# Q-Learning for Simple Heading

**Environment:**

Square shaped grid world as of our need. It includes land, green water and dynamic space. While building environment, we have to declare the reward function as well. We can use this function for reward formulation.

**heading.activate(grid_size,green_water,land,theta, test = "simple heading")**

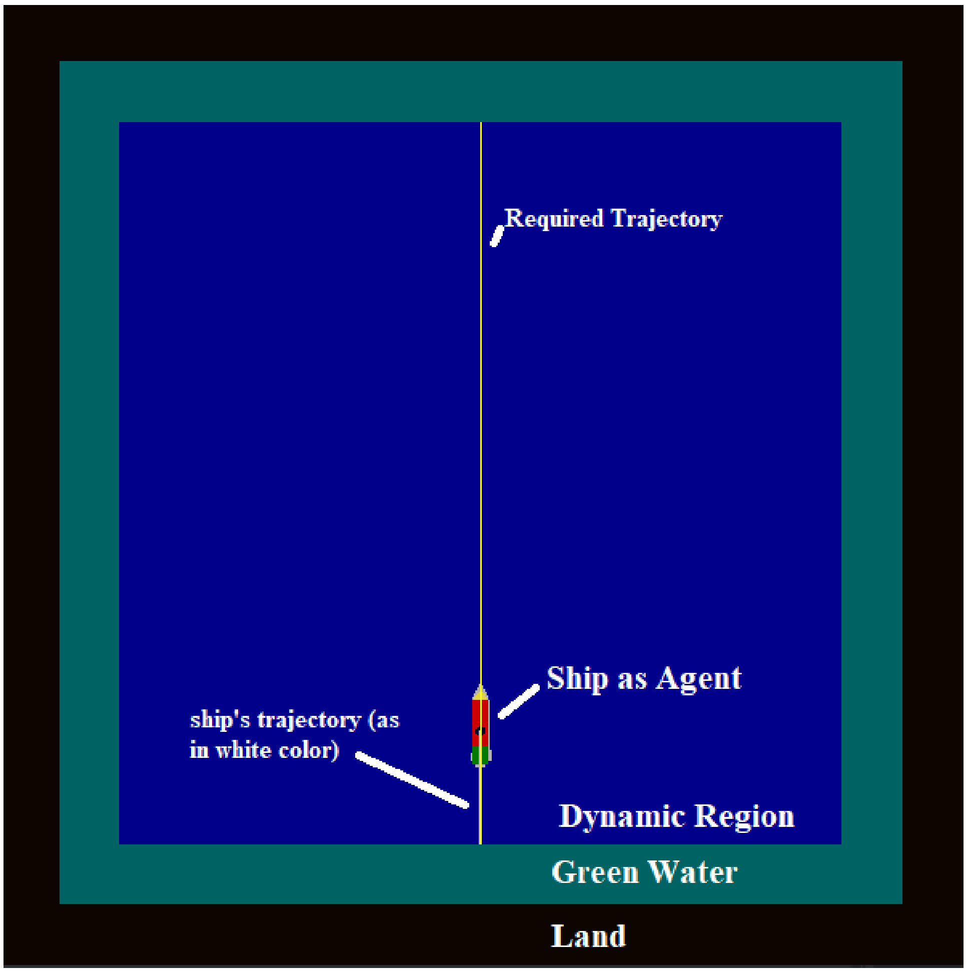

**Fig.1 Environment Formation**

theta- required heading angle
it will return Reward matrix, path reward points((x,y) coordinate of cell). We declare the reward as following,

| | |
|---|---|
| land | = -100 |
| green water | = -1 |
| dynamic region | = 0 |
| path | = 100 |

while declaring the path's reward, we are giving small reward for adjacent cells as exploration purpose.

| | | | | | | |
|---|---|---|---|---|---|---|
| **3** | **4** | **5** | 100 | **5** | **4** | **3** |
| (Xn-3,Yn-3) | (Xn-2,Yn-2) | (Xn-1,Yn-1) | (Xn,Yn) | (Xn+1,Yn+1) | (Xn+2,Yn+2) | (Xn+3,Yn+3) |

Heading Path

**Agent:**

Agent is functioning as first order nomoto model with 15 different actions. It takes ship's current state [u,r,x,y,psi,delta,n] and will return next state as observation.

**observation, reward,done,_ = env.step(action)**

**Action space:**

| -7 | -6 | -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| -35 | -30 | -25 | -20 | -15 | -10 | -5 | 0 | 5 | 10 | 15 | 20 | 25 | 30 | 35 |

[Action: Rudder angle]

While taking a step by agent it will return 4 output for us.

| | |
|---|---|
| Observation | - [u,r,x,y,psi,delta,n] of next state and then updating the current state to next state. |
| Reward | - reward value for respective cell |
| Done | - True or False (whether agent reached final state or not ) |
| _ | - message about agent's position (whether it is in correct path or not) |

As we consider first order nomoto model as agent's character, we need to do sanity check on designed agent. Here, we did sanity check by giving "0" action and "7" action. As result, agent moves in straight line and formed circle for constant action through time.

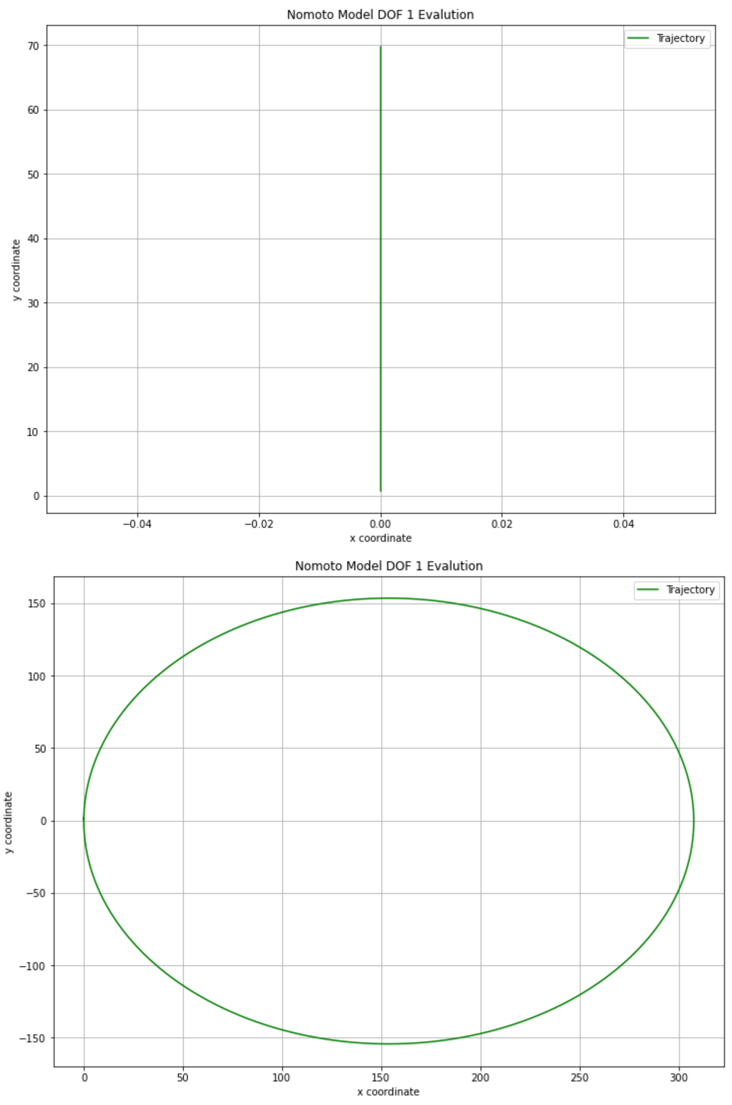

**Fig.2 Nomoto Model Evaluation by action "0" and action "7"**

Here, Agent's starting position is considered as **[5,0,int(grid_size/2),int(land+ green_water),0,0,120]** and 0.1 seconds as time interval.

**Q – Learning :**

- As we know that the agent has 15 different actions, at every cell in the gird, It can take 15 different kind of actions . So, we need to form dictionary which has value for each action at every cell(only in dynamic region). i.e    Q_table : {"(0,0)" : {dictionary_00},
  "(0,1)" : {dictionary_01},………………………………upto last cell}

- At every cell, agent has a choice of taking 15 actions based on value function. So, every cell has dictionary which has value for every action.
  Dictionary_00    : {"-7" : v1, "-6":v2 , "-5":v3……………………."7": v15}

- Once we formed the Q-table, we can start our agent's training by many episodes based on simple q-learning. Here, training take three parameters (epsilon, alpha and gamma). At every action in every episode, we can update the value function by our q learning formula.

$$V(S_t) \leftarrow V(S_t) + \alpha\Big[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\Big]$$

- the parameters we have taken are playing pivot role in agent's training ,
    - epsilon - for exploring action of agent
    - alpha - giving important for reward (preferably in range (0.9-0.99))
    - gamma - accounting the next state's value (preferably in range (0.05-0.3))
- Here, Description of input parameters
    - Heading Angle - 10
    - Grid size - 100
    - Land - 5
    - Green water - 5
    - Alpha, Gamma - 0.9, 0.1
    - Epsilon - 0.05,0.2,0.35,0.5 (for comparing exploration rate)
- Episode termination criteria:
    - When agent reaches termination point (or)
    - When agent goes to green water region(or)
    - Average cumulative reward < 95 (if passes through exact path, average cumulative reward always 100)

**After Training :**

- After training, we will get trained **q table ,value function and optimal action function** in 100*100 matrix format
- By plot, we can see the trained results

## Q-learning Results and Plots

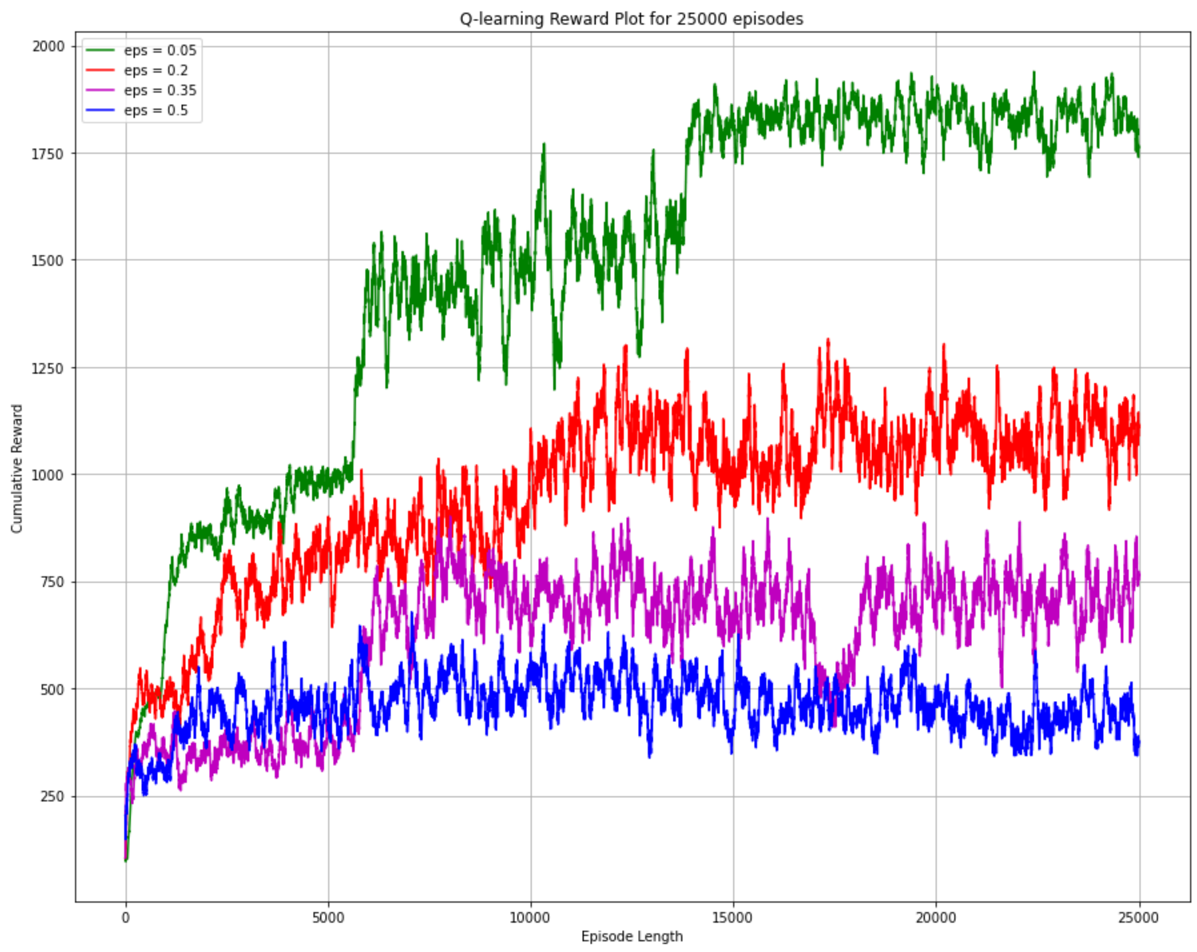

**Fig 3. 25000 episodes training for different epsilon value**

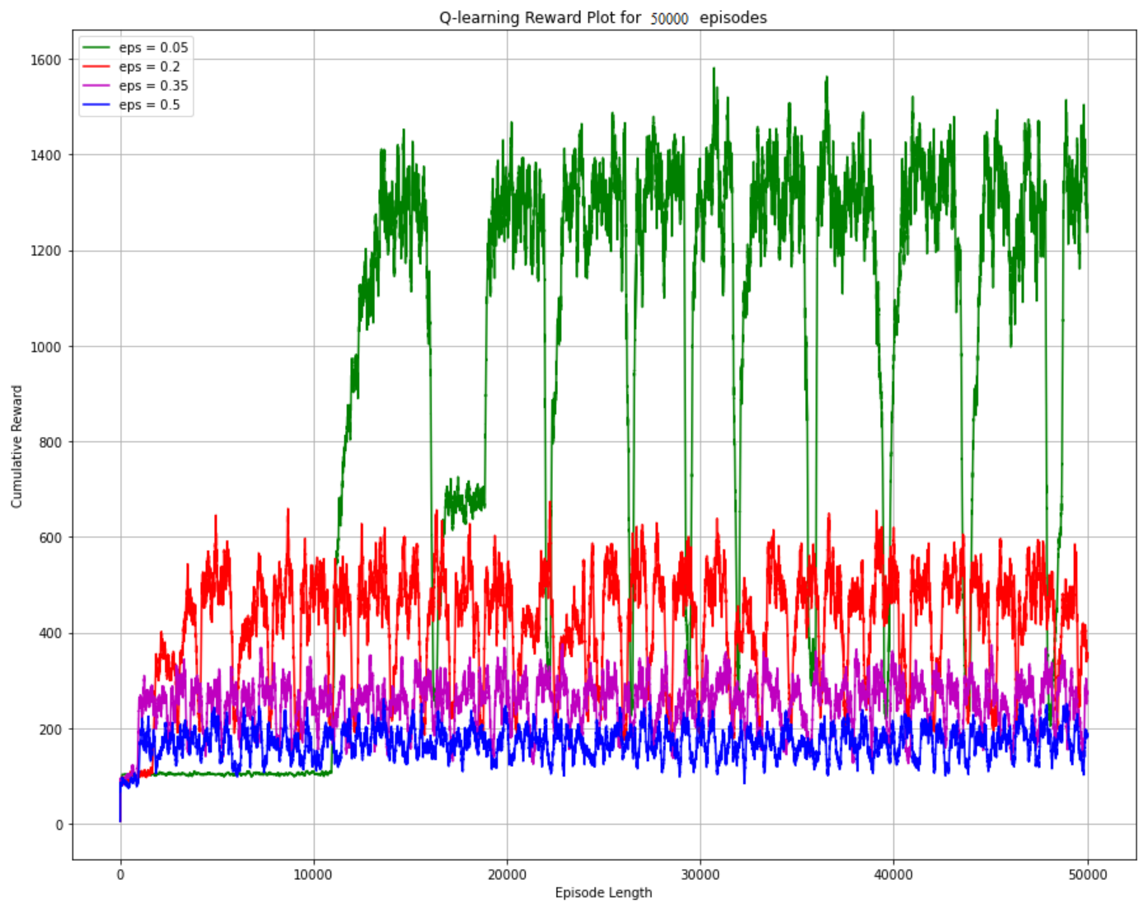

**Fig.4 50000 episodes training**

**Value function Plot :**

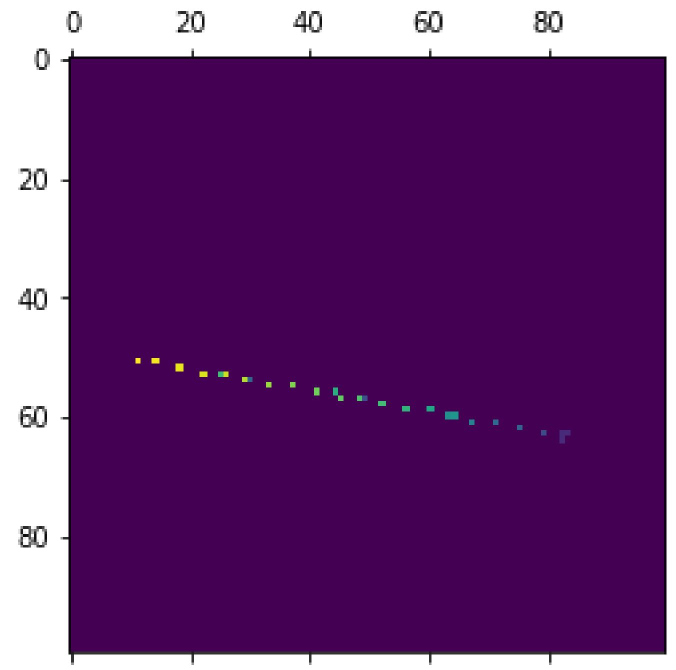

Fig.5 Value function Plot for eps = 0.05

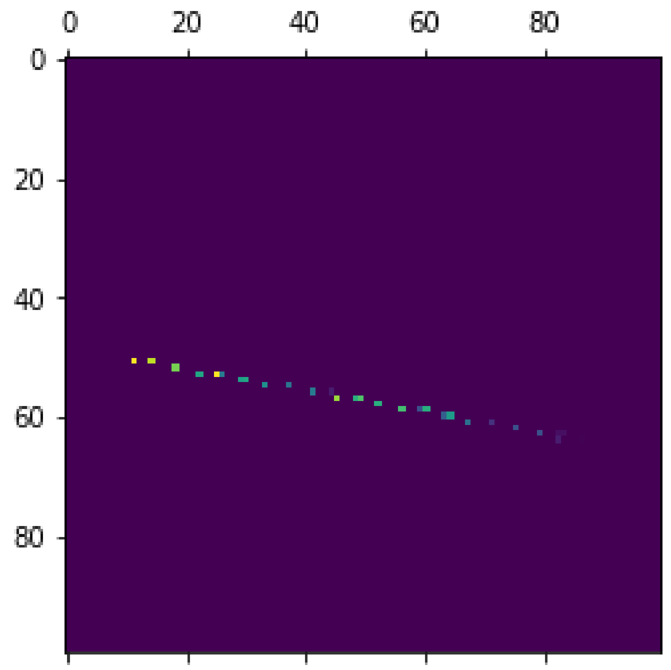

Fig.6 Value function Plot for eps = 0.2

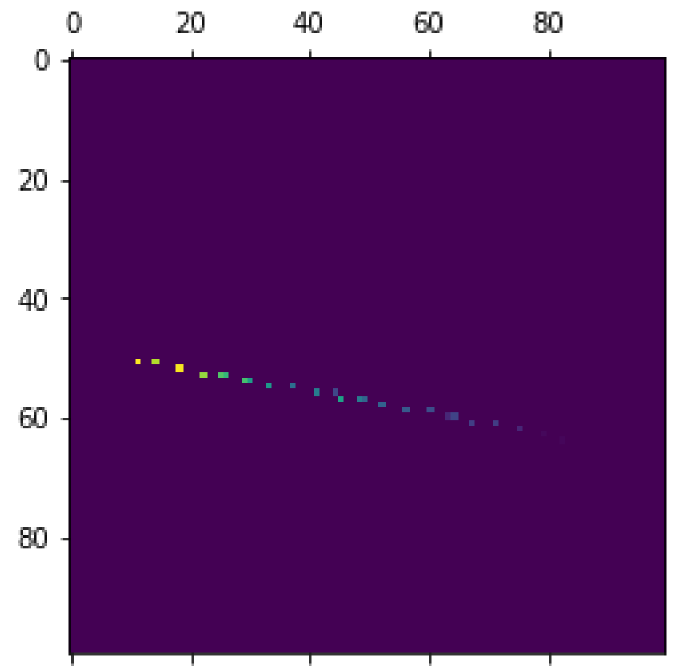

Fig.5 Value function Plot for eps = 0.35

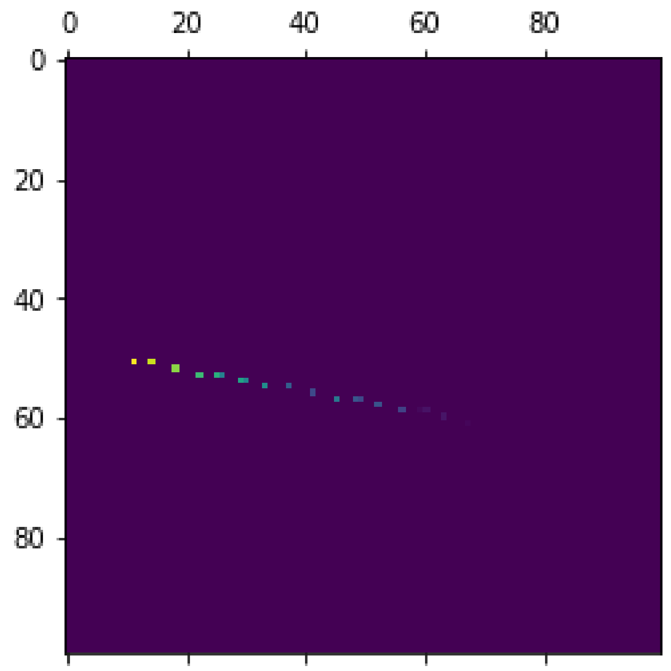

Fig.5 Value function Plot for eps = 0.5

**Observation:**

- From results and plot epsilon value of 0.05 and 0.2 is optimal

**Action Map :**

From trained Q table we can find that the optimal rudder angle action at each state cell.