Name: Yaru Niu GTID: 903525523

Problem Set 3
Deadline: 12:00pm EDT, 17 September 2019

Assistant Professor Matthew Gombolay
CS 8803 - Interactive Robot Learning

September 17, 2019

Instructions: Write your name in the top, left-hand corner. **You may work with others (collaborate) to complete this assignment.** Here, "collaborate" means that you talk about the assignment, teach/learn from each other, and even compare answers. However, you *must* write your own code, but you *may* help debug each other's code. The execution of the coding must be done on your own. Finally, you must list the names of the people with whom you collaborated. Sign here acknowledging adherence to completing this assignment according to these instructions:

Signature: _____

Collaborators: _____

Problem 1. For this homework, we are going to build upon your Wumpus World code from PSet 2. You should now:

1. Implement the full state space (i.e., binary variables in each cell for agent occupancy, visited, ok, breeze, stench, Wumpus, pit, and gold) for a full state space size of $(2^8)^{16}$. However, you do not need to enumerate the whole state space! You will end up constructing the states as you need them, as you will see in problem 2.2.

2. Make the Wumpus dynamic by moving up/down/left/right/stay with uniform probability. Only consider feasible actions. For example, if the Wumpus is in cell $(1, 1)$, it cannot move down or to the left, so it picks right/stay/up with probability of one-third each. If the Wumpus and agent ever occupy the same sell in state s, the game is over. Assume the Wumpus is impervious to pits.

Problem 2. Implement the "Q-learning" algorithm presented in Lecture on 03 SEP 2019. Set $\gamma = 0.95$ and $\epsilon = 0.1$. Run the algorithm for 10,000 episodes with a max episode duration of $T = 16$. Note: Each episode starts with s_o with the agent in $(1, 1)$ as shown in Figure 7.2 and the Wumpus in $(1, 3)$ except that the world has not been made visible to the agent.)

1. Show an x-y plot where the x-axis is the episode number of the learning algorithm e and the y-axis is the loss for that episode

$$L_e = \sum_{t=1}^{T_e} \left[(r_{t,e'} + \gamma \max_{a'} Q(s_{t+1,e'}, a')) - Q(s_{t,e'}, a_{t,e'}) \right]^2$$

current episode e, and T_e is the number of time steps that transpired in episode e.

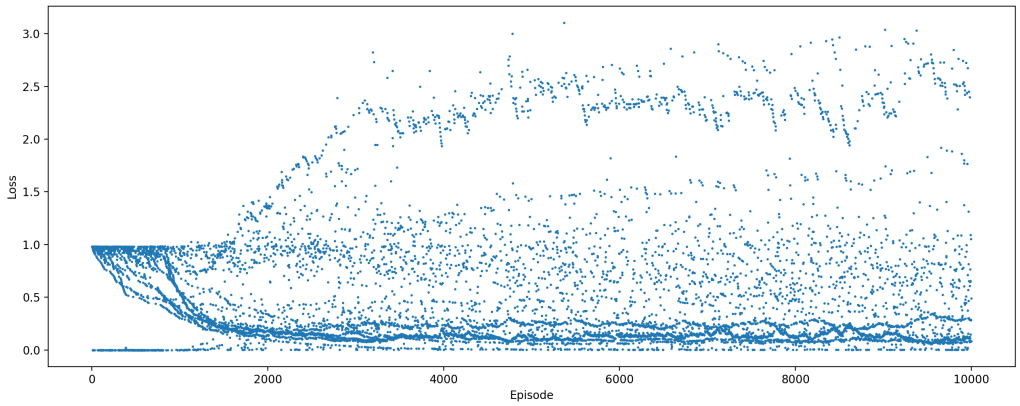

Figure 1: Loss in each episode.

2. Show an x-y plot where the x-axis is the iteration number of the learning algorithm and the y-axis is the number of *unique* states explored so far.

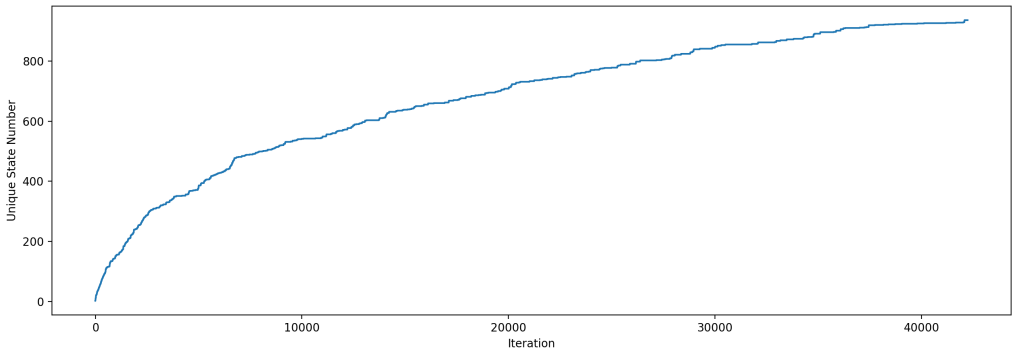

Figure 2: Explored unique state.

3. Report the value of q-values of the learned policy for each of the three actions available to the agent in s_o. Report the value to three decimal places.

   (a) $Q(s_o, \text{“up”}) = -0.090$

   (b) $Q(s_o, \text{“right”}) = 0.453$

   (c) $Q(s_o, \text{“stay”}) = 0.321$

Problem 3. For this problem, you will use the “slice-y-sample” and “slice-dy-sample” csv data sets available on Canvas. These data sets contains the angular positions and velocities, respectively, of a Rethink Robotics Sawyer's seven (7) degrees of freedom (DoF) sampled at 100 hz. Because Sawyer has 7 DoF, you would actually need to reason in multiple dimensions for our DMP. As such, we would have $h_{i,j}, c_{i,j}, w_{i,j}, \forall i \in 1, ..., m$ and $j \in 1, ..., \text{DoF}$. Do we need multiple α_y terms for each joint? That would be a design parameter! Rather than dealing with all of this confusion for a single homework assignment, let us just pick DoF number 7 (i.e., the seventh column of each data set).

For this problem, assume that $m = 8$ (i.e., there are eight Gaussian kernels in the forcing function, f), $\alpha_y = 8$, $\beta_y = 10$, and $\alpha_x = 1$. Furthermore, assume that $h_i = 1$ and $c_i = \frac{i}{m}$. Finally, assume that y_g is the final, recorded position in the data file. The last thing you'll need is to impute values for the acceleration, $\ddot{y}$, of the joint, as the robot does not record this value. You should use the following method to approximate this value: $\ddot{y}_t = \frac{1}{2}(\dot{y}_t + \dot{y}_{t-1})$, with $\ddot{y}_0 = 0$.

Your task is to learn a DMP controller by applying the closed-form expression

$$w_i = \frac{s^T \psi_i \mathbf{f}}{s^T \psi_i s}$$

where

$$s = \begin{bmatrix} x_{t_o}(y_g - y_o) \\ \vdots \\ x_{t_N}(y_g - y_o) \end{bmatrix}, \psi_i = \begin{bmatrix} \psi_i(t_o) & \dots & 0 \\ 0 & \ddots & 0 \\ 0 & \cdots & \psi_i(t_N) \end{bmatrix},$$

and

$$\mathbf{f} = \begin{bmatrix} \ddot{y}_{t_o} - \alpha_y(\beta_y(y_g - y_{t_o}) - \dot{y}_{t_o}) \\ \vdots \\ \ddot{y}_{t_N} - \alpha_y(\beta_y(y_g - y_{t_N}) - \dot{y}_{t_N}) \end{bmatrix}.$$

Ok, so you have your answers. How do we visualize what the trajectory might look like? Use Euler's method to compute the trajectory plot the results on single plot of the angular position where the horizontal axis is time and the vertical axis is the angular position of the 7^{th} DoF of Sawyer. Also include the raw data, which makes a total of two curves. Include a legend denoting which curve is for the closed-form method and the raw method. Note that Euler's method would look like this:

$$y_t = y_{t-1} + \dot{y}_{t-1} * dt$$

$$\dot{y}_t = \dot{y}_{t-1} + \ddot{y}_{t-1} * dt$$
$$\ddot{y}_t = \alpha_y(\beta_y(y_g - y_{t-1}) - \dot{y}_{t-1}) + f(x_t)$$

where x_t is from your canonical function.

$w = [-174.4778151284968, -167.71840572508518, -161.40225435946243, -155.5170032141128,$

$-150.04784873624612, -144.9779440548627, -140.28881698743527, -135.96078697766234]$

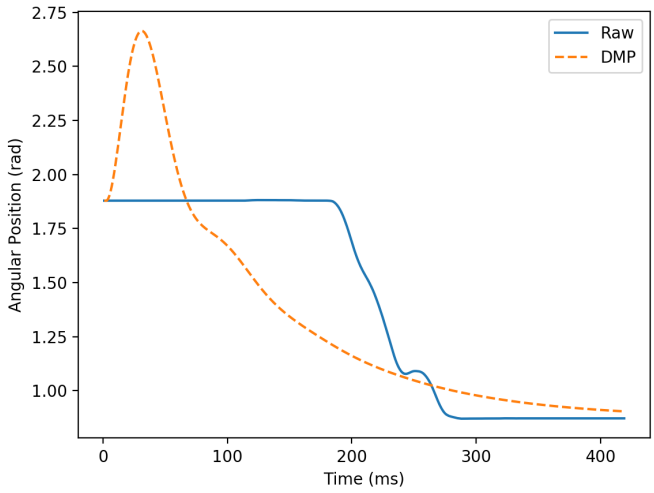

Figure 3: Trajectories generated by raw data and DMP.

Problem 4. [Bonus: 10% of PSet 3 Grade] Use gradient descent to solve for the values of $h_i, c_i, w_i, \forall i \in 1, ..., m$ for DoF 7. The loss you wish to minimize is the sum of the squared difference of the desired vs. learned accelerations. Use a learning rate of 0.01, and iterate for 1,000 steps. Include this curve in the plot you generated for Problem 3.