EL2805 HT22 Lab2 Report

Philipp Katterbch, 20000510-T472
philipp.katterbach@rwth-achen.de

Markus Pietschner, 19990814-T378
markus.pietschner@rwth-achen.de

Abstract

Report for Lab2 of EL2805 Reinforcement Learning.

1 Task 1

1.a

Task done.

1.b

The replay buffer is needed to avoid correlations between samples used for training. Since a random choice of samples is gathered this effect gets achieved.

The target network is needed since we use a semi-gradient method, which assumes, that $\max(Q(s_{t+1}))$ is (nearly) constant.

1.c

Task done. ExReBu and Network taken from Lab0. We ignored the agent file and implemented everything in the DQN-problem file.

1.d

The network has 8 input nodes for the state. Those output using a ReLU activation to a 32 node hidden layer. The hidden layer applies another ReLU and then transmits to the output layer. We use optim.adam as it is recommended. The discount factor is chosen to be 0.99 as we don't use eligibility traces and have an environment with sparse rewards. Smaller discount factors lead to the starting choices not being updated on landing/crashing. Buffer size is chosen as 20000 and we simulate over 500 episodes. Those are chosen this way to be in the middle of the recommended ranges. Training batch size is 100 resulting in an update time of the target network of $C = L/N = 200$. Epsilon decays linearly from 0.99 to 0.05.

No modifications.

1.e

1.e.1

In figure 1 the reward stays mor or less constant for the first 200 episodes. After this, it increases steadily to three different peaks, after which it decreases for a short while. After 500 episodes it reaches the maximum reward of around 120.

Preprint. Under review.

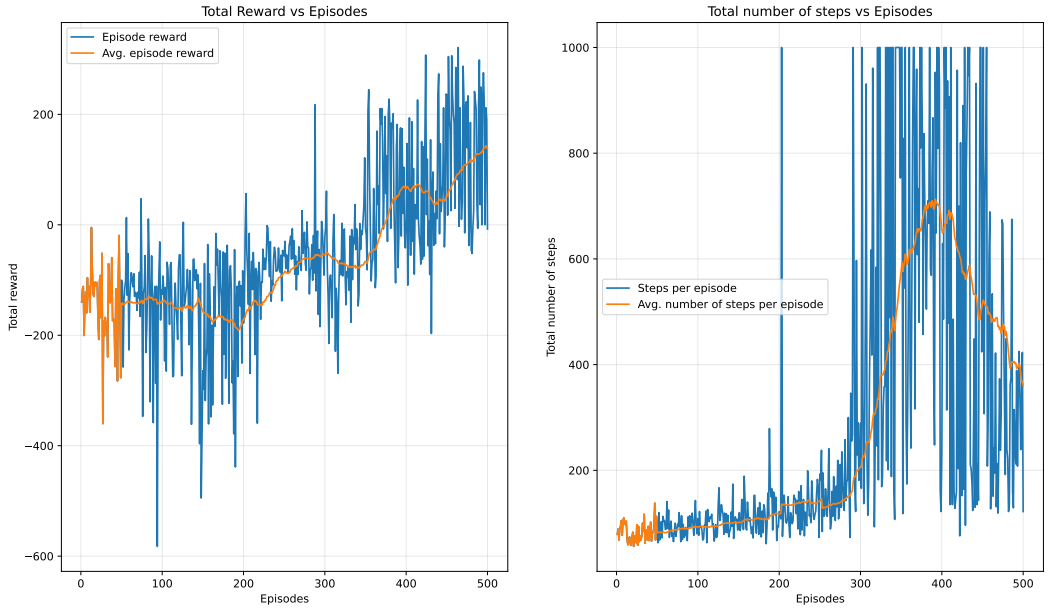

Figure 1: Total episodic reward and total number of steps taken per episode for standard parameters.

The number of steps stays below 200 for the first 300 episodes, after which it increases up to the maximum of 1000 steps in some cases. This can be explained with the lunar lander figuring out that it is not recommended to crash into the moon surface and therefore flying during the whole operation. In the end it figures out that the maximum reward can be achieved with a quick, precise landing, which decreases the number of steps to around 400.

1.e.2

For $\gamma = 1$ the reward decreases below -500 during the learning process. The number of steps stays very low. To sum it up, the RL-algorithm does not achieve a good result with this parameter

For $\gamma = 0.2$ the reward increases steadily until around -100, the number of steps also grows up to around 300. We can see in figure 3, that the learning process happens, but much slower as in figure 1.

1.e.3

With an increased number of episodes the average reward climbs up to 160 at around 650 episodes in figure 4, but decreases afterwards under 100. The number of steps per episode grows first and decreases again later on. For a lower number of episodes as in figure 5 the average reward does not reach a positive value, the average number of steps grows until a value of around 800. We notice, that a value for the number of episode which is either too high or too low, decreases the quality of our result.

For a lower batch size as in figure 6 the training becomes more stable, but also slower than with the standard parameters. For a higher batch size as in figure 7 the average reward fluctuates a lot more than with a lower batch size. The curve for the average number of steps per episode looks very similar.

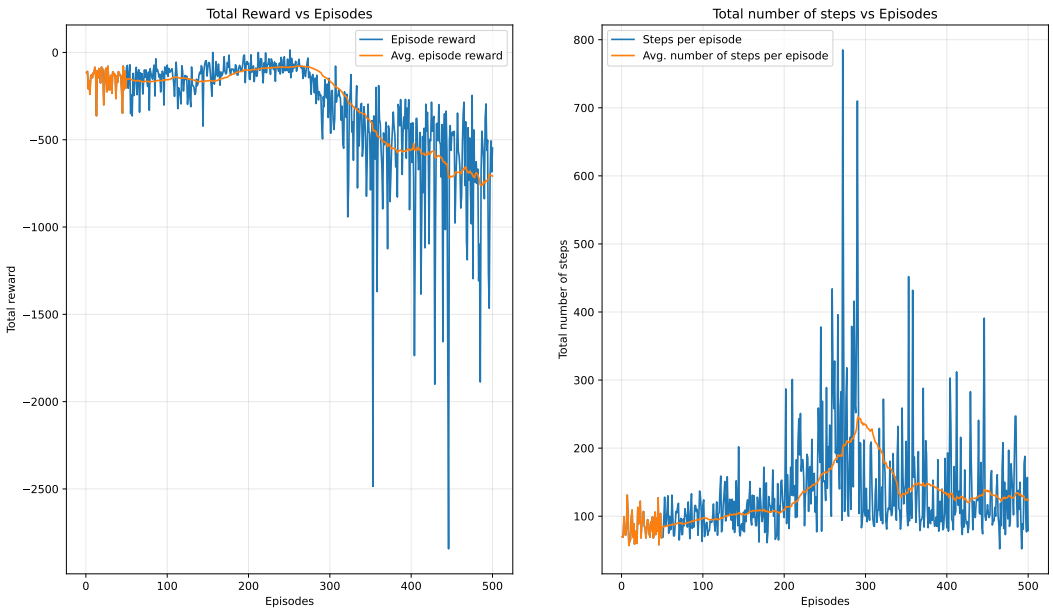

Figure 2: Total episodic reward and total number of steps taken per episode for $\gamma = 1$.

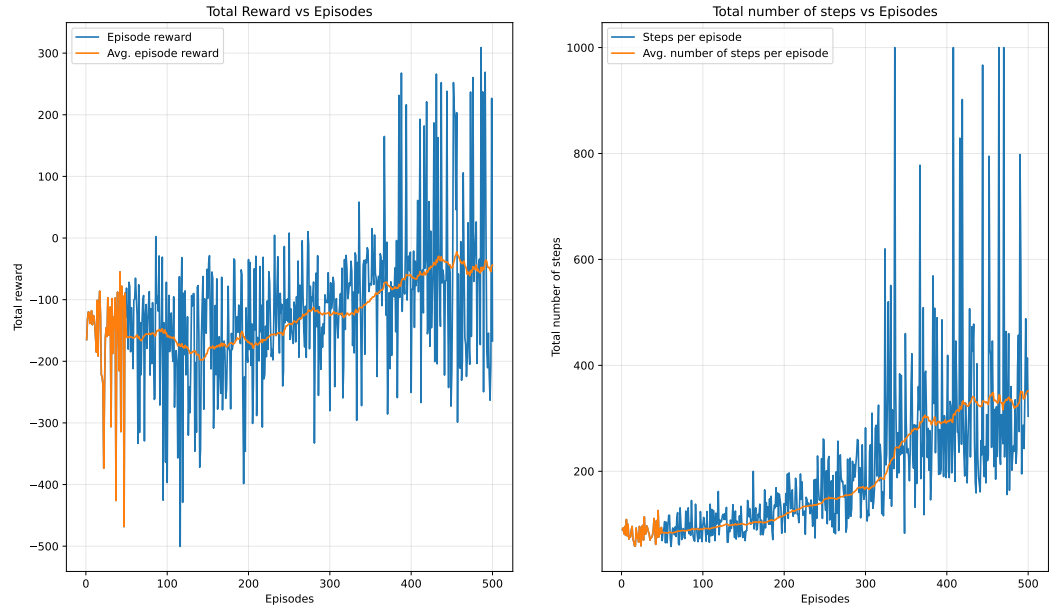

Figure 3: Total episodic reward and total number of steps taken per episode for $\gamma = 0.2$.

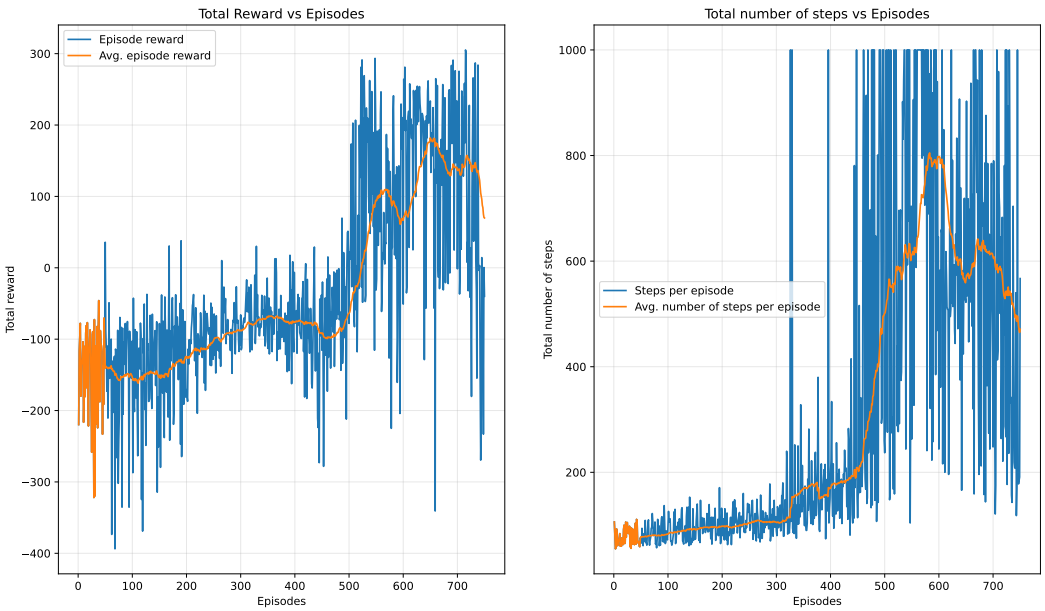

Figure 4: Total episodic reward and total number of steps taken per episode for num_episodes $= 750$.

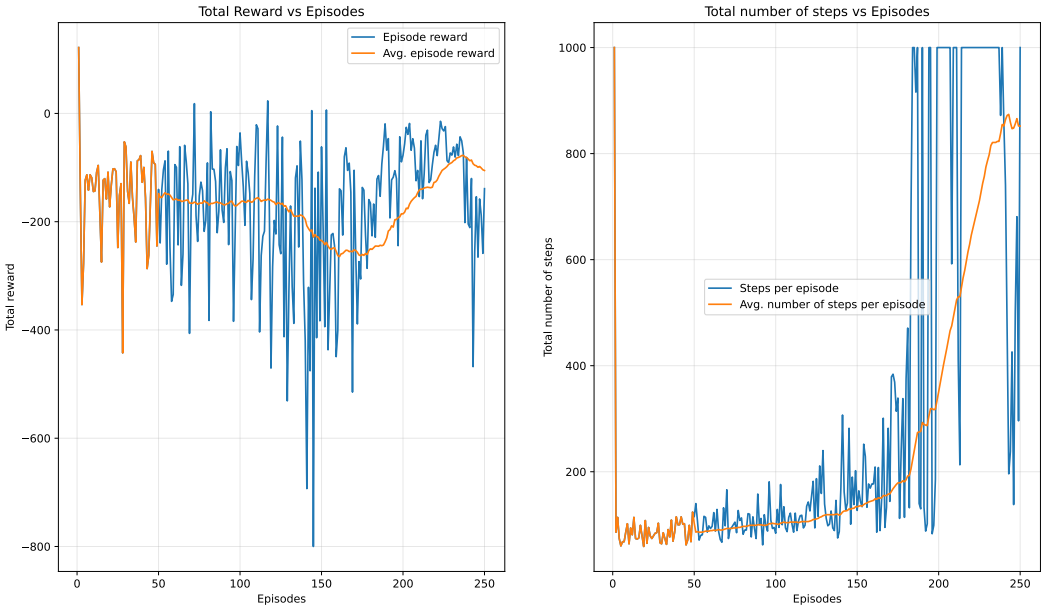

Figure 5: Total episodic reward and total number of steps taken per episode for num_episodes $= 250$.

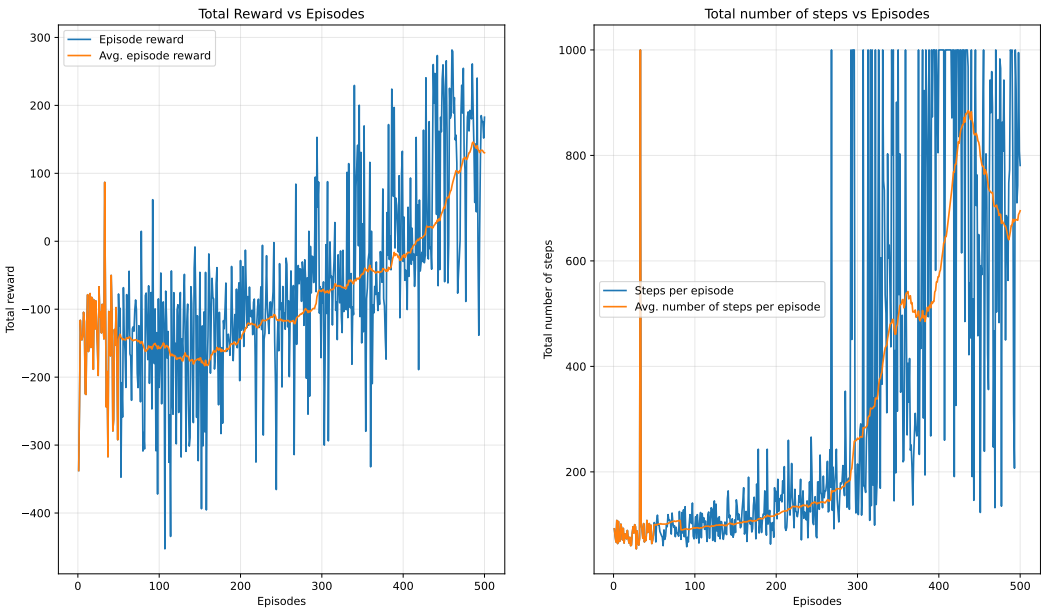

Figure 6: Total episodic reward and total number of steps taken per episode for batch_size $= 10,000$.

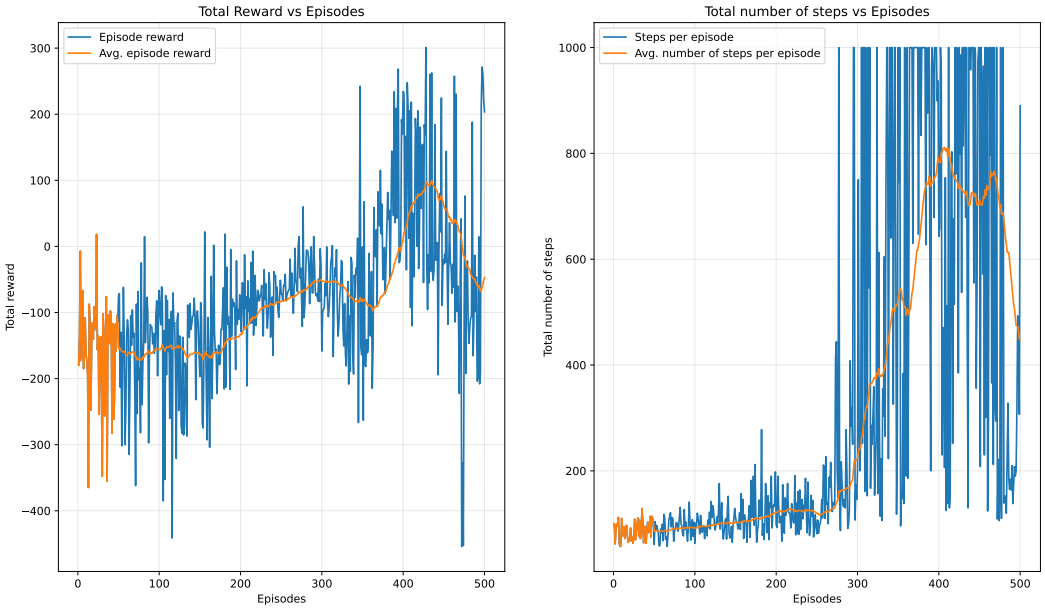

Figure 7: Total episodic reward and total number of steps taken per episode for batch_size $= 30,000$.

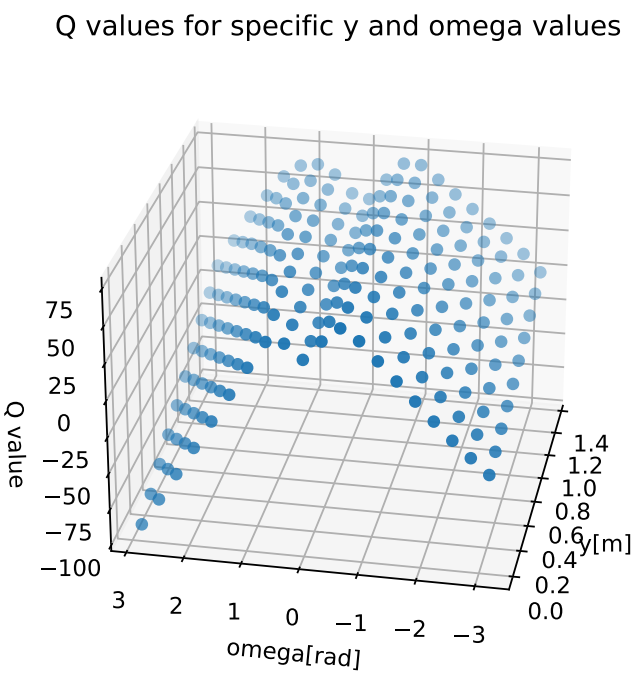

Figure 8: Q value

1.f

1.f.1

In figure 8 it is recognizable that an omega close to 0 increases the Q value. Furthermore, a higher y increases the Q value as well, but not by a lot.

1.f.2

Figure 9 shows a seperation in actions at omega $= 0$. When omega is less than 0, the lander tries to steer to reach omega $= 0$, the same goes for omega greater than 0, but in this case the steering direction is different. This makes sense as the lander tries to stabilize at omega $= 0$. The y value does not have any influence on the action.

1.g

In figure 10 and figure 11 the results of the trained and random agent are compared. With a few exceptions, the trained agent is able to achieve a result over 150. The random agent achieves total rewards below -100 in most cases. Therefore, the trained agent achieves a superior result.

1.h

Task done.

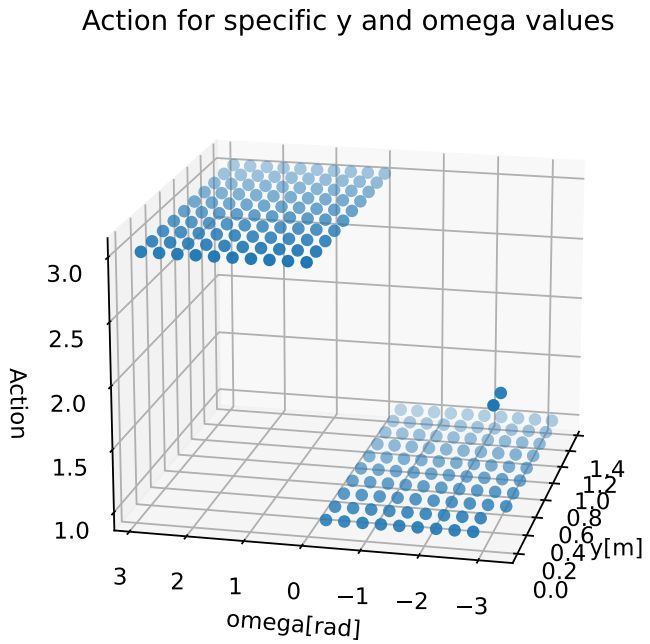

Figure 9: Actions

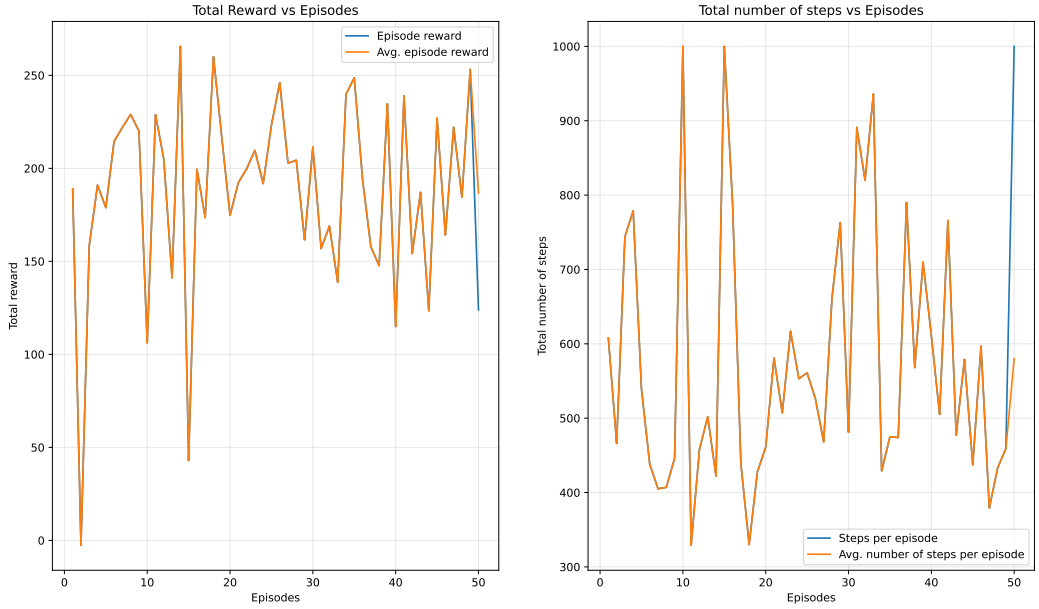

Figure 10: Result of the trained agent

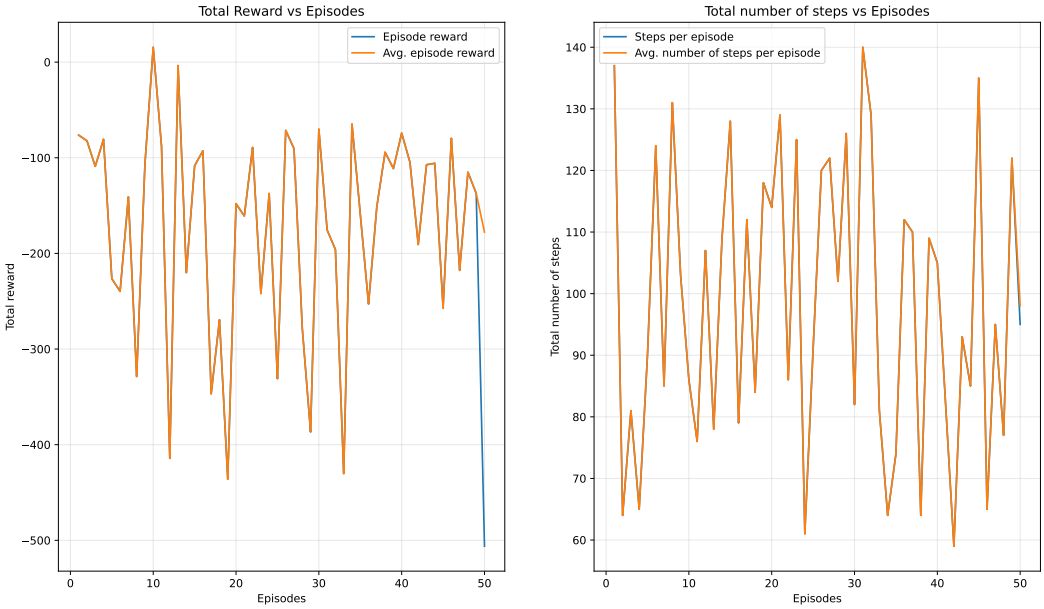

Figure 11: Result of the random agent