Mikaël Salasca (miksa599)
Jonas Michelberger (jonmi043)

Report TDDC17-Lab5

Task 2:

Question 1 (theory): In the report, a) describe your choices of state and reward functions, and b) describe in your own words the purpose of the different components in the Q-learning update that you implemented. In particular, what are the Q-values?

State function: To implement the getStateAngle function we initialized a variable which we assigned the discretized angle value. To make computations fast we used 10 angle states, but also experimented with 100 states which took as way longer for training. As a result, we get 10 states for the different angles.

Reward function: Since we want our rocket to point vertically upwards our reward function is rewarding a angle that is close to zero. A function to implement this is Reward = Pi – Angle, while the angle is limited between [-Pi, Pi]. If the angle is large the reward tends to zero. The closer the angle is to 0 the bigger the reward.

Qlearning update:
For the Q-learning update we have to know what our previous Q-Value was. This can be obtained by calling the QValue – get function. Additionally, we need the learning rate alpha, which we get from the previous state action and the last reward. The central element of the Qlearning algorithm is to obtain the Q max action. Therefore, we call the getMaxActionQValue and find the highest Qvalue of any action in the given state. Finally, we apply the Q-value function we learned in the reinforment learning lecture to generate our new Q-value for this state. The formula we are using is:

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Big(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \Big)}^{\text{temporal difference}}$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{new value (temporal difference target)}}$$

Source: https://en.wikipedia.org/wiki/Q-learning

The Q-values are previous states of the environment which serve as input to estimate what action is best to take next for the current state (iterative approach). The purpose of the update is to calculate the correct Q-value for the action performed which is improved with every step and gets closer to the real value.

Question 2: Try turning off exploration from the start before learning. What tends to happen? Explain why this happens in your report.

It starts to spin and fall down or is falling down at a very bad angle. It's the best it knows to this point and therefore he tries to exploit this behaviour. The algorithm has not learned what is good (high reward) and what is bad (small reward) so he is just using the best action he already knows until now without exploring anymore.

Task 3:

Description of our solution and interesting observations

For the implementation of the hover behaviour we had to specify not only a learning algorithm for the angle but also consider the vx and vy speed to get the hover behaviour. This constitutes trade-off because the different goals are influencing each other. Therefore, parameter optimization got very important to fine tune our values for the reward function.

For this part of the lab we have in total 400 states (10 for angle, 10 for vy and 4 for vx). Again, we experimented with several arbitrary numbers but since the size of the state-space and therefore the Q-table increases exponentially with the number of variables we include we did not choose too high values. For these parameters we reached a sufficiently good behaviour of the algorithm, but amore granular state space could further improve our implementation.

What is interesting is that due to the goal trade off (angle, horizontal and vertical speed) we can observe the tendency of the hover to fly vertically straight up because he has to speed up to correct angle and horizontal speed which then leads in many case to neglecting the vertical speed goal. Therefore, we tried to give the vertical speed goal a higher weight in the reward function to balance out our goals. In a space without gravity we don’t expect this behaviour.