# Project: Trajectory Planning Based on Artificial Intelligence

This project is about developing an agent with 2DOF which is able to chase a randomly moving target. In this project, we should use reinforcement learning algorithms that are able to automatically find good decision policies in continuous state and action spaces.

In this report, we talk about implementation of the desired environment. Then, we will explore different RL algorithms for continuous state and action spaces.

# Problem Statement

In this project, we should develop a 2-joint planar robot arm system that can track an object with random velocity.

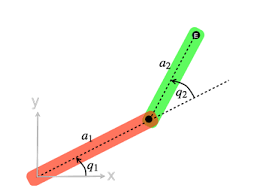


Figure 2-joint planar robot arm [1]

The system has 2 degrees of freedom (). In each timestamp, the angle of both arms can change a specific amount within a particular range (for example between (. There is also an object in the environment that has a constant bounded random velocity which will change randomly after the object reaches the system’s infeasible region.

## Feasible Region

The end effector position of the system in xy plane can be derived using the following equation:

The region that the end effector of the system can reach can be calculated using the above equations.

# Environment

The desired environment is implemented in gym library. The user should specify the length of both links and also the episode length. As discussed in the previous section, the environment’s action space has size 2. The observation space includes the angle of the first and second arm, location, and speed of the target (size= 6).

In the step method, the angle difference will be added to the current angles of the system. In addition, a valid point with the aforementioned criteria will be produced using point generator function. Then, the end effector position and the reward of the step will be calculated.

In the render method, a visual representation of the system and the object will be produced.

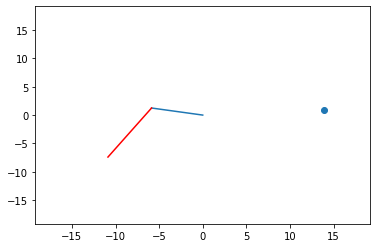


Figure 2 visual representation of the system

# Advanced Actor-Critic Methods [2]

## DDPG

Deep deterministic policy gradient (DDPG) is approximately the same as DQN. it uses a replay buffer to train an action-value function in an off-policy manner, and target networks to stabilize training. However, DDPG also trains a policy that approximates the optimal action. Because of this, DDPG is a deterministic policy-gradient method restricted to continuous action spaces.

## TD3

DDPG has been one of the state-of-the-art deep reinforcement learning methods for control for several years. However, there have been improvements proposed that make a big difference in performance. Twin-delayed DDPG (TD3) is a collection of improvements that together form a new algorithm. TD3 introduces three main changes to the main DDPG algorithm. First, it adds a double learning technique, similar to double Q-learning and DDQN, but this time with a unique “twin” network architecture. Second, it adds noise, not only to the action passed into the environment but also to the target actions, making the policy network more robust to approximation error. And, third, it delays updates to the policy network, its target network, and the twin target network, so that the twin network updates more frequently.

## SAC

## PPO

# Next Steps

Currently, I tried to implement the DDPG algorithm for the 2-joint planar robot arm. The problem is the average score of the agent does not improve significantly over the training episodes. In the following week, I will examine the implementation carefully to improve the agent’s performance. I may also try other continuous space and action methods to check and compare their overall performance.

# References

|  |  |
| --- | --- |
| [1] | fl0under, "Robotics," [Online]. Available: : https://robotacademy.net.au/lesson/analyzing-a-2-joint-planar-robot-arm/. |
| [2] | M. Morales, "Advanced actor-critic methods," in *Deep Reinforcement Learning*, Shelter Island, Manning Publications Co., 2020. |