Actor-Critic Reinforcement Learning

Mark Doughten [MD1875]
Vijayendra Sai Chennareddy [VC545]
Shubham Patil [SP2484]

November 20, 2024

1 Introduction

The objective is training a robot on how to reach a specific goal area using reinforcement learning, specifically an Actor-Critic framework. The report shows how the robot was trained and the rewards associated with each action. The framework requires using two neural networks for policy learning and value estimation with a reward function designed to encourage goal seeking. The transition function outlines the expected behavior. As a guide, the transition function is defined as follows:

$$\begin{aligned}
\dot{x}_{t+1} &= \dot{x}_t + (f_{x,t} - \rho_{x,t})\Delta t, \\
\dot{y}_{t+1} &= \dot{y}_t + (f_{y,t} - \rho_{y,t})\Delta t, \\
x_{t+1} &= x_t + \dot{x}_t \Delta t, \\
y_{t+1} &= y_t + \dot{y}_t \Delta t.
\end{aligned}$$

where $\rho_{x,t}$ and $\rho_{y,t}$ are small independent noises, sampled from $\mathcal{N}(0, 0.1)$ at each time step t, and Δt is set to 0.1 seconds. The transition function contextualizes the inputs into the actor neural network and transfers easily over to MuJoCo since the position is recorded through the episodes and force can get applied.

2 Map

The environment is a course featuring obstacles and boundaries. The initial state is obtained through sampling the position of the robot uniformly in the workspace of the robot and setting the initial velocity to 0. The goal position is fixed on (0.8, 0.0) and the epsilon is 0.1. Overall, this is a challenging configuration since the robot needs to learn how to travel above or below the obstacle in the middle. An example of how the points are selected during training sessions, showing the initial starting positions are random.

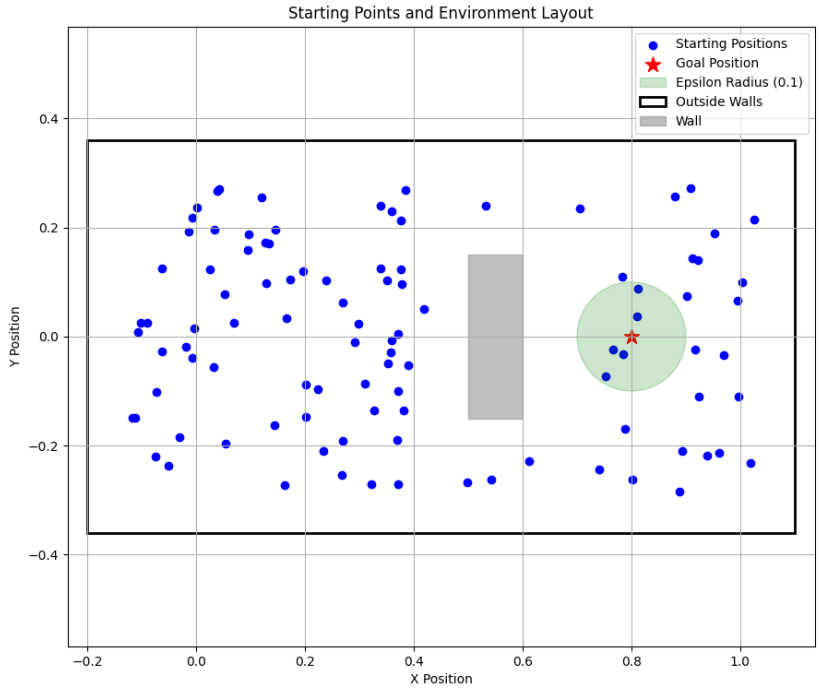

Figure 1: Spawn locations during the training process.

There are instances when the robot spawns on the goal and receives an immediate reward. An issue when the goal reward is high is the robot not exploring adequately since it was accustomed to reward nearby.

3 Actor-Critic Model

The Actor-Critic architecture comprises two neural networks:

- **Actor Network:** Outputs an action in the form of a mean (μ) and standard deviation (σ) for each step in the episode.
- **Critic Network:** Estimates the value function and provides feedback to improve the actor's policy.

3.1 Actor Network

The actor network processes the robot's state (x, y, v_x, v_y) through three fully connected layers. The output represents the mean and standard deviation for a Gaussian policy, a is the action vector consisting of control forces in the x- and y-directions.

$$\pi(a|s) \sim \mathcal{N}(\mu(s), \sigma(s)), \tag{1}$$

3.2 Critic Network

The critic network uses the same input as the actor and a single scalar value that represents the estimated state value $V(s)$. This value guides the actor policy by estimating the advantage of actions. The two networks work together to achieve the optimal actions based on the policy gradient.

3.3 Loss Functions

- **Actor Loss:** Encourages actions that maximize expected rewards:

$$L_{\text{actor}} = -\mathbb{E}[\log \pi(a|s) \cdot \text{Advantage}(s, a)].$$

- **Critic Loss:** Minimizes the temporal difference (TD) error:

$$L_{\text{critic}} = \frac{1}{2}(\text{TD} - V(s))^2.$$

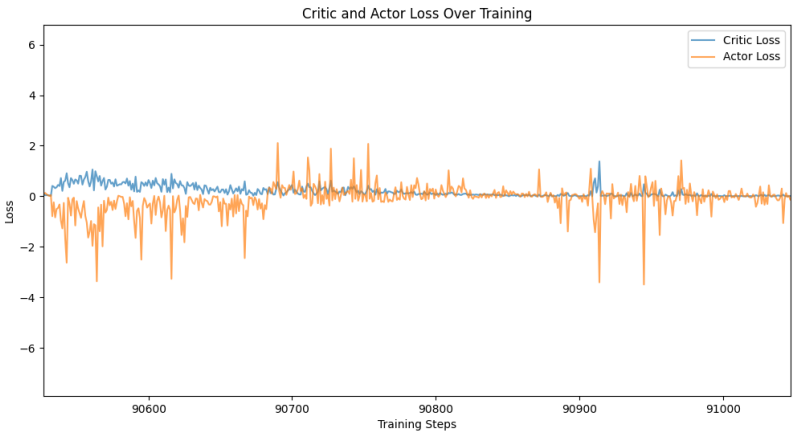

Figure 2: Recorded loss during a training session.

4 Training

4.1 Reward Function

The reward function incentives reaching the goal and penalizes wasting time. Since the obstacle add complexity, a simple reward function like distance to the goal not beneficial since the robot finds an optimal point on the closest side to not incur the cost to go around. The reward function balances three objectives:

- **Distance to Goal:** Encourages progress toward the goal with a linear penalty proportional to the distance.
- **Time Penalty:** Incentives finding the goal quicker during each episode.

4.2 Method

The Actor-Critic model in this experiment is based on Hado van Hasselt's lecture.

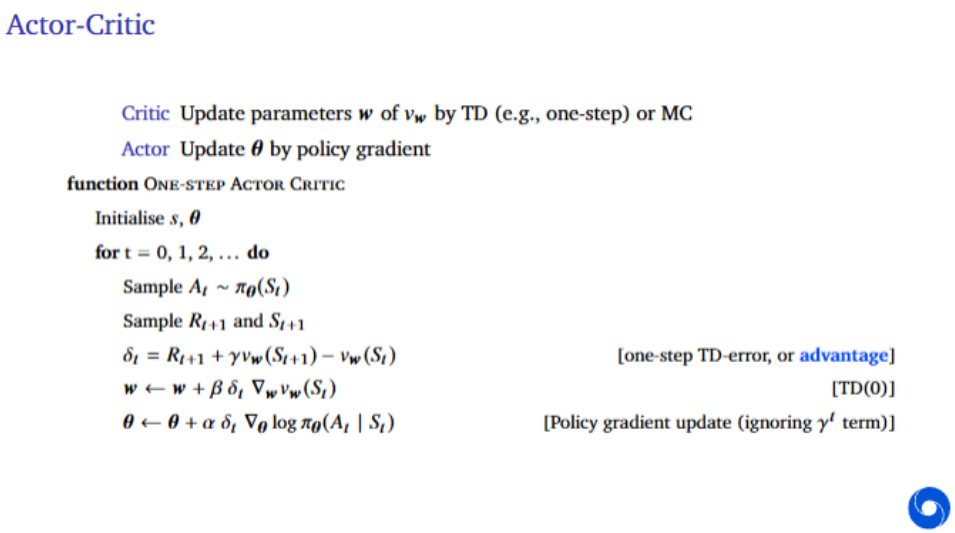

Figure 3: Pseudocode

The training process allows the user to set the duration and the number of steps. In our case, we typically set the number of steps to 1800 and episodes vary based on the session. While we can train in batch, a user can view a training session in between the batches to see the progression. Likewise, we have the option to run from a fixed point and The training loop iteratively performs the following steps:

1. Initialize the state s_0.
2. Use the actor network to sample an action a_t.
3. Apply a_t in the environment and record the next state s_{t+1} and reward r_t.
4. Compute the temporal difference (TD):

$$\text{TD} = r_t + \gamma V(s_{t+1}),$$

   where $\gamma = 0.99$ is the discount factor.

5. Backpropagation for the actor and critic networks using their respective loss functions.

5 Results

5.1 Learning Curve

The model achieves consistent improvements in reward over episodes. The recorded average reward per episode during training.

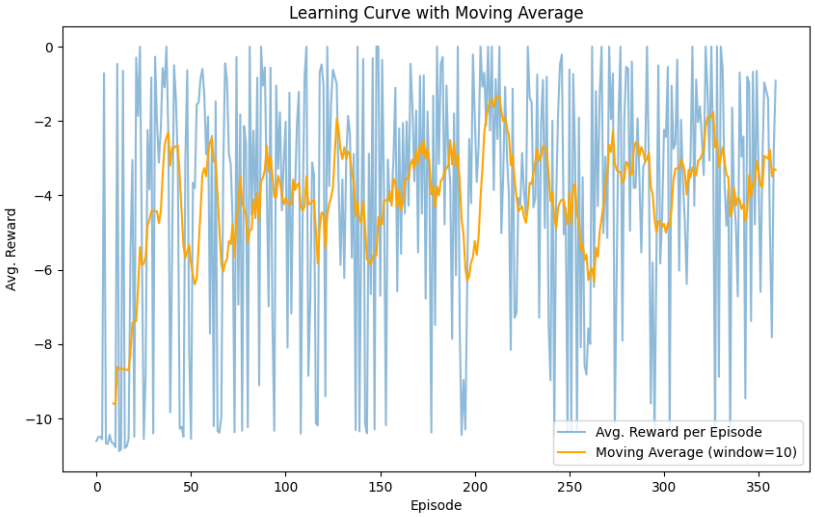

Figure 4: Average reward per episode during training.

6 Conclusion

The Actor-Critic reinforcement learning framework demonstrates effective navigation to the goal in a simulated environment. Future work involves extending this approach to submersibles and implementing sensors obstacle.

References

[1] van Hasselt, H. Lecture 9: Policy Gradients and Actor Critics; UCL, 2021.

[2] Jaramillo-Martínez, R.; Chavero-Navarrete, E.; Ibarra-Pérez, T. Reinforcement-Learning-Based Path Planning: A Reward Function Strategy. *Applied Sciences* 2024, 14 (17), 7654. `https://doi.org/10.3390/app14177654`.

[3] Larsen, T. N.; Teigen, H. Ø.; Laache, T.; Varagnolo, D.; Rasheed, A. Comparing Deep Reinforcement Learning Algorithms' Ability to Safely Navigate Challenging Waters. *Frontiers in Robotics and AI* 2021, 8, 738113. `https://doi.org/10.3389/frobt.2021.738113`.

[4] Manuelli, L.; Florence, P. Reinforcement Learning for Autonomous Driving Obstacle Avoidance using LIDAR. *Massachusetts Institute of Technology.* Available online: `https://www.peteflorence.com/ReinforcementLearningAutonomousDriving.pdf`.