Reward 1

 # ==

 # # Reward 1

 # if nextState[0] >= 0.5 or nextState[0] > episodeMaxDist:

 # reward += 5

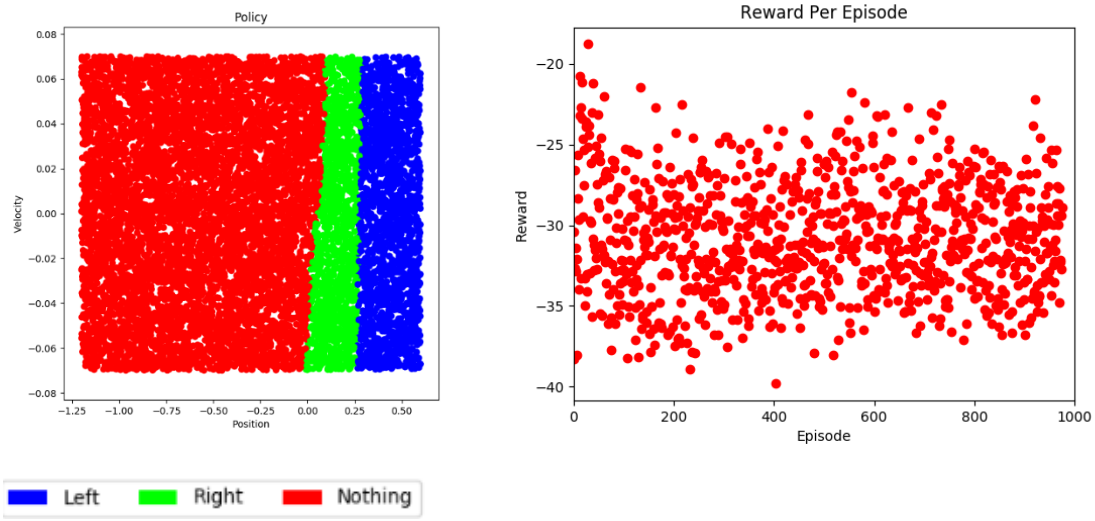
 # else:

 # reward = nextState[0] + 0.5

 # ==

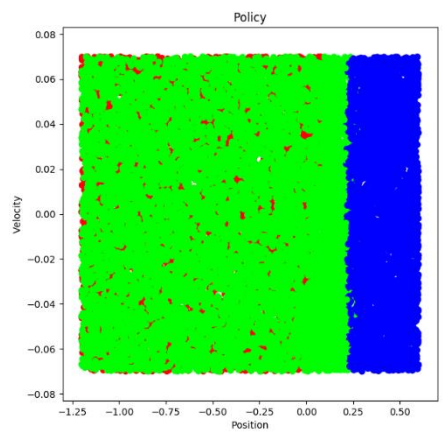
NN Architecture 1 with Reward 1

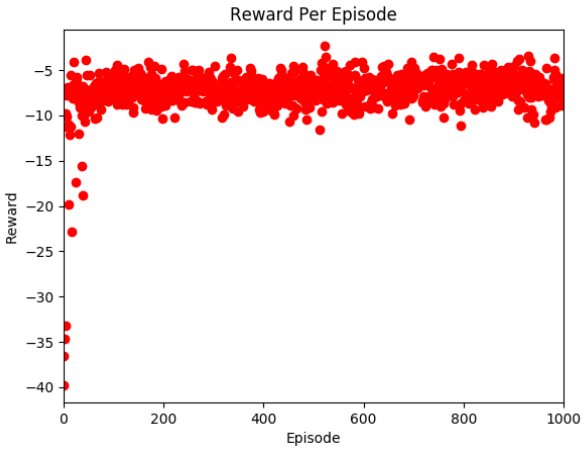
2 * 50 * 3

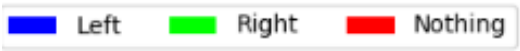


NN Architecture 2 with Reward 1

2 * 128 * 52 * 3

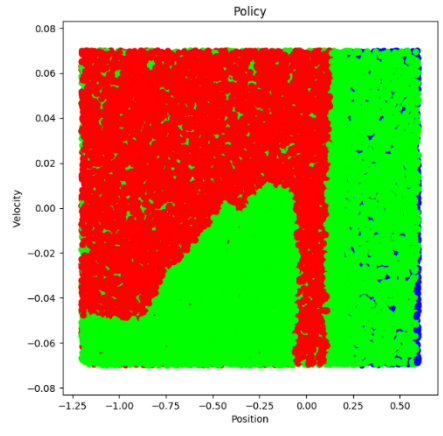






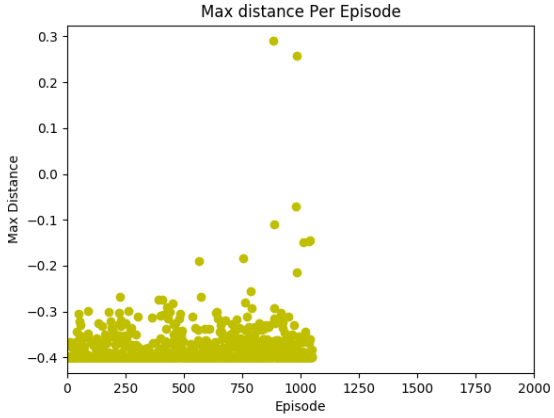
NN Architecture 3 with Reward 1

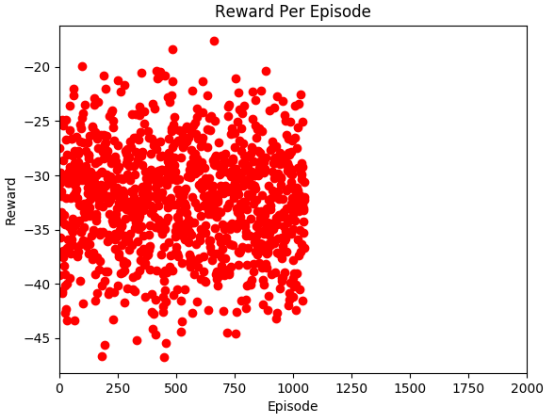
2 * 34 * 31 * 21 * 19 * 10 * 4 * 3

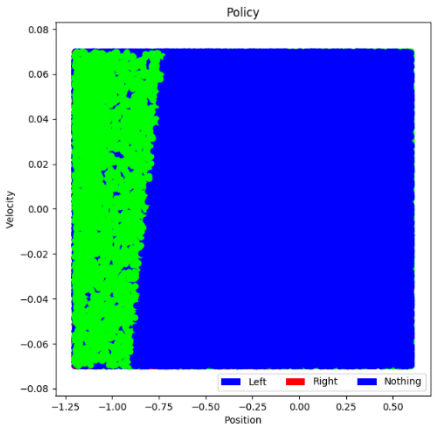


Reward 2 with NN Architecture 1

```
# ================================================================================
#       # Reward 4
#       sign = np.array([-1.0,0.0,1.0])
#       if nextState[1]*sign[action] >= 0:
#           reward = nextState[0] + 0.5
#       else:
#           reward = nextState[0] - 0.5
# ================================================================================
```

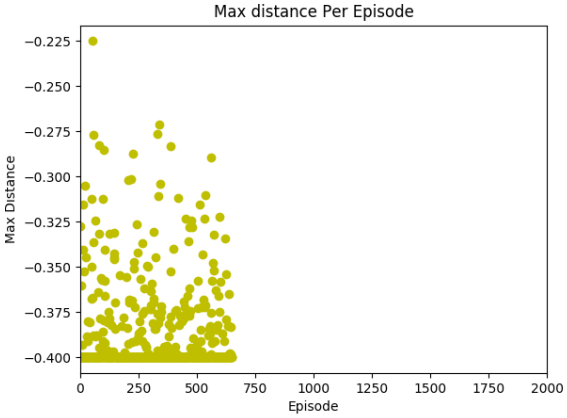


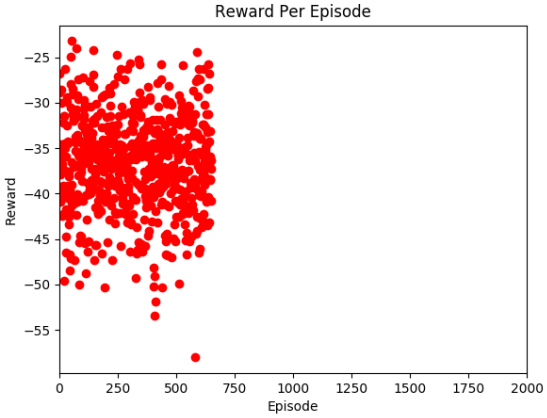


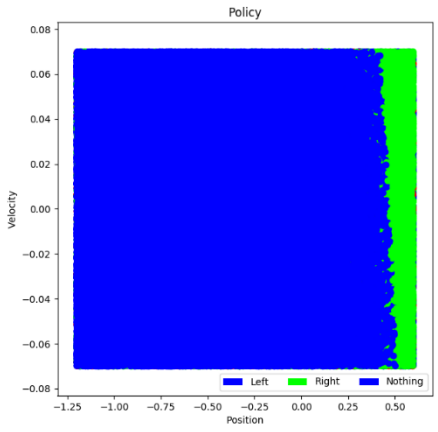


Reward 3 with NN Architecture 1

```
# ================================================================================
#       # Reward 5
#       sign = np.array([-1.0,0.0,1.0])
#       if currState[1]*sign[action] >= 0:
#           reward = nextState[0] + 0.5
#       else:
#           reward = nextState[0] - 0.5
# ================================================================================
```







Reward 4 with NN Architecture 1

 # ==

 # # Reward 6

 # sign = np.array([-1.0,0.0,1.0])

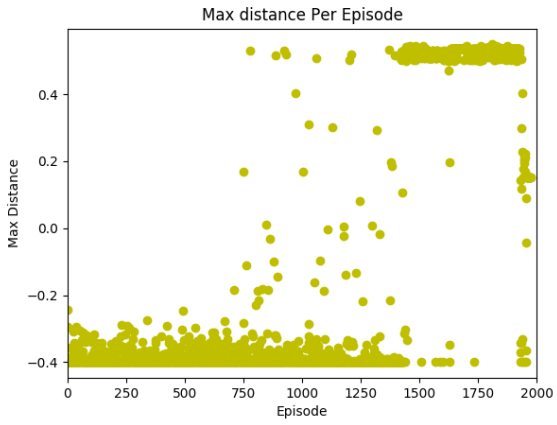
 # if currState[1]*sign[action] >= 0:

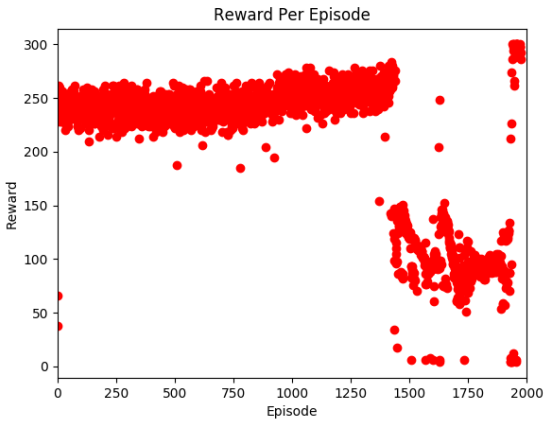
 # reward = 1

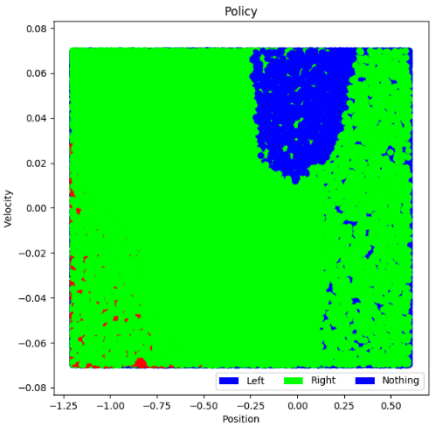
 # else:

 # reward = -1

 # ==







Reward 5 with NN Architecture 1

```
# ================================================================================
#       # Reward 7
#       sign = np.array([-1.0,0.0,1.0])
#       if currState[1]*sign[action] >= 0:
#           reward = 1
#       else:
#           reward = -1
#       reward = (0.999**step) * reward
# ================================================================================
```