Algorithm Description:

Algorithm used: Deep Deterministic Policy Gradients (DDPG)

Networks used:

- Actor: Two hidden layers (400, 300), Relu activation in the hidden layers with BatchNorm and Tanh at the output layer
- Critic: Two hidden layers (400, 300), Relu activation in the hidden layers with BatchNorm and Linear activation at the output layer

Hyperparams:

SEED = 10 # Random seed

NB_EPISODES = 10000 # Max nb of episodes

NB_STEPS = 1000 # Max nb of steps per episodes

UPDATE_EVERY_NB_EPISODE = 4 # Nb of episodes between learning process

MULTIPLE_LEARN_PER_UPDATE = 3 # Nb of multiple learning process performed in a row

BUFFER_SIZE = int(1e5) # replay buffer size

BATCH_SIZE = 200 # minibatch size

ACTOR_FC1_UNITS = 400 # Number of units for the layer 1 in the actor model

ACTOR_FC2_UNITS = 300 # Number of units for the layer 2 in the actor model

CRITIC_FCS1_UNITS = 400 # Number of units for the layer 1 in the critic model

CRITIC_FC2_UNITS = 300 # Number of units for the layer 2 in the critic model

NON_LIN = F.relu # Non linearity operator used in the model

LR_ACTOR = 1e-4 # learning rate of the actor

LR_CRITIC = 5e-3 # learning rate of the critic

WEIGHT_DECAY = 0 # L2 weight decay

GAMMA = 0.995 # Discount factor

TAU = 1e-3 # For soft update of target parameters

CLIP_CRITIC_GRADIENT = False # Clip gradient during Critic optimization

ADD_OU_NOISE = True # Add Ornstein-Uhlenbeck noise

MU = 0. # Ornstein-Uhlenbeck noise parameter

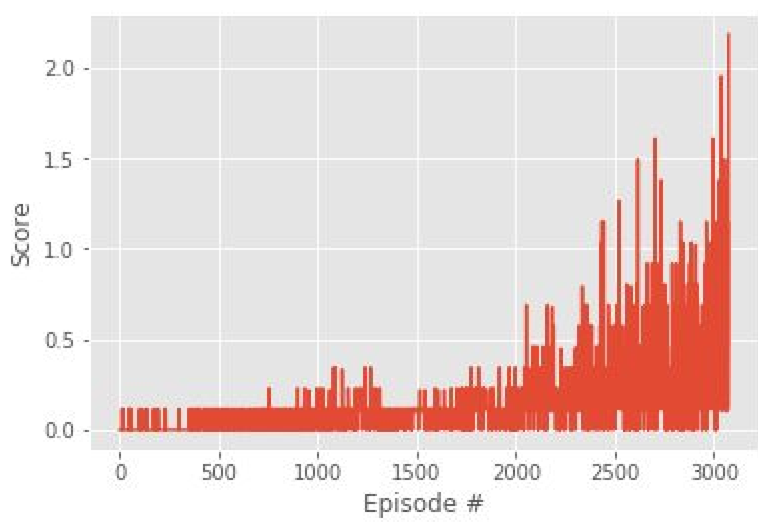
THETA = 0.15 # Ornstein-Uhlenbeck noise parameter

SIGMA = 0.2 # Ornstein-Uhlenbeck noise parameter

NOISE = 1.0 # Initial Noise Amplitude

NOISE_REDUCTION = 1.0 # Noise amplitude decay ratio

Rewards Plot:



Future Work:

Optimization of hyperparams through grid search or SMBO. Optimization of network architectures with deeper and more robust neural nets