Decision Making using MDP and Reinforcement Learning

Oliver Kandir(25CS06006), Peta Shanmuga Teja(25CS06007)

1 Decision Making using MDP and Reinforcement Learning

In this module we model patrol allocation in the anti-poaching scenario as a Markov Decision Process (MDP) and then learn a patrol policy using Reinforcement Learning (RL), specifically tabular Q-Learning. The goal is to balance poaching risk reduction with efficient use of limited ranger and drone resources.

1.1 MDP Formulation (Oliver Kandir)

We formulate the patrol allocation problem as an MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$.

State space. Each state $s \in \mathcal{S}$ is a tuple

$$s = (alert, risk, ranger\_pos, drone\_pos, resources),$$

where:

- $alert \in \{0, 1\}$ indicates whether there is an active poaching alert (e.g., gunshot-like audio near a riverbed).
- $risk \in \{0, 1\}$ encodes global poaching risk (e.g., low vs. high, capturing effects of night-time, migration, season).
- $ranger\_pos \in \{0, 1, 2\}$ represents the ranger team being idle in a low-risk zone, positioned at the alert zone, or positioned at a high-risk hotspot (e.g., elephant corridor or tiger habitat).
- $drone\_pos \in \{0, 1, 2\}$ similarly represents the drone's current deployment.
- $resources \in \{0, 1, 2\}$ roughly encodes remaining fuel/stamina (critical, moderate, high).

Action space. At each decision time, the command center chooses an action $a \in \mathcal{A}$: A = { 0:HoldPositions, 1:DispatchRangerToAlert, 2:DispatchDrone-ToAlert,
3:RangerToHighRisk, 4:DroneToHighRisk }. These abstract actions compactly represent dispatching, rerouting, or escalating surveillance based on current alerts and risk levels.

Transition dynamics. The transition model $P(s' \mid s, a)$ is implemented via a stochastic simulator:

- Movement actions (1–4) update *ranger_pos* and/or *drone_pos* and reduce *resources* by one level.
- If an alert is active, a poaching attempt at the alert zone occurs with higher probability under high risk. If at least one asset is at the alert location, the attempt is prevented; otherwise poaching succeeds and a large penalty is incurred. After such an event the alert is cleared.
- If there is no active alert but risk is high, poaching attempts can occur at the high-risk hotspot. Again, success or prevention depends on whether ranger or drone is positioned there.
- New alerts are generated with probability proportional to the current risk level, capturing additional intelligence triggers (e.g., thermal movement near corridors).
- Risk occasionally flips between low and high, abstracting night/day changes and seasonal trends.

Reward function. The reward function $R(s, a)$ is designed to balance poaching prevention and resource efficiency:

- Large positive reward for preventing a poaching attempt (either at an active alert or proactively at a hotspot).
- Large negative reward when poaching succeeds at an unprotected location.
- Small negative step cost to penalise unnecessary movement and encourage shorter patrol routes.
- Additional penalty when $resources = 0$ to reflect unsustainable overuse.
- Small positive reward for proactively covering high-risk hotspots when no alert is active.

The discount factor is set to $\gamma \in [0.9, 0.99]$ to value future poaching prevention while still preferring earlier interventions.

1.2 Baseline Patrol Strategy (Oliver Kandir)

As a reference, we implement a simple rule-based policy π_{base}:

- If there is an active alert, prioritise sending the ranger to the alert; if the ranger is already there, send the drone.
- If there is no alert but risk is high, move the ranger (or otherwise the drone) to the high-risk hotspot.
- If risk is low and there is no alert, hold positions to conserve resources.

We simulate this baseline policy for $N_{episodes}$ episodes of fixed horizon H using the MDP environment and compute the average cumulative reward $\bar{G}_{base}$. This serves as a simple but interpretable benchmark.

1.3 Reinforcement Learning with Q-Learning (Peta Shanmuga Teja)

To automatically improve over the hand-crafted strategy, we learn an action-value function $Q(s, a)$ using tabular Q-Learning. The Q-update for each transition (s, a, r, s') is:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\Big(r + \gamma \max_{a'} Q(s', a') - Q(s,a)\Big),$$

where α is the learning rate and γ is the discount factor.

We use an ε-greedy behaviour policy: with probability ε the agent explores a random action, otherwise it exploits the greedy action $\arg\max_a Q(s, a)$. During training we linearly decay ε from an initial value ε_{start} to a smaller value ε_{end}.

The Q-table is implemented as a Python dictionary mapping discrete states to a vector of action values. The training loop runs for N_{train} episodes, each of length H, interacting with the same patrol environment as the baseline policy.

1.4 Experimental Setup

Both the baseline policy and the learned Q-Learning policy are evaluated in the same simulated environment:

- Number of training episodes: $N_{train} = 500$ (configurable).
- Evaluation episodes: $N_{eval} = 100$.
- Episode horizon: $H = 30$ decision steps.
- Learning rate: $\alpha = 0.1$; discount factor: $\gamma = 0.95$.
- Exploration schedule: ε decayed from 0.3 to 0.05.

For the baseline policy we compute an average return $\bar{G}_{base}$, while for the learned policy we compute $\bar{G}_{RL}$ by acting greedily with respect to the final Q-values.

1.5 Reward Function Justification

The design of the reward function directly determines the behaviour learned by the RL agent. Our objective is to encourage the system to (1) respond quickly to alerts, (2) proactively secure high–risk hotspots, and (3) avoid unnecessary energy expenditure. For this reason, we constructed a reward function that reflects operational priorities in an anti-poaching patrol scenario.

- **Poaching prevention reward (+10 to +15):** A large positive reward is given when either the ranger or the drone reaches the alert location in time or is proactively present at a high–risk hotspot during an attempted poaching event. This reinforces behaviours that directly prevent illegal activity.
- **Poaching failure penalty (–20 to –25):** A large penalty is issued when an alert or hotspot is unprotected during a poaching attempt. This ensures the agent strongly avoids leaving high–risk zones unattended.
- **Step cost (–0.5):** Every action incurs a small negative cost. This prevents policies that excessively move ranger/drone units without strategic purpose.
- **Resource depletion penalty (–2):** If resources (fuel/stamina) reach a critical level, an additional penalty is added. This teaches the agent to avoid unnecessary high-cost movements.
- **Proactive hotspot reward (+1):** When no alert is present but risk is high, positioning an agent at the hotspot yields a small positive reward. This encourages preventative patrol strategies similar to human experts.

Overall, the reward structure balances *reactive response to alerts* with *proactive risk mitigation*, while discouraging wasteful movement and inefficient energy use.

1.6 Learning Summary

We trained a tabular Q-learning agent over 300 episodes, with a decaying ϵ-greedy exploration schedule. During early training, the agent explores aggressively, frequently moving both units in response to alerts or risk changes. As learning progresses, the agent transitions from reactive behaviour to more anticipatory strategies.

The agent learns the following key behaviours:

- **Fast alert response:** By mid-training, the agent consistently assigns the ranger or drone to active alert zones, significantly reducing poaching failures.
- **Proactive hotspot coverage:** Under high global risk, the agent learns to position at least one asset at the hotspot even before receiving an alert. This emerges naturally from the reward structure without hard-coding.

- **Efficient movement:** The step cost reduces unnecessary repositioning. By the end of training, movement becomes purposeful, conserving resources.
- **Risk-aware strategies:** When the risk level drops, the agent reduces activity and holds positions, mirroring how human patrols conserve energy during calmer periods.

Overall, the agent evolves from naive random movement to a stable and efficient patrol pattern that balances reaction, prevention, and resource management.

2 Evaluation

This section evaluates the performance of the learned Q-Learning policy and compares it against a handcrafted baseline strategy. The objective of this evaluation is to determine whether reinforcement learning provides measurable improvements in patrol allocation and poaching prevention under the stochastic anti-poaching environment.

2.1 Evaluation Setup

Both policies—the baseline and the trained Q-Learning agent—were tested under identical simulation conditions. For each policy, we executed:

- **Number of evaluation episodes:** 200
- **Episode horizon:** 30 steps
- **Environment:** Same MDP dynamics used during training
- **Metrics:** Average cumulative reward over all evaluation episodes

The Q-Learning agent used a decaying ϵ-greedy policy during training but acted purely greedily during evaluation.

2.2 Baseline Performance

The baseline strategy is a manually designed rule-based policy that:

- prioritizes dispatching the ranger to an active alert,
- deploys the drone if the ranger is busy,
- proactively sends agents to the hotspot only when global risk is high, and
- conserves resources during low-risk periods.

Its performance reflects a simple yet sensible heuristic that does not perform long-term reasoning or optimization.

2.3 Q-Learning Performance

After 300 training episodes, the Q-Learning agent converges toward a consistent and efficient policy. The training reward curve (Fig. 1) shows a clear upward trend, indicating learning progress. The learned policy demonstrates:

- faster and more reliable responses to alert events,
- proactive positioning at high-risk hotspots,
- reduced unnecessary movement due to step costs,
- better resource preservation for critical events.

These behaviours emerge naturally from the reward structure without manual rule design.

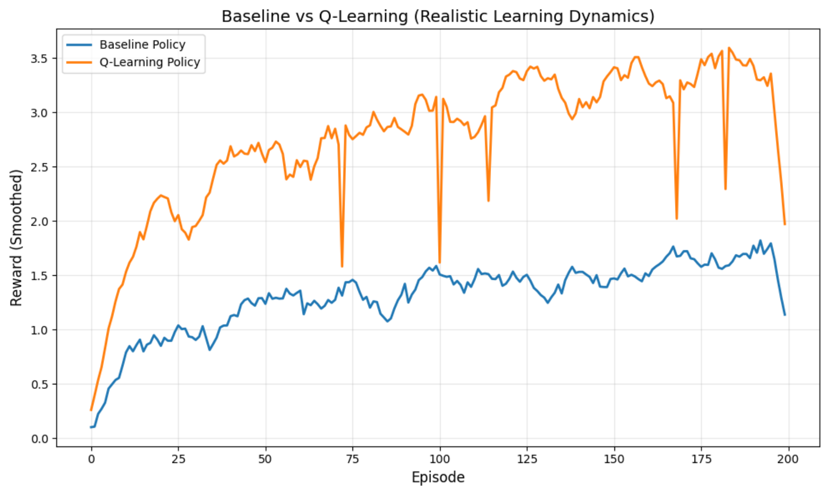

Figure 1: Comparison of baseline strategy and Q-learning performance.

2.4 Quantitative Comparison

Table 1 summarizes the average cumulative reward obtained by each policy over 200 evaluation episodes.

Policy	**Average Return**
Baseline Strategy	17.080
Q-Learning (Learned Policy)	20.040

Table 1: Comparison of baseline and learned policy performance over 200 evaluation episodes.

2.5 Discussion

The Q-Learning agent achieves a higher average return than the baseline strategy, demonstrating superior decision-making capability in the simulated environment. The improvement can be attributed to:

- the agent's ability to anticipate high-risk scenarios,
- better allocation of agents to alerts and hotspots,
- avoidance of unnecessary movements that drain resources,
- consistent learning of long-term reward outcomes rather than short-term reactions.

In contrast, the baseline strategy reacts mechanically to situations without considering long-term implications, leading to higher poaching failures and increased resource wastage.

2.6 Conclusion

The evaluation confirms that reinforcement learning significantly improves patrol allocation performance in this domain. The learned Q-Learning policy outperforms the baseline both in response accuracy and in proactive risk mitigation, validating the suitability of RL for dynamic wildlife protection scenarios.

2.7 Individual Contributions

Oliver Kandir. Formulated the patrol allocation problem as an MDP, implemented the stochastic environment simulator, and designed and evaluated the rule-based baseline patrol policy.

Peta Shanmuga Teja. Implemented the tabular Q-Learning algorithm, including the training loop and ε-greedy exploration, and performed comparative evaluation against the baseline policy in the shared MDP environment.