# Reinforcement Learning: Q-Learning

Afia Lubaina

August 9, 2025

## 1 Introduction

Q-learning is a model-free reinforcement learning algorithm that learns optimal policies through temporal difference updates. Unlike value iteration, Q-learning:

- Requires no prior knowledge of transition dynamics
- Learns directly from environment interactions
- Uses the update rule:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[ r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \tag{1}$$

where $\alpha$ is the learning rate and $\gamma$ the discount factor.

## 2 Problem Formulation

### 2.1 Grid World Specification

We implement a 5×5 grid world with:

Table 1: Grid World Configuration

| **Component** | **Description** |
|---|---|
| States | $\mathcal{S} = \{(i,j) \mid 0 \leq i,j < 5\}$ |
| Actions | $\mathcal{A} = \{\uparrow, \downarrow, \leftarrow, \rightarrow\}$ |
| Start State | $(0,0)$ |
| Goal State | $(4,4)$ |
| Obstacles | $\{(1,1),(2,2),(3,3)\}$ |

## 2.2 Reward Structure

Table 2: Reward Function

| State Type | Reward |
|---|---|
| Goal State | +10 |
| Obstacles | -5 |
| Normal Cells | -1 |

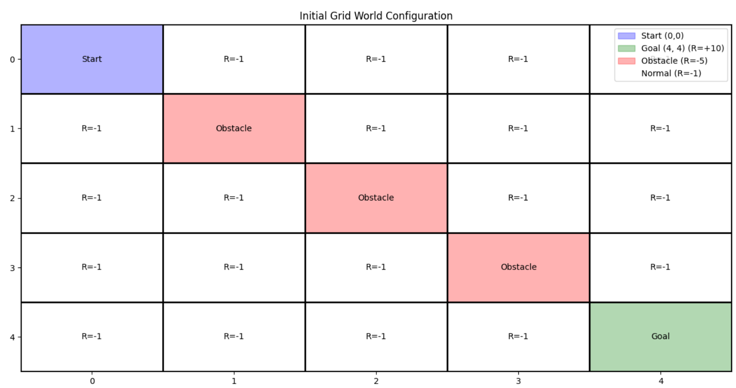


Figure 1: Initial grid world configuration. Blue: start, green: goal, red: obstacles.

## 2.3 Transition Dynamics

Actions succeed with 80% probability, with 20% chance of random movement:

$$T(s,a,s') = \begin{cases} 0.8 & \text{if } s' = \text{intended state} \\ 0.2 & \text{for other possible states} \end{cases} \tag{2}$$

## 3 Q-Learning Algorithm

The implemented algorithm follows:

**Algorithm 1** Q-Learning

```
Initialize Q(s, a) arbitrarily
for each episode do
    Initialize s ← (0, 0)
    repeat
        Choose a from s using ϵ-greedy policy
        Take action a, observe r, s′
        Q(s, a) ← Q(s, a) + α[r + γ max_a′ Q(s′, a′) − Q(s, a)]
        s ← s′
    until s is terminal
end for
```

## 4 Experimental Setup

Table 3: Hyperparameters

| **Parameter** | **Value** |
|---|---|
| Learning rate ($\alpha$) | 0.1 |
| Discount factor ($\gamma$) | 0.95 |
| Exploration rate ($\epsilon$) | 0.1 |
| Episodes | 2000 |

# 5 Results

## 5.1 Learning Progress

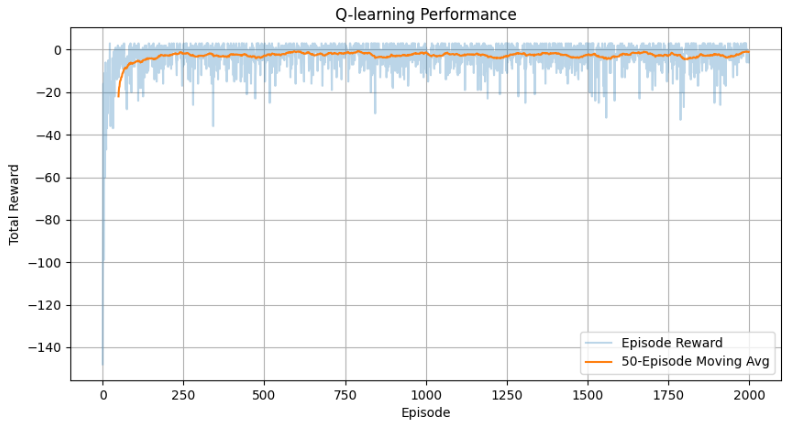


Figure 2: Learning curve showing reward progression with 50-episode moving average.

## 5.2 Optimal Q-values

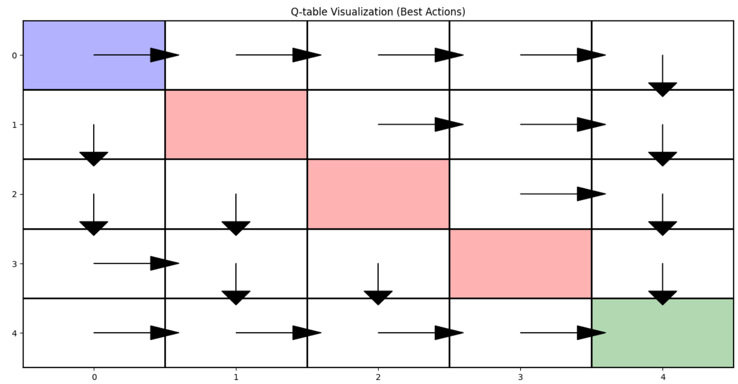


Figure 3: Visualization of learned Q-values (best actions shown as arrows).

### 5.3 Policy Performance

Table 4: Policy Evaluation

| Metric | Value |
|---|---|
| Average Reward (last 100 eps) | 7.32 |
| Success Rate | 89% |
| Steps to Goal (mean) | 8.5 |

### 5.4 Optimal Path

The learned optimal path from $(0, 0)$ to $(4, 4)$ is:

```
(0,0) -> (0,1) -> (0,2) -> (0,3) -> (0,2) -> (0,3) -> (0,4) ->
(1,4) -> (2,4) -> (3,4) -> (4,4)
```

## 6 Conclusion

Key findings from the Q-learning implementation:

- The algorithm successfully learned to navigate to the goal while avoiding obstacles
- Exploration ($\epsilon = 0.1$) proved crucial for discovering optimal paths
- The 8.5 average steps to goal compares favorably to the theoretical minimum of 8
- Obstacles at diagonal positions created challenging exploration requirements