Robotics-Reinforcement Learning with Function Approximation

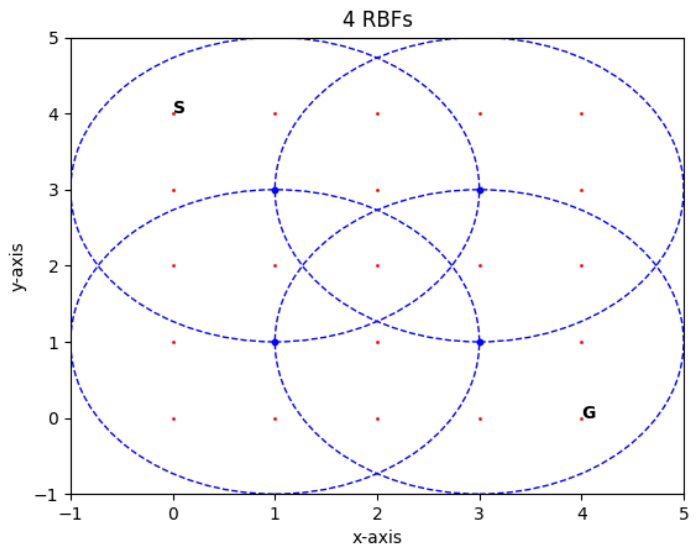
Submit the project report to cover all of the requirements.

Submit your source code with instructions to run your code.

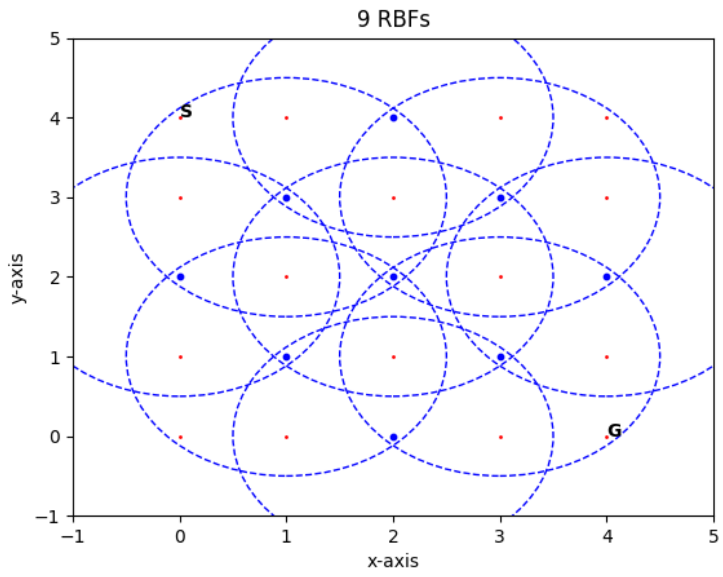
Project 2: Deadline 2/26/25

The problem is the same as Project 3a. Using Radial Basis Functions (RBF) to approximate the state space. You can use 4 RBF to approximate the state space. Something is similar to these figures.

4 RBF cover:

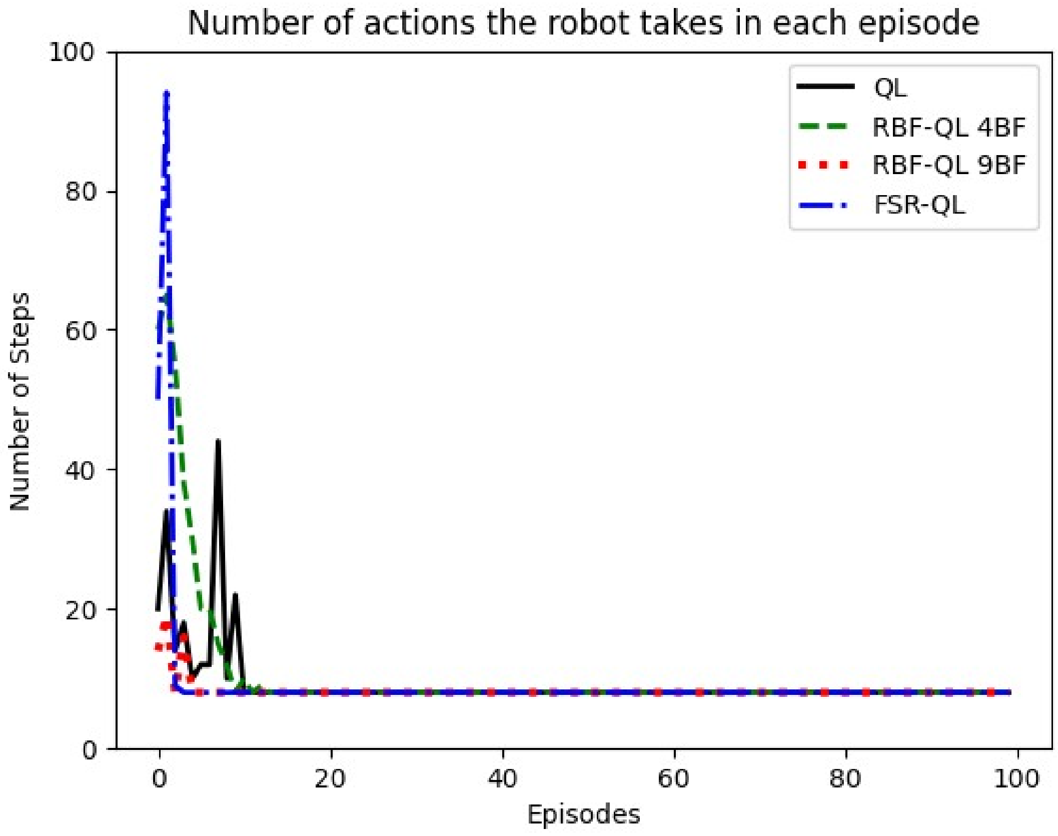


9 RBF cover:

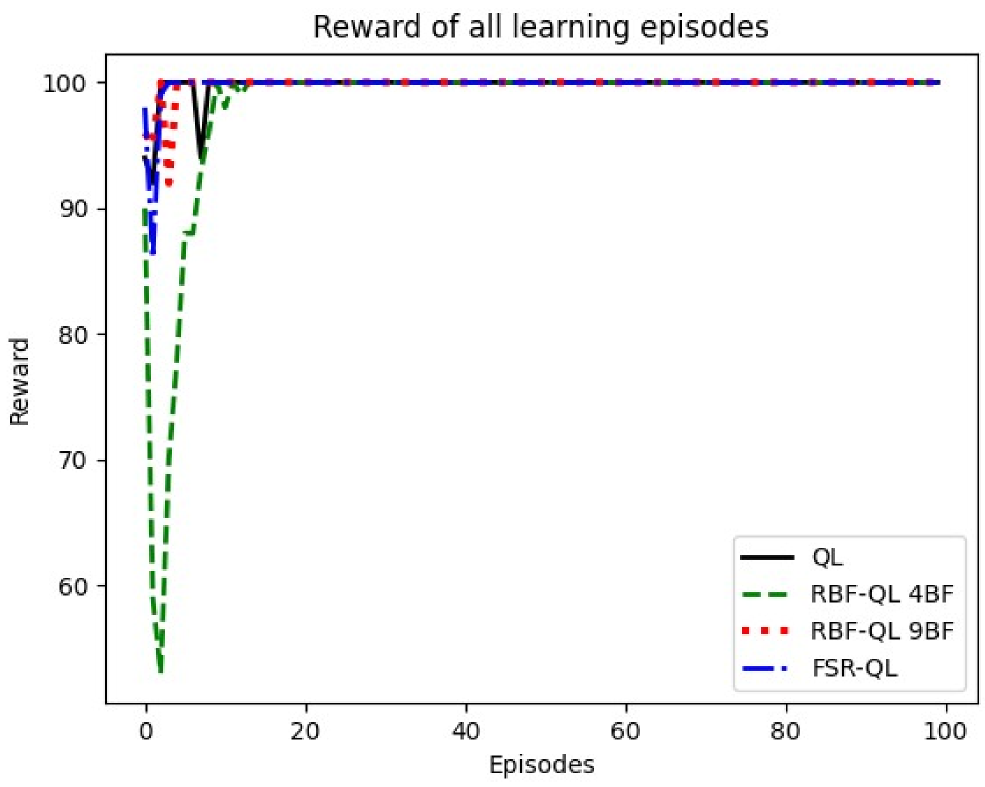


Requirement:

1. (UG: 50 points, G: 40 points) Plot the number of actions the robot takes in each episode for normal Q learning (QL), RBF-QL with 4BF, and RBF-QL with 9BF. Something is similar to this figure


Number of actions the robot takes in each episode
Number of Steps
Episodes
QL
RBF-QL 4BF
RBF-QL 9BF
FSR-QL
0
20
40
60
80
100

2. (UG: 50 points, G: 40 points) Plot the reward of all learning episodes for each method: Q learning (QL), RBF-QL with 4BF, and RBF-QL with 9BF.



3. (Grad: 20 points) For Grad students (CPE671 only): Implement the FSR and plot its results to compare to the RBF-QL with 9BF (see the figures in Sections 1 and 2). Undergrad students are welcome to implement this FSR with +10 points bonus.

Appendix:

The normal Q-learning equation is here:

$$Q_{k+1}(s_k, a_k) \leftarrow (1-\alpha)Q_k(s_k, a_k) + \alpha[r_{k+1} + \gamma \max_{a'} Q_k(s_{k+1}, a')], \quad (1)$$

where $0 \leq \alpha \leq 1$ and $0 \leq \gamma \leq 1$ are learning rate and discount factor of the learning algorithm, respectively.

The full pseudo code of the approximated Q-learning (just need to use equation 4) is here:

Algorithm 1: APPROXIMATED Q-LEARNING.

```
   Input: $\alpha, \gamma \in (0, 1]$, $\epsilon$, number of episode $N$
1  Initialize $\theta_0(s_0, a_0) \leftarrow 0$, $\forall s_0 \in S$, $\forall a_0 \in A$;
2  for $episode = 1 : N$ do
3      Measure initial state $s_0$
4      for $k = 0, 1, 2, \ldots$ do
5          Choose $a_k$ from $A$ using greedy policy with
            probability $p$:
6
           $a_k = \begin{cases} arg\max_{a'}(\phi^T(s_k, a')\theta_k), & p \geq \epsilon; \\ \text{a random action in } A, & \text{otherwise.} \end{cases}$
           Take action $a_k$ that leads to new state $s_{k+1}$:
7          Observe immediate reward $r_{k+1}$
8          Estimate $\phi(s_k, a_k)$ based on equation (3) or (4)
9          Update:
10
           $\theta_{k+1} \leftarrow \theta_k + \alpha[r_{k+1} + \gamma \max_{a' in A}(\phi^T(s_{k+1}, a')\theta_k) - \phi^T(s_k, a_k)\theta_k]\phi(s_k, a_k).$
11     until $s_{k+1} \equiv G$
```

For RBF-QL, each element of function ϕ is defined as:

$$\phi_l(s_k, a_k) = \begin{cases} e^{-\frac{s_k - c_l}{2\mu_l^2}}, & if\ a_k = A_j \in A \\ 0, & otherwise, \end{cases} \quad (4)$$

where c_l and μ_l are the center and radius of the RBF l, which has the shape of a Gaussian bell.

Note that, Equation (4) is for one dimension state. For 2 or more dimensions, use the equation in the lecture slide, or below:

$$\phi_i(s_k, C_i, \sigma_i) = e^{-\left(\frac{¿\vee s_k - C_i \vee ¿}{\sqrt{2}\sigma_i}\right)^2}$$

For 9 RBF Matlab code

```
clc;  clear;

n = 5;
state = states(n);
gamma = 0.9;
alpha = 0.65;

action = {[1,0];[-1,0];[0,1];[0,-1]}; %up,down,right,left
goal = [n,n];
goal_index = search(goal,state);

reward = initialize_rewards(n,state);

next_state = [0,0];
tf = 0;

iterations_9rbf = [];
reward_func_9rbf = [];

center = [[1 1]; [1 5]; [2 2]; [2 4]; [3 3]; [4 2]; [4 4]; [5 1]; [5 5]];
sigma = 1.2;

theta = rand(length(center)*4,1); % theta dimension is 36x1; number 4 hear means 4
actions


%=============================================================================
```

For 4 RBF Matlab code:

```
clc;  clear;

n = 5;
state = states(n);
gamma = 0.9;
alpha = 0.89;

action = {[1,0];[-1,0];[0,1];[0,-1]}; %up,down,right,left
goal = [n,n];
goal_index = search(goal,state);

reward = initialize_rewards(n,state);

next_state = [0,0];
tf = 0;

iterations_4rbf = [];
```

```
reward_func_4rbf = [];

center = [[2 2]; [2 4]; [4 2]; [4 4]];
sigma = 1.425;

theta = rand(length(center)*4,1); % theta dimension is 16x1; number 4 hear means 4
actions
```