# CS 237B: Principles of Robot Autonomy II
# Problem Set X

Name:
SUID:

## Problem 1

(i) Initializing all weights as zero may result in finding local optima, as opposed to using randomized weight selection. Conversely, the results may be more reproducible if using zero weights.

(ii) Xavier initialization attempts to keep the variance across the weights of each layer to be the same. The purpose is to prevent what is termed "exploding or vanishing gradients". Exploding gradients can be understood as when weights are initialized too large, the output of each layer increases exponentially. This leads to poor accuracy of the network, as the target oscillates around a minima (the cost oscillating around the minimum value). Similarly, when the weights are initialized to small, the network may converge too quickly, as the cost gets exponentially lower ("vanishing").

(iii) Adam is a a gradient descent optimization algorithm for stochastic objective function, thus it iteratively used a gradient-based step to find optimal values of parameters for minimizing an objective function. Based on the original paper, Adam achieves faster results in finding optimal parameters for linear/logistic regression, neural networks and convolutional neural networks, as compared to other optimization methods such as SGDNesterov (an accelerated form of stochastic gradient descent) and AdaGrad. The advantages of Adam include working on stochastic (non-stationary) objective and sparse gradients (by using an adaptive learning rate per-parameter). A potential disadvantage has been proposed of Adam being poor at generalizing, compared to SGD for example. [TODO add reference]

## Problem 2

(i) Using the associative property of matrix multiplication (for rearranging the torque m

(ii) See code.

(iii) The training procedure for the left and right goals were the same. As directed in the code comments, each training step consisted of a forward pass followed by a backpropagation of the gradients of the weights. The neural network consisted of three hidden layers (1024, 512, 64 units, respectively) each with ReLU activation functions. Xavier initialization was used for each layer. The output layer was a densely connected layer matching the output size (2). The loss function used is discussed in the latter questions. By experimentation, it was found that 200 epochs achieved optimization of the training loss, where the loss oscillated around a minimum with any increase in the number of epochs. The learning rate used was the default for Adam optimizer: 0.001. For the straight goal, the same NN was used; the only change was to the number of epochs and the learning rate. For the straight goal, the network was trained on 150 epochs and a learning rate of 0.002. These were decided on through looking at minimization the training loss over each epoch, as mentioned previously.

(iv) The table below shows the success of the NNs on the test set:

| Goal | Test Accuracy |
|---|---|
| left | 0.99 |
| right | 0.60 |
| straight | 0.32 |

The loss used was the Mean Squared Error (MSE). It was found through experimentation that this produced the best minimization, in comparison to the Euclidean norm for example. A weighting was chosen for the error, where a 4:1 ratio was used to penalize the steering error. This weighting had a considerable impact on the training loss optimization.

TODO: Is it bounded? TODO: Write out equation?

## Problem 3

(i) See code.