Reinforcement Learning for Two-Dimensional Satellite Collision Avoidance

Adam Abdin

March 5, 2025

1 Introduction

The rapid growth of satellite deployment in recent years has led to an increasing concern over space debris and collision risks. Since 2004, the number of space launches has steadily risen, with over 100 satellites being placed into orbit annually. This trend is expected to accelerate due to large-scale projects such as OneWeb and Starlink, which aim to establish vast satellite constellations for global internet coverage.

As more objects populate Earth's orbit, the probability of satellite-debris collisions increases significantly. Even a small piece of debris traveling at orbital speeds can cause catastrophic damage to a satellite, leading to service disruptions, financial losses, and the creation of even more debris. Effective collision avoidance strategies are therefore essential for ensuring the long-term sustainability of space operations.

In this project, we formulate the Satellite Collision Avoidance (SCA) problem as a two-dimensional decision control task. The satellite must navigate its orbit while avoiding collision with dynamically appearing debris, making real-time trajectory adjustments using limited fuel. The goal is to develop an RL-based controller capable of maximizing mission success while minimizing fuel consumption and deviation from the planned orbit.

Students will explore the application of RL techniques to space operations by:

- Formulating the problem as a Markov Decision Process (MDP).
- Designing an RL environment that simulates satellite motion and debris interactions.
- Implementing and training an RL agent to make optimal avoidance decisions.
- Evaluating the trained agent's performance under different initial conditions.

2 Problem Description

To study collision avoidance in a controlled and computationally feasible setting, we simplify the Satellite Collision Avoidance (SCA) problem to a two-dimensional representation. In this setup, the satellite, referred to as the agent, follows a predefined orbital trajectory while navigating a dynamic environment containing space debris.

2.1 Satellite Motion Model

At the initial time step $t = 0$, the satellite's planned trajectory is projected onto a two-dimensional space $\mathbb{R}^2$ and defined as:

$$O^t = (V_x t, 0), \quad \forall t \geq 0, \tag{1}$$

where O^t represents the satellite's intended position at time t. The satellite starts from an initial position:

$$S^0 = (S_x^0 = 0, S_y^0 = 0), \tag{2}$$

with an initial velocity:

$$V^0 = (V_x, V_y^0 = 0). \tag{3}$$

Here, V_x is a positive, fixed velocity component that determines the forward motion along the orbital path. The satellite has the ability to adjust its trajectory by modifying its vertical velocity V_y^t, enabling evasive maneuvers when necessary.

2.2 Debris Observation and Interaction

Throughout its mission, the satellite must avoid collisions with space debris, which are randomly distributed and move unpredictably. At any time step $t \geq 0$, the satellite's onboard system observes the spatial environment and records:

- N: The number of detected debris clusters.
- $D_n^t \in \mathbb{R}^2$: The position of the nth debris cluster at time t.
- $W_n^t \in \mathbb{R}^2$: The velocity of the nth debris cluster at time t.

Each debris cluster follows a unique trajectory, and the likelihood of collision between the satellite and debris n at time t is modeled by the probability function:

$$\Pr\{\text{Satellite collides with debris } n \text{ at time } t\} = P(S^t, D_n^t). \tag{4}$$

The survival probability of the satellite at time t, considering the presence of N debris clusters, is given by:

$$\mathbb{P} = \prod_{n=1}^{N} (1 - P(S^t, D_n^t)). \tag{5}$$

This equation assumes that collisions with different debris clusters are independent events, meaning that the total probability of survival is obtained by considering the probability of avoiding each debris individually.

2.3 Challenges and Objectives

The satellite must make real-time decisions to balance three competing objectives:

1. Collision avoidance: Preventing impact with space debris.
2. Fuel efficiency: Conserving onboard fuel for mission longevity.
3. Service quality: Minimizing deviations from the planned trajectory to maintain operational effectiveness.

The challenge lies in the fact that collision avoidance maneuvers consume fuel and may cause deviations from the intended orbit, which could affect the satellite's ability to perform its mission. An optimal policy must balance survival probability with fuel consumption and service efficiency.

3 Collision Avoidance

3.1 Control Mechanism

To avoid potential collisions, the satellite controller can execute trajectory adjustments by activating its propulsion system. This allows the satellite to modify its velocity along the y-axis, denoted as V_y^t, at the cost of fuel consumption. The motion dynamics of the satellite are governed by:

$$V_y^{t+1} = V_y^t + u^t \Delta t, \tag{6}$$
$$S_x^{t+1} = S_x^t + V_x \Delta t, \tag{7}$$
$$S_y^{t+1} = S_y^t + \frac{V_y^t + V_y^{t+1}}{2} \Delta t + \epsilon, \tag{8}$$

where:

- u^t represents the instantaneous acceleration along the y-axis,
- Δt is the discrete time resolution,
- ϵ is a Gaussian random error:

$$\epsilon \sim \mathcal{N}(0, \beta^2 |V_y^t + V_y^{t+1}|^2). \tag{9}$$

The propulsion system allows discrete thrust levels $\{1, 3, 5\}$ in either direction, resulting in 7 possible control actions at each timestep.

3.2 Objective

To maximize the satellite service and the fuel efficiency while avoid the collision, the controller must carefully decide the actions for each time step. Specifically, there are following operational conditions:

- The satellite may temporarily drift away from the planned orbit O^t thus hamper the service. Consider the standard profit as p_0 per Δt. The relationship between current profit and satellite position is defined as

$$p^t = p_0 \left(1 - \frac{|S_y^t|}{S_{max}}\right) \tag{10}$$

- For each mission, the initial fuel is f^0. The relationship between acceleration u^t and fuel consumption is $\Delta f = 0.1|u^t|\Delta t$. The mission must end once fuel depletion or collision happens.

4 Your Task

The goal of this project is to develop a Reinforcement Learning-based decision policy that enables the satellite to autonomously adjust its trajectory to maximize its survival probability while minimizing deviations from its intended mission orbit.

4.1 Task 0: Establishing the Environment

Define the RL problem using the standard MDP framework with:

- State space $\mathcal{S}$,
- Action space $\mathcal{A}$,
- Transition dynamics $P(s, a, s')$,
- Reward function $R(s, a, s')$.

Your first task is to create an environment that:

- Initializes with a given satellite position, velocity, and an initial debris distribution.
- Properly represents the state at each timestep.
- Implements the transition function to evolve the system from state s^t to s^{t+1} given action a^t.
- Defines clear termination conditions: fuel depletion, collision, or reaching the maximum timestep.
- Ensure the termination condition reflect the reality: termination by collision is worse compared to fuel depletion and maximum time step.

4.2 Task 1: Learning a Discretized Policy

Once you have defined the environment, the next task is to initialize a policy π for the agent and improve it by learning how to control in such an environment.

For simplicity, in this task, you are required to construct a discretized policy. This means the policy:

- Takes a state s from a discrete state space $\mathcal{S}$ as input.
- Outputs an action a from a discrete action space $\mathcal{A}$.

The specific requirements for this task are:

- Specify how you discretize the state space.
- Define the policy and describe how it will be improved.
- Once the policy is well-defined and learned, evaluate its performance by testing it in multiple scenarios with different initial states.

4.3 Task 2: Learning a Semi-continuous Policy

In contrast to the previous task, you are now required to construct a semi-continuous policy π, meaning it can handle continuous state representations while still choosing actions from a discrete action space $\mathcal{A}$. Specifically, you must:

- Define how the policy selects an action a^t for a given state s^t.
- Specify how the policy is improved during learning.
- Evaluate the trained policy on different initial conditions.

5 Data and Hints

5.1 Debris Observation

In real-world scenarios, detecting and tracking space debris is a complex and computationally demanding task. To make the problem more manageable within an RL framework, we introduce the following simplifications for debris generation and observation.

1. Initialization: At the start of the simulation, four debris clusters are uniformly distributed within the region $x \in [1, 5]$, $y \in [-2, 2]$. Each debris cluster is assigned an initial position:

$$D_n^0 = (D_{n,x}^0, D_{n,y}^0). \tag{11}$$

2. Velocity Calculation: Each debris cluster is assigned a random collision time t_n, drawn from a uniform distribution over $[50\Delta t, 100\Delta t]$. The velocity of each debris cluster is computed using:

$$S_x^0 + V_x t_n = D_{n,x}^0 + W_{n,x} t_n, \tag{12}$$

$$S_y^0 + V_y^0 t_n = D_{n,y}^0 + W_{n,y} t_n. \tag{13}$$

   This ensures that if no evasive action is taken, the nth debris cluster will collide with the satellite exactly at time t_n.

3. Emergence of New Debris: Additional debris clusters may appear over time. At each time step, the probability of detecting a new debris cluster at the detection boundary ($x = S_x^t + 5$) increases. Specifically:
   - At $t = \Delta t$, the probability of a new debris cluster emerging is 0.01.
   - This probability linearly increases over time, reaching 1 at $t = 100\Delta t$.
   - Once a new debris cluster appears, the probability resets to 0, and the process repeats.

The y-coordinate of a newly emerging debris cluster is uniformly sampled from the range $[-2, 2]$, and its velocity is computed using the same method as in step 2 by randomly sampling a collision time t_n from $[50\Delta t, 100\Delta t]$. After time t_n, the x and y coordinates of the satellite and the new debris will be identical if satellite keeps the velocity V^t afterwards.

4. Managing Dynamic Debris Information: Since the number of debris clusters changes dynamically, tracking all debris in real-time may be impractical. A possible approach is to consider only the top-K most dangerous debris clusters for decision-making. From an RL perspective, this means that the agent does not need to use all debris information as input. Instead, students are encouraged to experiment with different values of K to assess how filtering debris observations impacts policy performance.

5.2 Data Description

The following parameters define the environment and satellite dynamics in the simulation:

- Satellite velocity: $V_x = 0.5$.
- Mission duration: Each mission ends at $t = 200\Delta t$.
- Fixed constants:
  - Uncertainty scaling factor: $\beta = 0.01$.
  - Time step resolution: $\Delta t = 0.1$.
  - Maximum allowable orbit deviation: $S_{\max} = 2$.
  - Standard operational profit: $p_0 = 5$.
  - Initial fuel supply: $f^0 = 5$.

5.3 Collision Probability Model

The probability of collision with a specific debris cluster n at any given time is given by:

$$D'_n = \|S - D_n\|_2, \tag{14}$$

$$\mathbb{P}_n = \begin{cases} 0.005, & \text{if } D'_n \leq 0.1, \\ 0.005 \exp\left(-\frac{\log 1000}{4.9}(D'_n - 0.1)\right), & \text{otherwise.} \end{cases} \tag{15}$$

where D'_n represents the Euclidean distance between the satellite and debris cluster n.