Q-Learning for World Grid Navigation

EE5904/ME5404 PART II: PROJECT 2

REPORT DUE ON 25/04/2025

TA: XUBO GU

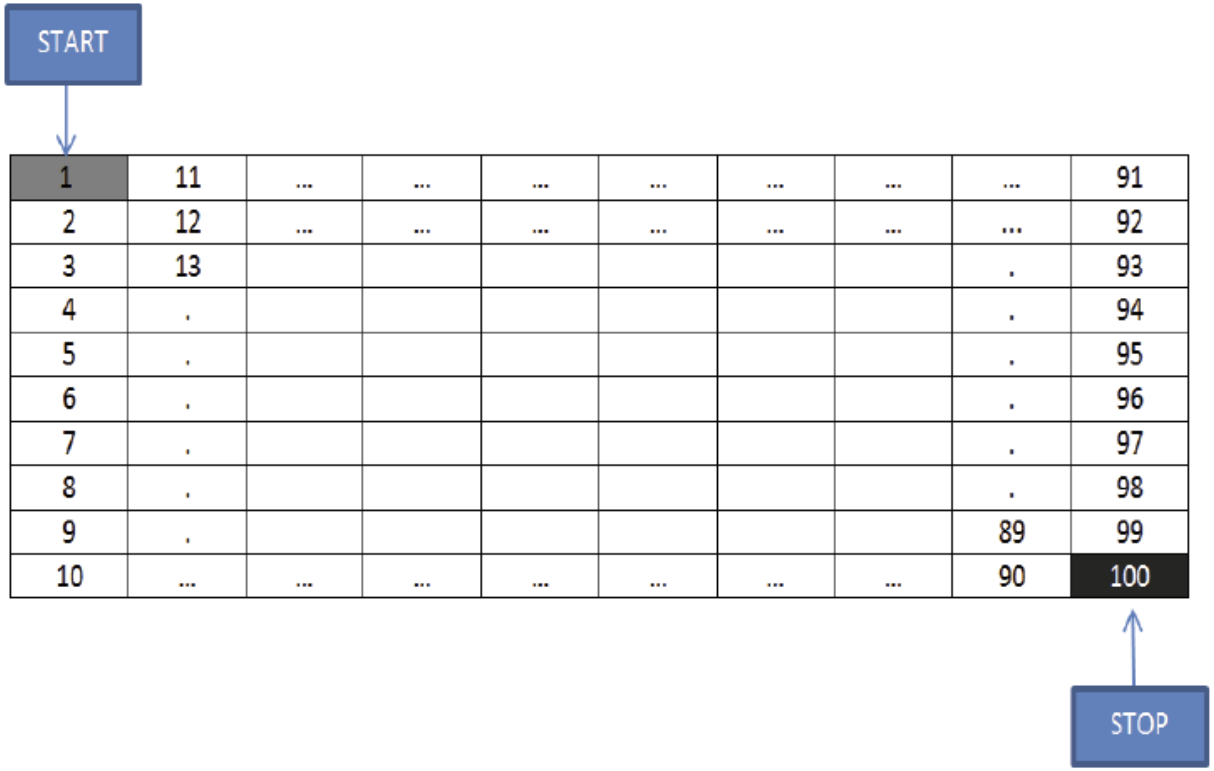
XUBO@U.NUS.EDU

Outline

- Project Description
- Project Implementation
- Important Notes

Project Description - Objective

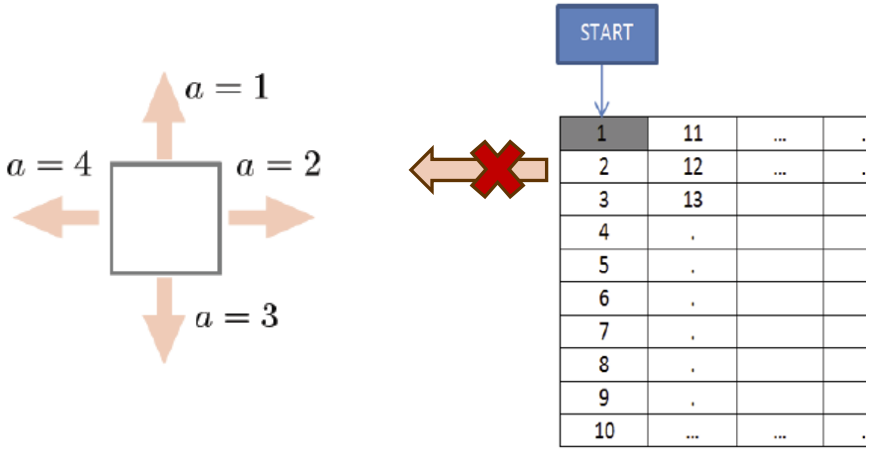
The robot is to move from the initial state (s = 1) to the goal state (s = 100) with the maximum total reward of the trip.



1	11	…	…	…	…	…	…	…	91
2	12	…	…	…	…	…	…	…	92
3	13							.	93
4	.							.	94
5	.							.	95
6	.							.	96
7	.							.	97
8	.							.	98
9	.							89	99
10	…	…	…	…	…	…	…	90	100

Project Description: State Transition

- At a state, the robot can take one of the four actions
- The state transition model is **deterministic**
  - In this project, you are required to implement **Q-Learning with ϵ-greedy** to find the optimal policy.
- Some of the actions are not allowed - states on the boundary: e.g., the robot can not go left in state=2

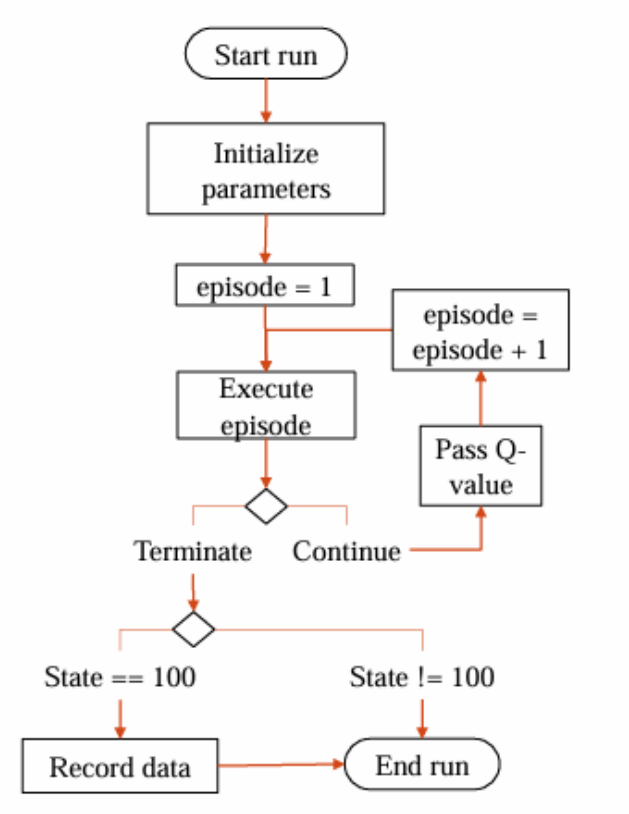


Project Description: Reward Function

- Reward is given as a matrix
  - Task 1 → saved in “task1.mat” [Given]
  - Task 2 → saved in “qeval.mat” [**Not** given]
- Reward Matrix
  - Dimension: 100 x 4 (States x Actions)

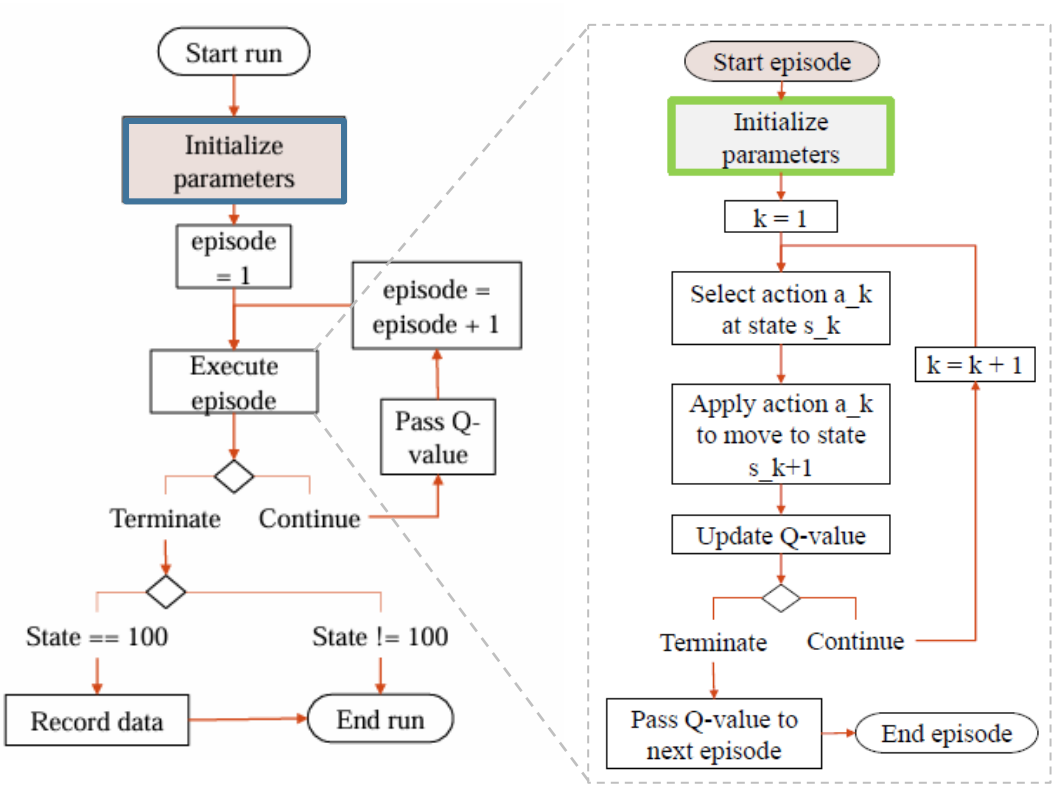
Project Description: Learning

- In this project, you are required to implement **Q-Learning**
- The robot learns the optimal policy in one run with N episodes
- Each episode starts from state s=1
- The updated Q values are passed to the next episode
- Repeat the episodes in one run, until the trigger of the termination



Implementation

Implementation - initialization



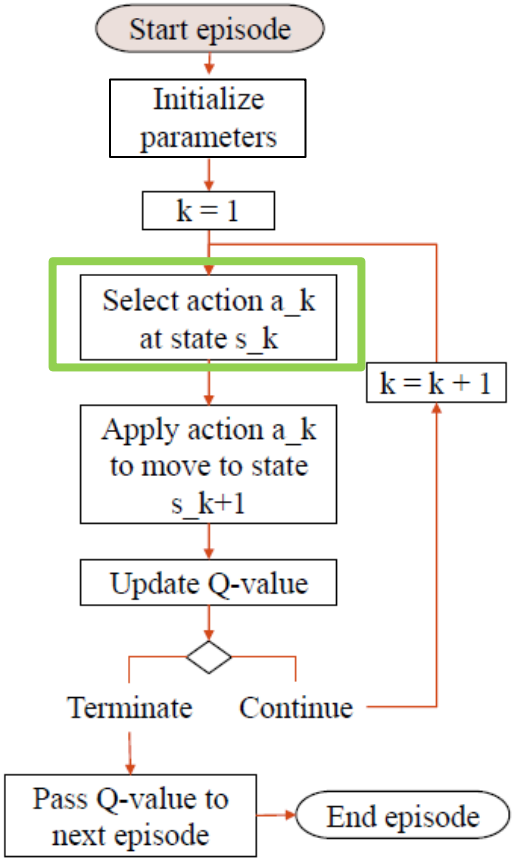
Initialization in each run:

- Initial Q-function $Q_1 \leftarrow 0$ | Optional

Initialization in each episode

- Discount factor γ
- Exploration probability ϵ_k
- Get Q-function
- Learning rate α_k
  - $\alpha_k = \epsilon_k$ in this project
- Initial state s_1 = 1
- Time step k = 1

Implementation – choose action

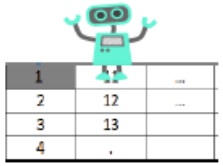


- Choose actions with ϵ −greedy exploration

Example:

For $k = 3$,

- $\epsilon = 1 - \frac{1}{k}$
- $s_3 = 11$



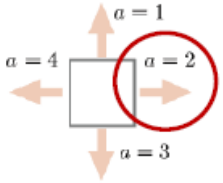
1		...
2	12	...
3	13	
4	.	

Recall that:

$$a_k = \begin{cases} a \in \arg\max_{\hat{a}} Q_k(s_k, \hat{a}) & \text{with probability } 1 - \epsilon_k \\ \text{an action uniformly randomly selected from all other actions available at state } s_k & \text{with probability } \epsilon_k \end{cases}$$

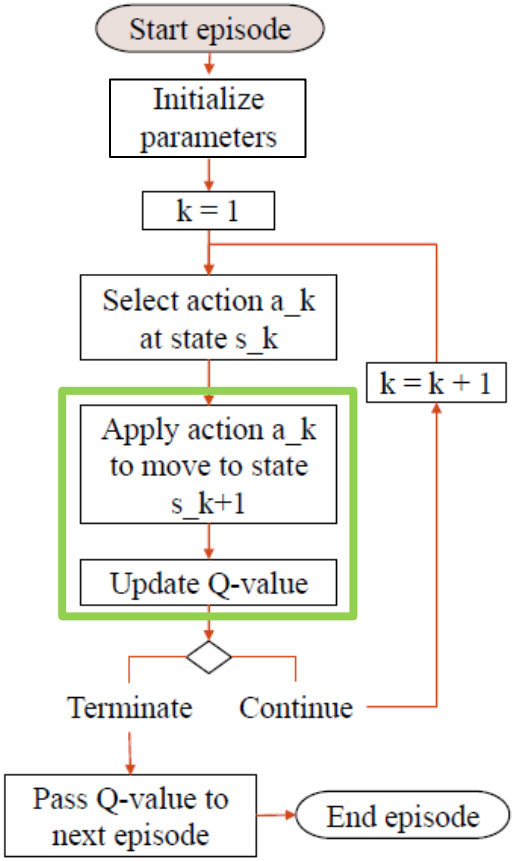
Current best action is 2

- Exploitation: $a_3 = 2$ each has $1 - \epsilon = \frac{1}{3}$ probability to be selected
- Exploration: $a_3 = 3, 4$ each has $\epsilon = \frac{2}{(2)3}$ probability to be selected



*** $a_3 = 1$ **cannot be** selected due to the **boundary**

Implementation – choose action

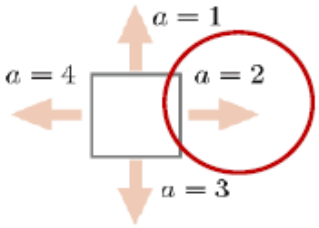


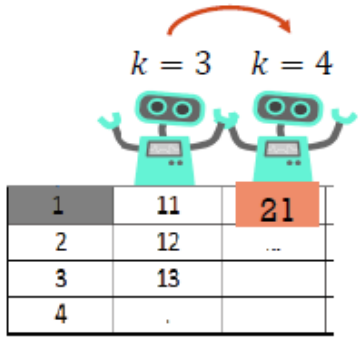
- Execute the action and update Q-value

Example (continue):

For $k = 3$,

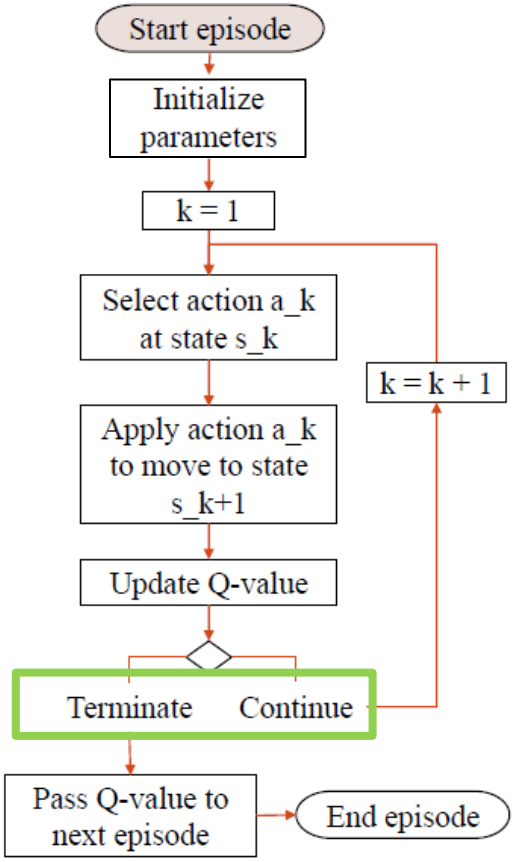
- $s_3 = 11$
- **Selected action is** $a_3 = 2$
- $s_4 = 21$





- **Receive reward** $r_{11\,2}$
- $Q_4(11,2) = Q_3(11,2) + \alpha_3(\boldsymbol{reward}(11,2) + \gamma * max(Q_3(21,:)) - Q_3(11,2))$

Implementation – termination



Termination condition for <u>each episode</u>:

- Robot reaches goal state $s_k = 100$ [Ideal Case]
- learning rate $\alpha_k < 0.005$

Continuation condition for <u>each step</u>:

- Otherwise

Important Notes: Task 1

1. Implement Q-learning algorithm in MATLAB
2. Reward function is in task1.mat
3. Discount factor γ and exploration probability ϵ_k are given in Table 1
4. For each set of parameter values
    - Set 10 runs (each has max episode = 3000)
5. Complete report
    - Number of goal reaching runs:
        - Out of 10 runs, count the runs that the robot ends at state 100.
    - Execution time:
        - Average the recorded execution time for those goal reaching runs.

TABLE I: Parameter values and performance of Q-Learning

ϵ_k, α_k	No. of goal-reached runs		Execution time (sec.)	
	$\gamma = 0.5$	$\gamma = 0.9$	$\gamma = 0.5$	$\gamma = 0.9$
$\frac{1}{k}$	?	?	?	?
$\frac{100}{100+k}$	?	?	?	?
$\frac{1+log(k)}{k}$	?	?	?	?
$\frac{1+5log(k)}{k}$	?	?	?	?

Important Notes: Task 1 – outputs

In your report, this optimal policy is to be presented in three ways:

1. As a column vector, with the position in the column corresponding to a state, and the entry for that position representing the action selected by the optimal policy at that state.
2. As a 10×10 grid diagram with arrows indicating the action selected by your **optimal policy** at each state.
3. As a 10×10 grid diagram showing **an (optimal) path** taken by the robot as it moves from the initial state to the goal state according to your optimal policy, plus the reward associated with this optimal path.

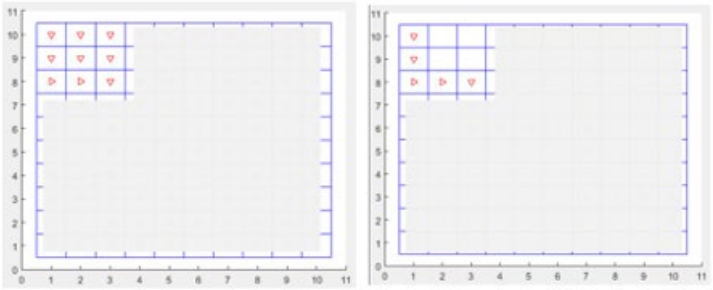
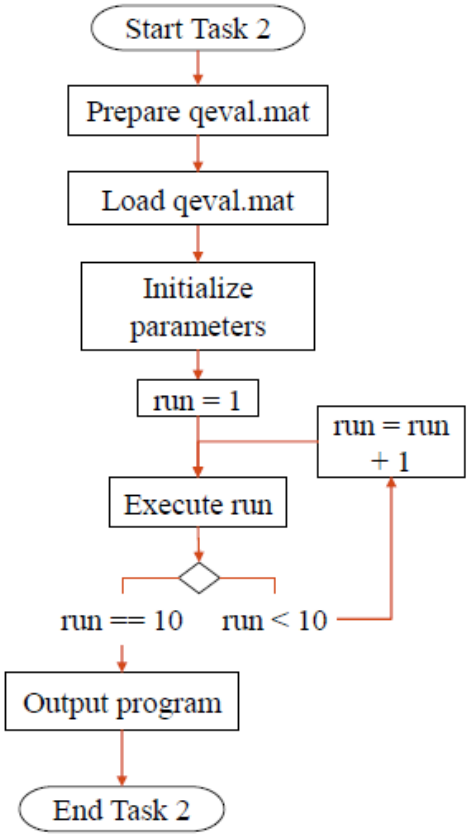


Fig. 3: Sample diagram illustrating an optimal policy (left) and the optimal path (right).

Important Notes: Task2

❖ Need to deal with unknown rewards

1. Implement Q-learning algorithm in MATLAB.
2. Decide <u>your own</u> discount factor γ and exploration probability ϵ_k.
3. Complete report.
4. "qeval.mat" will be used (as a replacement for the reward you have from task1.mat) to evaluate your RL program.



Note: You can test its execution on your own by making up a qeval.mat file containing dummy sample values.

Important Notes: Assessment

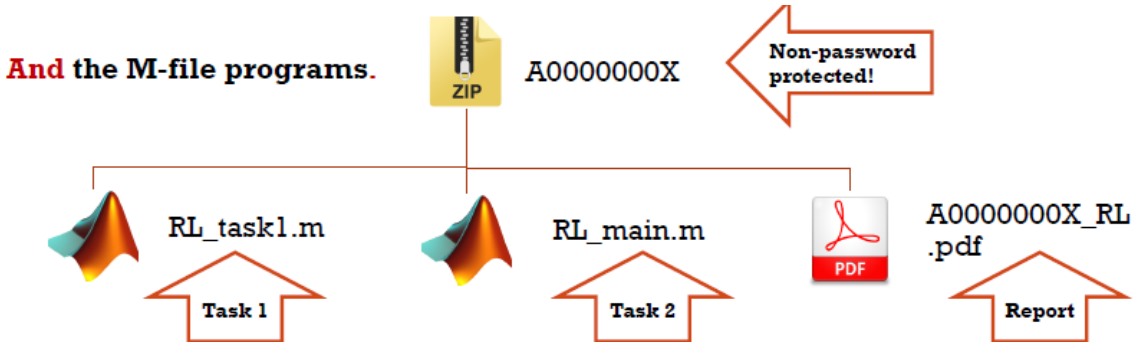
The project will be assessed based on the following criteria:

1. **Comments** (with supporting argument) on the results obtained in Task 1
2. **Presentation of the report.** This includes good report style, clarity, and conciseness
3. **Performance of your M-file program** for Task 2 in finding an optimal policy based on the reward function specified in unknown file qeval.mat

Important Notes: Submission

A report (in a PDF file with name the report as: StudentNumber_RL.pdf) describing the implementation and the results. It must contain a cover page showing:

- Student's name
- Student number
- Student's email address
- Name of the module
- Project title



Report due on **25/04/2025 Singapore time**

APPENDIX –USEFUL MATLAB CODES

MATRIX	
Create 2 x 3 matrix	[1 2 3; 4 5 6]
Create 4 x 3 matrix of zeros	zeros(4, 3)
Find number of rows and columns of matrix A	size(A)
Get element at 1st row and 1st column of matrix A	A(1, 1)

REWARD	
Load 'task1.mat'	load task1.mat
Create 'qeval.mat'	save('qeval.mat', 'qevalreward')

DATA	
Find the time taken of a block of code	tic % block of code toc
Display string with variable	disp(['This is ' variable 'variable.'])

TRAJECTORY PLOT	
Plot coordinate x and y with arrows	• plot(x, y, '^'); % action 1 • plot(x, y, '>'); % action 2 • plot(x, y, 'v'); %, action 3 • plot(x, y, '<'); % action 4
Set axis min and max	axis([0 10 0 10])
Format title	title(['Execution of optimal policy with associated reward = ' total_reward])
Show grid	grid on
Start grid from top left corner	set(gca,'YDir','reverse')

https://www.mathworks.com/help/matlab/getting-started-with-matlab.html

Thank you!