Reinforcement Learning and Optimal Control for Robotics ROB-GY 6323 Exercise Series 4

December 20, 2024

JITHIN GEORGE
Univ ID: N10719458
Net ID: jg7688

Reward Function Design

1. State and Action Variables

- `self.state`: Represents the current state of the quadrotor, likely including position, velocity, and possibly orientation.
- `action`: Represents the control input applied to the quadrotor, treated as the deviation from the hover thrust (`self.u_grav`).

2. State and Control Differences

- `u`: The control input applied to the system, which is the sum of the gravitational control thrust (`self.u_grav`) and the action (`action`).
- `state_diff`: The difference between the current state (`self.state`) and the desired state (`self.x_des`).
- `control_diff`: The difference between the control input (`u`) and the desired gravitational thrust (`self.u_grav`).

3. Reward Function Design

The reward function is based on the deviation of the state and control from the desired values. It includes penalties for large deviations and for unsafe behaviors such as collisions or going out of bounds.

State and Control Costs

The cost for the state and control deviations is calculated as follows:

$$\text{cost\_state} = \frac{1}{2}(\text{state\_diff}^\top Q \text{state\_diff})$$

$$\text{cost\_control} = \frac{1}{2}(\text{control\_diff}^\top R \text{control\_diff})$$

The total reward is calculated as:

$$\text{reward} = \exp\left(-(\text{cost\_state} + \text{cost\_control})\right)$$

This reward incentivizes the quadrotor to minimize its state and control deviations from the desired values.

Penalties and Termination Conditions

- **Collision Penalty**: If the quadrotor collides with an obstacle (`quadrotor.check_collision(self.state)`), the reward is reduced by 1. This penalizes unsafe behavior and encourages collision avoidance.
- **Out-of-Bounds Penalty**: If the quadrotor goes outside predefined bounds, a large penalty of -100 is applied, and the episode is marked as **terminated**.
- **Episode Length**: The episode terminates if the step count exceeds a predefined maximum (`self.max_steps`), which is set to 200. This limits the maximum duration of each episode.

Trajectory Plot

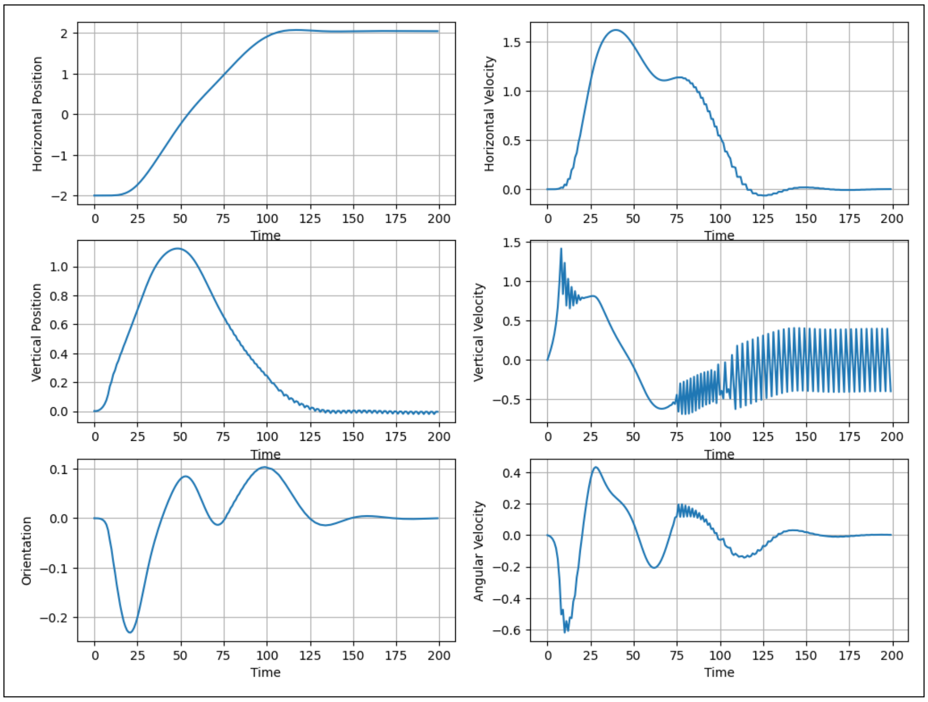

Figure 1: State Variance Over Time

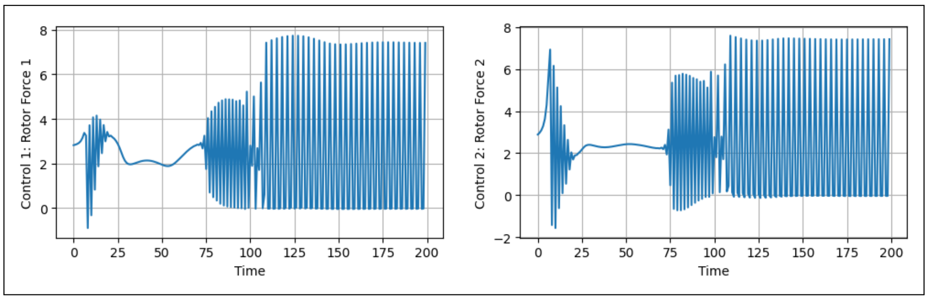

Figure 2: Control Variance Over Time