Design implementation

Note: References lines in `agent.py`.

In terms of the network structure, 3 layers were used. Layer 1 has dimensions [90, 125], Layer 2 has dimensions [125, 90] and Layer 3 (output layer) has dimensions [90, 4] (Lines 318-321). A learning rate of $\alpha = 0.002$ is also used to ensure that new training data do not severely outweigh previous experiences (Line 336).

For the Bellman equation, a discount factor of $\gamma = 0.95$ is used (Line 30). A mini-batch size of 850 transitions was chosen to update the Q function evenly across the environment space (Line 17). The initial episode length is set to 150 and is incremented to 175 when the number of episodes exceeds 30, and to 200 when the number of episodes exceeds 45 (Lines 55-59) to give the agent more chances for exploration.

Before any training was done, the experience replay buffer was filled with 15000 transitions (Lines 162-171) under the initial policy [0.275, 0.3, 0.275, 0.15] (Lines 259-263), indexed by [North, East, South, West], which encourages an overall movement to the East in order to explore the environment. Once the buffer is filled, training begins.

The agent is able to take one of four actions which each has a magnitude of 0.02 based on the current policy (Lines 266-274). This transition is stored into the buffer with a new weight that depends on the current maximum weight (Lines 148-160). Then, weighted random sampling is performed to obtain a mini-batch of 850 transitions which are used to train the Q network (Line 176). Here, a double deep Q-learning algorithm is used to update the network parameters (Lines 350-376). The new weights of the transitions are then updated based on delta which depend on: (i) the loss associated with that transition, and (ii) a small positive constant that is equal to 5% of the current largest weight (Lines 371-372).

During training, the target network is updated to have the same parameters as the Q network every 10 steps (Lines 189-191). The greedy policy ($\varepsilon = 0$) is tested every 15 steps (Lines 193-198). If the agent successfully reaches the goal, training stops (Line 202). Otherwise, training continues as usual (Line 174).

At the end of each episode, the policy is updated (Lines 113-124) based on a GLIE ε-greedy algorithm (Lines 83-110). The function used is $\varepsilon_{dec} = \frac{\varepsilon}{k^{0.35}}$, where ε_{dec} is the decreased epsilon, $\varepsilon = 0.6$, and k is the episode number. k is reset to 1 after each greedy policy is executed to keep ε_{dec} relatively high.

Finally, the reward associated with a transition is based on (i) R1: the distance to the goal (D), and (ii) R2: the x-position of the agent (X) as shown in Equation 1 where $\beta_1 = -1.5$, $\beta_2 = -2$ and the weighting terms are $a = 0.35$, $b = 0.65$.

$$R = aR_1 + bR_2 = a\left(\frac{\beta_1}{D^{\beta_1}}\right) + b\left(\beta_2(1 - X)\right)$$