Homework 2: **Due on 02/23/2025**

ESE 559 – Learning and Planning in Robotics

Problem 1: Consider a robot residing in the grid world environment of Fig.1. The actions of the robot are A = {up, down, left, right, idle}. The task of the robot is to eventually reach either the green or the blue cell with the smallest number of steps, while avoiding obstacles (red). The robot does not need to stay in the green/blue region once it arrives there. In other words, what happens after the robot reaches these regions is irrelevant. Therefore, in your implementation, you can treat the blue, green, and red cells as terminal MDP states (i.e., once you reach one of these states, you stay there forever). For simplicity, assume that the available actions at each state are only the ones that can keep the robot inside the workspace. When the robot chooses the action "idle" it stays with probability 1 in its current state s. For all other actions, the probability of going to the correct state is 0.7 and the probability of going to a neighboring state is 0.3/m(s) where m(s) is the number of neighboring states of the current state s. For instance, for the state s, shaded with gray color, we have that (i) the only available actions are {up, left, right, idle} and (ii) m(s)=3, P(s,`left', s1)=0.7, P(s,`left', s3)=0.15, and P(s,`left', s2)=0.15.

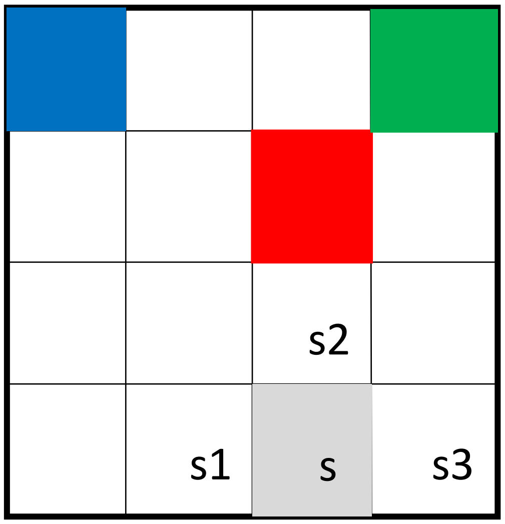

Figure 1: 4x4 grid world

A) Design rewards for the considered task. Explain your reward design.
B) Implement value iteration (Chapter 4.4) to compute the optimal policy. Provide (i) a figure showing how the state value function V evolves for the highlighted state s with respect to iterations; (ii) and a figure showing the optimal policy at each state (i.e., plot an arrow on the grid world showing the optimal action/direction).
C) Implement policy iteration (Chapter 4.3) and provide a figure showing the optimal policy at each state. Does policy iteration compute the optimal policy faster?

Submit also the code implementing B), and C).

Problem 2: Prove the Bellman equation for the state-value function that holds for any policy π (half of the proof was provided in class), i.e.

$$V_\pi(s) = \sum_a \pi(s,a) \left[\sum_{s'} P^a_{ss'} (R^a_{ss'} + \gamma V_\pi(s')) \right]$$

Provide a detailed explanation of your steps.

Problem 3: Consider two deterministic policies π and π′. The policy π′ is designed as follows:

$$\pi'(s) = \mathrm{argmax}_a Q_\pi(s,a)$$

Assume that the state-value functions of these functions are the same for all states s, i.e.,

$$V_\pi(s) = V_{\pi'}(s)$$

Using the Bellman equations, show that both π and π′are optimal policies (this was briefly proved in class). Provide a detailed explanation of your steps.

D) Implement the On policy Monte Carlo control algorithm discussed in class (Chapter 5.4 – to be covered on 02/19/2024) and apply it to learn the optimal policy. Provide a figure showing (i) the optimal policy at each state and (ii) the evolution of $Q(s,\pi^{*}(s))$ over iterations, where s is shaded with gray color in Fig. 1.