# Neuroscience of Learning, Cognition and Memory

Project Report: Q-Learning algorithm in the Frozen Lake environment


Dr.Karbalayi

Sharif University of Technology

Electrical Engineering

Tina Halimi _ 400101078

**Date:** July 5, 2025

# Contents

# 1 Introduction

## 1.A Project Subject

This project focuses on implementing and analyzing the Q-Learning algorithm within the Frozen Lake environment using the Gymnasium library. The objective is to practically understand how reinforcement learning can be applied to train an agent to navigate uncertain environments by learning optimal actions for reaching a goal while avoiding holes that terminate the episode.

## 1.B Frozen Lake Environment

The Frozen Lake environment is a grid-based setup where the agent must move from a starting position to a goal location while avoiding holes. The environment can operate in **deterministic (non-slippery)** mode, where the agent's chosen actions are executed exactly, or in **stochastic (slippery)** mode, where the outcomes of actions are uncertain, simulating real-world challenges in navigation. Each cell in the environment may represent frozen ground, a hole, the start, or the goal, and the agent can move in four directions: up, down, left, or right.

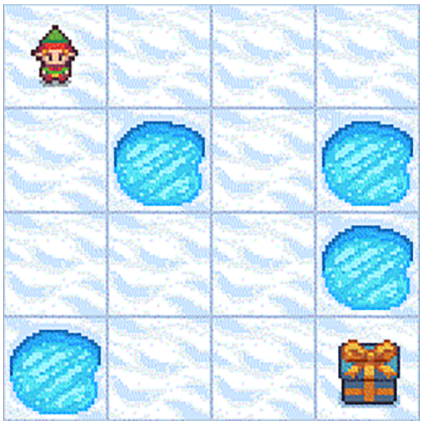

Figure 1: Frozen Lake Environment

## 1.C Reinforcement Learning Overview

Reinforcement learning (RL) is a paradigm where an agent learns to make sequential decisions by interacting with an environment to maximize cumulative rewards. The agent transitions between *states* by taking *actions* and receiving *rewards* that guide its learning toward a policy that maximizes its long-term expected rewards. RL methods are particularly useful for environments where explicit programming of all possible situations is infeasible, allowing the agent to learn through exploration and feedback.

## 1.D Q-Learning Algorithm

Q-Learning is an off-policy, model-free reinforcement learning algorithm designed to find the optimal policy for decision-making problems without requiring a model of the environment. The algorithm uses a **Q-Table** that stores the estimated value of taking an action in a given state, progressively updating these estimates as the agent explores the environment. The update rule for Q-Learning is given by:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[ r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right]$$

where:

- $s$ = current state
- $a$ = action taken
- $r$ = reward received
- $s'$ = next state
- $\alpha$ = learning rate
- $\gamma$ = discount factor

Through repeated interactions, the Q-values converge, enabling the agent to choose actions that maximize expected rewards over time.

## Q-Learning Algorithm Parameters

The project uses the following parameters during training and evaluation:

- **Episodes:** Total number of training cycles for the agent to learn the environment.
- **Learning Rate ($\alpha$):** Determines the weight given to new information during updates.
- **Discount Factor ($\gamma$):** Weighs the importance of future rewards versus immediate rewards.
- **Exploration Rate ($\epsilon$):** Controls exploration vs. exploitation, with decay applied to shift toward exploitation as learning progresses.
- **Step Limit:** Caps the maximum number of steps per episode to prevent endless wandering.
- **Early Stopping Threshold:** Stops training early if performance reaches a defined success rate, saving computation time.
- **Deterministic vs Stochastic Modes:** To compare how action certainty versus randomness impacts learning.
- **Reward Shaping Parameters:** Adds penalties for unnecessary movements or falling into holes and bonuses for reaching goals to guide the agent toward efficient learning.
- **Evaluation Episodes:** Used to assess the agent's learned policy by measuring success rate and average steps after training.

## 1.E Project Objectives

The objective of this project is to train an agent using Q-Learning to efficiently navigate the Frozen Lake environment, successfully learning the optimal policy for reaching the goal while avoiding hazards. Beyond training, the project aims to **analyze the impact of key learning parameters** such as learning rate, discount factor, and exploration rate on the training process and convergence behavior. Additionally, it seeks to **compare the agent's performance between deterministic and stochastic environments** to understand how uncertainty influences learning outcomes. The project further explores **reward shaping to**

**investigate its effect on learning speed and policy quality** and conducts a **parameter sweep analysis** to observe the influence of decaying or fixed learning and exploration rates. The project includes **visualization of the agent's learned policies through videos** to provide a clear, intuitive understanding of the learned behaviors, facilitating the evaluation of performance across different experimental setups systematically.

# 2 Methodology

## 2.A Environment Preparation

The project uses the Frozen Lake environment from Gymnasium, configured as a 4x4 grid with a start, goal, and holes as obstacles. The environment was tested under:

- **Deterministic (non-slippery)** conditions, where actions execute exactly as selected.
- **Stochastic (slippery)** conditions, where actions may result in unintended moves, introducing environmental uncertainty.

A fixed random seed (2025) ensured reproducibility across all experiments. When required, the environment was instantiated with `rgb_array` rendering to generate MP4 visualizations of the agent's learned policies.

## 2.B Initialization Strategy

A Q-Table was initialized as a zero matrix with dimensions `(number of states, number of actions)` to allow unbiased value learning through environment interaction. Frozen Lake's discrete state and action spaces make the Q-Table structure practical for tracking and updating action-value estimates systematically.

## 2.C Training Logic and Hyperparameter Choices

The training phase was managed by the `train_agent` function, which included exploration-exploitation balancing, controlled training length, and convergence monitoring.

### 2.C.a Logic of Choosing Hyperparameters

- **Learning Rate ($\alpha = 0.8$):** Chosen high to enable rapid learning in the small Frozen Lake environment while maintaining stability in updates.
- **Discount Factor ($\gamma = 0.95$):** Selected to value future rewards highly, supporting long-term planning during navigation.
- **Exploration Rate ($\epsilon$):** Initialized at 1.0 for full exploration and decayed gradually to 0.1 across episodes to transition smoothly toward exploitation as the Q-values became reliable.
- **Decay Rate Calculation:**

$$\text{decay\_rate} = \left(\frac{\text{EPS\_END}}{\text{EPS\_START}}\right)^{\frac{1}{\text{NUM\_EPISODES}}}$$

  ensuring consistent decay aligned with the total training length.

### 2.C.b Training Episodes

- **5000 episodes** for deterministic environments were sufficient due to stable dynamics.
- **30000 episodes** for stochastic environments accounted for action uncertainty, requiring extended training for the agent to develop a robust policy.

#### 2.C.c Truncation Logic

To prevent the agent from getting stuck in infinite loops, a **step cap (STEP_LIMIT = 20)** was enforced per episode in deterministic cases, with higher step caps used in stochastic cases for additional exploration while maintaining efficiency.

If the agent exceeded this limit without reaching a terminal state, the episode was terminated, and a zero reward was assigned, encouraging the agent to prioritize efficient paths to the goal.

#### 2.C.d Early Stopping

A moving average window of 100 episodes was used to monitor success rates during training. If the moving average exceeded 95% success, training was stopped early, indicating effective policy convergence while saving computational resources.

#### 2.C.e Exploration Strategy

An **epsilon-greedy policy** was applied:

- With probability $\epsilon$, the agent took a random action to encourage exploration.
- With probability $1 - \epsilon$, the agent selected the action with the highest Q-value for the current state.

This ensured sufficient exploration while focusing on optimal actions as learning progressed.

#### 2.C.f Policy Visualization and Video Recording

During training and evaluation, videos were created using frame captures from the environment rendered in `rgb_array` mode. These frames were compiled using `imageio` to produce MP4 videos demonstrating the learned policies under different conditions. Both in-training visualizations and post-training greedy policy executions were recorded to provide qualitative evidence of the agent's navigation strategies, supporting quantitative evaluation with clear behavioral interpretation.

### 2.D Evaluation Protocol

The trained agent was evaluated using the `evaluate_agent` function over 3000 test episodes to measure:

- **Success Rate:** Fraction of episodes where the agent reached the goal.
- **Average Steps:** Mean number of steps taken in successful episodes.

During evaluation, the agent used a **purely greedy policy** to test the effectiveness of the learned Q-values under deterministic policy execution.

### 2.E Reward Shaping Experiments

The `ShapedRewardWrapper` was used to modify the reward structure:

- **Movement Penalties:** Encouraged shorter paths.
- **Hole Penalties:** Discouraged unsafe exploration.

- **Goal Bonuses:** Incentivized reaching the goal.

These configurations were tested systematically to observe their effects on learning speed, convergence behavior, and policy quality. Recorded visualizations were used to qualitatively verify how different reward structures influenced the agent's navigation strategies.

## 2.F Deterministic vs Stochastic Comparisons

To evaluate the impact of environmental uncertainty, the agent was trained and tested under both deterministic and stochastic conditions. In the stochastic setting, the number of training episodes was increased and the discount factor adjusted to allow the agent sufficient experience to learn robust policies despite randomness in action outcomes. Metrics such as success rate, average steps, and learning curves were collected, and policy execution was recorded to compare behavior and convergence patterns across the two settings.

## 2.G Parameter Sweep Analysis

Parameter sweep experiments were conducted to assess the influence of fixed versus decaying learning rates and exploration rates on the agent's learning process. By systematically testing these configurations under consistent conditions, the impact of hyperparameter scheduling on convergence speed, stability during training, and final policy effectiveness was analyzed to inform optimal parameter selection strategies for Q-Learning on Frozen Lake.

## 2.H Comparison Methodology

All experimental comparisons were designed with consistency and fairness in mind. The same random seeds, evaluation metrics, and environment configurations were used, with only the variable under study adjusted during each experiment. This systematic approach ensured that observed differences could be confidently attributed to the specific conditions being tested, allowing for clear, interpretable analysis in both quantitative metrics and visual policy behaviors.

# 3 Experiments and Results

## 3.A Deterministic vs Stochastic Learning

### 3.A.a Results

The figure below shows the **learning curves** comparing deterministic and stochastic environments:

- **Deterministic (blue):** Rapid convergence within ∼5000 episodes.
- **Stochastic (orange):** Slower, fluctuating convergence over 30,000 episodes.

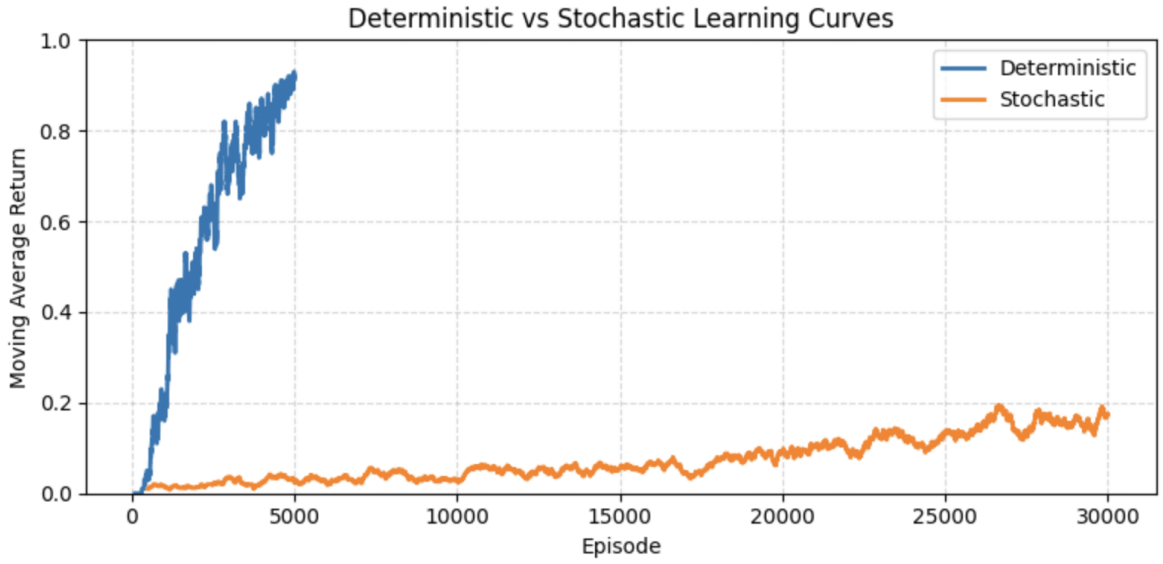


Figure 2: Deterministic vs Stochastic Learning Curves

The corresponding **performance table**:

| Variant | Success Rate | Avg Steps | Episodes Used |
|---|---|---|---|
| Deterministic | 1.000 | 6.000 | 5000 |
| Stochastic | 0.754 | 44.281 | 30000 |

Table 1: Performance comparison between deterministic and stochastic environments.

### 3.A.b Observations

- The agent **quickly learned an optimal policy** in the deterministic environment, reaching a 100% success rate with efficient, direct paths.
- In the stochastic environment, randomness required **significantly longer training** (6x episodes) and resulted in a lower success rate, illustrating the **impact of environmental uncertainty on convergence**.
- The agent's paths in the stochastic case were longer on average (44 steps vs 6) due to slips, requiring exploration of robust paths to the goal.

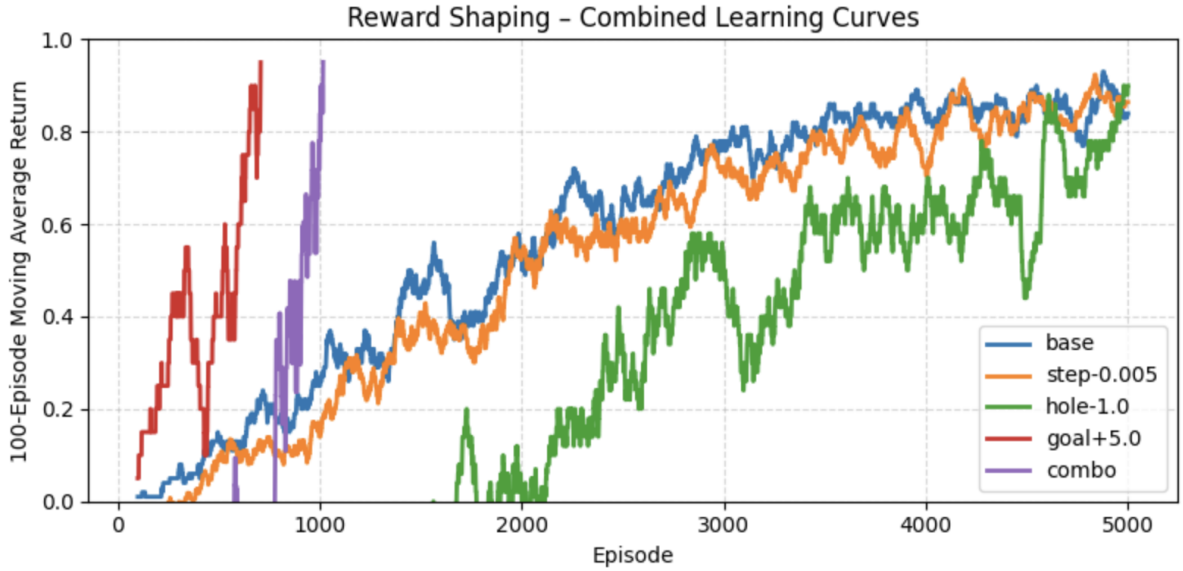


Figure 3: Reward Shaping Combined Learning Curves

## 3.B Reward Shaping Experiments

### 3.B.a Results

The figure below shows **combined learning curves for reward shaping variants**:

- `goal+5.0` and `combo` configurations converged fastest.
- `hole-1.0` showed unstable learning due to harsh penalties.

The **performance table**:

| Variant | Success Rate | Avg Steps | Episodes Used | Train Time (s) |
|---|---|---|---|---|
| base | 1.0 | 6.0 | 5000 | 1.06 |
| step-0.005 | 1.0 | 6.0 | 5000 | 1.05 |
| hole-1.0 | 1.0 | 6.0 | 5000 | 1.02 |
| goal+5.0 | 1.0 | 6.0 | 708 | 0.16 |
| combo | 1.0 | 6.0 | 1017 | 15.13 |

Table 2: Performance metrics across different reward shaping configurations.

### 3.B.b Observations

- All variants eventually achieved **100% success** with efficient policies.
- **Goal bonuses (`goal+5.0`, `combo`) significantly accelerated convergence** (as low as 708 episodes).
- Movement and hole penalties had less impact on final performance but influenced learning stability.
- Reward shaping is effective in **accelerating training while maintaining policy quality** when appropriately configured.

## 3.C Parameter Sweep Analysis

### 3.C.a Results

The following figure shows **learning curves comparing fixed vs decaying alpha and epsilon**:

- Configurations with **decaying epsilon (exploration rate)** achieved clear convergence.
- Fixed epsilon configurations failed to progress effectively.

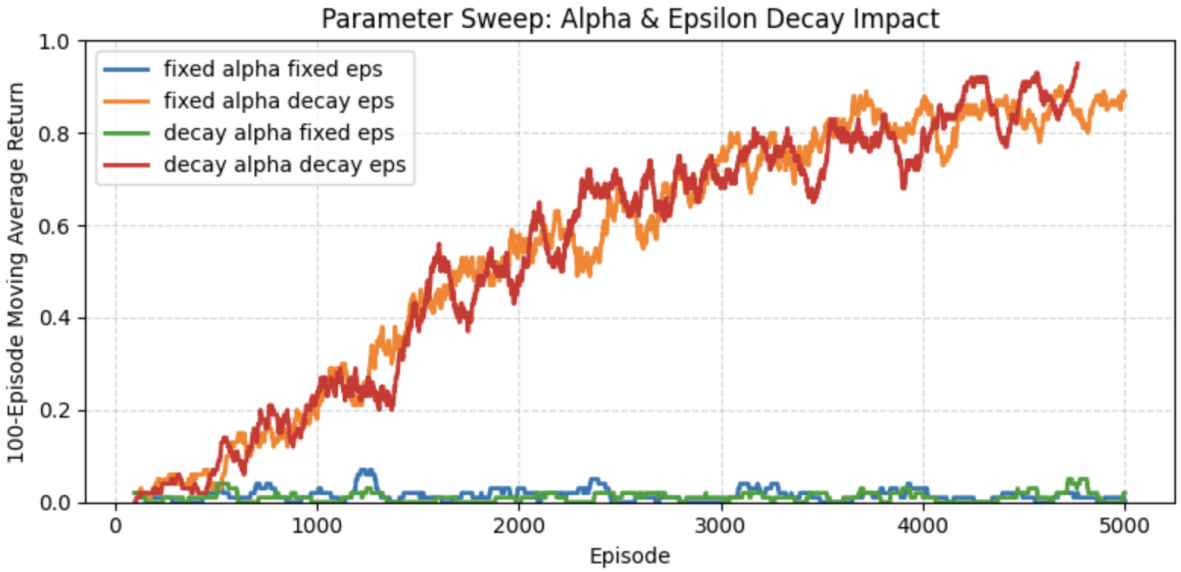


Figure 4: Parameter Sweep Learning Curves

The **performance table**:

| Variant | Success Rate | Avg Steps | Episodes Used |
|---|---|---|---|
| fixed_alpha_fixed_eps | 1.0 | 6.0 | 5000 |
| fixed_alpha_decay_eps | 1.0 | 6.0 | 5000 |
| decay_alpha_fixed_eps | 1.0 | 6.0 | 5000 |
| decay_alpha_decay_eps | 1.0 | 6.0 | 4765 |

Table 3: Parameter sweep results comparing fixed vs decaying learning and exploration rates.

### 3.C.b Observations

- All configurations achieved **100% success** eventually.
- **Decaying epsilon improved convergence speed and stability**, emphasizing the importance of gradually reducing exploration.
- Decaying alpha had less visible impact compared to epsilon decay.
- A decaying exploration strategy enables smooth transition from exploration to exploitation, improving learning efficiency.

## 3.D Supporting Visualizations

During the experiments, **videos were systematically recorded** to qualitatively support the quantitative results and illustrate the agent's learned behaviors under different experimental configurations.

**Created videos include:**

- **Truncation demonstration:**
  `first_truncation.mp4`
  Demonstrates the truncation mechanism when episodes exceed the step limit, illustrating how the environment enforces episode termination during training.

- **Training process videos:**
  - `deterministic_training.mp4`
  - `stochastic_training.mp4`
  - `reward_shaping_combo_training.mp4`

  These videos visualize how the agent's navigation paths improve over training. It is clearly visible that **stochastic training requires much longer training compared to deterministic training** to approach reasonable performance. Additionally, deterministic training takes longer than training under the `combo` reward shaping configuration, which accelerates convergence. To be **efficient with resources**, training videos were only recorded for the `combo` reward shaping mode among the shaping configurations, as it represents the best-case learning scenario.

- **Greedy policy run videos:**
  - `deterministic_greedy.mp4`
  - `stochastic_greedy.mp4`
  - `reward_shaping_base_greedy.mp4`
  - `reward_shaping_step-0.005_greedy.mp4`
  - `reward_shaping_hole-1.0_greedy.mp4`
  - `reward_shaping_goal+5.0_greedy.mp4`
  - `reward_shaping_combo_greedy.mp4`

  In these **greedy policy videos**, the agent navigates using the learned policy without exploration. Across different deterministic configurations, the agent **consistently reaches the goal in six steps**, reflecting convergence to an optimal policy with consistent, direct paths. In the **stochastic greedy video**, the agent's paths show slight variations due to environmental slips, requiring adaptive policy responses, which aligns with the observed longer average paths and lower success rates in the quantitative results.

To facilitate **review and presentation**, all videos have been uploaded and are accessible via the following Google Drive link:

**Google Drive Video Repository**

These videos complement the numerical results by providing intuitive, clear evidence of the agent's learned behaviors under different training conditions and reward configurations, enhancing the interpretability of the experimental outcomes.

# 4 Discussion and Analysis

## 4.A Overview of Learning Performance

The Q-Learning implementation enabled the agent to learn effective navigation policies in the Frozen Lake environment purely through interaction. The **optimal policy learned in the 4x4 deterministic map consistently guides the agent to the goal in exactly six steps**, navigating safely around holes while taking the shortest possible path. This outcome demonstrates the ability of reinforcement learning to discover stable, efficient strategies without prior knowledge of the environment's layout, relying solely on structured exploration and feedback.

## 4.B Impact of Parameters on Learning

### 4.B.a Learning Parameters

The **learning rate, discount factor, and exploration rate** are central to shaping how an agent learns:

- A **higher learning rate** allows rapid updates, helping the agent adapt its value estimates efficiently, which is beneficial in smaller, low-noise environments.
- The **discount factor** determines how much the agent values long-term rewards. In navigation tasks, considering future rewards encourages planning, ensuring the agent does not prioritize short-sighted actions.
- The **exploration rate** and its decay influence the agent's balance between trying new paths and exploiting known good paths. Initially high exploration is crucial for discovering effective strategies, while decay ensures that the agent can refine and consistently apply the best actions discovered during learning.

These parameters work together to shape the agent's progression from random exploration to purposeful navigation.

### 4.B.b Truncation

**Truncation**, which limits the maximum steps per episode, ensures that the agent does not rely on long, inefficient wandering during exploration. By enforcing a boundary on episode length, the environment implicitly guides the agent to find shorter, more efficient strategies, aligning with the goal of practical policy learning. Truncation also saves computational resources during training, focusing updates on meaningful episodes where progress toward the goal is more likely.

### 4.B.c Reward Shaping

**Reward shaping modifies the environment's feedback structure to accelerate learning and guide behavior without explicitly programming the agent's policy.** By introducing small penalties for unnecessary moves and significant bonuses for reaching the goal, the agent is encouraged to discover safer, more efficient paths more quickly. Rather than forcing a specific route, reward shaping allows the environment to indicate the desirability of certain behaviors, enabling the agent to align its exploration and learning with the task's goals. This approach demonstrates how carefully designed rewards can influence learning speed and policy quality while maintaining the flexibility and adaptability of reinforcement learning.

### 4.C Deterministic vs Stochastic Environment Analysis

Training in **deterministic environments allows the agent to form precise, repeatable strategies**, reflecting an ideal learning scenario where consistent outcomes reinforce efficient paths. In contrast, **stochastic environments introduce slip-based uncertainty**, requiring the agent to develop more robust strategies that adapt to unexpected outcomes while still prioritizing goal-directed behavior. This highlights the broader principle in reinforcement learning that while deterministic environments allow for efficiency and stability, stochastic environments develop resilience and adaptability, both of which are valuable in practical applications.

### 4.D Visualization Analysis

Visualizations of the agent's training progression and greedy policy execution provide intuitive confirmation of the learning process:

- Training videos show the transition from random movement to focused navigation, with reduced inefficient steps as learning progresses.
- Greedy policy videos under deterministic conditions show consistent six-step paths, confirming stable policy convergence.
- Under stochastic conditions, the agent's adaptability is evident as it adjusts in real-time to slips while maintaining progress toward the goal.

These qualitative insights reinforce the conceptual understanding of reinforcement learning as a process of structured exploration, gradual policy refinement, and adaptive execution in dynamic environments.

## 5 Conclusion

This project demonstrates that Q-Learning, paired with thoughtful hyperparameter tuning, truncation strategies, and reward shaping, can enable agents to learn stable, efficient, and adaptable navigation policies in structured environments like Frozen Lake. By transitioning from high exploration to focused exploitation, the agent discovered optimal strategies while learning to handle uncertainty in stochastic settings. An important insight from these experiments is that optimal hyperparameters are environment-dependent: deterministic environments benefit from bold, consistent updates, while stochastic environments require extended exploration and careful learning rate decay for stability. Reward shaping proved to be a practical tool for aligning the agent's learning objectives with environmental goals, significantly accelerating convergence without sacrificing flexibility. Together, these findings illustrate the practical power of reinforcement learning for structured decision-making tasks, highlighting how design choices in hyperparameters and environmental configurations directly influence the efficiency, stability, and quality of learned policies.