Assignment 2 Report: Escape Through the City

Amirthalingam R
Student ID: 24816

October 12, 2025

1. Implemented Functions (TODOs)

All required methods in `agents.py` were completed. Below are the key implementations.

Value Iteration

```
max_q = -np.inf
for action in self.env.get_actions():
    q_value = self._calculate_q_value(state, action)
    max_q = max(max_q, q_value)
new_value_table[state] = max_q
```

Value Iteration - Optimal policy

```
for state in self.states:
    if state == self.env.goal or state in self.env.obstacles:
        continue
    best_q = -np.inf
    for action in self.env.get_actions():
        q_value = self._calculate_q_value(state, action)
        if q_value > best_q:
            best_q = q_value
            current_policy[state] = action
```

Policy Evaluation

```
new_value_table[state] = self._calculate_q_value(state, self.policy[state])
```

Policy Improvement

```
for action in self.env.get_actions():
    if self._calculate_q_value(state, action) > self._calculate_q_value(state, best_action)
        best_action = action
self.policy[state] = best_action
```

2. Results

Stage 1: Sunny Day (No Obstacles, No Rain)

Convergence behaviour:

Discount factor	Iterations	Reward	Converged ?	Correct Policy ?
0.9	19	-11	Yes	Yes
0.7	19	-11	Yes	Yes
0.5	19	-11	Yes	Yes
0.3	13	-11	Yes	Yes
0.1	7	-100	Yes	No

Table 1: Convergence behavior of Value Iteration

Discount factor	Iterations	Reward	Converged ?	Correct Policy ?
0.9	19	-11	Yes	Yes
0.7	19	-11	Yes	Yes
0.5	19	-11	Yes	Yes
0.3	19	-11	Yes	Yes
0.1	17	-11	Yes	Yes

Table 2: Convergence behavior of Policy Iteration

Observations:

- Both algorithms converged to the same optimal policy in all cases except when the discount factor was 0.1.
- Value Iteration did not learn the correct policy when the γ value is small. This is because the agent becomes myopic and does not provide enough weightage to long term rewards.
- Policy Iteration always converged and learned the optimal policy.
- Lower γ gives faster convergence. But, can sometimes result in learning sub optimal policy.

Learned Policies

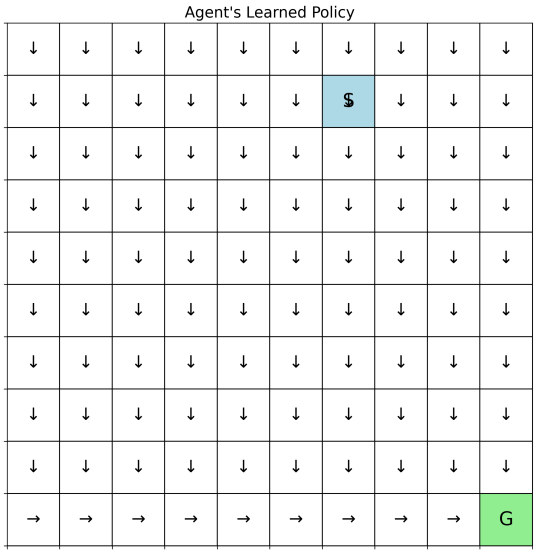

Figure 1: Value Iteration, γ=0.9

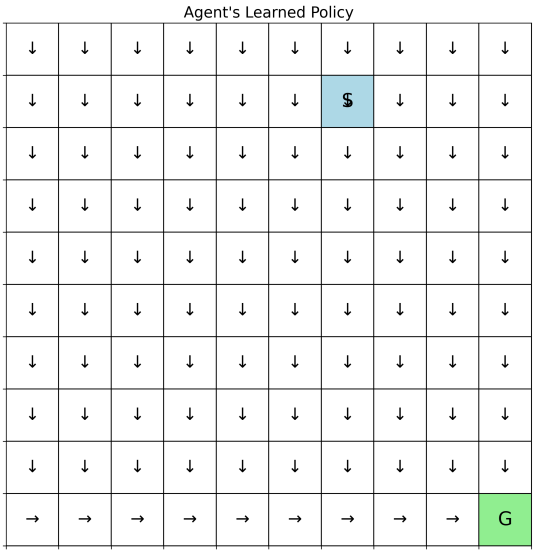

Figure 2: Policy Iteration, γ=0.9

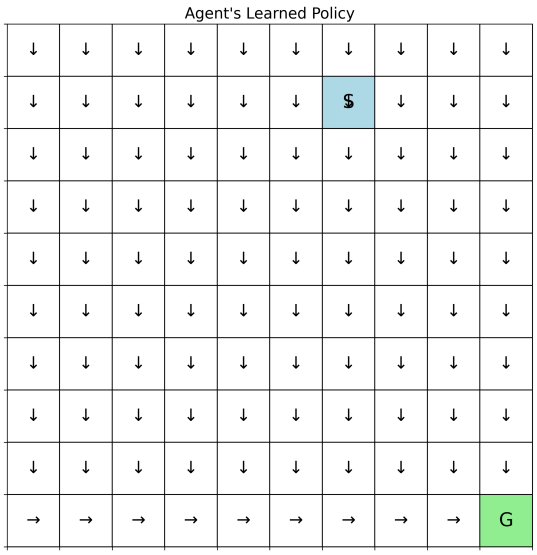

Figure 3: Value Iteration, γ=0.7

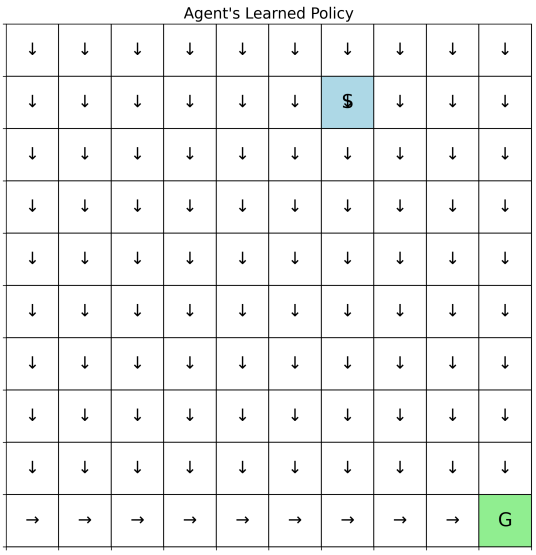

Figure 4: Policy Iteration, γ=0.7

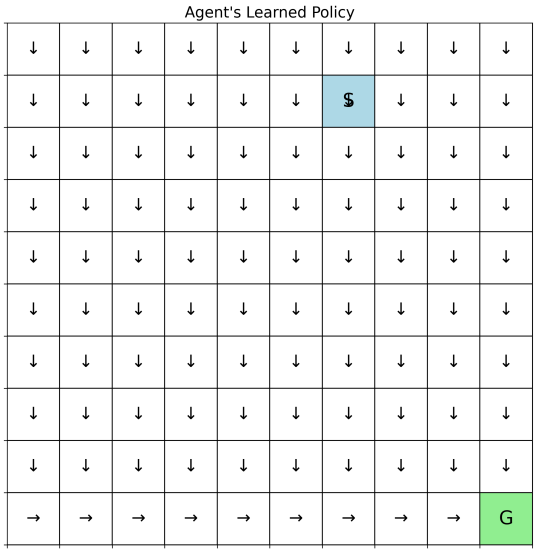

Figure 5: Value Iteration, γ=0.5

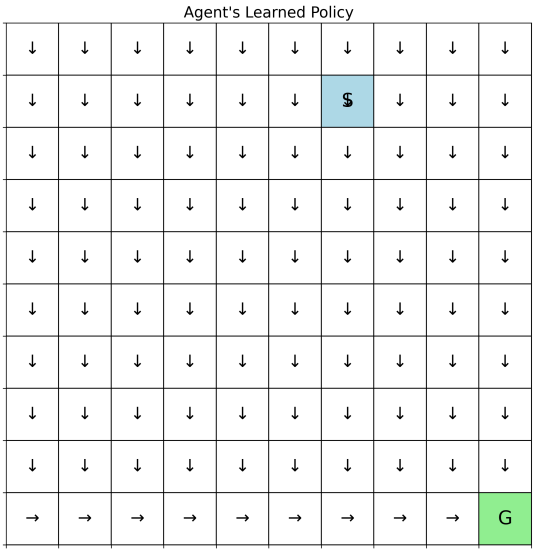

Figure 6: Policy Iteration, γ=0.5

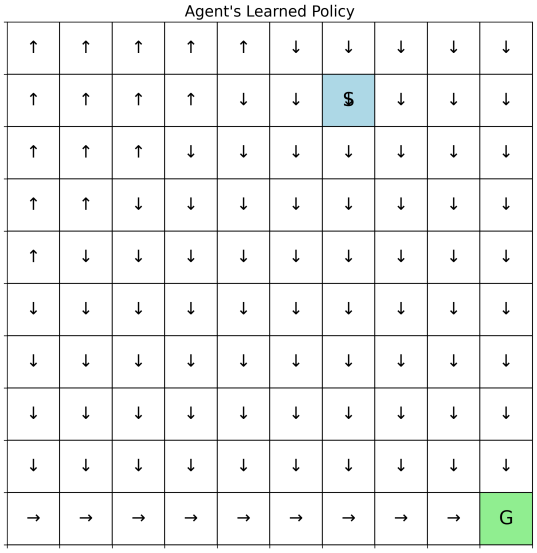

Figure 7: Value Iteration, γ=0.3

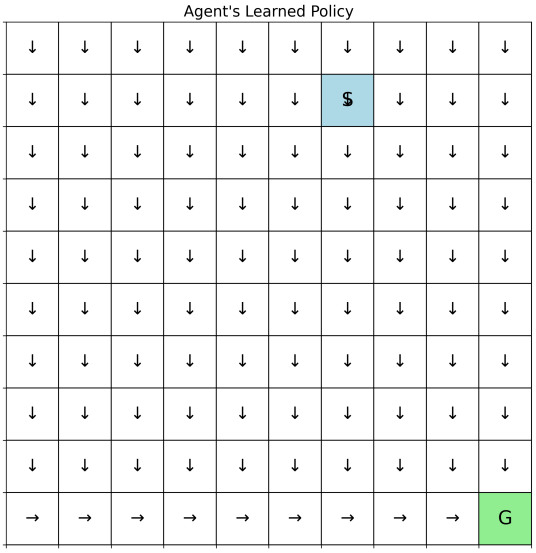

Figure 8: Policy Iteration, γ=0.3

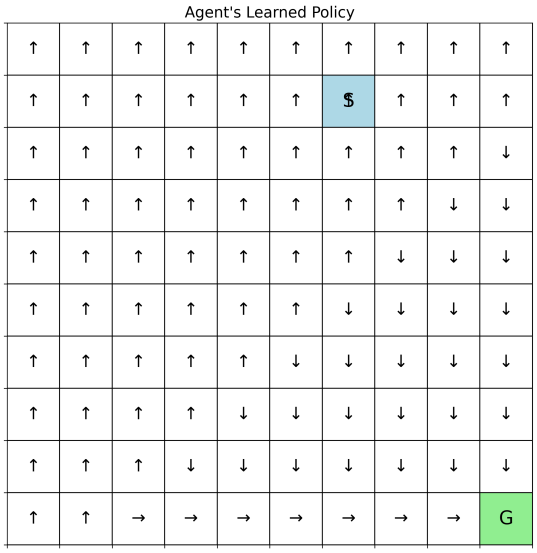

Figure 9: Value Iteration, γ=0.1

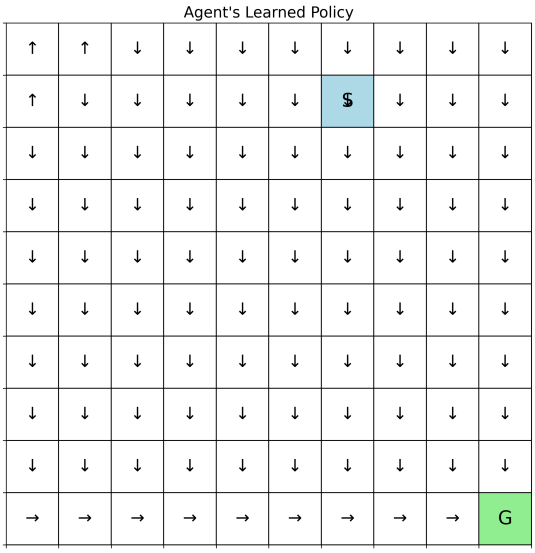

Figure 10: Policy Iteration, γ=0.1

Stage 2: Obstacles Added

Convergence behaviour:

Discount factor	Iterations	Reward	Converged ?	Correct Policy ?
0.9	19	-11	Yes	Yes
0.7	19	-11	Yes	Yes
0.5	19	-11	Yes	Yes
0.3	13	-11	Yes	Yes
0.1	7	-100	Yes	No

Table 3: Convergence behavior of Value Iteration

Discount factor	Iterations	Reward	Converged ?	Correct Policy ?
0.9	19	-11	Yes	Yes
0.7	19	-11	Yes	Yes
0.5	19	-11	Yes	Yes
0.3	19	-11	Yes	Yes
0.1	17	-11	Yes	Yes

Table 4: Convergence behavior of Policy Iteration

Observations:

- Both algorithms successfully avoided obstacle cells. Did not note any difference in convergence speed when compared to the previous stage.
- Value iteration with γ=0.1 did not learn the optimal policy.
- In other cases, the agent learned the shortest path while still avoiding the obstacles.

Learned Policies

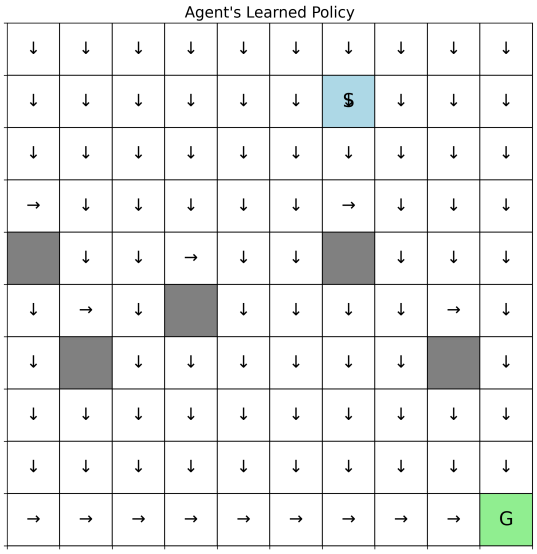

Figure 11: Value Iteration, γ=0.9

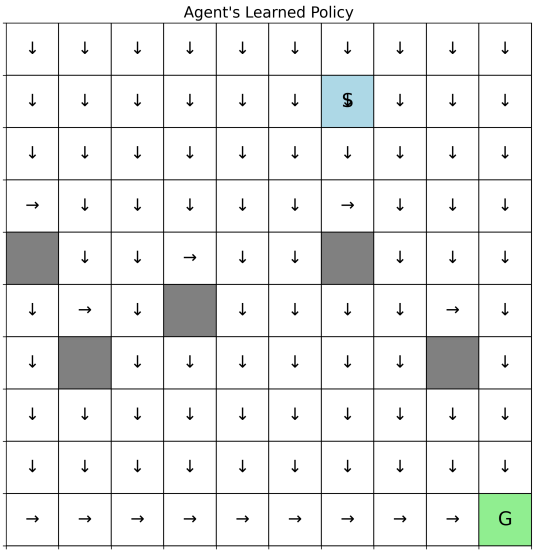

Figure 12: Policy Iteration, γ=0.9

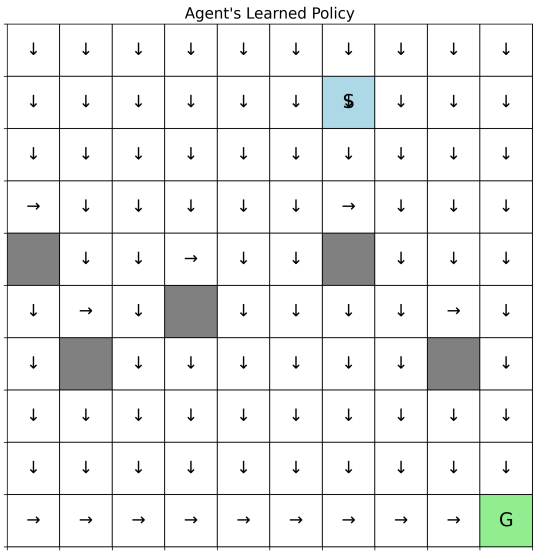

Figure 13: Value Iteration, γ=0.7

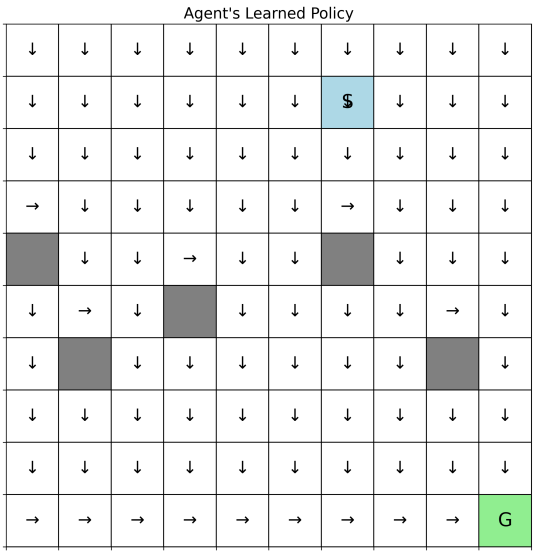

Figure 14: Policy Iteration, γ=0.7

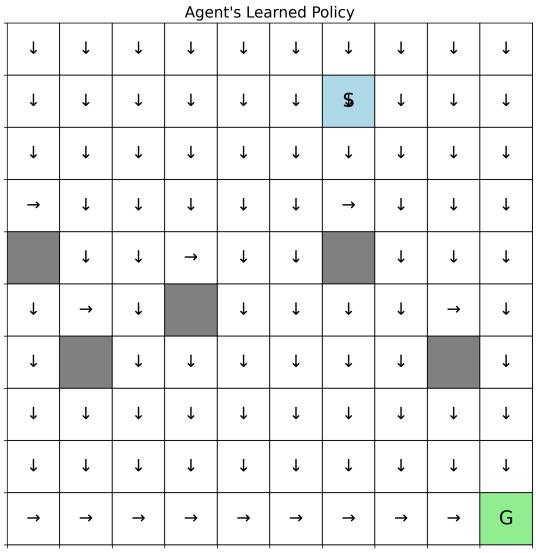

Figure 15: Value Iteration, γ=0.5

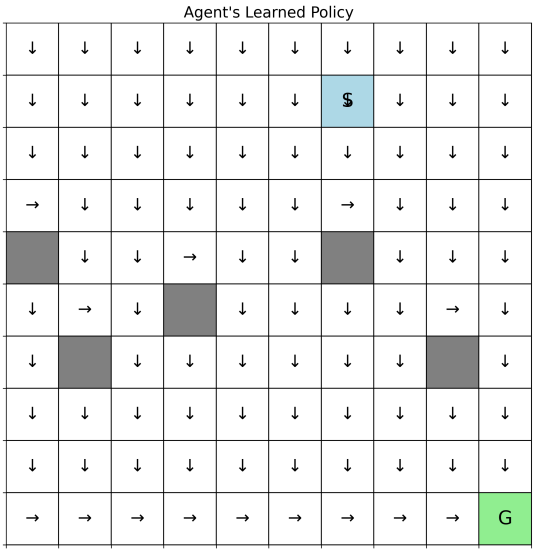

Figure 16: Policy Iteration, γ=0.5

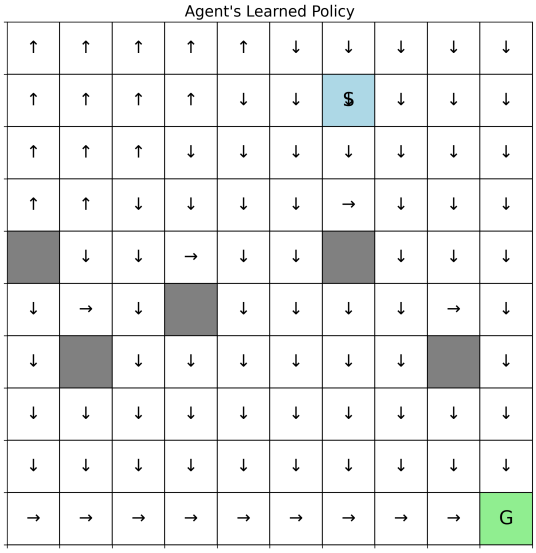

Figure 17: Value Iteration, γ=0.3

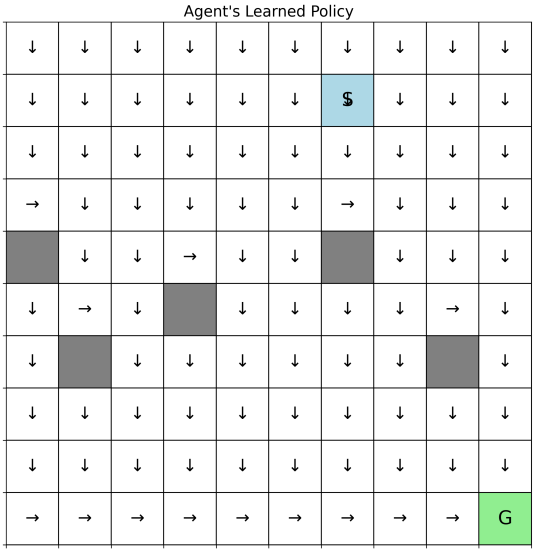

Figure 18: Policy Iteration, γ=0.3

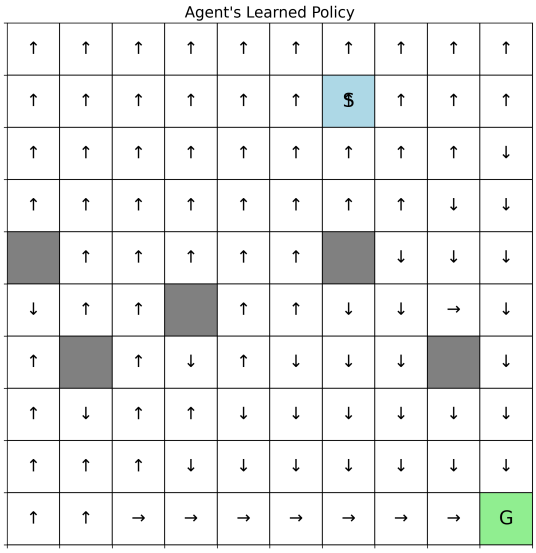

Figure 19: Value Iteration, γ=0.1

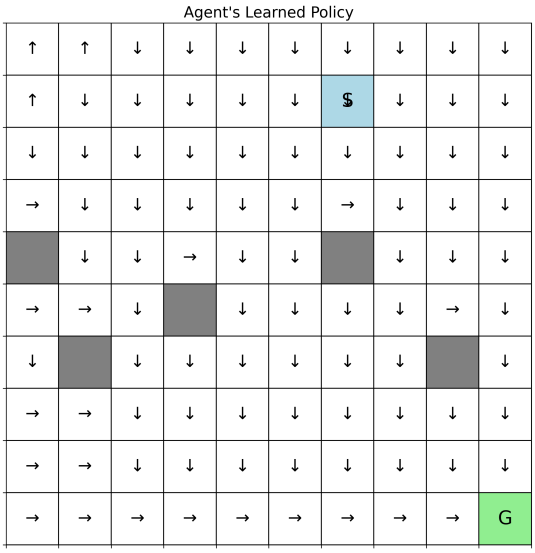

Figure 20: Policy Iteration, γ=0.1

Stage 3: Rainy Day (Stochastic Slips)

Convergence behaviour:

Slip prob	Theta	Discount factor	Iterations	Reward	Converged ?
0.0001	1e-6	0.8	21	-11	Yes
0.001	1e-6	0.8	92	-11	Yes
0.01	1e-3	0.8	229	-11	Yes

Table 5: Convergence behavior of Value Iteration

Slip prob	Theta	Discount factor	Iterations	Reward	Converged ?
0.0001	1e-6	0.8	10	-11	Yes
0.001	1e-6	0.8	9	-11	Yes
0.01	1e-3	0.8	235	-11	Yes

Table 6: Convergence behavior of Policy Iteration

Observations:

- As slip probability increased, convergence slowed down noticeably.
- Value Iteration remained more stable, while Policy Iteration oscillated.
- For slip = 0.01, both Value Iteration and Policy Iteration converged with higher θ tolerance only (1e-3).

Learned Policies

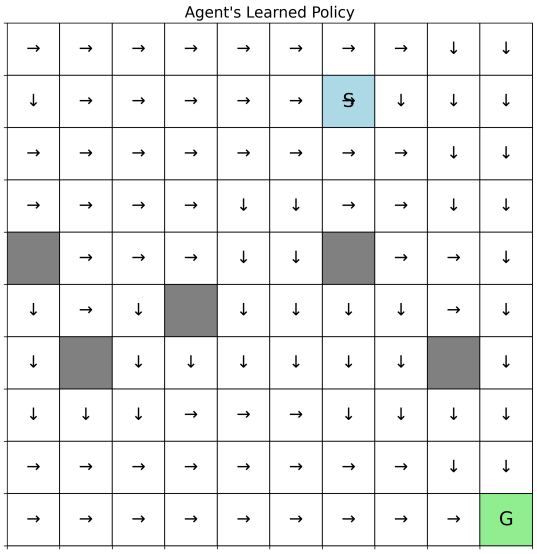

Figure 21: Value Iteration, slip=0.0001

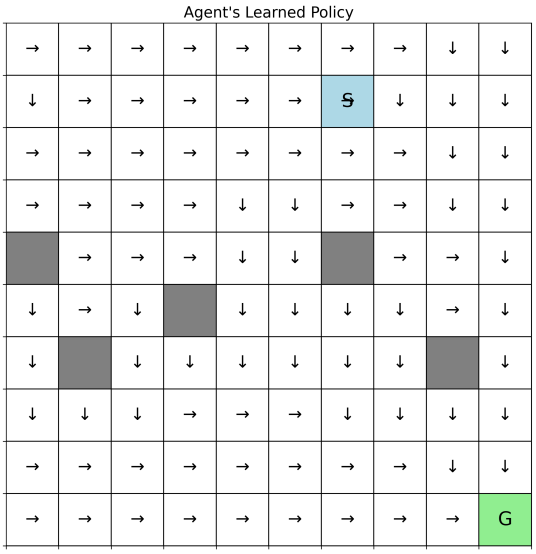

Figure 22: Policy Iteration, slip=0.0001

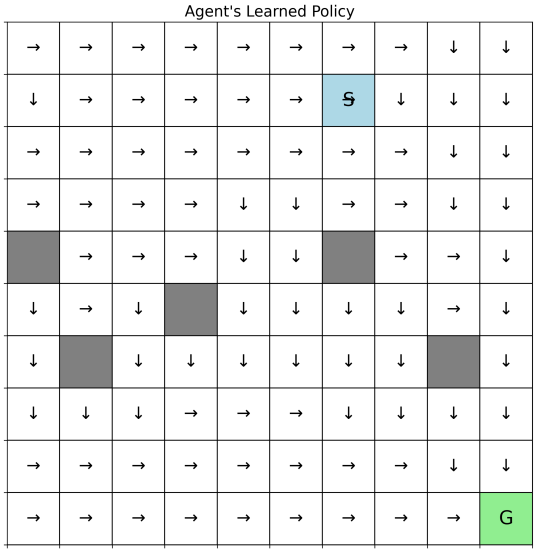

Figure 23: Value Iteration, slip=0.001

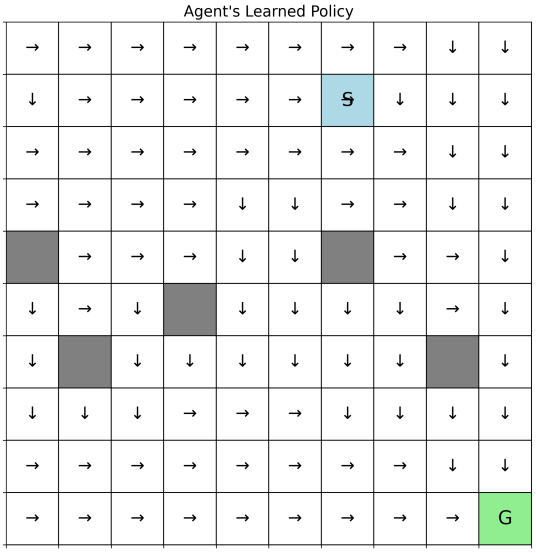

Figure 24: Policy Iteration, slip=0.001

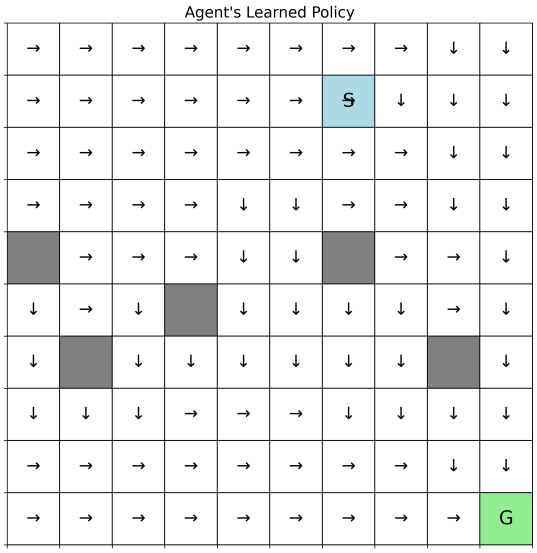

Figure 25: Value Iteration, slip=0.01

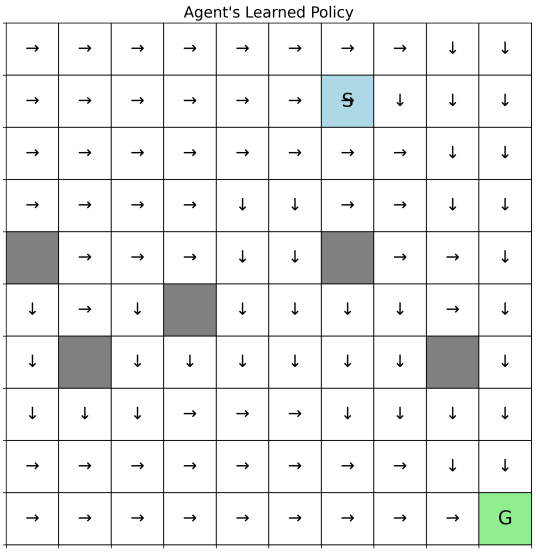

Figure 26: Policy Iteration, slip=0.01

3. Insights

- **Discount factor (γ):** Lower γ values led to faster but greedier convergence; higher γ yielded more optimal long-term paths.
- **Obstacles:** Forced exploration of alternative routes; both algorithms handled penalties well.
- **Stochasticity:** Introduced instability; tuning θ improved convergence for noisy environments.
- **Performance:** Policy Iteration converged faster in deterministic cases, while Value Iteration (with higher γ) was more robust under uncertainty.

4. Summary

- Both algorithms produced optimal paths in all stages except value iteration with lower γ.
- Value Iteration (with higher γ) offered better stability with noise and uncertainty.
- Policy Iteration was efficient when transitions were deterministic.
- Increasing complexity (obstacles + rain) demonstrated how dynamic programming scales to uncertain environments.