REINFORCEMENT LEARNING FOR CHAOTIC CONTROL: STABILIZING THE THREE-BODY PROBLEM IN STELLAR DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

The Three-Body Problem, a classic problem in celestial mechanics, is renowned for its inherent chaos and unpredictability. It describes the complex motion of three bodies interacting through gravitational force, where small changes in initial conditions can lead to drastically different outcomes, making long-term predictions challenging. This paper introduces a novel approach that leverages Reinforcement Learning (RL) to control the chaotic behavior of the Three-Body Problem. RL has been successfully applied to numerous control problems due to its ability to learn optimal policies in complex environments. In this study, we employ the Deep Deterministic Policy Gradient (DDPG) algorithm, a powerful RL method, to stabilize the motion of three bodies on an unstable periodic orbit in the presence of noise. Our results demonstrate the efficacy of the proposed RL-based control method in maintaining the stability of the Three-Body System under perturbations, providing a new perspective on the potential of machine learning techniques for tackling chaotic control problems in celestial mechanics.

1 INTRODUCTION

As an avid reader of science fiction literature, I was captivated by Liu Cixin's acclaimed novel, ”The Three-Body Problem”. In the novel, the Trisolarans, a civilization living in a three-star system, struggle with unpredictable and extreme climate changes due to the chaotic motion of their three suns. The inability to predict the motion of the three celestial bodies is not only a crucial plot device in the novel, but also a reflection of a longstanding scientific conundrum known as the Three-Body Problem in celestial mechanics. This novel reignited my interest in this complex problem and inspired me to explore novel approaches to predict and control the motion of the three bodies in a star system.

The Three-Body Problem, first posed by Isaac Newton, refers to the difficulty in predicting the motion of three bodies interacting through gravitational forces alone. While two-body problems can be solved analytically, the three-body problem is inherently chaotic; small changes in initial conditions can lead to drastically different outcomes, making long-term predictions challenging. This problem is important not only in astrophysics but also in other fields, such as atomic and molecular physics. Despite its chaotic nature, certain stable and periodic solutions, known as Lagrange and Euler points, have been discovered, which kindle the hope of finding control methods for this chaotic system.

To understand the motion of three bodies, numerical simulations are often employed. These methods, including Runge-Kutta and Euler's method, allow us to approximate the solutions of the differential equations governing the system's behavior. However, these methods do not necessarily provide insight into how we might control or stabilize the system's chaotic dynamics.

Enter Reinforcement Learning (RL), a subfield of artificial intelligence that deals with how an agent should take actions in an environment to maximize some notion of cumulative reward. In RL, a problem is framed as a Markov Decision Process (MDP), where an agent interacts with an environment by taking actions based on its current state and receives rewards or penalties. RL has shown great promise in a variety of control problems.

However, traditional RL methods struggle with continuous state and action spaces, which are common in many real-world applications, including our Three-Body Problem. This has prompted the development of methods such as the Deep Deterministic Policy Gradient (DDPG) algorithm. DDPG, an Actor-Critic method, has been particularly successful in continuous control tasks. It uses two neural networks: an actor that decides the next action and a critic that predicts the future reward. DDPG and its successor methods have been applied in a range of fields, from robotics to power systems, demonstrating their potential for controlling complex dynamics.

In this paper, we apply DDPG to the Three-Body Problem. We employ DDPG to learn a control policy that can stabilize the motion of the three bodies on a periodic orbit, even in the presence of noise. By doing so, we demonstrate how reinforcement learning can be used to control the chaotic dynamics of the Three-Body Problem, potentially offering new insights into the control of other complex systems. In this quest, I hope to extend the frontier of our understanding and control over chaotic systems, inspired by a science fiction novel that brought to light the profound mysteries inherent in the cosmos.

2 General formatting instructions

2.1 Three-Body Problem

The Three-Body Problem involves the motion of three bodies interacting through gravitational forces. In two dimensions (i.e., assuming the bodies move in the same plane), the problem can be described by the following system of ordinary differential equations:

$$\frac{d^2x_i}{dt^2} = G\sum_{j\neq i} m_j \frac{x_j - x_i}{r_{ij}^3} \tag{1}$$

$$\frac{d^2y_i}{dt^2} = G\sum_{j\neq i} m_j \frac{y_j - y_i}{r_{ij}^3} \tag{2}$$

where $i, j \in \{1, 2, 3\}$, $r_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$ is the distance between bodies i and j, and G is the gravitational constant.

By introducing the velocities $v_{x_i} = \frac{dx_i}{dt}$ and $v_{y_i} = \frac{dy_i}{dt}$ as intermediate variables, we can rewrite the system as a first-order system:

$$\frac{dx_i}{dt} = v_{x_i} \tag{3}$$

$$\frac{dy_i}{dt} = v_{y_i} \tag{4}$$

$$\frac{dv_{x_i}}{dt} = G\sum_{j\neq i} m_j \frac{x_j - x_i}{r_{ij}^3} \tag{5}$$

$$\frac{dv_{y_i}}{dt} = G\sum_{j\neq i} m_j \frac{y_j - y_i}{r_{ij}^3} \tag{6}$$

In this paper, we focus on a particular unstable periodic orbit in the case where the three bodies have equal mass and start as an equilateral triangle. If the initial velocities are chosen such that the centripetal force equals the gravitational force, the bodies will follow a circular orbit for a period of time before eventually deviating due to the system's inherent instability. This unstable periodic orbit is of particular interest as it represents a delicate balance in the system that we aim to stabilize using reinforcement learning.

If we consider a circular orbit in the Three-Body Problem, where each body moves in a circle with constant speed, the forces on each body will be in equilibrium. In such a case, the gravitational force of the other two bodies is equal to the centripetal force required to keep the body in circular motion.

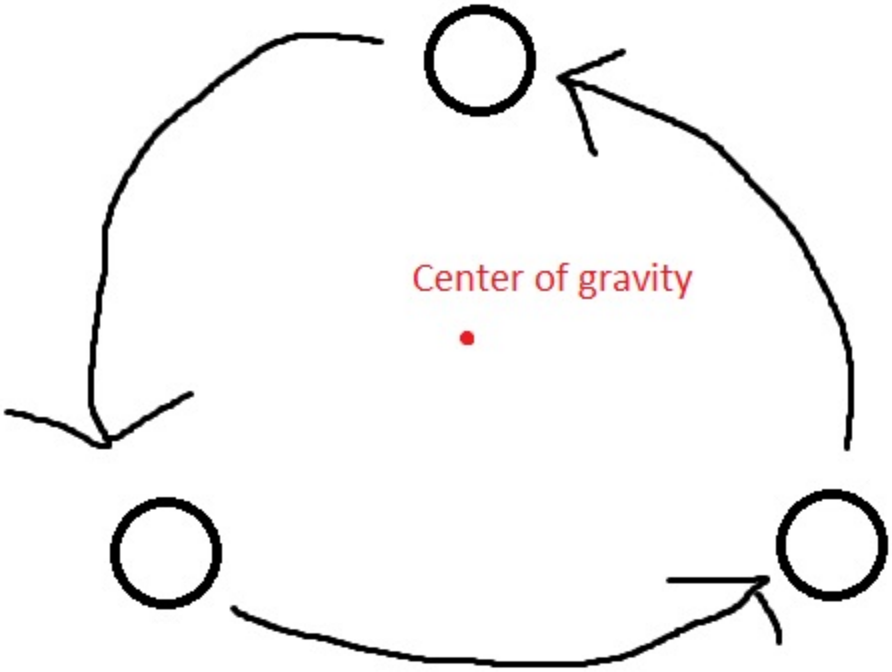


Figure 1: A sketch of the unstable periodic orbit.

We can write this equilibrium condition as:

$$F_{\text{gravity}} = F_{\text{centripetal}} \tag{7}$$

The gravitational force exerted on one body (let's call it body 1) by the other two bodies (body 2 and 3) is given by:

$$F_{\text{gravity}} = G\frac{m_1 m_2}{r_{12}^2} + G\frac{m_1 m_3}{r_{13}^2} = 2G\frac{m_1^2}{r^2} \tag{8}$$

Here, $m_1 = m_2 = m_3 = m$ due to equal masses, and $r_{12} = r_{13} = r$ is the distance between the bodies, which is equal to the radius of the circular orbit.

The centripetal force required to keep body 1 in a circular motion with speed v is given by:

$$F_{\text{centripetal}} = m_1 v^2/r = mv^2/r \tag{9}$$

Setting F_{gravity} equal to $F_{\text{centripetal}}$, we get:

$$2G\frac{m^2}{r^2} = mv^2/r \tag{10}$$

Solving for v, we find the speed that the bodies must have in order to maintain this unstable circular orbit:

$$v = \sqrt{2Gm/r} \tag{11}$$

2.2 Euler's Method

Euler's method is a simple numerical method for solving ordinary differential equations (ODEs) with a given initial condition. The general form of the forward Euler's method can be described as:

$$y_{n+1} = y_n + hf(t_n, y_n) \tag{12}$$

Here, y_n is the approximation of the function $y(t)$ at time t_n, h is the step size, and $f(t_n, y_n)$ is the derivative of the function $y(t)$ with respect to time t at time t_n.

For the Three-Body Problem, we can use the forward Euler's method to approximate the positions and velocities of the stars at each time step. We update the positions (x, y) and velocities (v_x, v_y) using the equations:

$$x_{n+1} = x_n + hv_{x_n} \tag{13}$$
$$y_{n+1} = y_n + hv_{y_n} \tag{14}$$
$$v_{x_{n+1}} = v_{x_n} + ha_{x_n} \tag{15}$$
$$v_{y_{n+1}} = v_{y_n} + ha_{y_n} \tag{16}$$

Here, a_{x_n} and a_{y_n} are the accelerations due to the gravitational forces acting on the stars at time t_n.

There are higher-order methods, such as the Runge-Kutta method, that can provide more accurate solutions to ODEs. However, these methods are more complex and computationally intensive, so we use the forward Euler's method for its simplicity and ease of implementation in this study.

2.3 Reinforcement Learning and Markov Decision Processes

Reinforcement Learning (RL) is a branch of machine learning that is concerned with how an agent should take actions in an environment in order to maximize a notion of cumulative reward. At the heart of RL is the concept of a Markov Decision Process (MDP).

A MDP is a mathematical model used to describe a decision-making problem in terms of states, actions, rewards, and transition probabilities. Formally, a MDP is defined by a tuple (S, A, P, R, γ), where:

- S is the state space, a set of all possible states that the agent can be in.
- A is the action space, a set of all possible actions that the agent can take.
- $P : S \times A \times S \rightarrow [0, 1]$ is the transition probability function. $P(s'|s, a)$ represents the probability of transitioning to state s' given that the agent is in state s and takes action a.
- $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function. $R(s, a, s')$ represents the immediate reward received after transitioning to state s' from state s due to action a.
- $\gamma \in [0, 1]$ is the discount factor, which determines the present value of future rewards.

In an MDP, the agent selects actions according to a policy $\pi : S \rightarrow A$, which is a mapping from states to actions. The goal of RL is to find the optimal policy π^* that maximizes the expected cumulative reward, defined as $E[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})]$, where s_t is the state, a_t is the action taken by the agent at time t, and s_{t+1} is the state at the next time step.

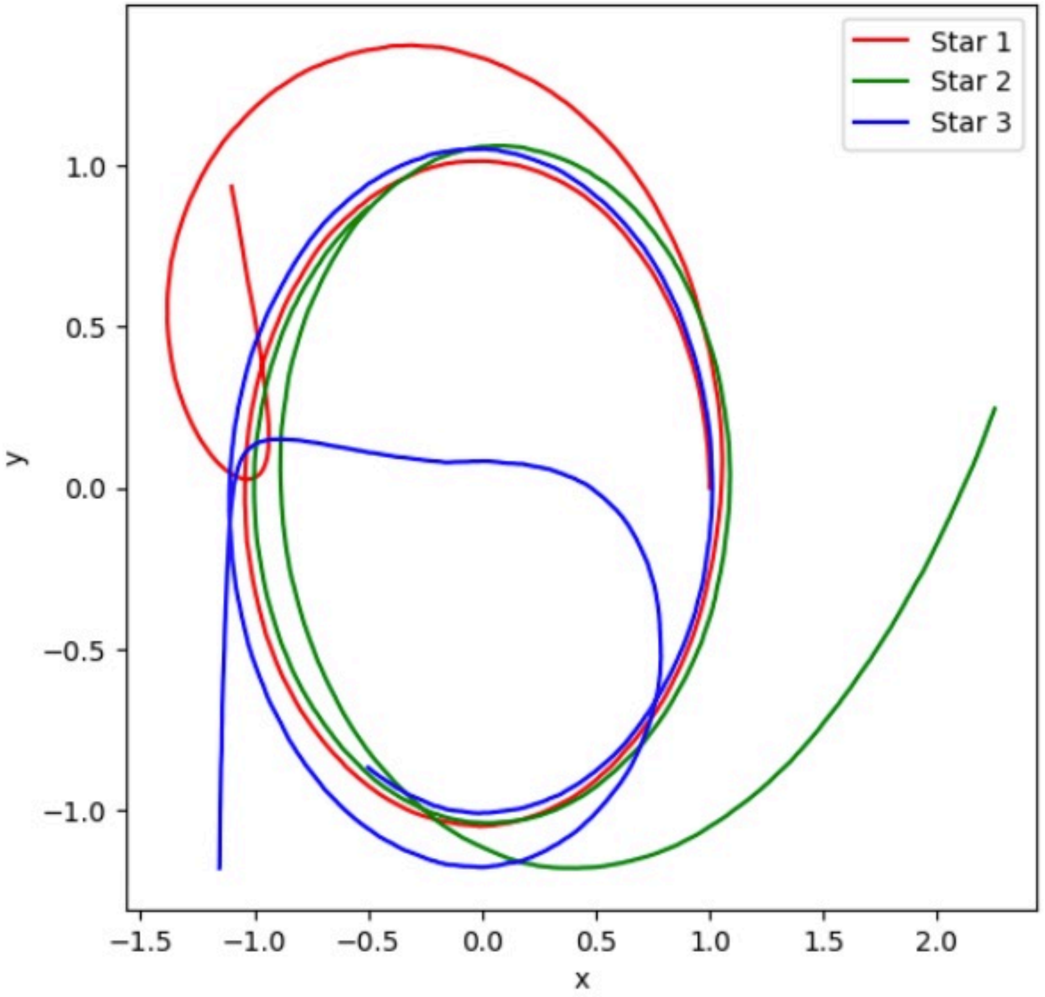


Figure 2: A sample run without control. The stars fly away as the injected noise gets magnified.

2.4 Reinforcement Learning with Continuous State and Action Spaces

Reinforcement Learning (RL) with continuous state and action spaces is of great significance in many domains, particularly in robotics and dynamic system control, where actions and states are often real-valued quantities. However, traditional RL methods like Q-Learning and SARSA, which are based on the tabular approach for value function approximation, are not directly applicable to these problems. The reason is that in a continuous space, it's practically impossible to construct a table for every possible state-action pair. Thus, we need function approximation methods to represent the value function in these cases.

Deep Deterministic Policy Gradient (DDPG)Lillicrap et al. (2019) is an actor-critic algorithm that extends the traditional actor-critic methods to deal with continuous action spaces. The actor and the critic are both represented by neural networks, which are used as function approximators.

The actor network maps states to actions, and the critic network estimates the Q-value function given a state-action pair. During training, the critic network learns the Q-value function using the Bellman equation as in Q-learning, and the actor network is updated based on the policy gradient derived from the learned Q-value function.

Here is the pseudocode for DDPG:

Algorithm 1 Deep Deterministic Policy Gradient (DDPG)

```
Require: Initial actor network parameters $\theta$, critic network parameters $\phi$, target network parameters
    $\theta^-$, $\phi^-$, replay buffer $\mathcal{D}$, batch size $B$, discount factor $\gamma$
Ensure: Trained actor network
  for each episode do
      Initialize state $s$
      while not done do
          Select action $a = \text{actor}(s) + \mathcal{N}$ with exploration noise $\mathcal{N}$
          Execute action $a$, observe reward $r$ and next state $s'$
          Store transition $(s, a, r, s')$ in replay buffer $\mathcal{D}$
          Sample a random minibatch of $B$ transitions from $\mathcal{D}$
          Compute target $y = r + \gamma\text{critic}(s', \text{actor}(s'))$ using target network parameters $\theta^-$, $\phi^-$
          Update critic by minimizing the loss: $L = \frac{1}{B}\sum(y - \text{critic}(s, a))^2$
          Update the actor policy using the sampled policy gradient:
          $\nabla_\theta J \approx \frac{1}{B}\sum \nabla_a \text{critic}(s, a)|_{a=\text{actor}(s)} \nabla_\theta \text{actor}(s)$
          Update the target networks:
          $\phi^- \leftarrow \tau\phi + (1 - \tau)\phi^-$
          $\theta^- \leftarrow \tau\theta + (1 - \tau)\theta^-$
          $s \leftarrow s'$
      end while
  end for
```

This algorithm has been widely used for various tasks in robotics and dynamics and is known for its efficiency and effectiveness in dealing with continuous state and action spaces.

3 Experiment

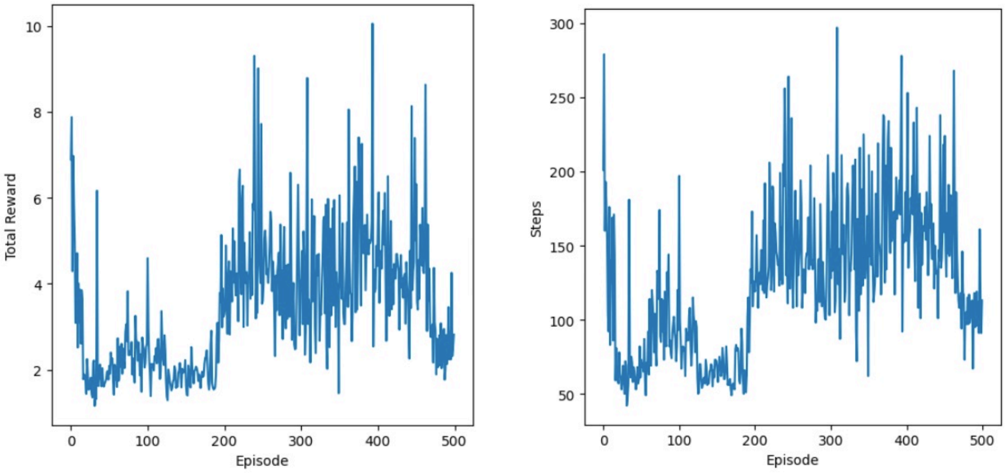
The experiment was conducted using a model of the three-body problem as the environment and an agent based on the Deep Deterministic Policy Gradient (DDPG) algorithm. The state of the environment includes the positions, velocities, and accelerations of the three bodies in the x and y directions, resulting in a state dimension of 18. The agent's action is to perturb the positions of the three bodies in the x and y directions, resulting in an action dimension of 6. The maximum allowable perturbation is 10^{-3}.

The agent was trained over 500 episodes, each consisting of 1000 time steps. In each time step, the agent selected an action based on its current policy, applied the action to the environment, and stored the resulting state, action, reward, and next state in a replay buffer. The agent's policy was then updated based on a batch of transitions sampled from the replay buffer. The total reward for each episode was calculated as the sum of the rewards over the time steps.

One important detail to note is that the time step used in the DDPG algorithm is not the same as the time step used in the numerical simulation of the three-body problem. Each DDPG time step actually corresponds to 10 time steps of the numerical simulation, meaning that the control is applied every 0.03 time unit, even though the simulation is performed with a discretization of 0.003 time units. This approach was chosen to reduce the complexity of the problem.

The reward function was designed to encourage the agent to maintain the three bodies in a stable periodic orbit. Specifically, the reward at each time step was calculated as $0.05 - \frac{1}{3}\sum|1 - r|$, where r is the radius of the bodies from the circular orbit. If the deviation exceeded 0.05, the episode was terminated.

To take a step in the environment, the agent's action was first added to the positions of the bodies. Then, the positions, velocities, and accelerations of the bodies were updated using 10 steps of the forward Euler method. If, at any point during these steps, the average deviation of the bodies from the circular orbit exceeded 0.05, the episode was terminated. Otherwise, the new state, the reward, and the done flag were returned.



(a) Rewards

(b) Steps persisted

Figure 3: Numerical results using DDPG to train for 500 episodes

4 Results and Discussion

5 Analysis

The results of the experiment demonstrate the challenges and complexities associated with controlling the chaotic three-body problem using reinforcement learning.

In the initial stages of the experiment, the agent was able to maintain the bodies in a stable periodic orbit for approximately 50 steps before they deviated significantly from the circular path. As the agent continued to learn over subsequent episodes, this figure gradually increased to around 150 steps. However, there was a high level of variability in the number of steps maintained across different episodes, reflecting the inherent instability and unpredictability of the three-body problem.

To further illustrate this point, we provide two graphical representations of the trajectories of the bodies during a single run of the simulation, one with the application of control and one without. Despite the use of a reinforcement learning algorithm designed to maximize the time the bodies remained in a stable orbit, there was no significant improvement in the trajectories when control was applied compared to when it was not.

These results underscore the difficulty of controlling the three-body problem, even when using sophisticated reinforcement learning algorithms. They also highlight the limitations of current approaches to reinforcement learning in the face of highly complex, chaotic systems. However, the fact that the agent was able to achieve any level of control over the system, albeit limited and inconsistent, is notable. It suggests that with further refinement and optimization, reinforcement learning could potentially be a viable approach to controlling the three-body problem and other similar chaotic systems.

This work also serves as a stepping stone for further research in this area. For example, future work could explore the use of different reinforcement learning algorithms, alternative reward structures, or more advanced numerical integration methods for simulating the three-body problem. It could also investigate ways to reduce the variability in the control performance across episodes, such as by incorporating a mechanism for the agent to learn from its past mistakes.

6 Future Directions

This research has laid the groundwork for a variety of future explorations in applying reinforcement learning to complex dynamical systems. There are several promising directions to extend this work:

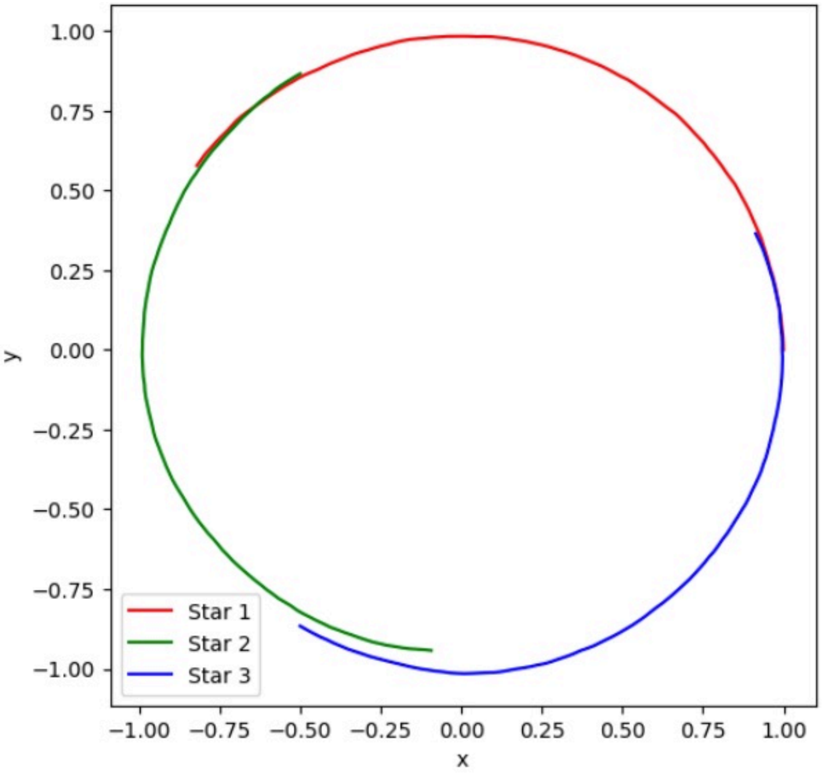


Figure 4: A demo run of the controller after training the controller. The process is terminated after deviating from the circular orbit.

1. **Exploration of Other Periodic Orbits:** While this study focused on a specific unstable periodic orbit of the three-body problem, there exist numerous other types of orbits, each with their own unique dynamics. Future work could explore the application of reinforcement learning to stabilize these different orbits, broadening the scope of this approach.

2. **Driving Systems to Desired States:** Instead of starting with a system already in an unstable orbit, it would be interesting to investigate whether a reinforcement learning agent could drive a system from arbitrary initial conditions to a specific orbit. This would represent a more general and challenging control task.

3. **Investigating Numerical Methods:** In this study, we used the Euler method for numerical simulation due to its simplicity. Future research could investigate the use of more advanced numerical methods. It would be insightful to see whether the control policy learned by the reinforcement learning agent transfers to these methods, or whether it is simply exploiting the specific errors introduced by the Euler method.

4. **Varying Mass Distributions:** We have considered the case where all three bodies have equal masses. However, in a more general scenario, the bodies could have different masses, which would significantly affect the dynamics of the system. Exploring the ability of reinforcement learning methods to handle such heterogeneity could be a fruitful direction for future research.

By following these directions, we can further expand our understanding of how reinforcement learning can be applied to control complex dynamical systems and manage chaos, opening up new possibilities in a range of fields from astrophysics to robotics.

7 Conclusion

This paper has demonstrated a novel application of reinforcement learning (RL) to control the chaotic dynamics inherent to the Three-Body Problem in celestial mechanics. We applied the Deep Deterministic Policy Gradient (DDPG) algorithm, a powerful actor-critic RL method, to stabilize an unstable periodic orbit of three bodies under perturbations. Our results show the potential of RL methods to not only learn but also maintain stability in complex and unpredictable dynamical systems, such as the Three-Body Problem. This study offers a unique contribution to the intersection of machine learning and celestial mechanics and highlights the significant potential of these techniques in controlling chaotic systems. We hope this work serves as a foundation for future research exploring the broader application of RL in dynamics and chaos control.

References

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.