Policies Learned

The implemented algorithm learns a hierarchical policy using SMDP Q-learning and intra-option Q-learning. The macro-policy first selects an option (one of the four color-coded subgoals) based on the Q-table for high-level decisions. Then, a sub-policy (trained Q-table for each option) guides the taxi to the respective destination. The learned policy effectively minimizes the steps taken by choosing the best available option for reaching the passenger and then dropping them at the destination.

The Q-learning updates ensure that the taxi learns to navigate efficiently by iteratively updating the Q-values based on observed rewards. The final learned policy follows an optimal sequence of actions, first moving towards the passenger and then towards the destination while avoiding unnecessary movements.

Alternate Set of Options

Instead of using only navigation-based options (moving towards the four predefined colored locations), an alternative approach could involve options based on **state-dependent actions**:

1. **Navigate to Passenger**: A high-level action that directs the taxi to the passenger's current location, regardless of where they are.

2. **Pick Up Passenger**: Once at the passenger's location, this action picks them up.

3. **Navigate to Destination**: A high-level action that directs the taxi towards the given drop-off point.

4. **Drop Off Passenger**: When at the destination, the passenger is dropped off.

These four options are **mutually exclusive** with the current approach because they focus on passenger-based goals rather than purely location-based movement.

If both approaches (color-based navigation and state-based macro-actions) are implemented and compared:

- The **color-based approach** may be more generalizable to random passenger-destination pairs.

- The **state-based approach** is likely to be more efficient, as it directly optimizes passenger pickup and drop-off.

Comparing their performance would involve running Q-learning with both sets of options and analyzing total steps taken, convergence speed, and final reward values.

Comparison Between SMDP Q-Learning and Intra-Option Q-Learning

Observations and Improvements with Intra-Option Q-Learning:

1. **Intra-option learning allows updates within an option execution** rather than only at termination. This means Q-values for sub-policies (like navigating to a color) get updated while they are being executed.

2. **Faster learning**: Because intra-option Q-learning updates both the macro and micro Q-values in parallel, it converges faster than SMDP Q-learning.

3. **Smoother transitions**: Intra-option Q-learning enables partial policy execution before switching, whereas SMDP Q-learning strictly waits for termination before updating.

Intra-option Q-learning generally provides better efficiency by enabling simultaneous updates at both macro and micro levels. It improves convergence speed and allows the learned policy to adapt more flexibly to varying initial conditions.