line

represents the baseline case with no maneuver applied; the red line represents the full plan case where all maneuvers have been implemented; and the dotted lines represent the ablation cases for the first 1–4 maneuvers. It can be observed that the initial peak collision probability at around 13 min exceeds 10^{-4} . After the first avoidance maneuver, the probability decreases significantly, but still has a small peak at around 65 and 80 min. After the second avoidance maneuver, these small peaks have further decreased. And it can be observed that although the

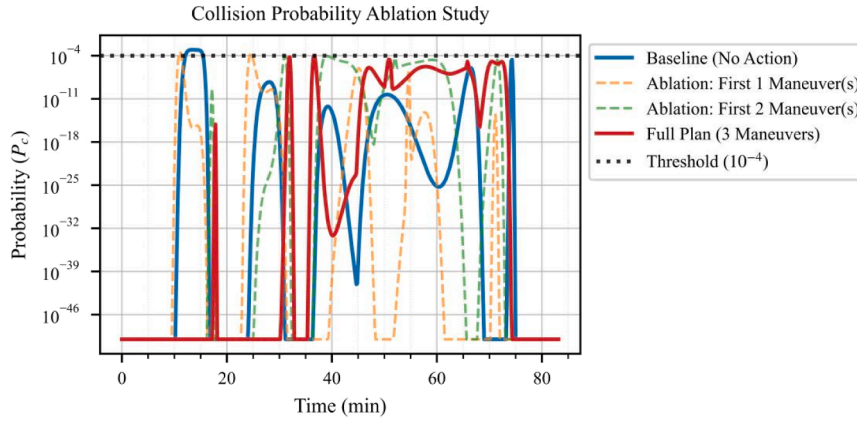


Fig. 21. Time evolution of collision probability for different maneuver configurations of MCTS method in the ablation study.

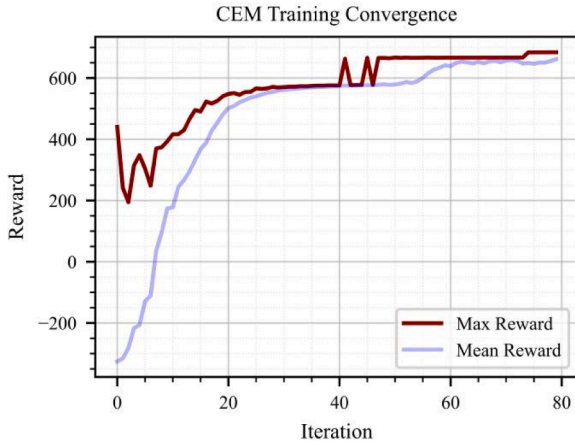


Fig. 22. Diagram of the learning curve of the CEM method in 5-stage strict multistage collision scenario.

Table 7
Maneuver vector table of CEM.

$t/(s)$	$dV_x/(m/s)$	$dV_y/(m/s)$	$dV_z/(m/s)$	Magnitude (m/s)
0	-2.101634	-1.562322	-0.361312	2.643532
1827.3181	0.213255	-2.231242	-0.072865	2.242594
3045.4106	-0.02812	-0.004058	0.035008	0.045086
3513.2168	0.007498	-0.007031	-0.011326	0.015295
5000	-0.758399	1.167303	-1.076193	1.759533

maneuvers 2–5 have decreased the orbital divation, they did not increase the probability of the collision. Ultimately, the CEM method demonstrated better planning performance compared to MCTS, showing better results in terms of safety, efficiency, and track stability.

Finally, we evaluate the performance of the PPO method with the planner random seed set to 1. To ensure better reproducibility of the results, we conducted five groups of synchronous training using seeds (1, 2, 3, 4, 5) in the same environment. The training curves are shown in Fig. 26. The dark orange line denotes the average reward, and the light-colored band represents the 1-sigma standard deviation band. The training process of the neural network is executed for 500,000 iterations. As the training process shown in Fig. 26, the algorithm completed the optimization in 21,030.96 s in average, achieving a cumulative final reward of around 608. During the evaluation phase, we used the trained neural network of seed 5 for inference. The model completed the inference in 2.13 s, and the peak collision probability is suppressed to 7.20×10^{-8} , and the total fuel consumption is 69.4348 m/s.

Furthermore, the final terminal state deviation is maintained at 1.395837, demonstrating that the PPO method tends to converge towards a more conservative feasible solution characterized by significantly higher fuel consumption, which deviates from the high-efficiency, near-optimal trajectories produced by MCTS and CEM methods. However, this conservative nature also yields its own benefit in terms of mission safety, as PPO achieves the lowest peak collision probability among all tested planners. Furthermore, a critical advantage of the PPO approach lies in its operational efficiency: while MCTS and CEM require intensive iterative computations during the planning phase, the pre-trained PPO policy enables near-instantaneous inference. This minimal computational latency makes PPO particularly suitable for real-time onboard deployment, where rapid response to dynamic orbital hazards is paramount.

The final maneuver vectors after the optimization are listed in Table 8, and the spatial positions and directions of the impulsive maneuvers are illustrated in Fig. 27a. Fig. 27b illustrates the temporal evolution of the cumulative reward, fuel consumption, and orbital deviation during the evaluation. Distinct from the previous methods, the PPO agent adopts a high-frequency control strategy in the initial phase. It can be observed that the cumulative reward ascends rapidly within the first 10 min, reflecting the agent's proactive approach to securing a safe trajectory early in the mission. Consistent with the data in Table 8, the fuel consumption exhibits a steep stair-step increase during the first 10 mins due to a sequence of dense impulsive maneuvers, followed by sparse, corrective impulses at approximately $t = 22, 39, 57$ min. Consequently, the orbital deviation increases sharply during the initial high-intensity maneuvering phase and is subsequently maintained at a stable level through these subsequent fine-tuning actions. This behavior demonstrates the PPO method's capability to converge on a robust, multi-impulse policy that prioritizes early safety while maintaining long-term trajectory stability.

Fig. 28 illustrates the propagation of position uncertainty for both the protected spacecraft and the five space debris objects throughout the collision avoidance scenario. The initial setting of the covariance is exactly the same as that in the MCTS and CEM method. It can be observed that the final propagation results and trend also largely consistent with those of MCTS and CEM.

Fig. 29 illustrates the collision probability ablation study for the 13-maneuver sequence optimized by the PPO agent. The blue line represents the baseline (no action), the red line denotes the full execution of the 13-maneuver plan, and the various dashed lines represent partial executions. As shown in the baseline case, the initial collision risk at approximately 13 min significantly exceeds the safety threshold of 10^{-4} . After implementing the first and second avoidance maneuvers, this primary peak is successfully suppressed. However, several secondary risk peaks still persist or emerge at subsequent collision stages around 23, 30, 40, and 70 min. As the number of executed maneuvers increases,

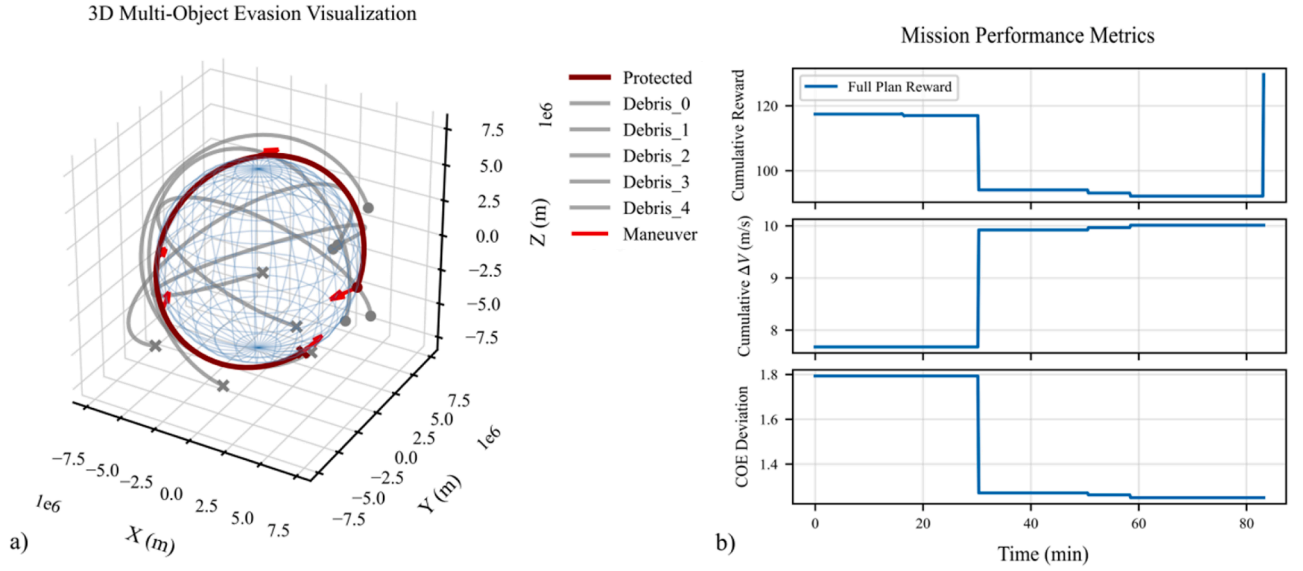


Fig. 23. Diagram of: a) the implementation positions and directions of each maneuver of the CEM method; b) Cumulative rewards, Δv and orbital deviation during the full-scenario simulation.

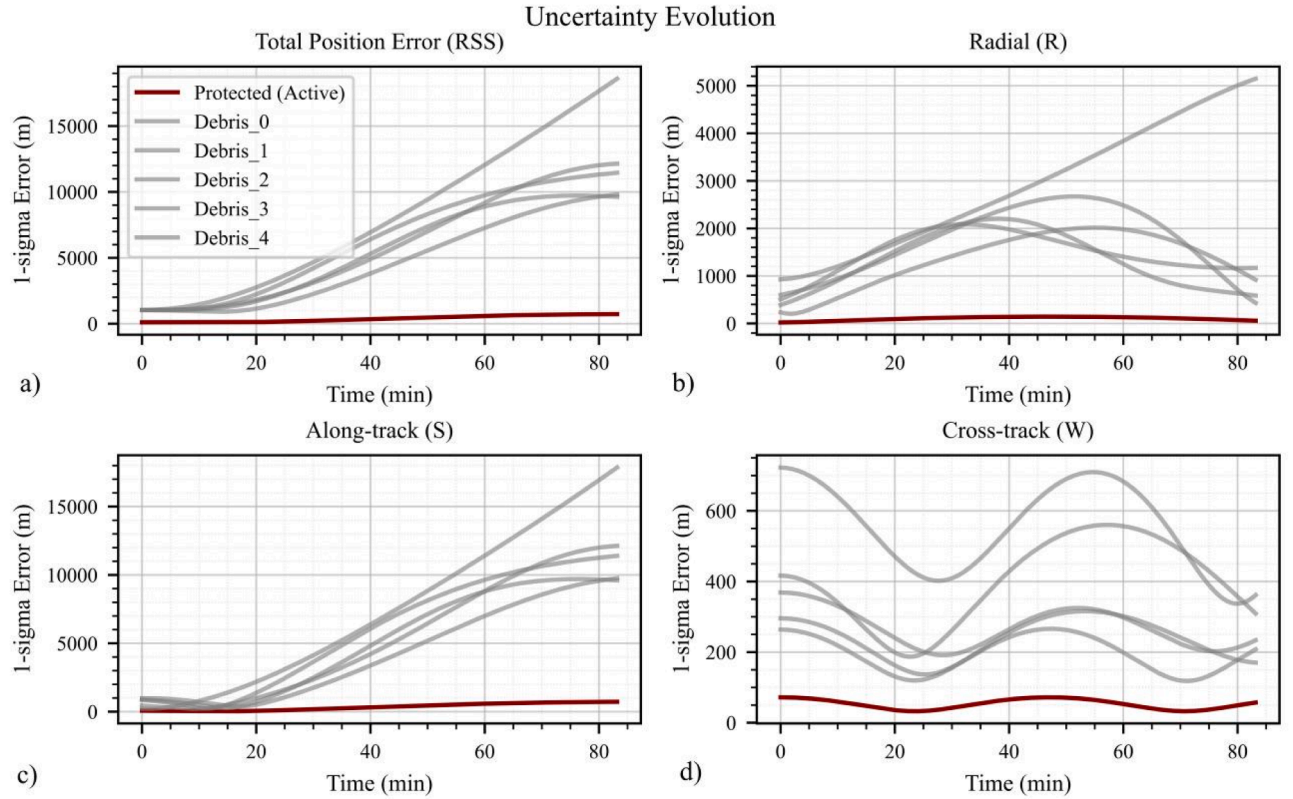


Fig. 24. Evolution of estimation errors for each space object: a) total position error over time; b) radial (R) error component over time; c) along-track (S) error component over time; d) cross-track (W) error component over time.

these residual peaks are progressively attenuated, demonstrating the necessity of the multi-impulse strategy in managing long-term risk. Ultimately, the full 13-maneuver plan (red line) achieves a robust safety margin, with all peak collision probabilities maintained well below 10^{-7} . This ablation study confirms that the PPO policy effectively coordinates a sequence of impulses to eliminate both immediate and latent collision risks throughout the mission duration.

6.2. Multi-Scenario performance comparison of three algorithms

In the experiment, we conduct a unified scaling-oriented comparison among heterogeneous decision-making approaches and employ the method described in Chapter 3 to construct four types of collision scenarios, and each type of the collision scenario utilize five random seeds: (1, 2, 3, 4, 5). In the complex collision scenario, 10% of the debris is generated by the strict multistage collision method described in [Section](#)

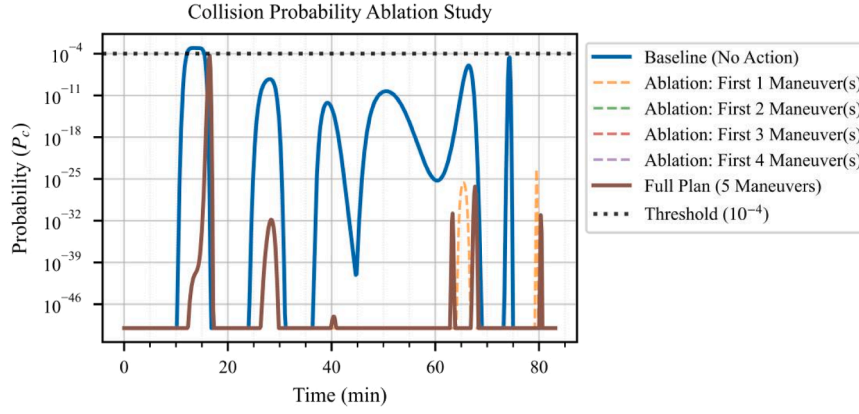


Fig. 25. Time evolution of collision probability for different maneuver configurations of CEM method in the ablation study.

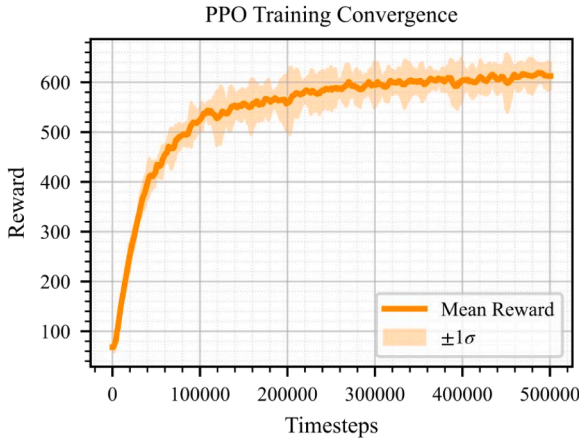


Fig. 26. Diagram of the learning curve of the PPO method in 5-stage strict multistage collision scenario.

Table 8
Maneuver vector table of PPO.

$t/(s)$	$dV_x/(m/s)$	$dV_y/(m/s)$	$dV_z/(m/s)$	Magnitude (m/s)
60	5	5	5	8.660254
120	4.300472	-4.059096	4.86997	7.66074
180	4.585153	-0.721923	3.386609	5.745774
240	5	-1.131902	5	7.161089
300	2.57167	-3.263431	1.72233	4.497765
360	3.23409	-3.269966	3.989524	6.088375
420	2.982093	-2.387233	4.075917	5.586131
480	3.053068	-1.299752	3.707714	4.975713
540	3.051367	-0.927394	3.637618	4.837682
1318.1027	3.055913	-1.81539	1.924431	4.04199
2372.1104	4.34029	-2.084447	2.400653	5.380165
3410.8279	2.507149	-2.67761	0.194747	3.673325
4660.8281	-0.125079	0.833513	-0.746417	1.125845

3.3, 30% by the probability-based multistage collision method in Section 3.4, and the remaining 60% is imported from the real space debris TLE data obtained by CelesTrak as detailed in Section 3.5 [42,46]. The average initial collision probability and the standard deviation of the initial collision probability for each type of scenario, based on the two-body dynamics model and considering perturbation conditions, are shown in Fig. 30 and Table 9. The number of space debris and the number of dangerous debris contained in each scene are also listed in Table 9. Here, dangerous debris refer to those space debris that remain after the warning-stage screening and risk evaluation, which are still likely to collide with the spacecraft. It can be seen that under the two

body dynamic, if only the maximum value of the collision probability is considered, among the four collision scenarios, the simple scenario has the highest collision probability, followed by the complex scenario and the strict scenario, while the probability scenario has the lowest collision probability. However, when J_2 perturbation was introduced, the probability of collisions based on probability significantly increased compared to the strict scenarios and complex scenarios. This might be because the addition of perturbation increased the uncertainty of the scene, thereby affecting the covariance propagation of the orbits and thus causing the probability of collision in the probabilistic-generated cloud-like collisions to increase significantly.

In order to compare the performance of different learning-based methods in a strongly coupled multistage collision scenario, we conduct a unified scaling-oriented comparison among heterogeneous decision-making approaches. In addition to training and optimizing the four types of collision scenarios using MCTS, CEM, and PPO methods mentioned above, we also introduce the Powell method as a representative of traditional numerical optimization to provide a baseline for local search performance without a learning mechanism. During the training and evaluation processes, each method uses five random seeds (1, 2, 3, 4, 5) to initialize 5 planners respectively. The statistical results of the optimization are shown in Table 10 and Fig. 31, where the bar chart represents the average value of the statistical data, while the line error band illustrates the standard deviation of the statistical values. The reduction rate of collision probability p_{red} is defined as the decrease in the peak collision probability after maneuvering divided by the peak collision probability at the initial stage. $\bar{f}$ represents the average fuel consumption and $\bar{d}$ represents the average orbital deviation, $\bar{N}_m$ represents the total count of impulsive velocity increments executed. The "Wall Clock" column separately records the execution practice and training duration of the planners. Among them, the training duration is unique to the deep reinforcement learning method PPO. This is because after obtaining the policy neural network through training, PPO can directly perform inference in new scenarios relying on its generalization ability. It can be seen that these methods have almost all accomplished the avoidance of multistage collisions, but there are differences in their avoidance effects and training time consumption. The detailed analysis of the key indicators is as follows:

Regarding the avoidance effect, each algorithm has its own advantages. We can see that the MCTS and CEM algorithms provide superior safety, achieving collision probability reduction rates exceeding 99.9% in the complex scenario. While the PPO algorithm maintains a stable performance around 90%, the Powell method's effectiveness is more sensitive to scenario types, even failing to provide a positive reduction rate in the "probability" scenario. It is worth noting that in the probabilistic scenario, the reduction effect of the collision probability for all four planning methods is not as good as that in the other three scenarios. Moreover, the Powell method even showed a poor result of an increased

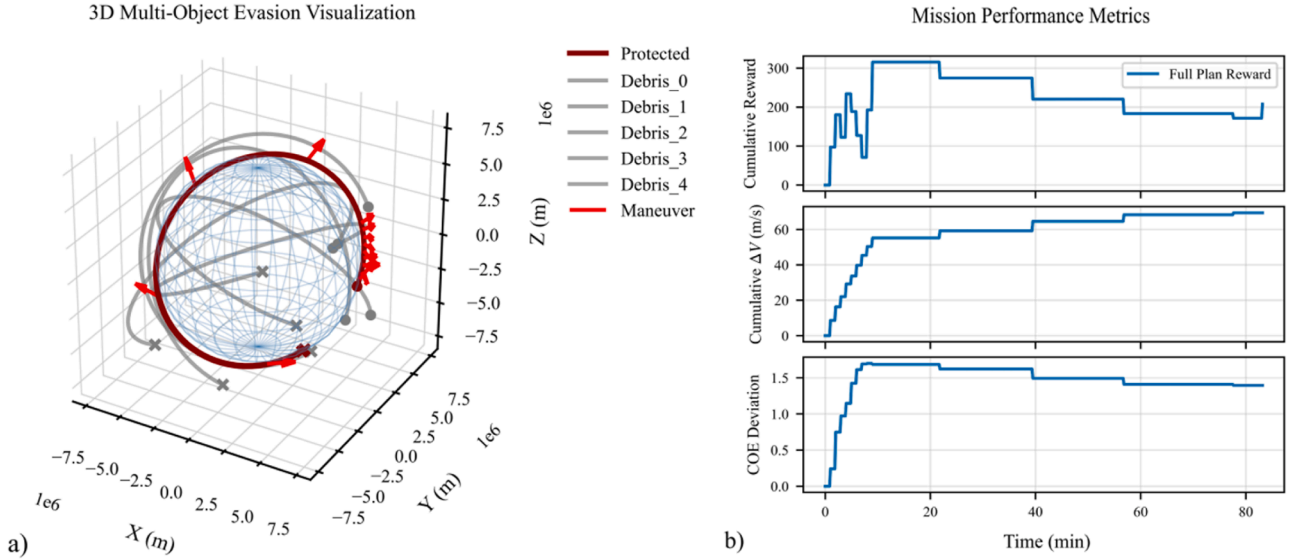


Fig. 27. Diagram of: a) the implementation positions and directions of each maneuver of the PPO method; b) Cumulative rewards, Δv and orbital deviation during the full-scenario simulation.

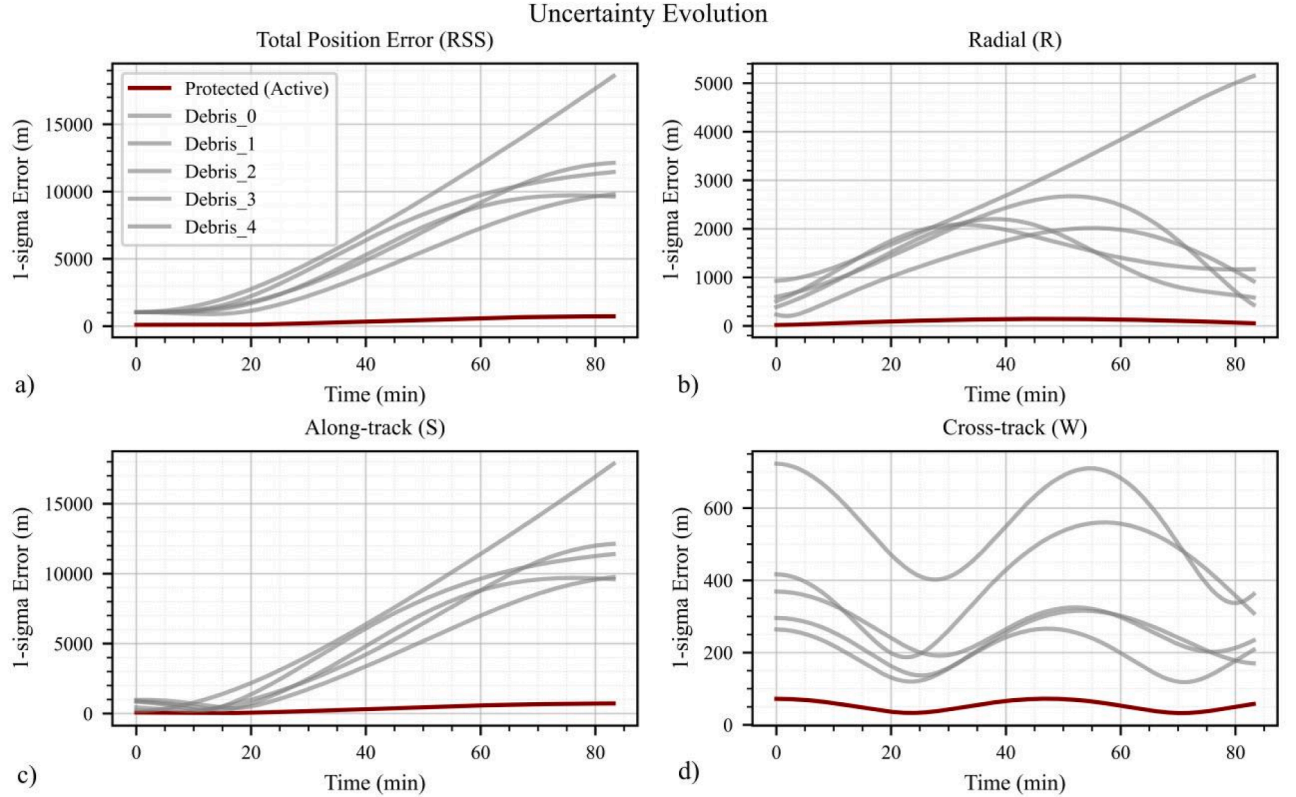


Fig. 28. Evolution of estimation errors for each space object: a) total position error over time; b) radial (R) error component over time; c) along-track (S) error component over time; d) cross-track (W) error component over time.

collision probability here. This might be because the initial collision probability in the probabilistic scenario is relatively low in the two-body dynamics environment, which to some extent may mislead or affect the optimization and learning of the algorithm in the initial stage.

Regarding the number of maneuvers and fuel consumption, algorithms generally increase the frequency of impulses as the number of potential collision events grows. Comparing the algorithms, PPO tends to generate significantly more maneuvers, reaching 15.5 in the complex

scenario, to maintain its policy, whereas the search-based MCTS and traditional Powell methods are more fuel-efficient, typically requiring fewer than 7 maneuvers. The data show that the fuel consumption of the PPO method is significantly higher than that of the other three methods, and this difference becomes even greater as the complexity of the environment increases. This phenomenon indicates that, due to its consideration of task safety and generalization requirements, the PPO method is likely to tend to generate conservative feasible solutions

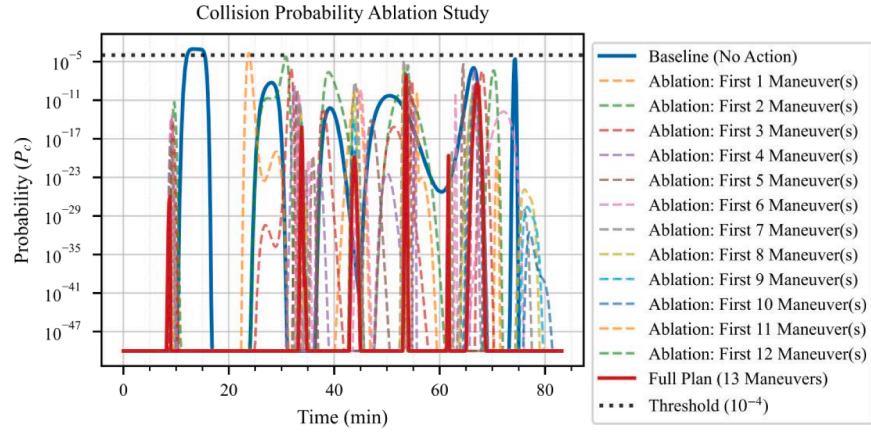


Fig. 29. Time evolution of collision probability for different maneuver configurations of PPO method in the ablation study.

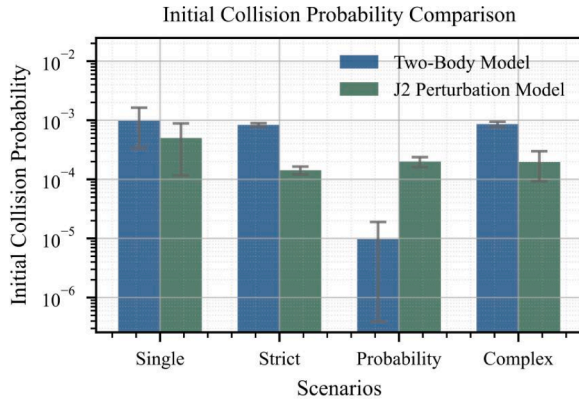


Fig. 30. Mean and standard deviation of the initial collision probability for each scenario type.

Table 9
Summary table of initial parameters for each scenario.

Scenario	Duration (s)	Debris Count	Dangerous Debris Count	Average Initial p ($\times 10^{-6}$) - Two Body	Std of Initial p ($\times 10^{-6}$) - Two Body	Average Initial p ($\times 10^{-6}$) - J_2	Std of Initial p ($\times 10^{-6}$) - J_2
Single Debris Collision	1000	1	1	986.378	646.594	501.022	384.679
Strict Multistage Collision	5000	5	5	834.586	58.7118	142.723	21.9725
Probability-Based Multistage Collision	2000	24	24	9.70514	9.31321	199.287	38.474
Complex Multistage Collision	5000	100	41	858.964	85.9672	195.858	103.245

Table 10
Summary table of the optimization results of different learning-based methods.

Methods	Scenario	$\bar{p}_{red}$	$\bar{f}$	$\bar{d}$	$\bar{N}_m$	Wall Clock (Exec/Train)(s)
Powell	single	9.32E-01	3.08E+00	9.95E-02	2.00	3.55E+00 / -
	strict	6.57E-01	3.95E+00	5.97E-01	7.00	1.73E+02 / -
	probability	-7.69E-01	3.69E+00	2.22E-01	3.00	7.02E+01 / -
	complex	8.81E-01	1.24E+01	1.13E+00	7.00	2.96E+03 / -
CEM	single	9.20E-01	1.82E+00	9.74E-02	2.00	3.30E+02 / -
	strict	1.00E+00	3.77E+00	2.17E-01	4.38	3.15E+03 / -
	probability	1.25E-01	1.57E-01	1.79E-02	2.00	5.58E+03 / -
	complex	1.00E+00	6.72E+00	3.42E-01	4.88	3.43E+04 / -
MCTS	single	9.66E-01	2.72E+00	2.13E-01	3.22	2.36E+02 / -
	strict	1.00E+00	2.58E+00	1.67E-01	3.06	3.23E+03 / -
	probability	9.28E-01	1.25E+00	1.01E-01	3.00	1.64E+04 / -
	complex	1.00E+00	4.27E+00	1.62E-01	3.56	3.55E+04 / -
PPO	single	8.91E-01	5.20E+00	2.08E-01	3.80	3.19E-02 / 1.12E+03
	strict	9.56E-01	3.54E+01	1.14E+00	11.4	2.22E+00 / 2.11E+04
	probability	6.17E-01	4.53E+01	1.01E+00	8.30	4.76E+00 / 8.75E+04
	complex	9.03E-01	6.00E+01	9.71E-01	15.5	2.45E+01 / 2.20E+05

rather than the optimal solution with the lowest fuel consumption.

Regarding the decision time consumption, with the increase of the complexity of the environment, the computational cost of all three algorithms has increased. Complex multistage collision scenarios consume the most time, followed by probability-based multistage collision and strict multistage collision scenarios, and single debris collision scenarios consume the least time. The experimental results show that when the scene complexity is low, the time consumption of the three algorithms is not significantly different, with the Powell method being the most efficient in the single debris scenario with a wall clock time of only 3.55 s. However, as the scene complexity increases, the computational burden of the Powell method grows significantly, reaching 2961.43 s in the complex scenario due to the high-dimensional objective function. In contrast, the PPO algorithm demonstrates a decisive real-time advantage; its online execution time remains under 25 s even in the complex scenario, although it requires a substantial offline training investment, which reaches 219,706.16 s in the complex case. The time consumption of the MCTS and CEM algorithms increases more sharply in dense

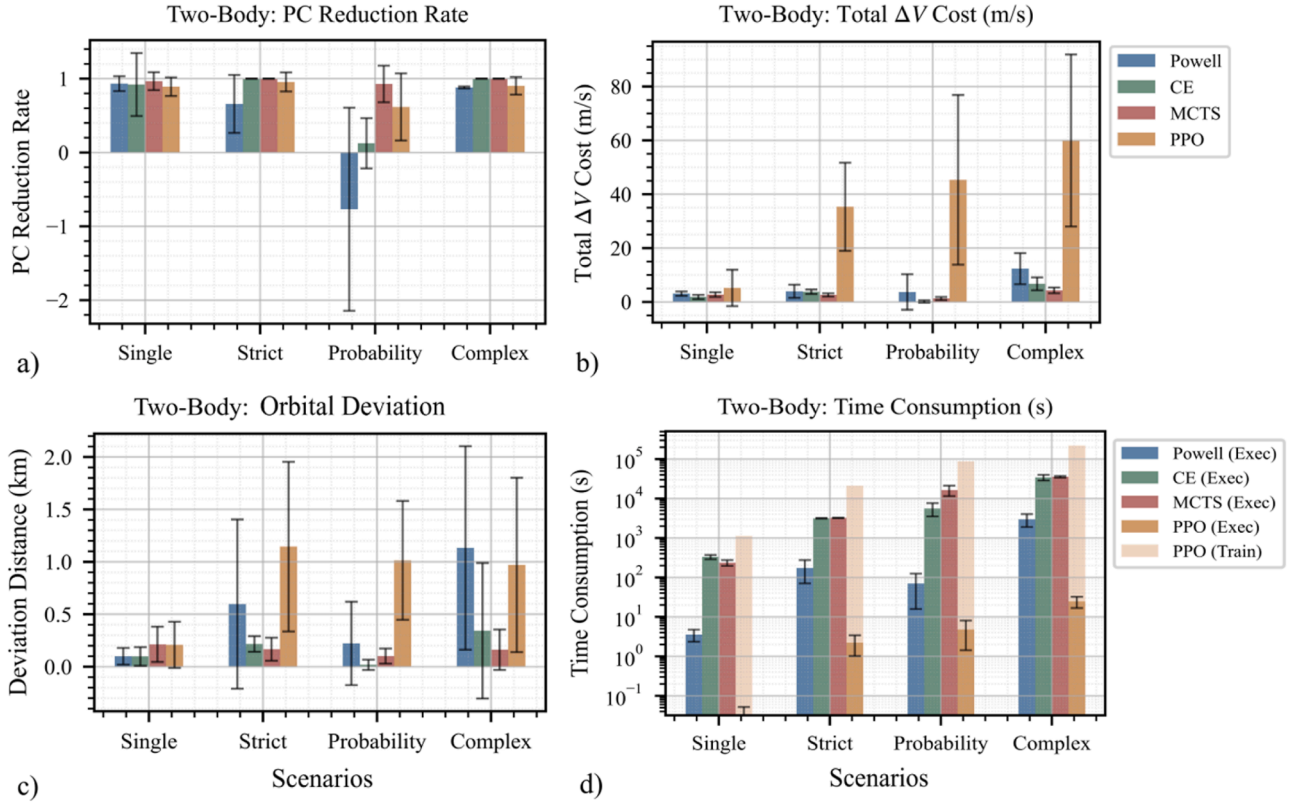


Fig. 31. Statistical comparison of the performance of the four algorithms in various scenarios: a) results of the reduction rate of collision probability; b) results of the total Δv cost; c) results of orbital deviation; d) results of time consumption.

environments, both exceeding 34,000 s in the complex scenario, which is significantly higher than the non-learning Powell method.

It can be observed that when the scale of space debris is small, Powell algorithm performs well in both effectiveness and speed. However, as the scenario complexity increases, its safety becomes less reliable compared to more robust search methods. Using PPO algorithm is a rational choice for missions demanding high-speed online response, provided that sufficient offline training time is available. For scenarios where maximizing safety is the primary objective, MCTS and CEM are the better choices despite their higher online computational costs. The CEM algorithm can provide a reference result when the debris' scale is relatively small but is less time-efficient than MCTS in high-complexity environments.

6.3. Sensitivity of algorithm performance to environmental perturbations

The sensitivity of the algorithm performance to environmental perturbations is analyzed by introducing the J_2 effect into the collision avoidance scenarios. As the primary non-spherical gravitational perturbation in Low Earth Orbit, the J_2 effect significantly complicates orbital propagation and state transition logic, thereby increasing the non-linearity of the decision-making landscape. The statistical results of the optimization are shown in Table 11 and Fig. 32.

It is observed that the introduction of such perturbations leads to a dramatic escalation in computational costs for all online optimization and search-based algorithms. Specifically, the Powell method demonstrates extreme sensitivity to the fidelity of the propagator; its wall clock time in the complex scenario explodes from 2961.43 s under the two-

Table 11
Summary table of the optimization results under J_2 perturbation.

Methods	Scenario	$\bar{p}_{red}$	$\bar{f}$	$\bar{d}$	$\bar{N}_m$	Wall Clock
Powell	single	9.43E-01	2.90E+00	1.45E-01	2.00E+00	8.65E+02
	strict	8.17E-01	5.49E+00	7.87E-01	7.00E+00	3.11E+04
	probability	7.89E-01	8.19E+00	5.32E-01	3.00E+00	1.69E+04
	complex	9.11E-01	1.32E+01	1.48E+00	7.00E+00	4.15E+05
CEM	single	9.99E-01	2.98E+00	1.16E-01	2.00E+00	2.87E+03
	strict	9.98E-01	5.34E+00	5.51E-02	4.67E+00	8.87E+04
	probability	7.71E-01	7.26E+00	2.70E+00	2.00E+00	4.20E+04
	complex	1.00E+00	9.35E+00	1.34E+00	4.98E+00	1.04E+05
MCTS	single	1.00E+00	3.15E+00	4.23E-02	1.80E+00	5.43E+03
	strict	1.00E+00	2.92E+00	1.63E-01	1.70E+00	2.19E+04
	probability	8.32E-01	2.35E+00	2.59E-01	1.50E+00	6.10E+04
	complex	9.99E-01	7.26E+00	1.93E-01	3.20E+00	1.05E+05
PPO	single	9.79E-01	2.53E+01	2.74E+00	5.63E+00	8.02E-01
	strict	9.47E-01	6.96E+01	1.46E+00	1.28E+01	2.11E+01
	probability	5.32E-01	4.31E+01	6.53E-01	9.83E+00	3.74E+01
	complex	6.56E-01	6.85E+01	2.10E+00	1.91E+01	1.85E+02

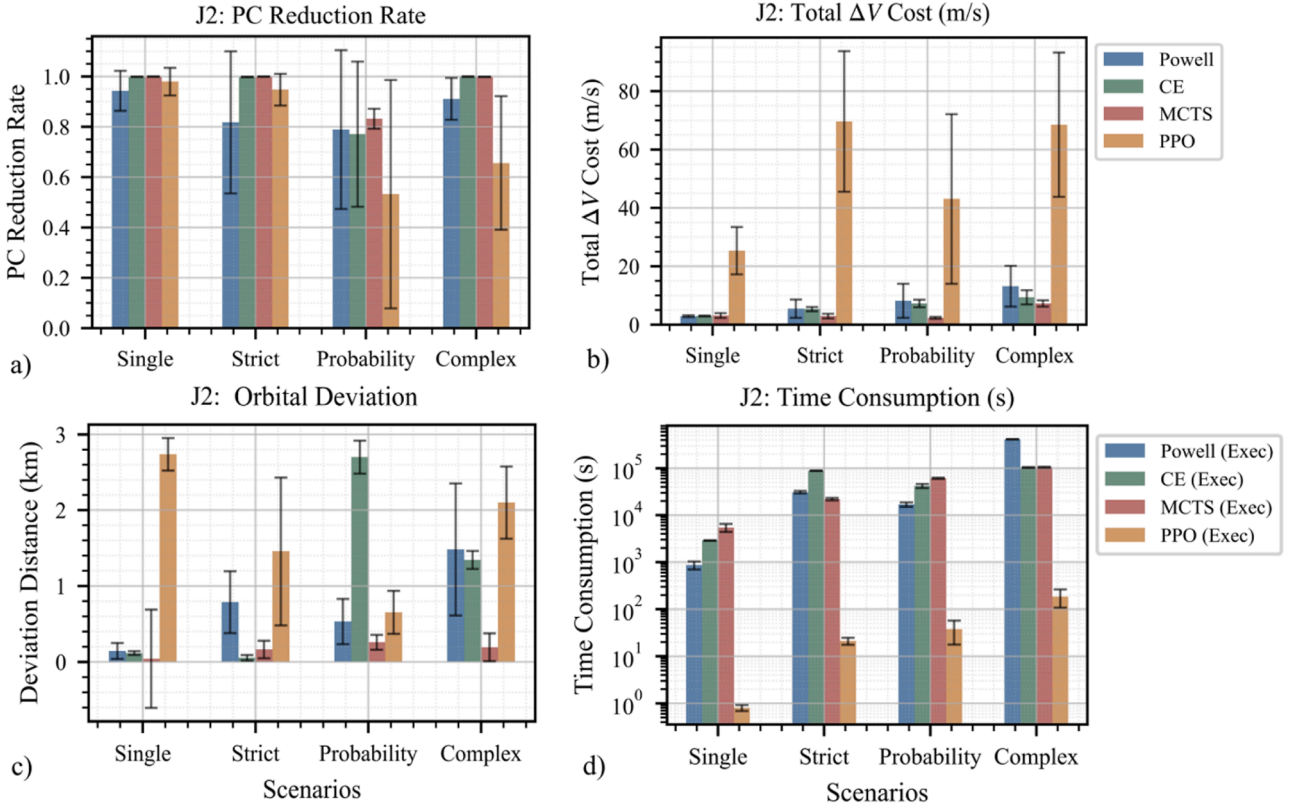


Fig. 32. Statistical comparison of the performance of the four algorithms under J_2 perturbation in various scenarios: a) results of the reduction rate of collision probability; b) results of the total Δv cost; c) results of orbital deviation; d) results of time consumption.

body model to a prohibitive 414,600.14 s under the J_2 model. Similarly, search-based algorithms such as MCTS and CEM experience a significant surge in computational burden, with execution times tripling to exceed 100,000 s in dense environments. This massive increase in temporal overhead underscores the computational bottleneck faced by classical optimization and online search methods when high-fidelity physics is involved. Regarding the deep reinforcement learning-based approach, a cross-model evaluation was conducted to assess the robustness of the PPO agent. Due to the immense computational burden of numerical integration in the J_2 environment, the PPO agent was not retrained; instead, the policy trained under the two-body model was directly evaluated in the perturbed environment to test its generalization capability under model mismatch. The results indicate that while the PPO execution time increases to 184.97 s in the complex scenario—primarily due to the increased overhead of reward calculation—it still maintains a significant speed advantage over search-based methods. However, the avoidance effectiveness of PPO exhibits moderate decay, with the collision probability reduction rate in the complex scenario dropping from 0.90 to 0.65. This performance degradation reflects the sensitivity of end-to-end policies to refined dynamical changes, yet the fact that the agent still achieves over 60% risk reduction without retraining demonstrates a notable degree of zero-shot generalization to more complex orbital environments.

In contrast to the fixed policy of PPO, MCTS and CEM maintain superior safety performance and fuel savings effect under J_2 perturbations, consistently achieving collision probability reduction rates exceeding 99.8%. This robustness stems from their online search mechanisms, which allow the algorithms to actively "explore" and adapt to the J_2 dynamics during the decision process. However, this high level of safety is achieved at the cost of extreme computational latency, making these methods less suitable for missions requiring instantaneous response.

Although the addition of perturbation has increased the initial collision probability of the probability scenario, the performance of each

method in this scenario has improved. However, the unique characteristics of each method have not undergone significant changes. The Powell method is an ideal choice with a faster speed in simple scenarios. When the scenario becomes increasingly complex and the task is not urgent, the CEM and MCTS methods are a more preferred choice. Although they may consume more time than the PPO method, they can generate higher-quality avoidance strategies using less fuel, while ensuring sufficient safety redundancy. However, when the avoidance task is very urgent or requires on-board autonomous avoidance calculations, due to the limitations of computing power, the advantage of PPO method in quickly generating feasible maneuvers gradually becomes apparent. It can transfer the main time-consuming part of the optimization training to the ground simulation training, and only needs to perform generalized reasoning during autonomous avoidance, greatly saving the time for strategy generation. The experimental results reveal a critical trade-off between real-time efficiency and physical fidelity: while search-based methods offer absolute robustness by incorporating high-fidelity models online, pre-trained policies like PPO offer a pragmatic alternative by decoupling decision speed from dynamical complexity, provided the system can tolerate moderate performance losses due to model mismatch. Future research could further explore transfer learning techniques to mitigate this performance decay without incurring the prohibitive costs of full retraining in perturbed environments by changing the quantity of space debris and the total number of steps of the simulation.

6.4. Time complexity under problem scale expansion

To systematically evaluate how algorithmic computational efficiency is affected by increasing problem scale, we designed the following two groups of experiments by changing the quantity of space debris and the total number of steps of the simulation.

In experiment one, we used the complex multistage collision

scenario, set the simulation step size to 10 s and total steps to 500. In this experiment, 10% of the debris is generated by the strict multistage collision method described in Section 3.3, 30% by the probability-based multistage collision method in Section 3.4, and the remaining 60% is imported from the real space debris TLE data obtained by Celestrak as detailed in Section 3.5 [42,46]. The total number of debris is gradually increased from 5 to 100 to demonstrate system scalability. In Fig. 33, the x-axis represents the total number of space debris objects in the environment. The left y-axis corresponds to the optimization time of each algorithm, with solid lines indicating results after screening-based early warning and dashed lines representing those without screening. The right y-axis indicates the number of the dangerous debris candidates actually involved in optimization, which is illustrated using transparent blue shaded areas. Data suggest that in this scenario, the dangerous debris screened out accounts for approximately half of the total, the implementation of the previously proposed dangerous debris screening and multistage collision warning significantly improved the optimization efficiency of all algorithms. This improvement is most pronounced in MCTS, followed by PPO and CEM. Among these methods, the PPO method requires the longest offline training time, significantly exceeding MCTS and CEM. However, its inference time is markedly shorter than other methods, as indicated by the yellow line at the bottom of the figure, which is 1–3 orders of magnitude shorter. As the amount of debris increases, the time consumption of CEM grows rapidly, while that of PPO and MCTS increases a little more slowly. MCTS tends to require more training time than CEM, and its time consumption grows faster as the quantity of space debris increases.

In experiment two, we also used the complex multistage collision scenario, set the amount of space debris to 60, the simulation step size to 10 s, and gradually increased the total number of simulation steps from 20 to 50,000. As shown in Fig. 34, the PPO method requires the longest duration in total training time, followed by MCTS and CEM. However, it is noteworthy that the PPO method's online inference time after offline training is significantly shorter than that of online optimization algorithms. Regarding time growth rate, the MCTS method shows the fastest increase with increasing time steps, followed by the CEM method. When the simulation steps increase to approximately 20,000, the online optimization time of MCTS even begins to exceed the training duration of PPO. The PPO method demonstrates a slower time growth rate, and the computational time of the PPO method even remains virtually constant during offline inference. This may be because, when solving the problem using the PPO algorithm, most of the computation time is spent on gradient calculations during neural network propagation. The optimization time is primarily constrained by the network architecture—namely, the number of layers and nodes—rather than by the total number of simulation steps. This gives the PPO algorithm distinct advantages

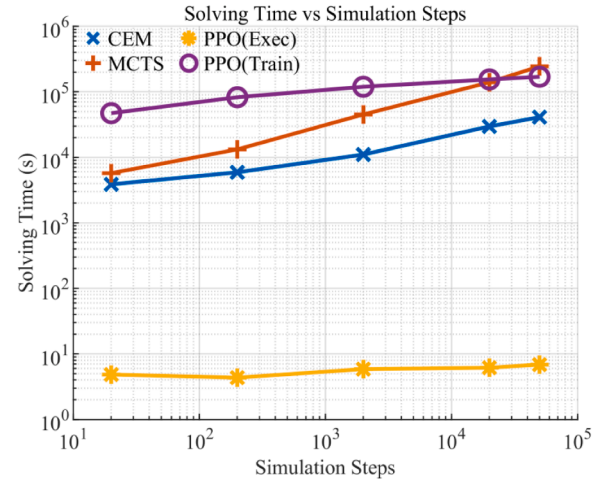


Fig. 34. Diagram of the solving time and the total number of simulation steps.

in multistage collision warning and avoidance scenarios involving long time horizons.

7. Conclusions and further discussion

This study investigates the problem of early warning and autonomous maneuver planning in the context of multistage space debris collisions in LEO. To address this, we propose a comprehensive solution framework that integrates a debris screening and multistage collision warning mechanism with reinforcement learning-based optimization strategies. Specifically, we propose a relatively systematic construction method for multistage collision scenarios of space debris, construct four representative multistage collision scenarios of different complexity levels, and apply three learning-based methods - MCTS, CEM, PPO - and a Powell baseline to learn and optimize avoidance maneuvers. To evaluate algorithm performance, we conducted extensive simulations across varying environmental scales, including different debris quantities and simulation duration.

The results demonstrate that all three learning-based approaches are capable of effectively handling multistage collision avoidance tasks under varying levels of environmental complexity, achieving substantial reductions in peak collision probability while maintaining feasible fuel consumption and orbital deviation. Among them, the search-based methods, namely MCTS and CEM, consistently exhibit superior performance in terms of safety and fuel efficiency, particularly in complex multistage scenarios involving a large number of debris objects. These methods are able to explore the maneuver space more thoroughly and adapt to dynamically evolving collision risks, leading to high-quality avoidance strategies with relatively compact maneuver sequences. However, this performance advantage comes at the cost of significantly increased computational burden, which grows rapidly with both the number of debris objects and the simulation horizon. In contrast, the PPO-based approach demonstrates a fundamentally different trade-off. While its avoidance strategies tend to be more conservative and fuel-intensive, PPO achieves highly stable performance across different scenarios and maintains a near-constant inference time regardless of problem scale. This property makes it particularly attractive for real-time or onboard applications where computational resources are limited and rapid response is required. Furthermore, the experimental results under J_2 perturbation indicate that although the performance of PPO degrades due to model mismatch, it still retains a reasonable level of effectiveness without retraining, suggesting a certain degree of generalization capability. Overall, the comparative analysis highlights that no single method dominates across all performance metrics. In addition, the Powell method, adopted as a representative local

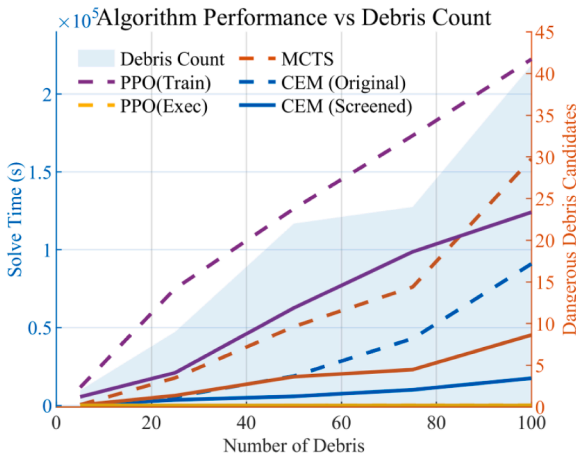


Fig. 33. Diagram of the relationship between algorithm performance and the number of debris.

optimization baseline, also demonstrates competitive performance in simple scenarios but shows limited scalability as the problem dimension increases. Instead, the choice of algorithm depends on the specific mission requirements, balancing safety, fuel efficiency, and computational constraints. The proposed framework provides a unified platform for evaluating these trade-offs in a scalable and systematically constructed multistage collision environment, thereby contributing to a clearer understanding of the applicability of different decision-making strategies in complex orbital scenarios.

Despite the results obtained in this study, several limitations of the present study should be acknowledged. First, the orbital dynamics model adopted in this work remains relatively simplified. Although the J_2 perturbation is incorporated in part of the experiments, higher-order perturbations and environmental effects are not considered, which may limit the fidelity of the simulation in real operational contexts. In addition, the reliance on high-frequency numerical integration introduces considerable computational overhead, particularly when higher-fidelity dynamics are involved, as evidenced by the significant increase in runtime under perturbed conditions. From a methodological perspective, the proposed screening and warning mechanism, while effective in reducing the dimensionality of the decision problem, is inherently heuristic and may overlook potential long-term collision risks that do not manifest as immediate orbital intersections. Furthermore, the constructed multistage scenarios are defined within finite time horizons and may not fully capture the open-ended and continuously evolving nature of the space debris environment. Regarding the algorithm comparison section, this study selected a group of representative methods for analysis. Using the classic local optimization method as the benchmark is not sufficient to comprehensively cover the performance of more advanced deterministic techniques, such as some methods using differential algebra that have already been able to perform polynomial planning with higher quality and greater speed [32]. As a result, the conclusions drawn in this work are primarily limited to the comparative behavior of the chosen algorithms within the proposed framework. Future work will aim to incorporate more efficient orbit propagation techniques and extend the framework to accommodate higher-fidelity dynamics and more diverse optimization strategies.

Data availability statement

Some or all data, models, or code that support the findings of this study are available from the corresponding author upon reasonable request.

CRediT authorship contribution statement

Liwen Yang: Writing – original draft, Visualization, Validation, Methodology, Investigation. **Jikai Wang:** Validation, Methodology, Investigation. **Ming Xu:** Supervision, Project administration. **Xue Bai:** Supervision, Methodology, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is supported by the Beijing Natural Science Foundation under Grant Number L252017 and 3264041, the Postdoctoral Fellowship Program and China Postdoctoral Science Foundation under Grant Number BX20250465.

References

- [1] Celestrak, "Satellite catalog (SATCAT) box score," 2025. [Online]. Available: <http://celestrak.org/satcat/boxscore.php>. [Accessed: Jul. 24, 2025].
- [2] P. Eichler, Analysis of the necessity and the effectiveness of countermeasures to prevent a chain reaction of collisions, *Acta Astronaut.* 26 (7) (1992) 487–495.
- [3] F.R. Hoots, P.W. Schumacher Jr., R.A. Glover, History of analytical orbit modeling in the U.S. space surveillance system, *J. Guid., Control, Dyn.* 27 (2) (2004) 174–185, <https://doi.org/10.2514/1.9161>.
- [4] D. Vallado, P. Crawford, R. Hujsak, T.S. Kelso, Revisiting spacetrack report #3, in: *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, 2006, <https://doi.org/10.2514/6.2006-6753>.
- [5] D.J. Fonte, Comparison of orbit propagation in the research and development Goddard trajectory determination system, in: *AAS/AIAA Astrodynamics Conference*, Halifax, Nova Scotia, Canada, 1995.
- [6] R.P. Patera, Quick method to determine long-term orbital collision risk, in: *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Reston, VA, USA, 2002. Paper No. AIAA-2002-1809.
- [7] M.R. Akella, K.T. Alfriend, Probability of collision between space objects, *J. Guid. Control Dyn.* 23 (5) (2000) 769–772, <https://doi.org/10.2514/2.4611>.
- [8] F.R. Hoots, L.L. Crawford, R.L. Roehrich, An analytic method to determine future close approaches between satellites, *Celest. Mech.* 33 (2) (1984) 143–158.
- [9] P.A. Dybczynski, T.J. Jopek, R.A. Serafin, On the minimum distance between two keplerian orbits with a common focus, *Celest. Mech.* 38 (1986) 345–356.
- [10] G.F. Gronchi, An algebraic method to compute the critical points of the distance function between two keplerian orbits, *Celest. Mech. Dyn. Astron.* 93 (1–4) (2005) 295–329.
- [11] R. Armellin, et al., Computing the critical points of the distance function between two keplerian orbits via rigorous global optimization, *Celest. Mech. Dyn. Astron.* 107 (3) (2010) 377–395.
- [12] D. Casanova, C. Tardioli, A. Lemaitre, Space debris collision avoidance using a three-filter sequence, *Mon. Notices Royal Astron. Soc.* 442 (4) (2014) 3235–3242, <https://doi.org/10.1093/mnras/stu1065>.
- [13] K. Merz, et al., ESA's collision avoidance service: current status and special cases, in: *Proceedings of the 8th European Conference on Space Debris*, ESA Space Debris Office, 2021.
- [14] K. Chan, Collision probability analysis for earth orbiting satellites, *Space Cooperation into the 21st Century*, Univelt, San Diego, CA, USA, 1997, pp. 1033–1048.
- [15] J.L. Foster and H.S. Estes, "A parametric analysis of orbital debris collision probability and maneuver rate for space vehicles," NASA, Houston, TX, USA, NASA Tech. Rep. JSC-26099, 1992.
- [16] S. Alfano, A numerical implementation of spherical object collision probability, *J. Astronaut. Sci.* 53 (1) (2005) 103–109.
- [17] V.A. Chobotov, D.E. Herman, C.G. Johnson, Collision and debris hazard assessment for a low-Earth-orbit space constellation, *J. Spacecr. Rockets* 34 (2) (1997) 233–238, <https://doi.org/10.2514/2.3198>.
- [18] X.Z. Bai, L. Chen, G.J. Tang, Explicit expression of collision probability in terms of RSW and NTW components of relative position, *Adv. Space Res.* 52 (6) (2013) 1078–1096, <https://doi.org/10.1016/j.asr.2013.05.034>.
- [19] S. Hongqiang, Z. Zhanyue, H. Ewen, Comparative research on early warning determination methods of spacecraft collision, *IOP Conf. Ser.: Mater. Sci. Eng.* 751 (1) (2020) 012018, <https://doi.org/10.1088/1757-899X/751/1/012018>.
- [20] G. Campiti, et al., Orbital kinematics of conjuncting objects in Low-Earth Orbit and opportunities for autonomous observations, *Acta Astronaut.* 208 (2023) 355–366.
- [21] G. Campiti, G. Brunetti, C. Ciminelli, Detectability of potentially colliding space objects via star trackers on at-risk satellites, *IEEE Trans. Aerosp. Electron. Syst.* (2025).
- [22] S. Alfano, Collision avoidance maneuver planning tool, in: *Proceedings of the 15th AAS/AIAA Astrodynamics Specialist Conference*, Lake Tahoe, CA, USA, 2005, pp. 7–11.
- [23] J. Mueller, Onboard planning of collision avoidance maneuvers using robust optimization, in: *Proceedings of the AIAA Infotech@Aerospace Conference and AIAA Unmanned... Unlimited Conference*, Reston, VA, USA, 2009, <https://doi.org/10.2514/6.2009-2051>. Paper No. AIAA 2009-2051.
- [24] J.B. Mueller, P.R. Griesemer, S.J. Thomas, Avoidance maneuver planning incorporating station-keeping constraints and automatic relaxation, *J. Aerosp. Inf. Syst.* 10 (6) (2013) 306–322, <https://doi.org/10.2514/1.54971>.
- [25] B. Kelly, S. De Picciotto, Probability based optimal collision avoidance maneuvers, in: *Proceedings of the AIAA Space 2005 Conference & Exposition*, Long Beach, CA, USA, 2005, <https://doi.org/10.2514/6.2005-6775>. Paper No. AIAA 2005-6775.
- [26] H.G. Lewis, G.G. Swinerd, Comparison of methods for predicting collision risk, *Sci. Technol. Ser.* 109 (2004) 207–226.
- [27] C. Bombardelli, J. Hernando-Ayuso, Optimal impulsive collision avoidance in low Earth orbit, *J. Guid. Control Dyn.* 38 (2) (2015) 217–225, <https://doi.org/10.2514/1.G000742>.
- [28] C. Bombardelli, Analytical formulation of impulsive collision avoidance dynamics, *Celest. Mech. Dyn. Astron.* 118 (2) (2014) 99–114.
- [29] J.L. Gonzalo, C. Colombo, P. Di Lizia, Analytical framework for space debris collision avoidance maneuver design, *J. Guid. Control Dyn.* 44 (3) (2021) 469–487.
- [30] R. Armellin, Collision avoidance maneuver optimization with a multiple-impulse convex formulation, *Acta Astronaut.* 186 (2021) 347–362.
- [31] M.F. Palermo, Numerically efficient methods for impulsive and low-thrust collision avoidance manoeuvre design. M.S. thesis, Department of Aerospace Science and Technology, Politecnico di Milano, Milan, Italy, 2021.

- [32] Z. Pavanello, L. Pirovano, R. Armellin, Efficient collision avoidance manoeuvres under multiple polynomial constraints, *Acta Astronaut.* 234 (Sep. 2025) 548–559, <https://doi.org/10.1016/j.actaastro.2025.05.002>.
- [33] X. Cao, et al., Intelligent sequential multi-impulse collision avoidance method for non-cooperative spacecraft based on an improved search tree algorithm, *Chin. J. Aeronaut.* 38 (4) (Apr. 2025). Art. no. 103205.
- [34] J.D. Seong, H.D. Kim, Collision avoidance maneuvers for multiple threatening objects using heuristic algorithms, *Proc. Inst. Mech. Eng. G J. Aerosp. Eng.* 229 (2) (2015) 256–268, <https://doi.org/10.1177/0954410014530678>.
- [35] E.H. Kim, H.D. Kim, H.J. Kim, A study on the collision avoidance maneuver optimization with multiple space debris, *J. Astron. Space Sci.* 29 (1) (2012) 11–21, <https://doi.org/10.5140/JASS.2012.29.1.011>.
- [36] H. Yuan, D. Li, Deep reinforcement learning for rendezvous guidance with enhanced angles-only observability, *Aerosp. Sci. Technol.* 129 (2022) 107812, <https://doi.org/10.1016/j.ast.2022.107812>.
- [37] Q. Qu, K. Liu, W. Wang, J. Lü, Spacecraft proximity maneuvering and rendezvous with collision avoidance based on reinforcement learning, *IEEE Trans. Aerosp. Electron. Syst.* 58 (6) (2022) 5823–5834, <https://doi.org/10.1109/TAES.2022.3180271>.
- [38] S. Liu, C. Mu, Z. Liu, Autonomous spacecraft collision avoidance with multiple space debris based on reinforcement learning, in: *Proceedings of the 2023 42nd Chinese Control Conference (CCC)*, Beijing, China, 2023, pp. 2329–2334, <https://doi.org/10.23919/CCC58697.2023.10241000>.
- [39] C. Mu, et al., Autonomous spacecraft collision avoidance with a variable number of space debris based on safe reinforcement learning, *Aerosp. Sci. Technol.* 149 (2024) 109131, <https://doi.org/10.1016/j.ast.2024.109131>.
- [40] A. Solomon, C. Paduraru, Collision avoidance and return manoeuvre optimisation for low-thrust satellites using reinforcement learning, in: *Proceedings of the 75th International Astronautical Congress (IAC)*, Milan, Italy, 2024, <https://doi.org/10.5220/0013249000003890>.
- [41] J. Woo, N. Kim, Collision avoidance for an unmanned surface vehicle using deep reinforcement learning, *Ocean Eng.* 199 (2020) 107001, <https://doi.org/10.1016/j.oceaneng.2020.107001>.
- [42] T.S. Kelso, Analysis of the Iridium 33-Cosmos 2251 collision, *Adv. Astronaut. Sci.* 135 (2) (2009) 1099–1112 [Online]. Available: <https://celestrak.org/publications/AAS/09-368/AAS-09-368.pdf>.
- [43] F.J. Krage, “NASA spacecraft conjunction assessment and collision avoidance best practices handbook,” NASA, Tech. Rep. NASA/SP-20230002470, Feb. 28, 2023.
- [44] F. Frey, C. Colombo, Simulating a breakup event and propagating the orbits of space debris, *Celestial Mech. Dyn. Astron.* (2024), <https://doi.org/10.1007/s10569-024-10205-3>.
- [45] N. Cimmino, et al., Tuning of NASA standard breakup model for fragmentation events modelling, *Aerospace* 8 (7) (Jul. 2021), <https://doi.org/10.3390/aerospace8070185>. Art. no. 185.
- [46] Celestrak, “Chinese ASAT test,” 2025. [Online]. Available: <https://celestrak.org/events/asat.php>. [Accessed: Jul. 19, 2025].
- [47] P. Kovář, P. Purič, K. Kovářová, Study of the two-line element accuracy by 1U CubeSat with a GPS receiver, *Sensors* 22 (8) (Apr. 2022) 2902, <https://doi.org/10.3390/s22082902>.