EEE 598 – Reinforcement Learning in Robotics

Fall 2025

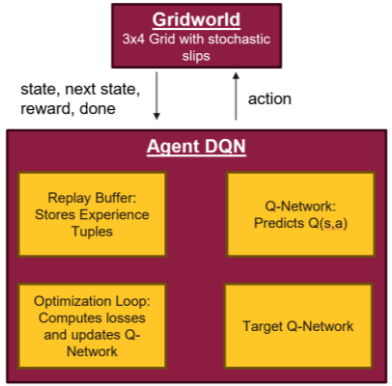
Assignment 2 – October 25th , 2025

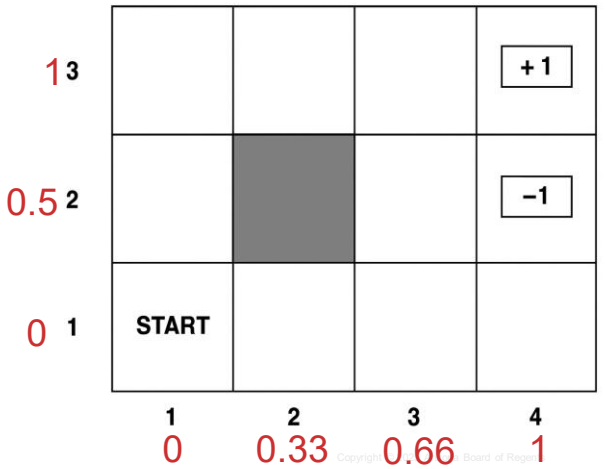
Eric Everett

CODE

Data Generation and Minibatch Training

- **Normalized state representation:** All environment states are normalized before being passed to the Q-network.
- **Transition collection:** Each environment step produces a tuple *(state, action, reward, next state, done)*.
- **Replay buffer:** These tuples are stored in a replay buffer to enable sample reuse and improve training.
- **Warm-up phase:** The buffer is first populated with random actions before the main training loop begins, new transitions are continually added as training proceeds.
- **Mini-batch sampling:** Each training iteration randomly samples 64 transitions from the buffer to update the Q-network.





Neural Network Design & Target Network

- **Network architecture:** The Q-network takes a **2-D normalized state vector** as input, passes it through **two hidden layers (64 neurons each)**, and outputs a **4-D vector of Q-values** corresponding to available actions.
- **Activation function:** ReLU is applied after each fully connected layer except the output, introducing nonlinearity without affecting the final value predictions.

- **Target network:** A **separate target network** with identical architecture is maintained to stabilize training.
- **Parameter synchronization:** The target network's weights are hard-updated every 500 steps to match the main Q-network parameters($\theta^- \leftarrow \theta$).

Neural Network Architecture:

Input layer $\in \mathbb{R}^2$

Hidden layer 1 $\in \mathbb{R}^{64}$

Hidden layer 2 $\in \mathbb{R}^{64}$

Output layer $\in \mathbb{R}^4$

Reinforcement Learning Hyperparameters:

Learning rate $\alpha = 1 \times 10^{-4}$

Discount factor $\gamma = 0.95$

Exploration rate $\epsilon = 0.9$

Number of episodes $N_{\text{ep}} = 2000$

Maximum steps per episode $T_{\max} = 50$

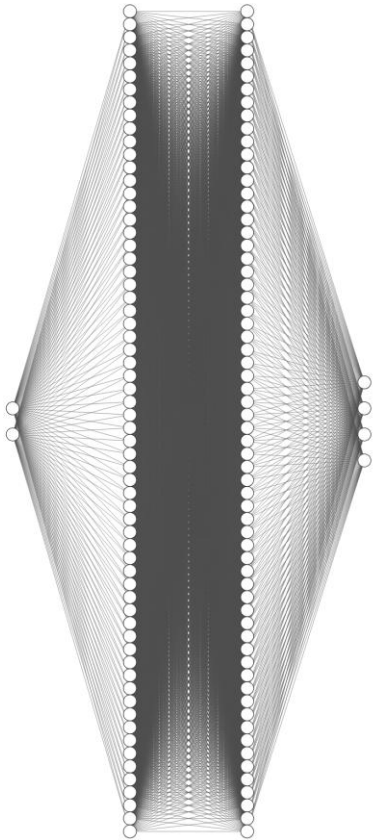
Exploration decay rate $\epsilon_{\text{decay}} = 0.995$

Minimum exploration rate $\epsilon_{\min} = 0.01$

Target network update interval $C = 100$

Replay warm-up steps $N_w = 100$

Gradient steps per environment step $G_s = 1$



Training Algorithm

- **Exploration strategy:** An ε-greedy policy is used for action selection, with a decaying exploration rate.
- **Target computation:** Targets are computed using the frozen target network:

$$y_i = r_i + \gamma(1 - d_i)\max_a(Q_{\bar{\theta}}(s_i', a_i))$$

- **Loss function:** The mean squared error (MSE) is used to minimize the difference between predicted and target Q-values:

$$L(\theta) = \frac{1}{B}\sum(Q_\theta(s_i, a_i - y_i)^2$$

- **Optimizer:** The Adam optimizer is employed, leveraging adaptive learning rates and momentum for efficient gradient descent.
- **Parameter update:** Backpropagation is performed to update network weights based on the computed gradients.
- **Stopping Conditions:** The stopping conditions are that there are 2000 episodes, with a maximum of 50 steps per episode. If during an episode done = True, then the episode will conclude and move to the next episode.
- **Evaluation of training:** Exploration is turned off, and model parameters do not update during evaluation runs to get final deployed model metrics.

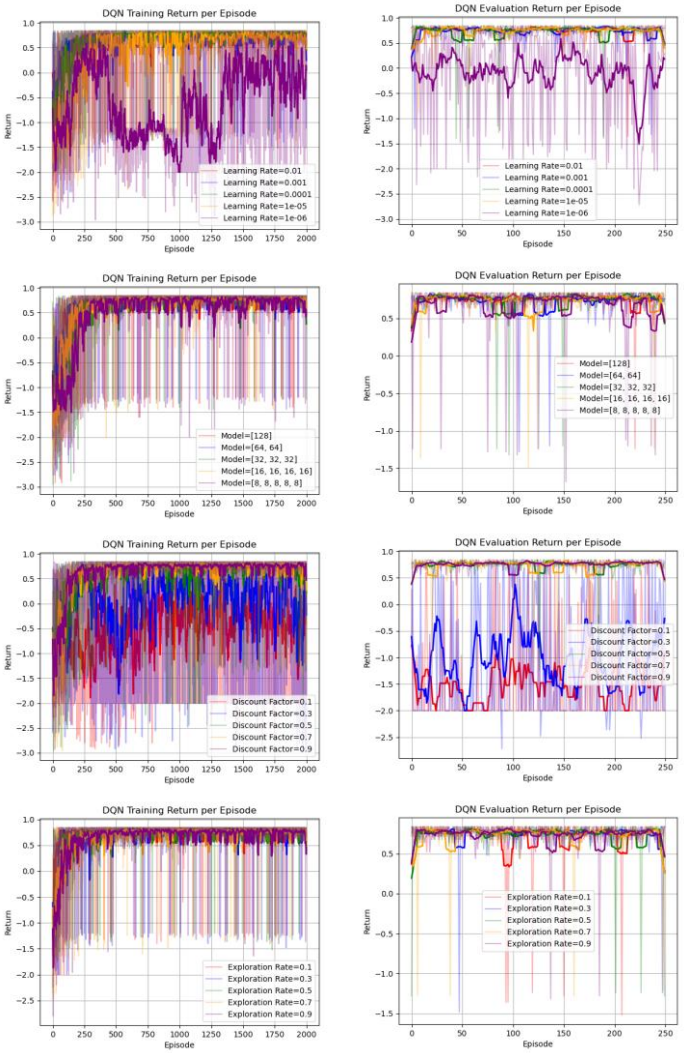
Algorithm 1 DQN Training

Require: Environment $\mathcal{E}$, Q-network $Q_\theta(s,a)$, target network $Q_{\theta^-}(s,a)$, replay buffer $\mathcal{D}$
Require: learning rate α, discount γ, exploration rate ϵ
Require: target update interval C, minibatch size B, warm-up steps N_w, total episodes N_{ep}
Require: Total reward R_{ep}, maximum steps per episode T_{max}

Initialize $\mathcal{D} \leftarrow \emptyset$, $\theta^- \leftarrow \theta$
Set random seed
for $t = 1$ to N_w **do** ▷ Replay warm-up with random actions
  Sample random a_t, observe $(s_t, a_t, r_t, s_{t+1}, done)$
  Store tuple in $\mathcal{D}$
end for
for episode $= 1$ to N_{ep} **do** ▷ Training loop
  Reset $\mathcal{E}$: $s \leftarrow (0,0)$, $R_{ep} \leftarrow 0$
  for step $= 1$ to T_{max} **do**
    Select $a \leftarrow \begin{cases} \arg\max_a Q(s,a), & \text{w.p. } 1-\varepsilon \\ \text{random}, & \text{w.p. } \varepsilon \end{cases}$
    Execute a in $\mathcal{E}$, observe and store $(s, a, r, s', done)$ in $\mathcal{D}$
    Sample minibatch $\{(s_i, a_i, r_i, s_i', d_i)\}_{i=1}^{B} \sim \mathcal{D}$
    $R_{ep} \leftarrow R_{ep} + r$
    Compute targets: $y_i = r_i + \gamma(1 - done_i)\max_{a'} Q_{\theta^-}(s_i', a')$
    Compute loss: $L(\theta) \leftarrow \frac{1}{B}\sum_{i=1}^{B}\left(Q_\theta(s_i, a_i) - y_i\right)^2$ ▷ MSE loss
    Update $\theta \leftarrow \theta - \alpha\nabla_\theta L(\theta)$ ▷ Adam Optimizer
    if step mod $C = 0$ **then** $\theta^- \leftarrow \theta$
    end if
    if *done* **then break**
    end if
  end for
  Decay $\epsilon \leftarrow \max(\epsilon \cdot \epsilon_{decay}, \epsilon_{min})$
  Record episode return and loss
end for
Set $\epsilon = 0$, freeze θ
for each eval episode **do** ▷ Evaluation loop
  Follow training loop without $\{\theta, \theta^-\}$ updates
  Record episode return and loss
end for
Plot loss and return curves

Varying Hyperparameters

- **Hyperparameter experiments:** Several examples of different hyperparameter configurations are shown (right), illustrating trends during both training and evaluation.
- **Additional results:** More hyperparameter tuning plots are available on the project's GitHub repository, including corresponding training and evaluation loss curves (omitted here for space).
- **Key sensitivities:** The learning rate and discount factor have the strongest influence on performance where extreme values for either lead to unstable or slow convergence.
- **Model selection:** Insights from these tuning experiments were used to determine the final hyperparameter values for the deployed model.



Final Model Results

- **Training and evaluation results:** The plots (right) show loss and return per episode for two random seeds for both training and evaluation. Additional seed experiments and performance plots are available in the project's GitHub repository.
- **Policy visualization:** The environment visualization below shows the agent's action choices per state. At grid locations (2, 3) and (1, 4), action choices differ across seeds, reflecting varying risk behavior near the high penalty region under stochastic slip conditions.

Trained Model Choices