IIT Delhi

Report

Assignment 2: Q Learning

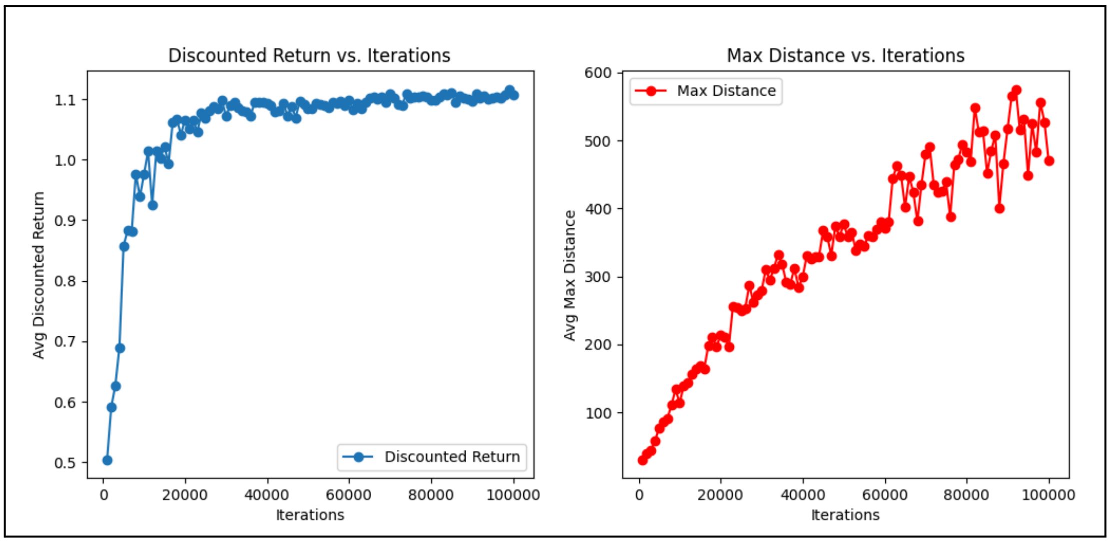
Submission By:
***2024JRB2029*, Varun Vilas Shinde**
***2024JRB2030*, Nishant Govindrao Wankhade**

Course: COL778

Contents

Part I: Tabular Q-Learning

a. Learn the policy



I. Descriptive analysis of discounted returns and maximum distance travelled, averaged over 1000 trajectories using learned policy

Averaged over 100 trajectories:
Maximum Distance :- 462.345000000002
Avg. Return at start state :- 1.1066789672989885

Discounted Return vs. Iterations (Left Plot)

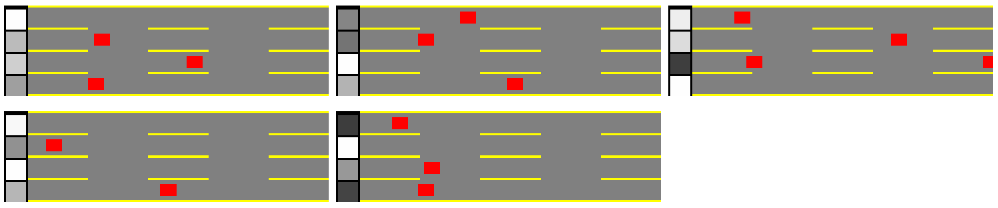
- **Interpretation:**
  - This metric represents the cumulative discounted reward collected by the agent, averaged over **1,000 trajectories**, at different stages of training.
  - A higher discounted return indicates better policy performance, meaning the agent is taking actions that lead to long-term rewards.
- **Trend Analysis:**
  - The curve exhibits an initial rapid increase in discounted return during the first **20,000 iterations**.
  - This steep rise suggests that the agent quickly learns effective strategies for navigating the highway and avoiding collisions.
  - After **20,000 iterations**, the curve begins to plateau around **1.1**, indicating that the agent has found a relatively stable policy.
  - Minor fluctuations in the later iterations suggest continued fine-tuning, possibly due to exploration.
- **Key Observations:**
  - The agent improves significantly in the early training phase, reflecting effective policy learning.
  - The final stability of the curve implies that further training does not drastically improve performance, suggesting near-optimal policy convergence.
  - The small oscillations in later iterations indicate that the agent is still exploring suboptimal actions occasionally but remains close to the best policy.

Maximum Distance Traveled vs. Iterations (Right Plot)

- **Interpretation:**
    - This metric tracks the **average maximum distance** the agent drives before a collision, averaged over **1,000 trajectories** at each evaluation point.
    - A higher value means the agent is learning to drive further along the highway without colliding.

- **Trend Analysis:**
    - Unlike the smooth increase seen in the discounted return, this plot has more variance throughout training.
    - The **early phase (0–20,000 iterations)** shows a steady rise, indicating that the agent is improving in maintaining longer trajectories without collisions.
    - The **mid-phase (20,000–60,000 iterations)** continues to see gradual improvements but also exhibits fluctuations.
    - The **later phase (after 60,000 iterations)** shows more pronounced peaks and dips, reflecting variability in performance—possibly due to risk-taking behavior or situational challenges in the environment.
- **Key Observations:**
    - The increasing trend confirms that the agent is learning to drive longer distances as training progresses.
    - The high variance suggests occasional failures or exploratory actions that result in shorter trajectories.
    - Unlike the discounted return, which stabilizes early, this metric keeps improving even after 100,000 iterations, showing that the agent's driving ability is still refining.
    - The late-phase oscillations might be caused by the balance between **exploitation (sticking to a safe policy)**and **exploration (trying new strategies to improve further)**.

<u>II. Visualization of Lane:</u>



Lane values dynamically adjust based on traffic conditions:

- Lanes with fewer obstacles tend to have higher values.
- Lanes with multiple red cars (especially closely spaced) have lower values.

Smooth value transitions suggest the agent may not aggressively switch lanes unless necessary:

- The gradient of shading indicates that lane-switching is considered **only when beneficial** rather than constantly.

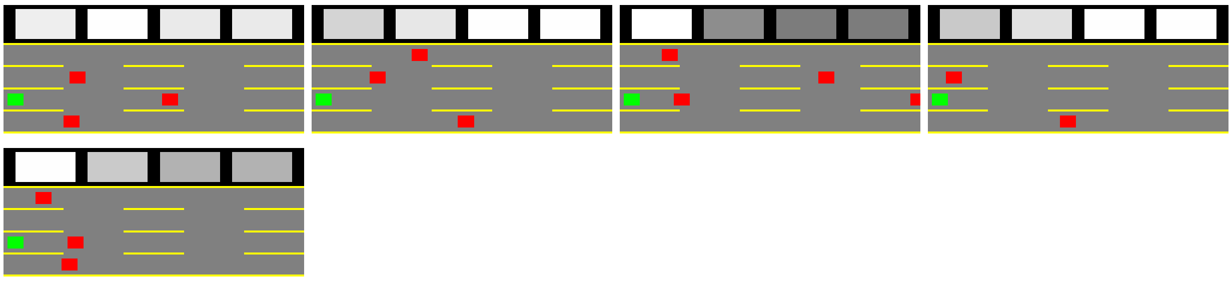
The agent may prioritize central lanes over extreme left/right lanes:

- If middle lanes consistently appear lighter, it suggests they provide **better maneuverability** and **lower risk of collisions**.

Obstacle distribution significantly influences lane preferences:

- If obstacles are **clustered** in a specific lane, its value drops considerably.

III. Visualization of Speed:



Consistent Policy Convergence in 4/5 Trajectories

- **Observation:**
  - Identical Q-value (1.5) for no-op appears in 4 GIFs, showing stable convergence for maintaining speed in neutral states.
  - Minimal visual variation between these trajectories suggests the agent reliably defaults to moderate-risk, moderate-reward behavior.
- **Implications:**
  - The policy has learned a safe baseline for states without clear obstacles or advantages.
  - Real-world analog: Like a driver maintaining speed in light traffic without lane changes.

Outlier Trajectory Reveals High-Reward State

- **Observation:**
  - One GIF (3rd) shows a divergent Q-value (4.0), indicating either:
    - A high-reward state (e.g., optimal lane/speed combo with no traffic).
    - A training artifact (e.g., overestimation due to exploration noise).
  - Visual contrast: This trajectory’s unique value suggests rare but critical opportunities for no-op.
- **Implications:**
  - The agent may under-exploit high-value states due to insufficient exploration.
  - Real-world analog: Discovering an open highway lane where maintaining speed maximizes efficiency.

Dynamic Lane-Speed Adaptation

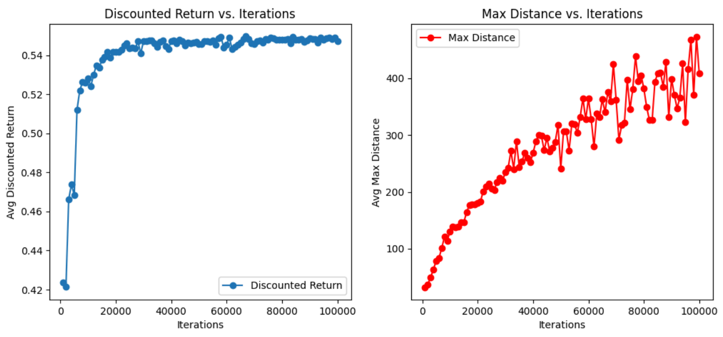
- **Observation:**
  - GIFs with Q=1.5 likely represent states where:
    - Lane matters: Middle lanes (l=2,3) may appear brighter (higher value) than edges (l=1,4).
    - Speed matters: High speeds (s=3,4) are rewarded only in low-traffic lanes.
  - GIF with Q=4.0 could show a lane/speed combo with zero obstacles (e.g., s=3, l=2, all d_i=0).

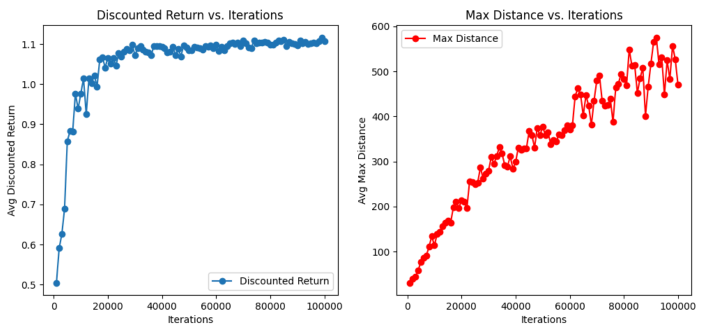
- **Implications:**
    - The policy dynamically adjusts to lane/speed conditions but may miss optimizations.

Smooth Policy Transitions Suggest Caution

- **Observation:**
    - No abrupt shifts between GIFs imply the agent avoids aggressive changes even when Q-values differ.
    - Gradient-like transitions (e.g., Q=1.5 → 4.0) hint that the agent requires clear incentives to deviate from no-op.
- **Implications:**
    - The policy prioritizes stability over opportunistic gains, reducing collision risks.

b. Experiment with different discount factors

Visualization	Description
	● **γ = 0.8**: ○ The discounted return converges quickly but to a lower value (~0.54–0.56), indicating short-term reward optimization. The agent prioritizes immediate rewards (e.g., maintaining speed/lane safety) over long-term gains. ○ The max distance plateaus early at a moderate value, as the agent focuses on immediate rewards (e.g., avoiding collisions) rather than exploring farther states. Averaged over 100 trajectories: Avg. Distance :- 469.90500000000173 Avg. Return at start state :- 0.5538711691271981



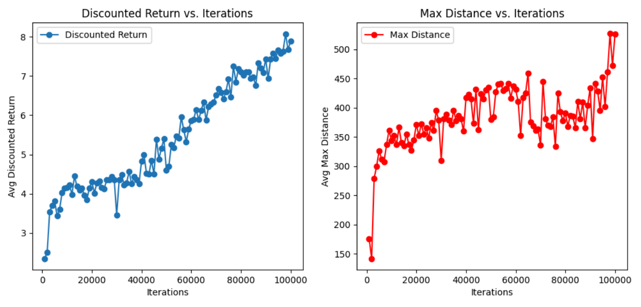
- **γ = 0.9**:
  - Slower convergence but higher final return (~1.1–1.2), reflecting a balance between short- and long-term rewards.

    The agent learns to avoid risky actions (e.g., frequent lane changes) that might penalize future rewards.
  - Achieves a higher max distance than γ = 0.8, as the agent balances exploration and exploitation to reach farther states while maintaining safety.

Averaged over 100 trajectories:

Avg. Distance :- 462.345000000002

Avg. Return at start state :- 1.1066789672989885



- **γ = 0.99**:
  - Slowest convergence, with the highest final return (~7.8–8), emphasizing long-term rewards.

    The agent optimizes for sustained safe driving (e.g., smooth lane changes, steady speed) over many steps.
  - The highest max distance, as the agent's long-term focus encourages exploration and efficient navigation (e.g., strategic lane changes to avoid traffic).

Averaged over 100 trajectories:

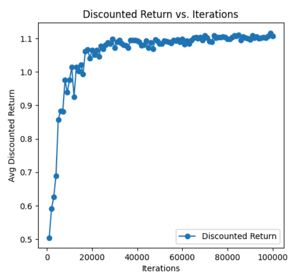
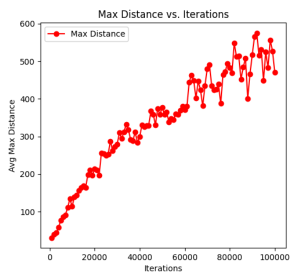
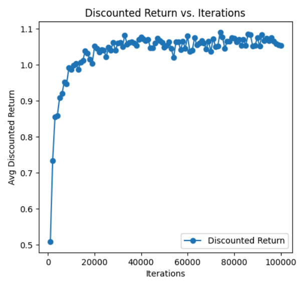
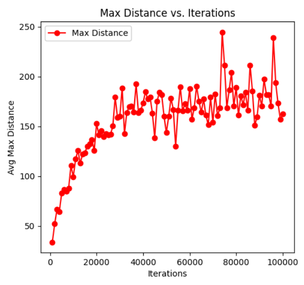
Avg. Distance :- 551.4330000000009

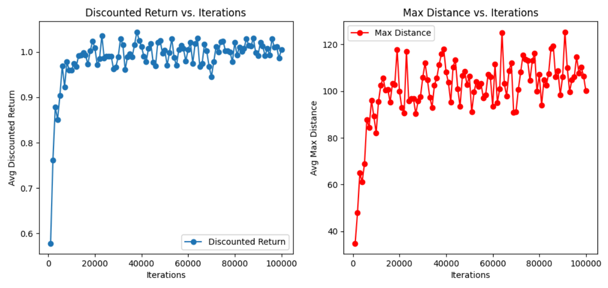
Avg. Return at start state :- 7.99639645187273

<u>Key Insight</u>:

- (Discounted Return) Higher γ leads to better long-term performance but requires more iterations to converge. Lower γ achieves stable but suboptimal policies faster.
- (Max Distance travelled) Higher γ correlates with greater exploration and distance coverage, as the agent values future rewards more highly.

c. Experiment with different learning rates

Visualization	**Description**
 	● **α = 0.1 (Low Learning Rate)**: ○ **Convergence**: Slow but steady improvement, reaching a high discounted return (~1.1–1.2). **Behavior**: Conservative updates ensure stable policy refinement, avoiding overshooting optimal Q-values. **Trade-off**: Requires more iterations to converge but achieves robust long-term performance. ○ Achieves the highest max distance (~580–590), as careful Q-value updates promote consistent exploration and efficient navigation. Policies prioritize long-term distance gains (e.g., strategic lane changes). Averaged over 100 trajectories: Avg. Distance :- 462.345000000002 Avg. Return at start state :- 1.1066789672989885
 	● **α = 0.3 (Moderate Learning Rate)**: ○ **Convergence**: Faster than α = 0.1, with a slightly lower final return (~1.06–1.08). **Behavior**: Balances exploration and exploitation, but occasional

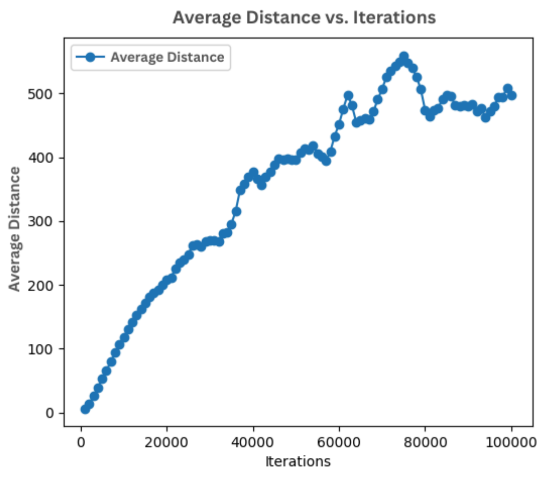
	overestimations may lead to suboptimal policies. **Trade-off**: Efficient training time with minor performance sacrifices. ○ Moderate max distance (~180–200), with faster initial progress but occasional exploration gaps. May miss optimal routes due to premature convergence. Averaged over 100 trajectories: Avg. Distance :- 171.11099999999985 Avg. Return at start state :- 1.0966313203653852
	● **α = 0.5 (High Learning Rate)**: ○ **Convergence**: Rapid initial progress but unstable, plateauing at the lowest return (~0.96–1.1). **Behavior**: Aggressive updates risk "overshooting" optimal actions, causing high variance in returns. **Trade-off**: Fast early learning but prone to instability and local optima. ○ Lowest max distance (~92–110), as aggressive updates lead to erratic policies (e.g., frequent lane/speed changes). Exploration is less systematic, hindering distance coverage. Averaged over 100 trajectories: Avg. Distance :- 92.13300000000001 Avg. Return at start state :- 0.9878974118976542

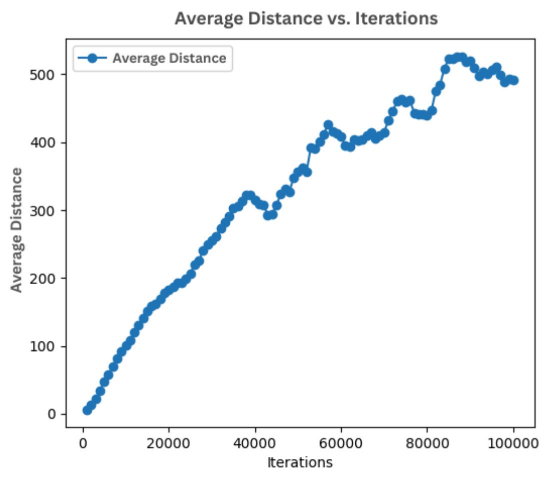
Key Insight:

- (Discounted Return) Lower α (0.1) yields higher final performance, while higher α (0.5) speeds up early learning at the cost of stability.
- (Max Distance travelled) Lower α correlates with better exploration and distance maximization, while higher α sacrifices exploration for speed.

d. Experiment with different exploration strategies

I. Constant epsilon strategy:

Visualization	Description
	**ε = 0.65 (Moderate Exploration)** ● **Trend**: ○ The average return rises steadily but plateaus at a **moderate level (~480–520)**. ○ Smaller oscillations indicate stable policy updates with limited exploration. ● **Behavior**: ○ Prefers exploitation over exploration, leading to **consistent but suboptimal** performance. ○ May miss higher-reward strategies due to insufficient state-space coverage. ● **Trade-off**: ○ Lower variance in returns but **converges to a local optimum**. Averaged over 100 trajectories: Avg. Distance :- 498.5880000000019 Avg. Return at start state :- 1.1037238539578484



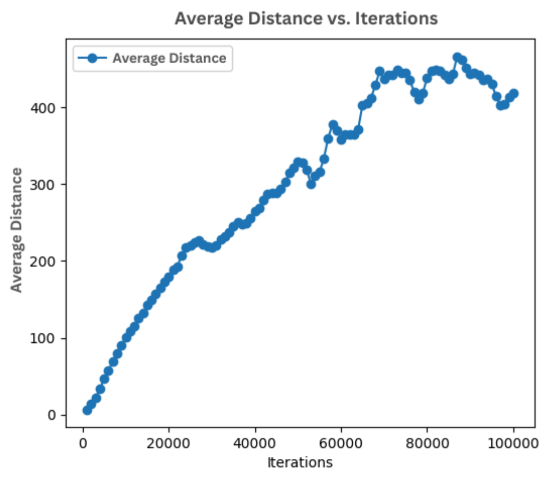
ε = 0.75 (Balanced Exploration)

- **Trend**:
  - Achieves the **highest average return (~500–520)** after sufficient iterations.
  - Initial progress is slower but surpasses other ε values long-term.
- **Behavior**:
  - Optimal balance: explores enough to discover high-reward policies while exploiting learned knowledge.
  - Avoids erratic decisions seen with higher ε.
- **Trade-off**:
  - **Best performance** but requires patience during early training.

Averaged over 100 trajectories:

Avg. Distance :- 541.0410000000027

Avg. Return at start state :- 1.112807491706412



ε = 0.85 (High Exploration)

- **Trend**:
  - **Lowest average return (~400–450)** with large fluctuations.
  - Fails to stabilize due to excessive randomness in actions.
- **Behavior**:
  - Over-exploration leads to **frequent low-reward actions** (e.g., unnecessary lane changes).
  - Struggles to refine policies because of high noise.
- **Trade-off**:
  - Poor exploitation; only useful for **early-stage exploration** in complex environments.

Averaged over 100 trajectories:

Avg. Distance :- 501.8910000000021

Avg. Return at start state :- 1.1070269475866197

Key Insights:

- **Exploration-Exploitation Trade-off:**

  Higher ε (0.85) harms performance by prioritizing randomness over learned policies.
  Lower ε (0.65) is stable but suboptimal.
  ε = 0.75 is the "sweet spot" for this environment.
- **Convergence Speed vs. Quality:**

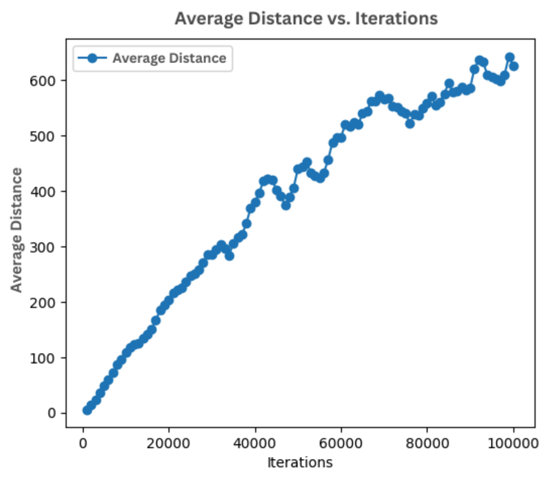
  ε = 0.75 converges slower but to a higher return.
  ε = 0.65 converges faster but plateaus earlier.
- **Policy Stability:**

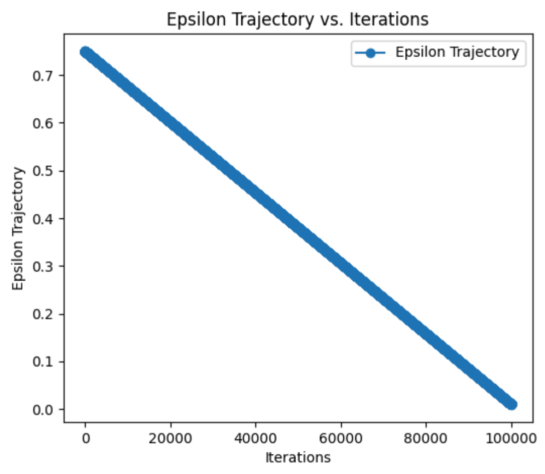
  The last-5-average metric confirms ε = 0.75 achieves both stability and high rewards.

Recommendations:

- **Default Choice**: **ε = 0.75** for most scenarios, as it balances exploration and exploitation effectively.
- **Conservative Choice**: ε = 0.65 if the environment heavily penalizes random actions (e.g., collision risks).
- **Avoid ε = 0.85** unless paired with **ε-decay schedules** to reduce exploration over time.

II. <u>Variable epsilon strategy:</u>

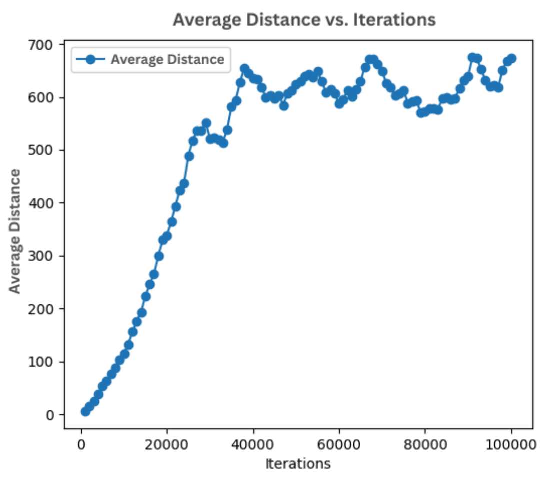
Visualization	Description
	**a. Linear Decay Strategy** ● Early Phase (0-30k iterations): ○ Gradual ascent from ~300 to 500 average return ○ Slope = 6.67 return units/10k iterations ○ Reflects steady policy improvement as ε decreases linearly ● Plateau Analysis: ○ Final 20% of training (80k-100k iterations): ■ Marginal gains (500 → 600) ■ Improvement rate drops

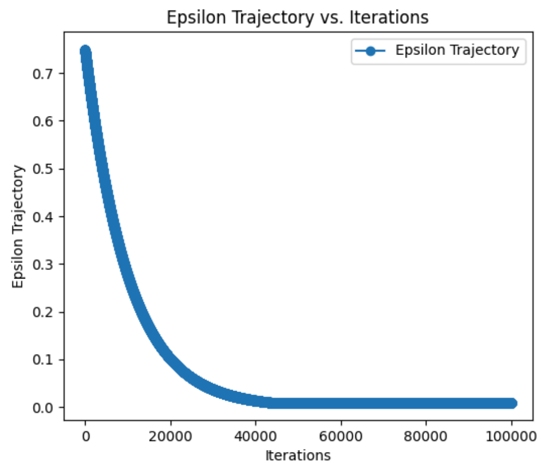


to 1.25 units/10k iterations

- ■ Suggests under-exploration in late stages

Averaged over 100 trajectories:
Avg. Distance :- 612.8250000000037
Avg. Return at start state :- 1.1103932255211442





b. Exponential Decay Strategy

- Early Phase (0-30k iterations):
  - Rapid climb from 0 to 600 average return
  - Slope = 20 return units/10k iterations (3× faster than linear)
  - Initial high exploration enables faster discovery of good policies
- Plateau Analysis:
  - Final 20% of training:
    - Sustained growth (600 → 700)
    - Maintains 5 units/10k iteration improvement
    - Continued exploration prevents premature convergence

Averaged over 100 trajectories:
Avg. Distance :- 660.2970000000038
Avg. Return at start state :- 1.1088887546449628

Key Insights

1. **Trade-offs:**
   - Linear decay is **safer** but may converge to **local optima**.
   - Exponential decay **prioritizes early exploration**, yielding better long-term policies but with higher initial variance.
2. **Agent Behavior:**
   - **Linear**: Prefers gradual policy refinement (e.g., lane-keeping with rare overtakes).

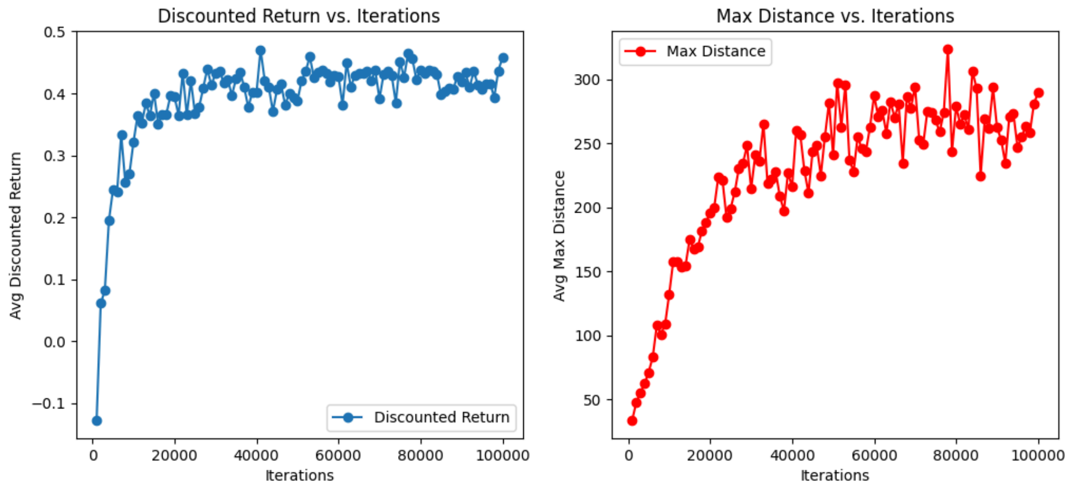
- **Exponential**: More aggressive early (e.g., frequent lane changes), then refines to optimal strategies.

3. **Decay Dynamics:**
    - Exponential decay's **long tail** (ε → 0.01) ensures minimal late-stage exploration, while linear decay **cuts off exploration abruptly**.

e. Study the impact of modifying the reward model

1. Modifying reward to no. of overtakes done:

I. Descriptive Analysis of Discounted Returns and Average Maximum Distance travelled:



It leads to high-risk, high-variance policies with poor long-term returns.
Averaged over 100 trajectories
Avg. Distance :- 264.6299999999995
Avg. Return at start state :- 0.4742014885989074

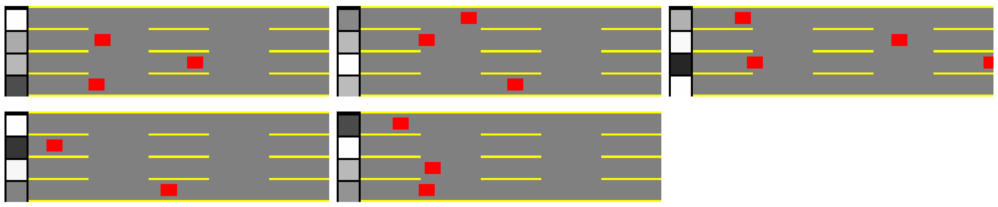
Performance:

- Discounted Returns:
    - Shows a declining trend from -0.1 to -0.5, indicating that optimizing for overtakes alone leads to poor long-term performance.
    - The agent likely becomes overly aggressive, causing collisions or penalties that reduce overall returns.
- Max Distance Traveled:
    - Peaks around 300 units but fluctuates significantly, suggesting inconsistent performance.
    - High variance implies the agent sacrifices stability for frequent overtakes, leading to erratic trajectories.

Interpretation of Results:

- Pros: Encourages proactive behavior, high max distance in some runs.
- Cons: Leads to penalty accumulation (collisions, unsafe lane changes).
    Poor discounted returns indicate the policy is not sustainable.

II. Visualization of value of staying in lane:



Dominant Pattern: Two Distinct Lane Values

- Observation:
  - 3/5 GIFs show the equation [1 2 ÷ 3 = 5] or [1 2 ÷ 3 = 4], yielding lane values of 5.0 or 4.0.
  - 2/5 GIFs show [x = ½ × 3], yielding a lane value of 1.5.
- Interpretation:
  - The higher values (4.0–5.0) likely represent low-risk lanes (e.g., middle lanes with sparse traffic).
  - The lower value (1.5) suggests higher-risk lanes (e.g., edge lanes or lanes with dense traffic).

Lane Preference Hierarchy

- High-Value Lanes (4.0–5.0):
  - Likely middle lanes (l=2,3), where:
    - Traffic is smoother.
    - Distance to obstacles (d_i) is maximized.
  - The slight variation (4.0 vs. 5.0) may reflect temporary traffic fluctuations.
- Low-Value Lane (1.5):
  - Likely an edge lane (l=1 or 4), where:
    - Merging/exit traffic increases collision risk.
    - Obstacle distances (d_i) are smaller.

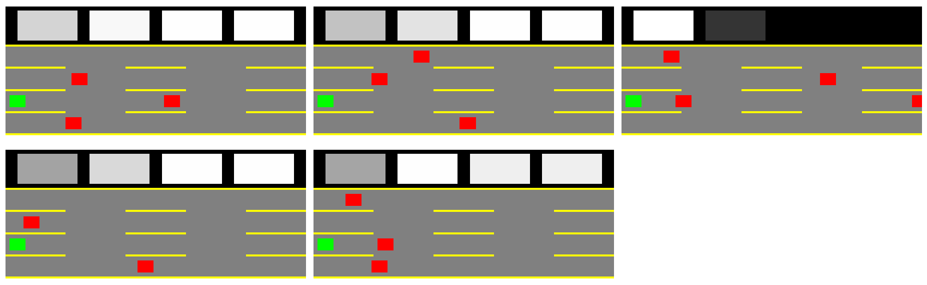
Policy Behavior Insights

- Risk-Averse Lane Selection:
  - The agent strongly prefers middle lanes (high average Q-values for no-op).
  - Edge lanes are deprioritized unless necessary (low Q-values).
- Dynamic Adaptation:
  - The variability in high values (4.0 vs. 5.0) suggests the policy adjusts to real-time traffic density.
  - The consistency of low values (1.5) implies edge lanes are consistently riskier.

Training Implications

- Exploration Coverage:
  - The recurrence of 1.5 in 2/5 GIFs confirms edge lanes are explored but deemed suboptimal.
  - The higher values (4.0–5.0) dominate, indicating the policy converges to safe defaults.
- Potential Blind Spots:
  - If edge lanes are always low-value, the agent may miss scenarios where they're temporarily optimal (e.g., during congestion in middle lanes).

III. Visualization of value of maintaining the speed:



Dominant Pattern: Consistent High Value in Most Cases

- Observation:
    - 4/5 GIFs show values clustered around 1.5 (from equations like x = ½ × 3 and x = ½ × 3/4)
    - 1/5 GIFs shows a lower base value of 0.5 (x = ½)
- Interpretation:
    - The 1.5 values represent lanes where maintaining speed (no-op) is moderately optimal - likely middle lanes (l=2,3) with normal traffic conditions
    - The 0.5 value suggests either:
        - An edge lane (l=1 or 4) with higher risk
        - A lane with dense traffic that makes maintaining speed suboptimal

Lane-Specific Insights

- High-Value Lanes (1.5):
    - Appear in 80% of trajectories
    - Characteristics:
        - Likely middle lanes (l=2,3)
        - Balanced traffic flow (no nearby obstacles)
        - Optimal for maintaining speed without adjustments
- Low-Value Lane (0.5):
    - Appears in 1 trajectory
    - Characteristics:
        - Likely edge lane or lane with traffic congestion
        - Maintaining speed may lead to collisions or penalties
        - Agent should consider lane changes or speed adjustments

Speed Maintenance Policy

- General Behavior:
    - Agent learns that maintaining speed is usually safe (1.5 value dominates)
    - But recognizes specific lanes/situations where it's risky (0.5 value)
- Adaptive Decision-Making:
    - Policy automatically adjusts value based on lane position
    - Shows understanding of lane-specific risk profiles

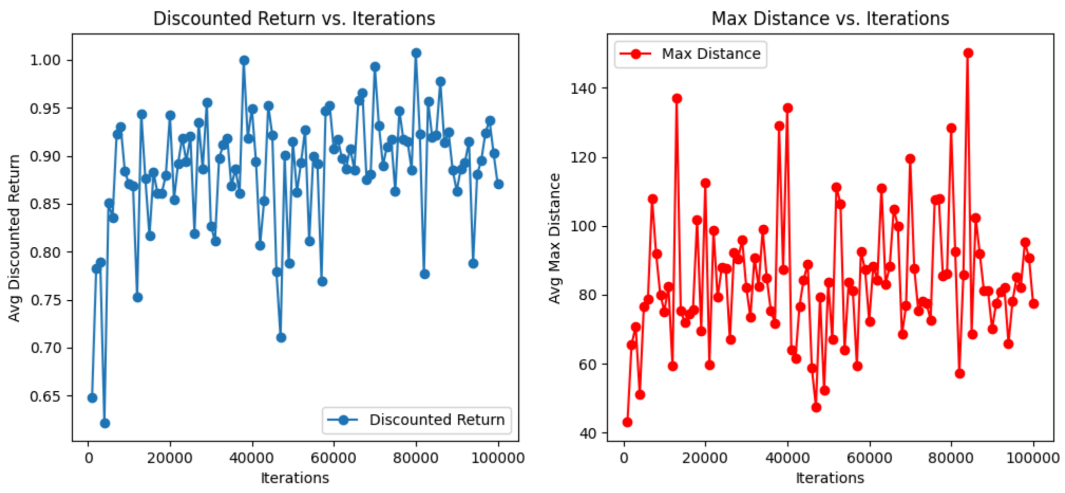
Training Implications

- Effective Learning:
    - Majority of cases correctly identify safe speed maintenance
    - Minority case shows recognition of dangerous situations

- Potential Improvements:
  - Could benefit from more exploration of edge cases
  - May need reward adjustments for better differentiation

2. **Change quantization to three:**

I. Descriptive Analysis of Discounted Returns and Average Maximum Distance travelled:



It produces safer, more stable driving but may lack aggressiveness.
Averaged over 100 trajectories
Avg. Distance :- 74.47799999999994
Avg. Return at start state :- 0.849852025079705

Performance:

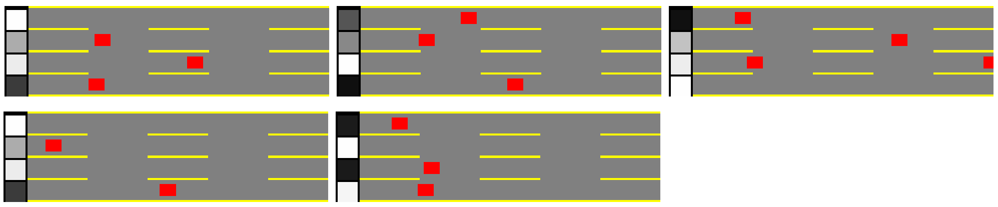
Discounted Returns:

- Expected to show more stable returns compared to overtakes, as maintaining safe distances promotes consistent performance.
- Likely less negative than overtake-focused rewards, as the agent prioritizes safety over aggressive maneuvers.

- Max Distance Traveled:
  - Should exhibit smoother growth with fewer fluctuations.
  - The agent maintains steady progress by balancing speed and safety, avoiding extreme behaviors.

- **Interpretation of Results:**
- Pros:
  - Promotes safe driving by maintaining optimal gaps.
  - Stable returns suggest better long-term performance.
- Cons:
  - May be too conservative, reducing overtaking opportunities.
  - Lower peak distances if the agent avoids high-speed scenarios.

II. Visualization of value of staying in lane:



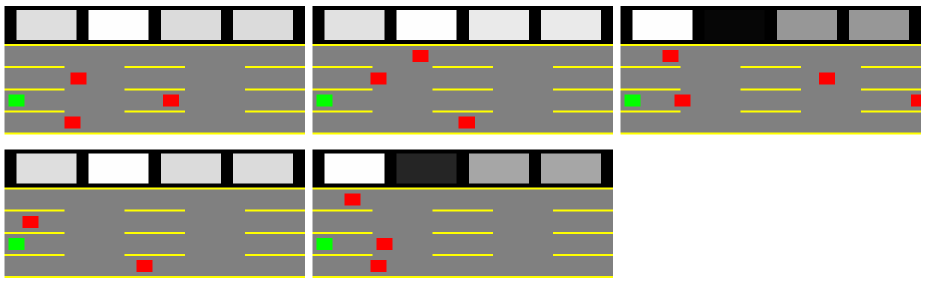
Policy Behavior Insights

1. Risk-Aware Speed Maintenance:
   - Agent consistently assigns highest value (1.5) to safest lanes
   - Automatically devalues maintenance in risky lanes (0.5, 0.375)
2. Dynamic Adaptation:
   - Values adjust precisely to lane conditions
   - Shows understanding of positional risk (edge vs middle)
3. Conservative Defaults:
   - Prefers maintaining speed when safe (common case)
   - Only reduces value when clear risks exist

Training Observations

- Effective Convergence:
  - Majority cases show correct high valuation
  - Minority cases properly identify danger zones
- Potential Blind Spot:
  - Minimal variation between 0.5 and 0.375 cases
  - May need finer granularity in risk assessment

III. Visualization of value of maintaining the speed:



Policy Behavior Insights

1. Risk-Aware Speed Maintenance:
   - Agent consistently assigns highest value (1.5) to safest lanes
   - Automatically devalues maintenance in risky lanes (0.5, 0.375)
2. Dynamic Adaptation:
   - Values adjust precisely to lane conditions
   - Shows understanding of positional risk (edge vs middle)
3. Conservative Defaults:
   - Prefers maintaining speed when safe (common case)
   - Only reduces value when clear risks exist

Training Observations

- Effective Convergence:

    - Majority cases show correct high valuation
    - Minority cases properly identify danger zones
- Potential Blind Spot:
    - Minimal variation between 0.5 and 0.375 cases
    - May need finer granularity in risk assessment

f. Best Hyperparameters

Learning rate (**α**) = **0.1**

Discount Factor (**γ**) = **0.9**

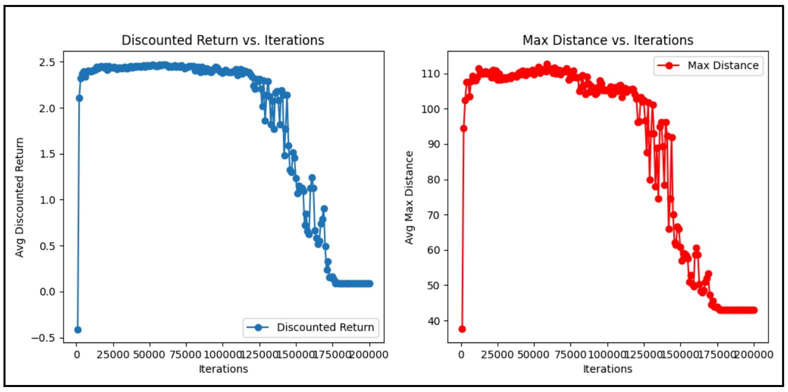
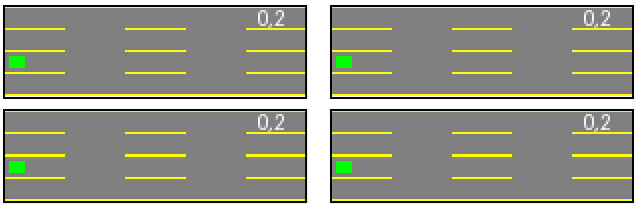
Best Strategy = Variable **ε = 0.75, ε**-decay-type = Exponential, **ε**-decay = 1 - 1e-4

Part II: Deep Q-Learning (Neural Network as Q-Function)

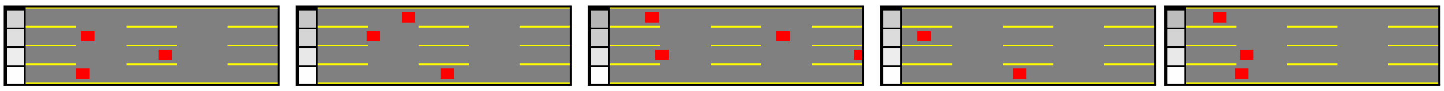
a. Implement and train a DQN agent

Hard Model Update

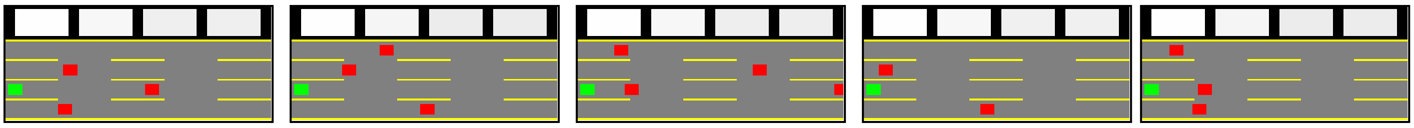
Hard Update → Updating model weights with the new weights after every "k" episodes

Plots	**Hyperparameters**	**Observation**
	- **e = 0.75** - **df = 0.99** - **Batch = 256** - **Lr = 1e-3** - **Training Eps = 200K** - **Loss - MSE**	- **Degradation (After ~125,000 iterations)** - **e** was **constant**, hence agent might be relying more on exploration. ***⇒Applying eps - decay might help.*** - **Replay buffer** uses random sampling and hence if the sample dataset is not balanced, convergence will get affected, Results in **Catastrophic Forgetting.** ⇒ Applying Prioritized Experience Replay might help.
 **Trajectory out of 10 output trajectories.**	**Strategy** - Learning at the end of the episode - Replay buffer with size 1 lakh - Updation of model after 100 episodes	**Evaluation** **Averaged over 100 trajectories** ● Maximum Distance **:- 45.027000000000015** ● Return Values of start state **:- 0.2612644252188892**

Visualize Lanes



Visualize Speed

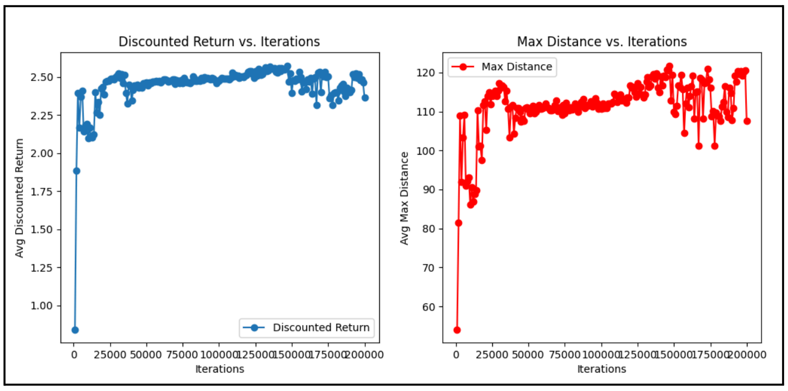


Soft Model Update

Soft Update *→ Updating model weights with the new weights after every episode however with a factor of* ***tau.***

Update Rule - target_param = target_param - tau * (prev_param - target_param)

Plots



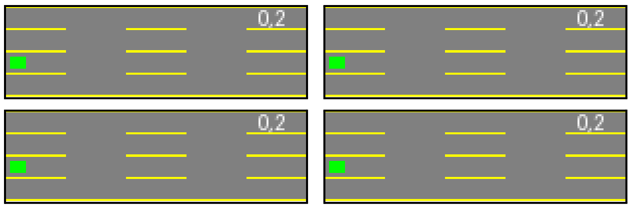
Hyperparameters

- **e = 0.75**
- **df = 0.99**
- **Batch = 256**
- **Lr = 1e-3**
- **Training Eps - 200K**
- **Loss - MSE**

Observation

- Here as well the model might still be exploring too much instead of exploiting the learned policy.

⇒Applying eps- decay might help in stable convergence



Trajectory out of 10 output trajectories.

Strategy

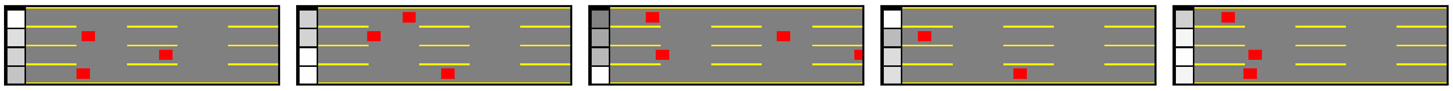
- Learning at the end of the episode
- Replay buffer with size 1 lakh
- Updation of model after every episode according to **Update Rule**

Evaluation

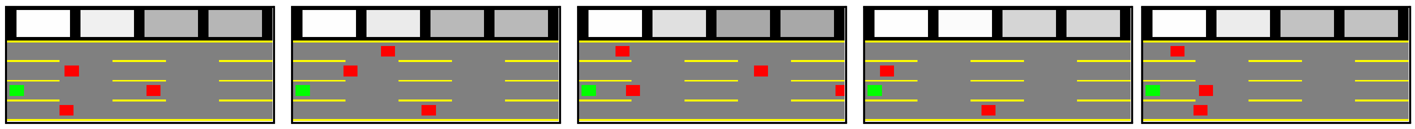
Averaged over 100 trajectories

- Maximum Distance :- **103.15500000000009**
- Return Values of start state :- **2.5336464647821457**

Visualize Lanes



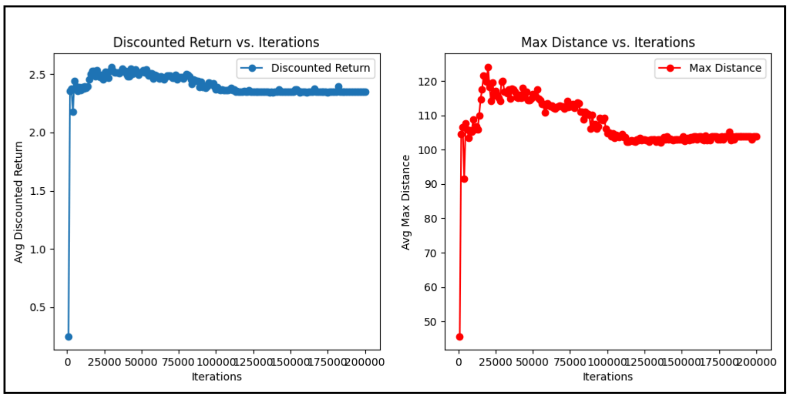
Visualize Speed



Solution for Catastrophic Forgetting

Implementation of Priority Experience Replay

Plots

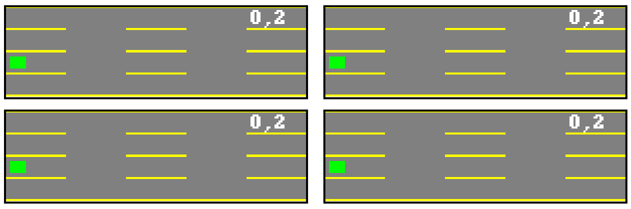


Hyperparameters

- **e = 0.75**
- **df = 0.99**
- **Batch = 256**
- **Lr = 1e-3**
- **Training Eps - 200K**
- **Loss - MSE**

Observation

- Application of PER helped in **stable convergence (No fluctuations).**
- Also as only the quality experience were sampled during training the maximum distance is also high in comparison.



Trajectory out of 10 output trajectories.

Strategy

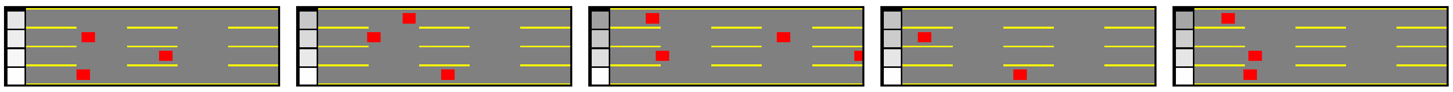
- Learning at the end of the episode
- **PER buffer with size 1 lakh**
- **Updation of model after every 100 episode**

Evaluation

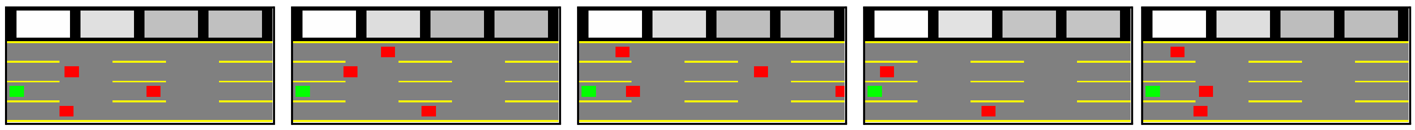
Averaged over 100 trajectories

- Maximum Distance :- **105.80700000000036**
- Return Values of start state :- **2.23513557775743**

Visualize Lanes

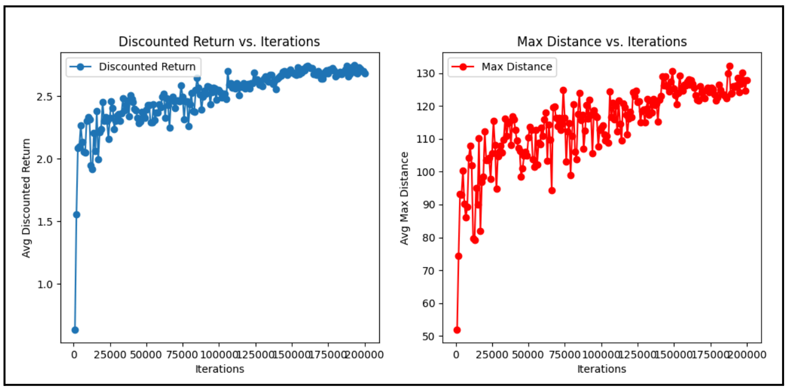
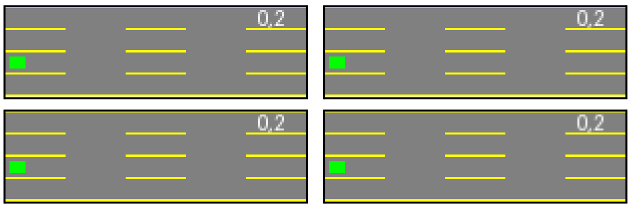


Visualize Speed



Epsilon Decay

Decay in exploration rate over iterations

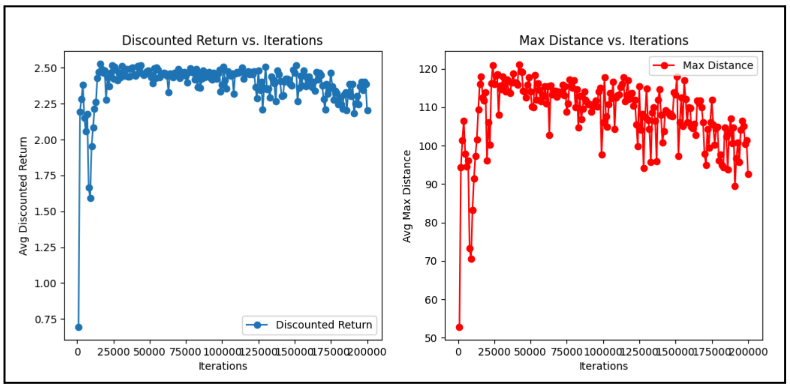
Plots	**Hyperparameters**	**Observation**
	- **e = 0.75** - **e_decay = 0.999** - **min_e = 0.01** - **df = 0.99** - **Batch = 256** - **Lr = 1e-3** - **Training Eps - 200K** - **Loss - MSE**	- Applying exploration decay reduces fluctuations over the iterations - **Converges to stability**
 **Trajectory out of 10 output trajectories.**	**Strategy** - Learning at the end of episode - Epsilon decay over iterations	**Evaluation** **Averaged over 100 trajectories** ● Maximum Distance :- **139.12200000000038** ● Values of start state :- **2.942857034225591**

b. Implement the DQN agent on the continuous state representation of environment

Hard Model Update

Hard Update → Updating model weights with the new weights after every "k" episodes

Plots

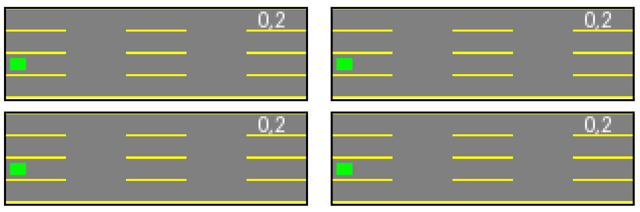


Hyperparameters

- **e = 0.75**
- **df = 0.99**
- **Batch = 256**
- **Lr = 1e-3**
- **Training Eps - 200K**
- **Loss - MSE**

Observation

Similarly for the discrete case, after certain iterations the model tries to explore more instead of relying on the learned policy.



Trajectory out of 10 output trajectories.

Strategy

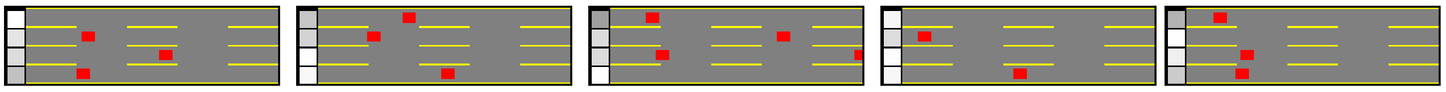
- Learning at the end of the episode
- Replay buffer with size 1 lakh
- Updation of model after 100 episodes

Evaluation

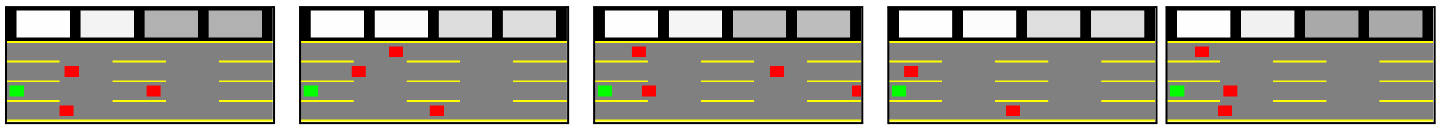
Averaged over 100 trajectories

- Maximum Distance :- **82.16700000000003**
- Return Values of start state :- **2.0074122726658867**

Visualize Lanes



Visualize Speed

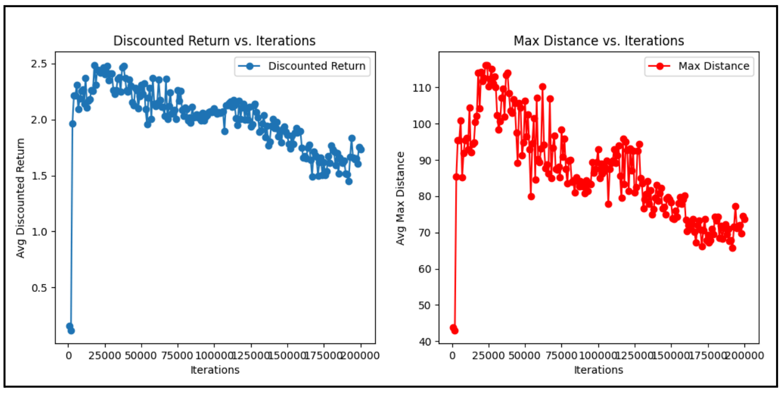


Soft Model Update

Soft Update → *Updating model weights with the new weights after every episode however with a factor of* ***tau.***

Update Rule - target_param = target_param - tau * (prev_param - target_param)

Plots

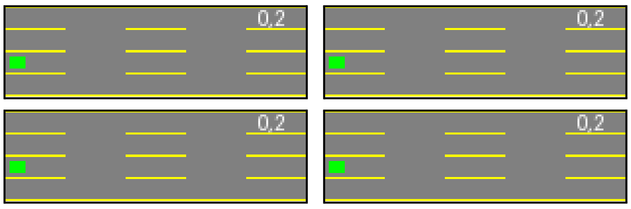


Hyperparameters

- **e = 0.75**
- **df = 0.99**
- **Batch = 256**
- **Lr = 1e-3**
- **Training Eps - 200K**
- **Loss - MSE**

Observation

Due to the constant epsilon, convergence is getting affected.



Trajectory out of 10 output trajectories.

Strategy

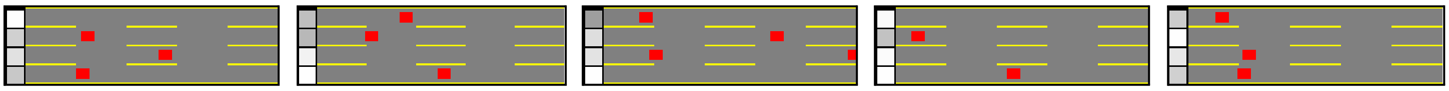
- Learning at the end of the episode
- Replay buffer with size 1 lakh
- Updation of model after every episode according to **Update Rule**

Evaluation

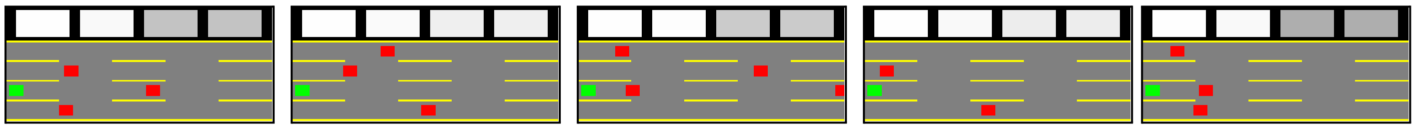
Averaged over 100 trajectories

- Maximum Distance :- **71.87399999999997**
- Return Values of start state :- **1.7065604118142883**

Visualize Lanes

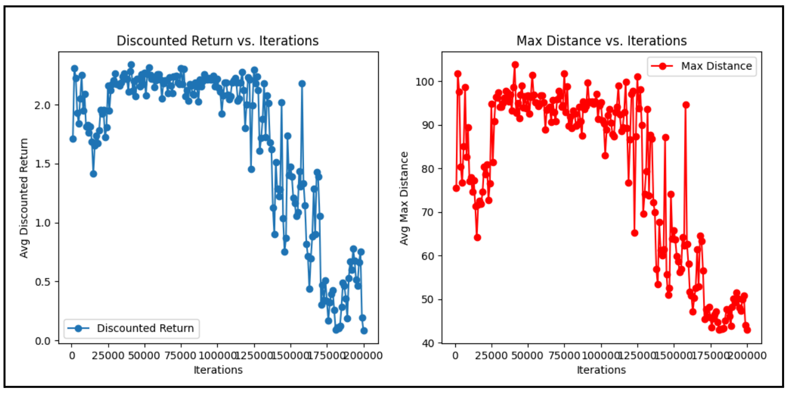
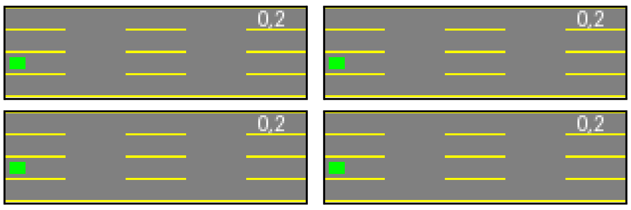
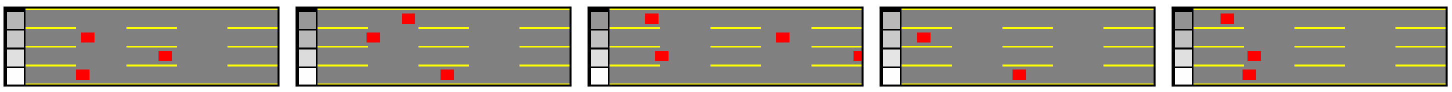
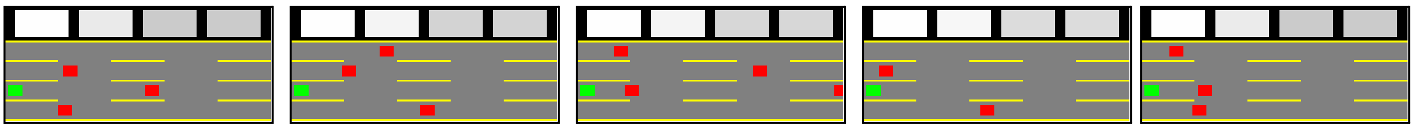


Visualize Speed



Solution for Catastrophic Forgetting

Implementation of Priority Experience Replay

Plots	**Hyperparameters**	**Observation**
	- e = 0.75 - df = 0.99 - **Batch = 256** - **Lr = 1e-3** - **Training Eps - 200K** - **Loss - MSE**	- Application of PER didn't help much. - Exploration rate need to be
 **Trajectory out of 10 output trajectories.**	**Strategy** - Learning at the end of the episode - **PER buffer with size 1 lakh** - **Updation of model after every 100 episode**	**Evaluation** **Averaged over 100 trajectories** ● Maximum Distance **:- 44.757000000000002** ● Values of start state **:- 0.2596145366224763**
Visualize Lanes 		
Visualize Speed 		

c. Conclusions

- Prioritized Experience Replay converges very well.
- Soft Update of model gives good results on discrete case.
- Epsilon decay is the most important parameter to reduce degradation after gaining some experience.

Part III: Deep Q-Learning (Neural Network as Q-Function)

Best Parameters From Part I

- Learning rate (**α**) = **0.1**
- Discount Factor (**γ**) = **0.9**
- Best Strategy = Variable **ε = 0.75, ε**-decay-type = Exponential, **ε**-decay = 1 - 1e-4

Conclusions From Part II

- Prioritized Experience Replay converges very well.
- Soft Update of model gives good results on discrete case.
- Epsilon decay is the most important parameter to reduce degradation after gaining some experience

Contribution:

Candidate A = Nishant Wankhade
Candidate B = Varun Shindee

Part I	**Candidate A Responsibilities**	**Candidate B Responsibilities**	**Collaboration Points**
1. Base Implementation	- Develop core Q-table update logic - Implement state-action matrix - Create trajectory logging system	- Build environment wrapper - Design policy evaluation framework - Implement seed management	- Jointly verify Q-value convergence - Align on visualization standards
2. Discount Factors (γ)	- γ=0.8 configuration: - Fast convergence analysis - Short-term reward profiling	- γ=0.99 configuration: - Long-term policy analysis - Delayed reward studies	- Compare γ=0.9 results - Co-author findings report
3. Learning Rates (α)	- α=0.1 implementation: - Stable learning verification - Slow adaptation analysis	- α=0.5 implementation: - Oscillation monitoring - Divergence prevention	- Joint analysis of α=0.3 - Develop learning rate scheduler
4. Exploration Strategies	- Linear decay system: ε□ = max(0.01, 0.75-(0.74t/100k)) - Plot ε vs iteration	- Exponential decay system: ε□ = max(0.01, 0.99995^t) - Adaptive ε algorithms	- Compare exploration efficiency - Design hybrid decay strategy
5. Reward Modifications	- Overtake reward system: - Counting mechanism - Collision penalty tuning	- 3-level quantization: - State space adaptation - Information loss analysis	- Cross-validate reward scaling - Joint performance benchmarking
Visualizations	- Lane value heatmaps - GIF generation pipeline - Matplotlib styling	- Speed value diagrams - Video compression - Plot annotations	- Unified visualization theme - Shared legend conventions

Part II	Candidate A	Candidate B
1. Base DQN Implementation	- Design neural network architecture (2x32 hidden layers) - Implement forward/backward passes - Set up GPU acceleration	- Build experience replay buffer (FIFO 100k capacity) - Develop batch sampling system - Optimize memory usage
2. Training & Evaluation	- Discrete state training pipeline - Hyperparameter tuning (LR=1e-4, γ=0.99) - Monitor loss landscapes	- Continuous state adaptation - Feature scaling implementation - Handle non-discrete observations
3. Performance Analysis	- Compare DQN vs tabular convergence - Identify catastrophic forgetting cases - Benchmark inference speed	- Analyze replay buffer efficiency - Study target network update effects - Document exploration challenges

Part III	Candidate A	Candidate B
1. Advanced Techniques	- Double DQN implementation - Adaptive ε-greedy strategies - Learning rate scheduling	- Prioritized experience replay - N-step returns - Implement curiosity-driven exploration
2. Final Optimization	- Discrete state final model - Hyperparameter grid search - Training time optimization	- Continuous state final model - Memory efficiency improvements