Exercise 1

Amr Alkhashab - 905833
ELEC-E8125 - Reinforcement Learning

September 17, 2020

1 Task 1

Whie training, we set time step to 200 and while testing, we set it to 500.

```
#setting test to 500 time step
env._max_episode_steps = 500
#setting training to 200 time step
env._max_episode_steps = 200
```

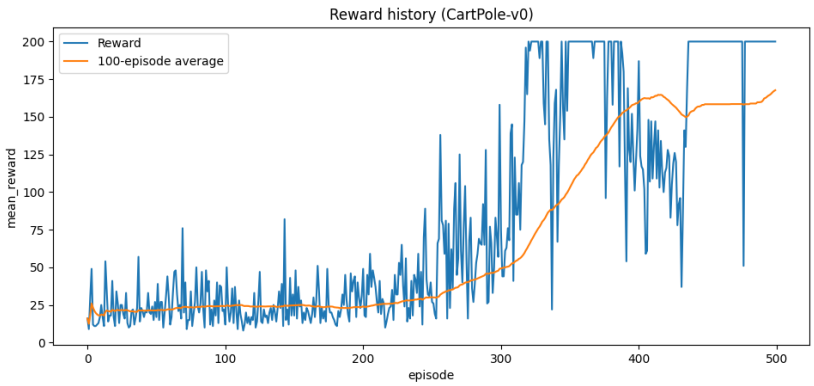

Figure 1: Reward for Task 1

Figure 2: Test position observation for Task 1

2 Question 1.1

For most cases, yes. Incases where the model learn to balance itself why oscillation without drifting too much like what can be observeed from Figure. 2 or drifting in a certain direction with low velocity, that it doesnot reach the termination state of obseravation[0]= +/- 2.4 for both the 200 and 500 timestep. This is entirely depend on how the agent was trained during those timesteps, and the training depends on the environment dynamics.

3 Task 2

The experiement has been repeared for 4 times.

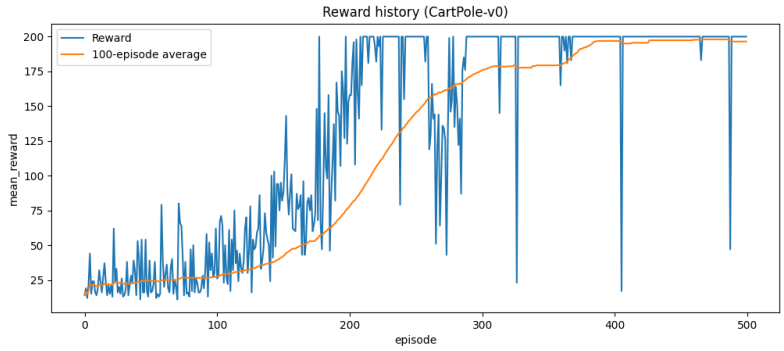

Figure 3: Reward for the first repetition

.

.

.

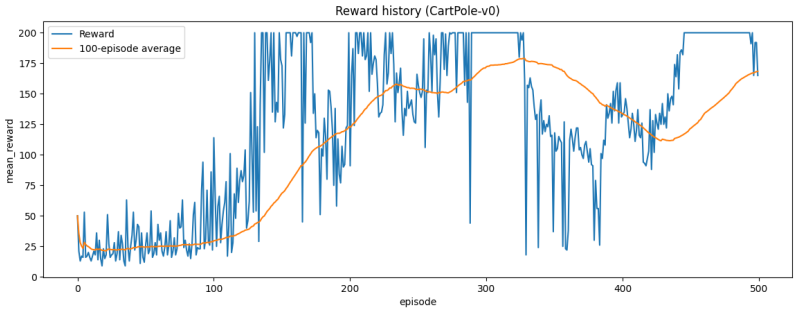

Figure 4: Reward for the second repetition

Results are:

1. Average test reward: 500.0 episode length: 500.0
2. Average test reward: 188.138 episode length: 188.138
3. Average test reward: 376.656 episode length: 376.656
4. Average test reward: 500.0 episode length: 500.0

4 Question 1.2

Behavior and performance of the repetitons are different. This can be seen clearly from Figure. 3, Figure. 2 and the above results. From the figures,we can observe that one was able to achieve a 100-episode avearge of about 200 at the end of episodes, while the other is very far away. This also can be seen also from the test results of the four repetition, where some obtain rewards less than 500 and terminates on avearge before the time step is completed. Even, those with same reward of 500 have different reward curves and learning approaches, in term of which convereges faster and as such.

5 Question 2

Since the agent is placed in an environment with many states, current states and action influence next states, and since the enivronment dynamics (transitions states) are intially unknown, the agent usually do random motion to try to explore the environment and learn it's dynamics to maximizing rewards, which the agent in some cases may fail to achieve thoughout the given training episodes. for some trials. The behaviors the agent need to adapt to balance the rod are many, in the sense that it may balance with high or low velocity or by changing postion in a strange manner. In conclusion, the robot can learn to achieve the max reward through different behaviors, and those behaviors are influenced by the state dynamics which the agent is trying to learn along the training and intial stocastic behavior. Reinforcement algorithm can be compared by doing many trials and measure the

average rate of convergance and consistance in variance and mean. As the number of trails goes to infinity, the results should converge to the real variance and mean produced by each model, the larger the avearge mean and smallerthe avearge variance means better results. Time of achieving this covergance is also important the faster the better, this means the result are much more consistant. .

6 Task 3

Each Task has been trained many times, the best result are only discussed in this section. A GIF is summited with the files for easier visulaization. A negative reward has been avoided since the agent seems to learn to terminates to minimize negative rewards and to avoid the zero reward return error.

6.1 Balance the pole close to the center of the screen (close to x = 0)

Code is as such: A reward is given every time the postion is observed to be between 0.2 and -0.2. Given a range instead of an exact number resulted in faster and more consistant training. Trained with 500 timestep and 1000 episode

```
def new_reward_zero(state):
    reward = 1
    x = state[0]
    if x > (-0.2) and x < (0.2) :
    reward = reward + 1
    return reward
```

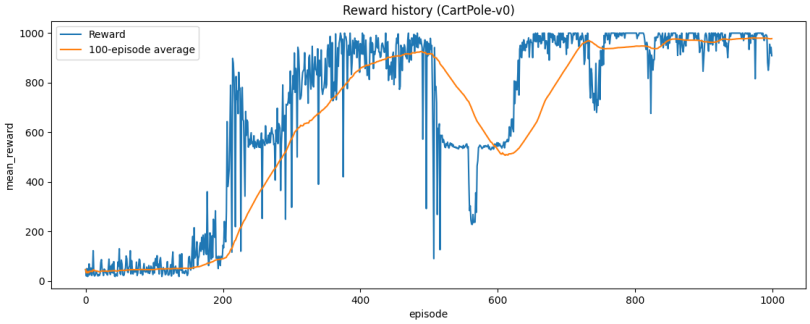

Figure 5: Reward for the first reward function

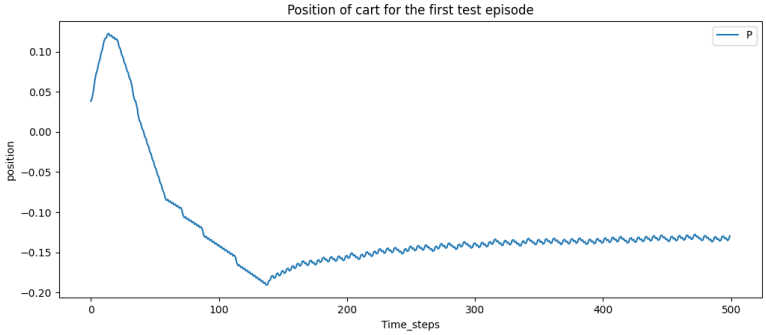

Figure 6: Position of first reward function for the first test episode

6.2 Balance the pole in an arbitrary point of the screen (x = x0)

Code is as such: A reward is given every time the postion is observed to be between 0.2 + desired position and -0.2 + desired position. Trained with 1000 episodes and 1000 timestep

```
def new_reward_position(state, position):
    reward = 1
    if x > (-0.2+position) and x < (0.2+position):
        reward = reward + 1
    return reward
```

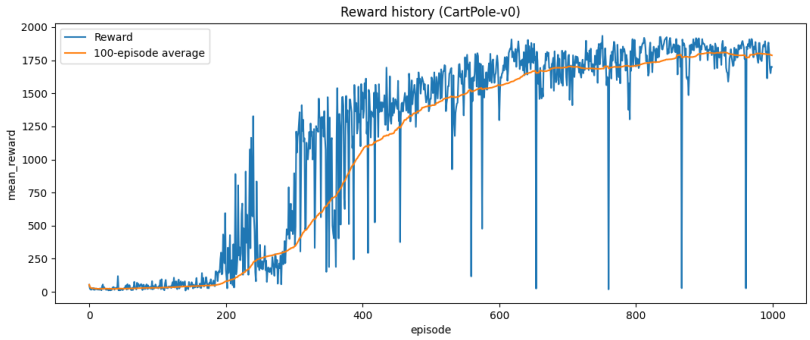

Figure 7: Reward for the second reward function for position equal -1

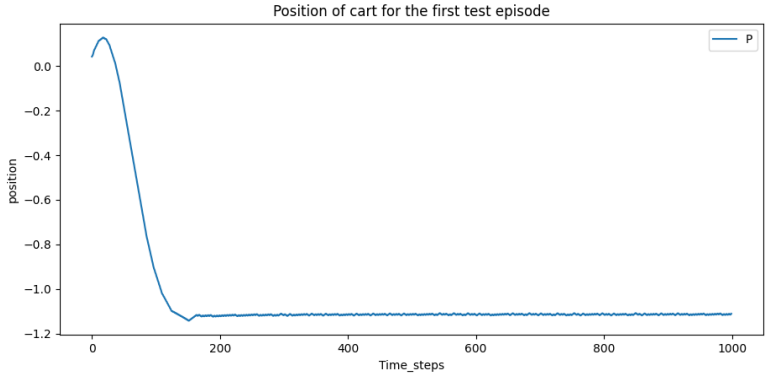

Figure 8: Position of second reward function for the first test episode for position equal -1

6.3 Keep the cart moving from the leftmost to rightmost side of the track as fast as possible

One approach is set a reward whever the velocity is larger than certain points, however implementation the agent doesnt not reach this velocity and just learn to balance the rod, so to guide the agent, a reward proportional to the velocity multiplied by a constant is given. Training an agent to achieve highest velocity is very difficult, many trail has been done before

the desired behavior is achieved. Trained for 1500 episode and 500 timestep.

```
def new_reward_speed(state):
    reward = 1
    v = state[1]
    reward = reward + abs(v)*4
    return reward
```

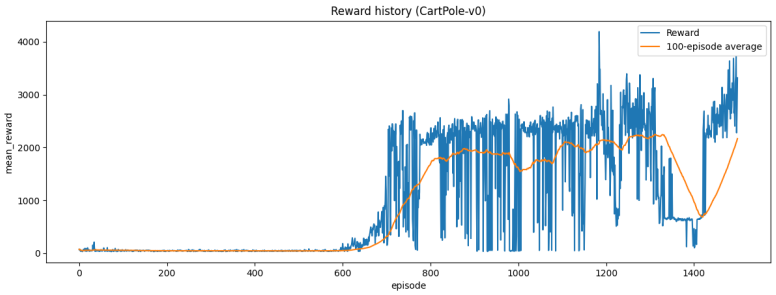

Figure 9: Reward for the third reward function

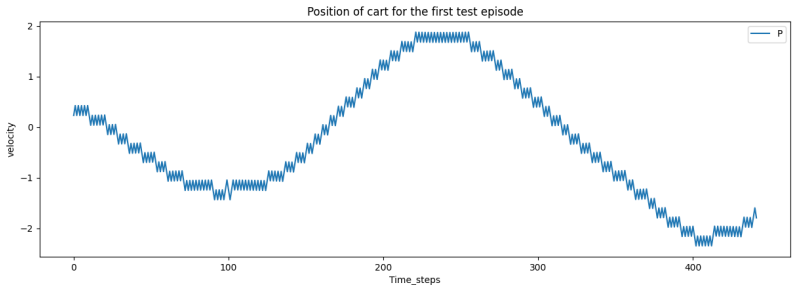

Figure 10: velocity of third reward function for the first test episode

7 Task 4

As we have mentioned before the agent training to achieve the object of the given rewards is very unstable. It require alot of trails and error, most of the time the epsiodes ends with a behavior of choosing to terminate at the boundries, or stucking at boundries after accelerating. 10 training trails of 1500 episodes has been done.

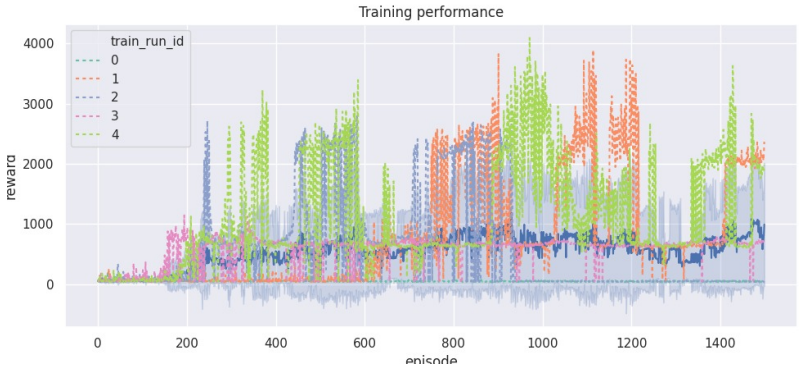

Figure 11: Multiple trainings for the third reward function

8 Question 3

The agent start accelerating from the starting point and then decelerate to avoid hitting the temination state, it then reverse back increasing it velocity as much as it can and stop at a point to prevent the rod of reaching it's terminating states, and then change direction and accelerates again. The velocity of the first episode can be seen from Figure. 10. The maximum velocity is about -2.2, the behavior is approximately the same accross all episodes.