Project Code: ANV2
Autonomous navigation with Discrete Car [Version 2]

Project Duration: 18-Aug-2024 ~~ 07-Sep-2024
Submission Information : (via) Google Form

Objective:

A car has to reach a desired xy-position (integers) and orientation (one of the eight directions N, S, E, W, NE, NW, SE, SW). It can select a velocity (1, 2, or 3) along with one of the steering actions: left, right or straight. Its orientation direction changes as per the steering action. E.g., if the previous orientation is 'E' and steers right, it changes to 'SE'. The position updates follows: x_next = x_prev + vel * x_sign, where x_sign = -1 if in {NW, W, SW}, 1 if in {NE, E, SE}, 0 if in {N, S} and y_next = y_prev + vel * y_sign, where y_sign = -1 if in {SW, S, SE}, 1 if in {NE, N, NW}, 0 if in {E, W}. Assuming limits on positions x and y to [-50, 50] be considered safe, build an agent using RL algorithms to help solve the problem. In particular, you shall do the following tasks:

1. Design an Environment Class that models the given specifications. You have to build the observation space and action space for the moves. Then, the reward structure will be designed following the structure of Gym Environment Classes. The reward structure can, however, change depending on the requirements of the algorithms.
2. Design a simple yet suitable visualisation to render the effects of the agent's interaction with the environment (using libraries such as matplotlib). It should clearly illustrate various features specific to your design environment and show the current state, action, and reward received.
3. Demonstrate the Policy Iteration (PI) and Monte-Carlo (MC) Learning algorithms using the Env class designed above. Also, run the Env with this Github repository algorithms (PI and MC).
4. Compare the results from the above algorithms and give insights on why the results are better or worse for one algorithm than others (only for your implementations). Also, visually compare the initial, intermediate, and final policies.

Bonus: *Report the maximum N for which the algorithms are in (4).*

Note: *The program can be written in Python, preferably, and should run on Linux.*

Submission Details: (to be submitted through Google Form)

ZIPPED Code Distribution [create separate Python file for the Env Class] A brief (2-4 page) report/manual of your work. Emphasise experiments, results, discussions, remarks/observations, and Machine configurations. (optional).

Submission Guidelines:

1. You should preferably use Python, which should run on a Linux Environment.
2. You should include a README file in the code distribution, which describes how to run the codebase and the essential files.
3. The submitted program file *should* have the following header comments: Group Number, Roll Numbers (Name of the member), Project Code, and Project Title
4. Submit through Google form only—link to form: https://forms.gle/G5Gts9MvWbVoRpmq5
5. You should name your file as <RollNo_ProjectCode.extension>. (e.g., 24CS10000_ANV2.zip for code and 24CS10000_ANV2.pdf for report)

You should not use any code available on the Web. Submissions found to be plagiarised will be awarded zero marks.

For any questions about the assignment, contact the following TA:
Suraj Singh (Email: suraj19995.ss@gmail.com)