Robust Reinforcement Learning Differential Game Guidance in Low-Thrust, Multi-Body Dynamical Environments

A Zero-Sum Reinforcement Learning Approach in Three-Body Dynamics

Ali Baniasad
Supervisor: Dr. Nobahari

Department of Aerospace Engineering
Sharif University of Technology

Outline

Research Motivation

- **Space missions** increasingly require autonomous guidance systems
- **Low-thrust spacecraft** operate in complex gravitational environments
- **Three-body dynamics** (Earth-Moon CRTBP) present inherent instabilities
- **Classical control methods** struggle with:
  - Model uncertainties
  - Environmental disturbances
  - Fuel efficiency requirements
- **Need for robust, adaptive guidance** without precise dynamic models

Central Question

How can we achieve robust spacecraft guidance in uncertain environments?

Problem Statement

Research Objective

Design a robust guidance framework for low-thrust spacecraft operating in Earth-Moon three-body dynamics under uncertainties.

System Characteristics:

- State: $\mathbf{x} = [x, y, \dot{x}, \dot{y}]^T$
- Control: $\mathbf{u} \leq u_{\max}$
- Dynamics: $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$

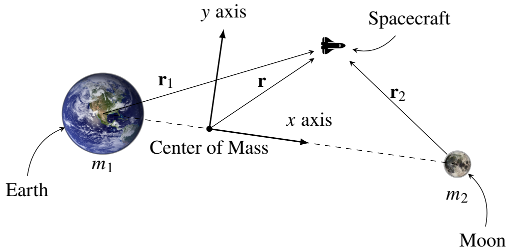
Mission Environment:

- Earth-Moon CRTBP
- Lyapunov orbit transfer
- Low-thrust propulsion

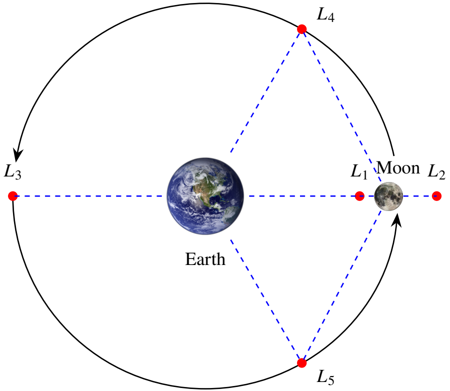
Mathematical Formulation

Optimal control problem with state-dependent uncertainties and adversarial disturbances

Planar CRTBP Model



(a) CRTBP Configuration



(b) Lagrangian points in the Earth-Moon system

Figure: CRTBP Model and Lagrangian Points

Reinforcement Learning Overview

- **Definition:** A type of machine learning where an agent learns to make decisions by taking actions in an environment to maximize cumulative reward.

- **Key Components:**
  - **Agent:** The learner or decision maker.
  - **Environment:** The external system with which the agent interacts.
  - **Actions:** Choices made by the agent to influence the environment.
  - **Rewards:** Feedback from the environment based on the agent's actions.

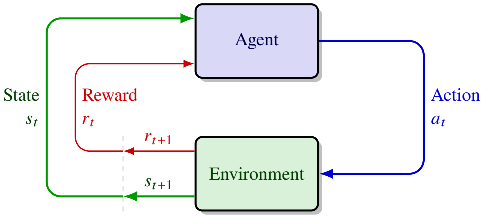


Figure: Agent-Environment Interaction Loop

State and Observations

- **State (s):** Complete description of the environment's condition
- **Observation (o):** Partial description of the state
  - May not contain all information
  - In fully observable environments: $s = o$
- **Action Space (a):** Set of all possible actions an agent can take
  - Can be discrete (finite set) or continuous (bounded range)

Policy

- **Policy:** Rules that an agent uses to decide which actions to take

- **Types:**
  - **Deterministic:** $a_t = \mu(s_t)$
  - **Stochastic:** $a_t \sim \pi(\cdot|s_t)$
- **Parameterized Policy:** Output is a function of policy parameters (neural network weights)
  - $a_t = \mu_\theta(s_t)$ or $a_t \sim \pi_\theta(\cdot|s_t)$
  - Parameters θ are optimized during learning

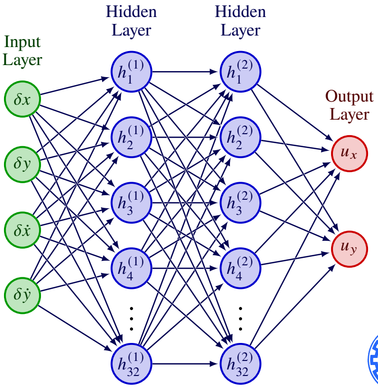


Figure: Policy Neural Network Structure

Trajectory and Reward

Trajectory:

- Sequence of states and actions: $\tau = (s_0, a_0, s_1, a_1, \ldots)$
- State transition: $s_{t+1} = f(s_t, a_t)$

Reward:

- $r_t = R(s_t, a_t, s_{t+1})$ or $r_t = R(s_t, a_t)$
- **Return:** Total accumulated reward
- Finite horizon: $R(\tau) = \sum_{t=0}^{T} r_t$
- Discounted: $R(\tau) = \sum_{t=0}^{\infty} \gamma^t r_t$

Value and Action-Value Functions

- **Value Function:** Expected return when following a policy

State Value Function:

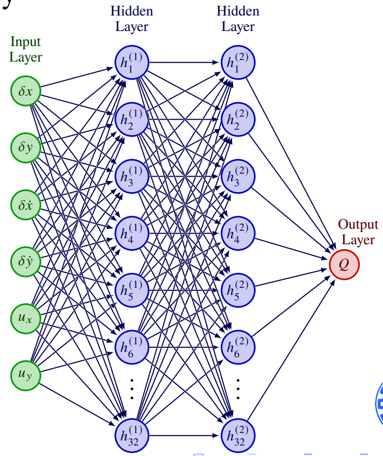
$$V^{\pi}(s) = \mathop{\mathbb{E}}_{\tau\sim\pi}[R(\tau)|s_0 = s]$$

Action-Value Function:

$$Q^{\pi}(s,a) = \mathop{\mathbb{E}}_{\tau\sim\pi}[R(\tau)|s_0 = s, a_0 = a]$$

Advantage Function:

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s)$$



Optimal Value Functions

Optimal State Value Function:

$$V^*(s) = \max_{\pi} V^{\pi}(s)$$

Optimal Action-Value Function:

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

Optimal Value Bellman Equation:

$$V^*(s) = \max_{a} \mathop{\mathrm{E}}_{s' \sim P} \left[r(s,a) + \gamma V^*(s') \right]$$

Optimal Q Bellman Equation:

$$Q^*(s,a) = r(s,a) + \gamma \mathop{\mathrm{E}}_{s' \sim P} \left[\max_{a'} Q^*(s',a') \right]$$

Key insight: The optimal policy π^* is greedy with respect to Q^*:

$$\pi^*(s) = \arg\max_{a} Q^*(s,a)$$

Bellman Equations

For Policy Value Functions:

$$V^\pi(s) = \mathop{\mathrm{E}}_{\substack{a\sim\pi \\ s'\sim P}}\left[r(s,a) + \gamma V^\pi(s')\right]$$

$$Q^\pi(s,a) = r(s,a) + \gamma \mathop{\mathrm{E}}_{s'\sim P}\left[\mathop{\mathrm{E}}_{a'\sim\pi}\left[Q^\pi(s',a')\right]\right]$$

For Optimal Value Functions:

$$V^*(s) = \max_a \mathop{\mathrm{E}}_{s'\sim P}\left[r(s,a) + \gamma V^*(s')\right]$$

$$Q^*(s,a) = r(s,a) + \gamma \mathop{\mathrm{E}}_{s'\sim P}\left[\max_{a'} Q^*(s',a')\right]$$