Continuous and Mapless Navigation for robots using Policy Gradient Algorithm

Avnish Gupta, Ashwij Kumbla, Anamika Kumari and Shreedhar Kodate

Motivation

We developed a reinforcement learning based motion planner that takes 90-dimensional range finding and target position with respect to robot as input and generates continuous steering command as output.

Task Overview

- Implemented a mapless motion planner which takes current information of the environment using laser range finder and goal location as shown in.
- **Action Space:** The 2-D action of every time step includes angular and linear velocities.Angular velocity range(-0.5, 0.5) using tanh function. Forward Linear velocity range is constrained in(0,0.5) using sigmoid function and backward linear velocity is constrained in(-0.5,0.5) using tanh function.
- **State Space:** The state vector is abstracted from 90-Dimensional laser range findings,the previous action and relative target position and all merged as 94-dimensional vector. The laser range findings are sampled between 0 and 360 degrees in a trivial and fixed angle dis- tribution of 4 degrees. Also, range information is normalised between(0,1)
- **Network:** The actor network takes 94 dimensional state input followed by five fully connected layers and produces an embedding of size two as output.

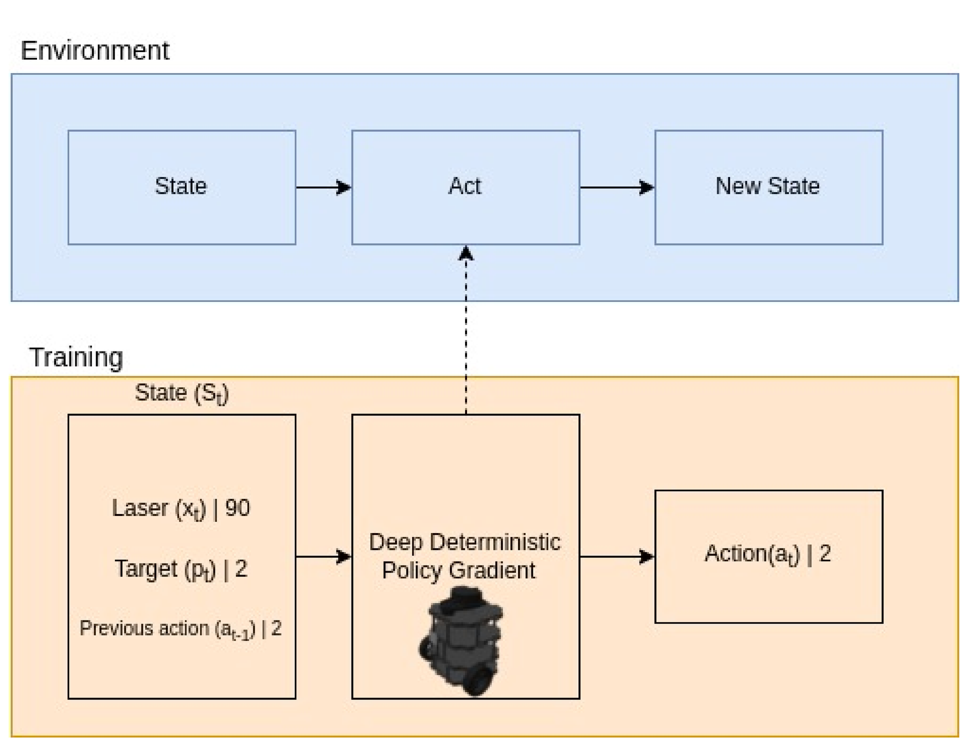

Figure 1:RL-Based mapless motion planner

Deep Deterministic Policy Gradient

- Deep Deterministic Policy Gradient (DDPG) is a model-free off-policy algorithm for learning continous actions.
- It combines ideas from DPG (Deterministic Policy Gradient) and DQN (Deep Q-Network).
- It uses Experience Replay and slow-learning target networks from DQN, and it is based on DPG, which can operate over continuous action spaces.

Important formulae:
Computing targets:

$$y(r, s', d) = r + \gamma(1-d)Q_{\phi_{target}}(s', \mu_{\theta_{target}}(s'))$$

Soft Update target networks:

$$\theta_{target} \leftarrow \rho\theta_{target} + (1-\rho)\theta$$

Environment

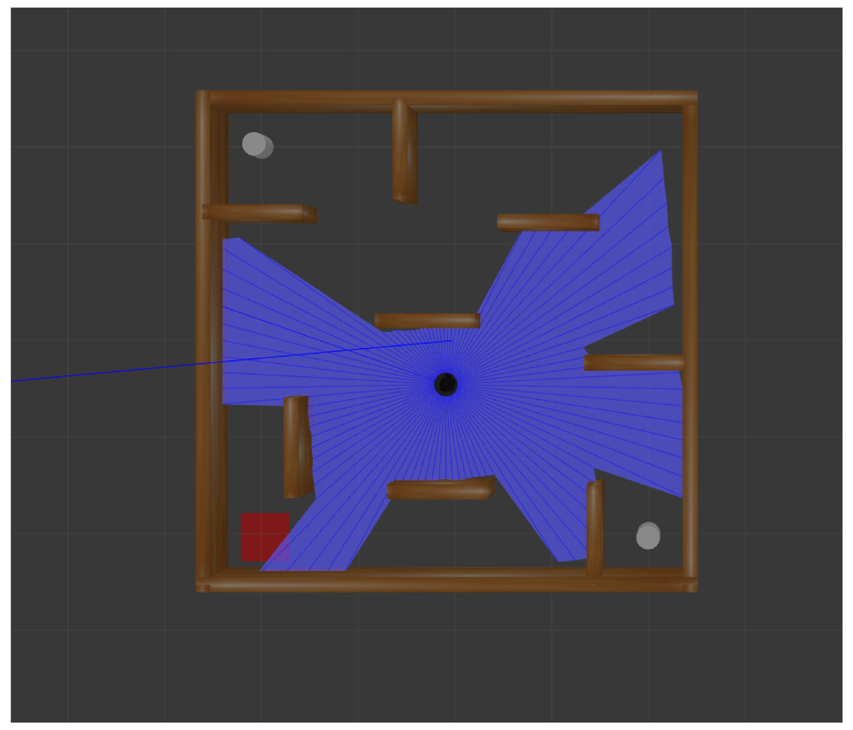

Figure 2:Environment Stage 4 - Complex with static obstacles

Training details

- Trained 2 actor models for forward (model_f)3 and forward-backward motion (model_fb).
- For model_f, the last layer has a sigmoid for producing positive linear velocity between 0 to 1 and a tanh for producing angular velocity between -1 to 1.
- For model_fb, tanh function is used for producing both linear and angular velocity outputs between -1 to 1.

Network Architecture

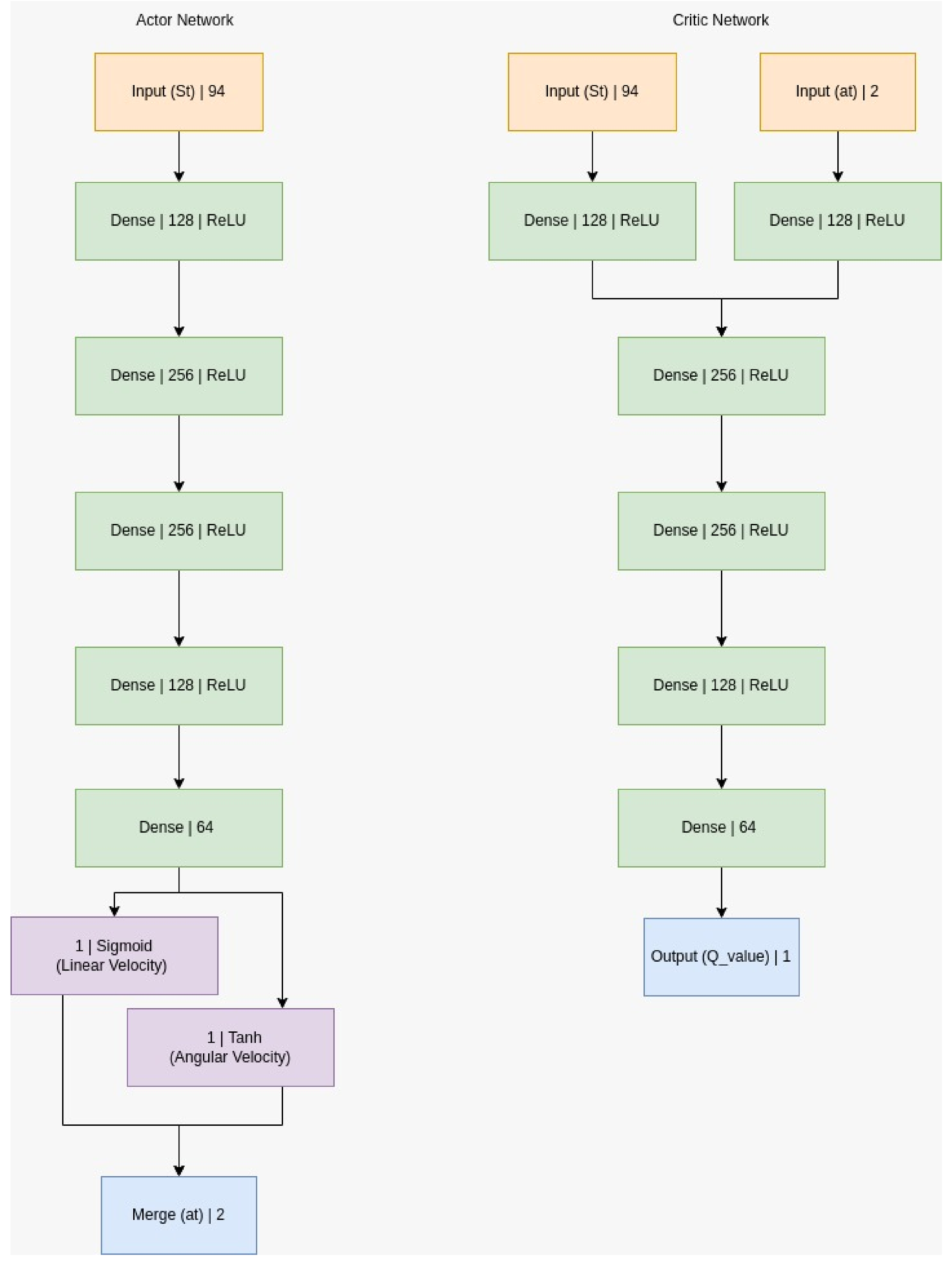

Figure 3:DDPG Actor Critic Forward motion.

Results

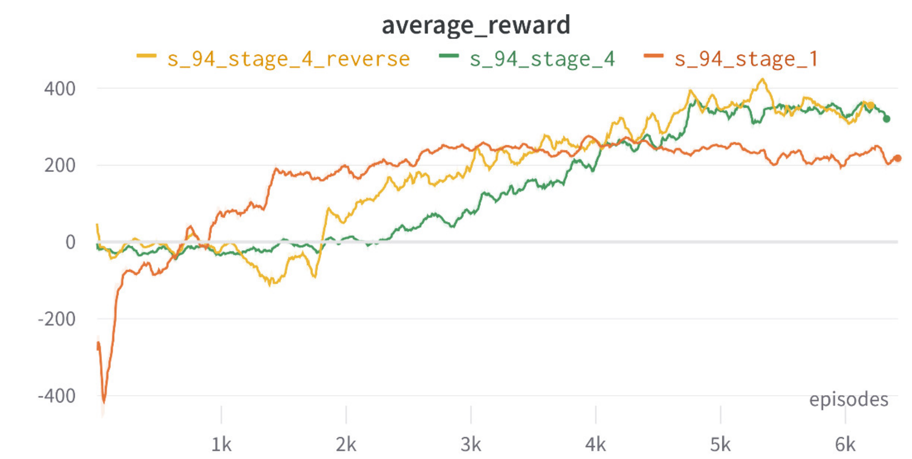

Figure 4:Average Reward while training.

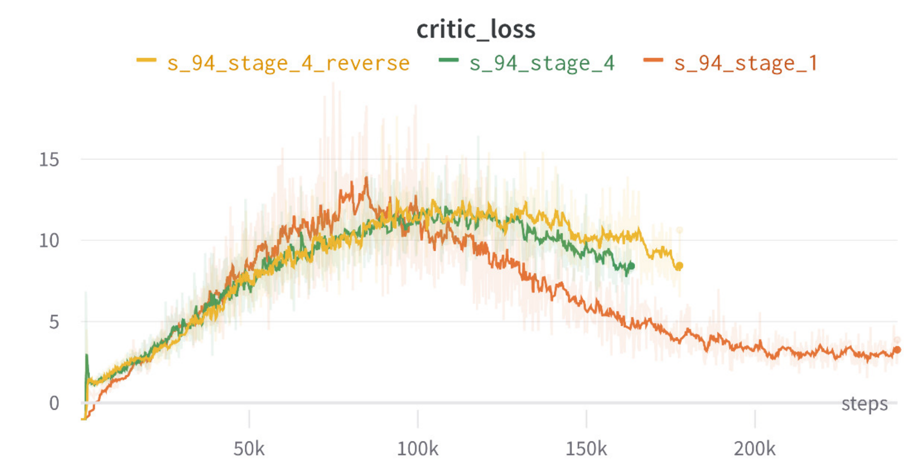

Figure 5:Critic Training loss

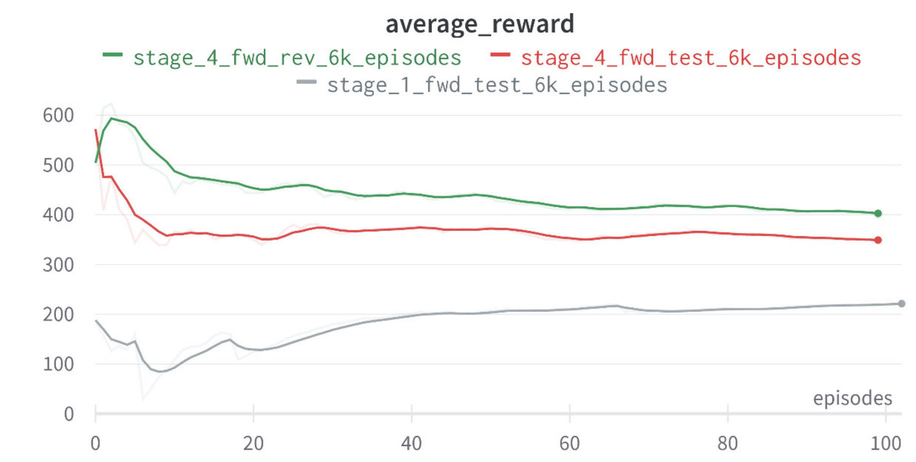

Figure 6:Average Test

Reward function

The robot can be in any one of the following states:

- Moving closer to the goal ($move\_closer$): 200*diff
- Moving away from the goal ($move\_away$): -8
- Collided with the obstacles ($collided$): -10
- Reached goal state ($reached\_goal$): +100

Conclusion

- The DDPG algorithm shows promising results for the Autonomous Mobile Robot control in different mapless environments.
- Because, the state space is of 94 dimensions, agent needs a large number of exploration steps in the episodes and hence, it starts performing better after 1000 episodes.

Future Work

To see if other algorithms can perform better and converge faster, training agents using other Policy Gradient algorithms:

- ADDPG - Asynchronous Deep Deterministic Policy Gradient: The Asynchronous training can help the model converge faster and update the network parameters frequently and gain by exploring different scenarios in parallel.
- MADDPG - Multi-agent Deep Deterministic Policy Gradient: A multi-agent system where the DDPG agents can learn to collaborate and synchronize their motions to achieve individual goals efficiently while maintaining system safety.