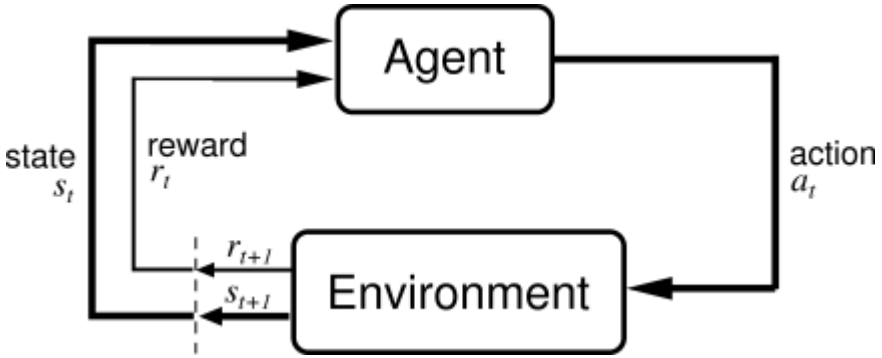
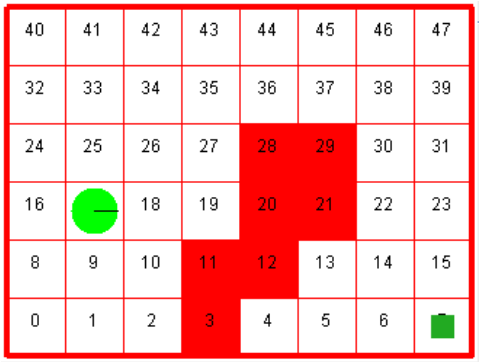
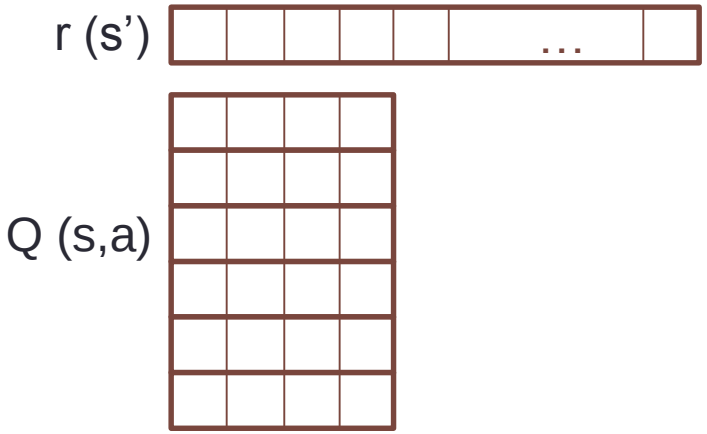
Q-LEARNING LAB

Q-Learning

- Simulation
- Learning
- Demonstration

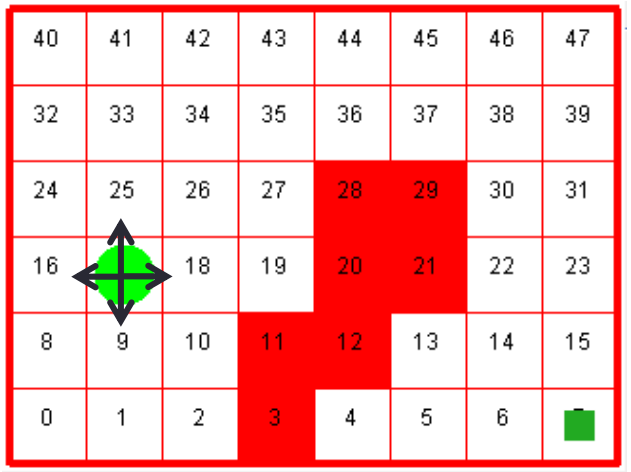






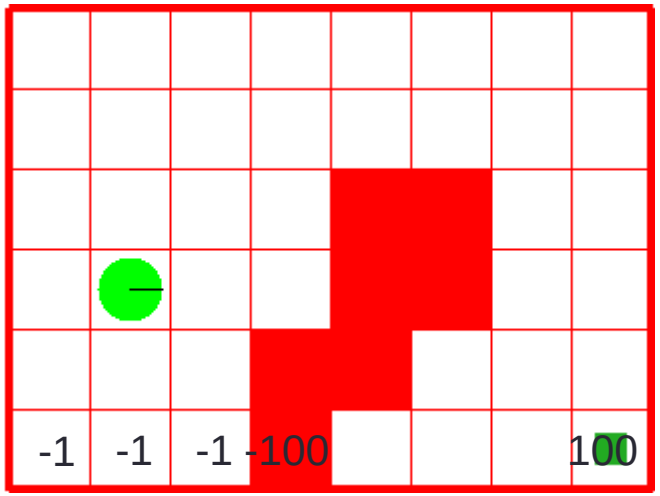
Simulation

- Randomly choose an action
  - a = rand.nextInt(4);
- Stochastically move in that direction
  - chance = rand.nextDouble();
  - if chance < 0.6
    - go in the direction of action
  - else if chance < 0.7
    - go to another direction unless wall (stay in place if wall)
  - ….
  - else
    - stay in place



Learning

- Construct the reward vector
  - -1 for empty cells
  - 100 for the goal state
  - -100 for wall cells (unimportant)



r (s')

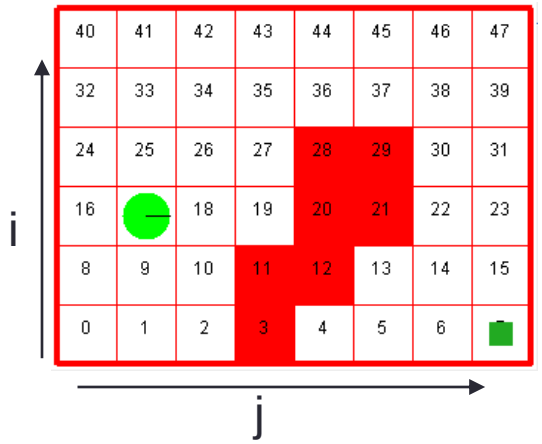
-1	-1	-1	-100				100	…	-1

Learning

- Construct the Q-Table
  - Each row → a state
  - Each column → an action
  - Initialize table with 0s
- Useful operations
  - i.e. $Q(s_3,E)$
  - i.e. $\max_{a'} Q(s_4', a')$
  - s = (roboti * colNum) + robotj
  - roboti = (int) (s / colNum)
  - robotj = s % colNum

Q (s,a)

	N	E	S	W
s_0	0	0	0	0
s_1	0	0	0	0
s_2	0	0	0	0
s_3	0	0	0	0
…	0	0	0	0
s_{n-1}	0	0	0	0



Learning

Algorithm 1 Q-Learning Algorithm

Define $\alpha = 0.65, \gamma = 0.99$
repeat
  Initialize s to robot or random location
  repeat
    Choose an action a from state s using random exploration policy
    Take the action a, observe reward $r(s')$ and next state s'
    $Q(s,a) \leftarrow Q(s,a) + \alpha[r(s') + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
    $s \leftarrow s'$
  until s reaches goal or max episode length
until number of episodes reached (e.g. 150 steps)

Single Episode

Learning

the state we ended up after performing action a

repeat
Choose an action a from state s using random exploration policy
Take the action a, observe reward $r(s')$ and next state s'
$Q(s,a) \leftarrow Q(s,a) + \alpha[r(s') + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
$s \leftarrow s'$
until s reaches goal or max episode length

choose action a' for which Q-value is maximized for state s'

→ expect to follow max policy after action a is taken at state s

- Current state: s_2
- Choose an action randomly: W
- Simulate the robot and **observe** the next state s'
- Assume next state: s_1

$Q(s_2,W) \leftarrow Q(s_2,W) + 0.65 * [\, r(s_1) + 0.99 * Q(s_1,a') - Q(s_2,W) \,]$

Demonstration

- Start from initial state
- Find the max valued action
- Use simulator to move the robot

- E.g. initial state: s_1

Q (s,a)

	N	E	S	W
s_0	-10	10	20	0
s_1	10	10	20	0
s_2	0	0	0	0
s_3	0	0	0	0
…	0	0	0	0
s_{n-1}	0	0	0	0

Questions ??

- Simulation
- Learning
- Demonstration