Part 1: QMDP

Consider the following discretized (Grid World) navigation problem where an agent is moving on a n × m grid with the actions "forward", "backward", "turn left", and "turn right" (thus the state representation has to include the orientation of the agent). The agent has unreliable actions where for "forward" and "backward" the probability that the agent actually moves to the intended grid cell (in the front or in the back, respectively) is 0.8 and the likelihood that the agent stays in the same cell is 0.2. Similarly, the turn actions turn the agent in the intended direction with a probability of 0.9 and stay in the same orientation with a probability of 0.1.

The environment contains a number of obstacles as well as a goal, both of which are unknown to the agent. If the agent's action would have it enter an obstacle cell (or leave the grid), the agent will stay in place (i.e. obstacles can not be traversed) and will receive a reward of -100. If the agent reaches the goal, the trial ends and the agent receives a reward of +100. All other rewards are 0 (i.e. there is no explicit cost to taking an action).

The agent can not perceive its location (and thus its state) but can only determine it with the following uncertainty: If the agent is in a particular location, the likelihood that it observes any location on the grid is 0 for any location with a Manhattan distance greater than 3 and uniformly distributed among all other locations. We design the state space, transition probability, and observation probability functions for the resulting POMDP and implement QMDP for this problem and evaluate its performance on a 15 × 25 world

Part 2: POMDP Learning

We consider a simplified grid-world as Part 1, we develop a modified version where the robot behaves the same except that it can no longer determine its position precisely. Instead it can only determine when it is hitting a wall or and obstacle and when it is sitting on top of the goal. Assuming that all the obstacle locations as well as the goal location are known beforehand, we design the POMDP for this problem and implement a competent POMDP learner (not QMDP) to learn a policy.