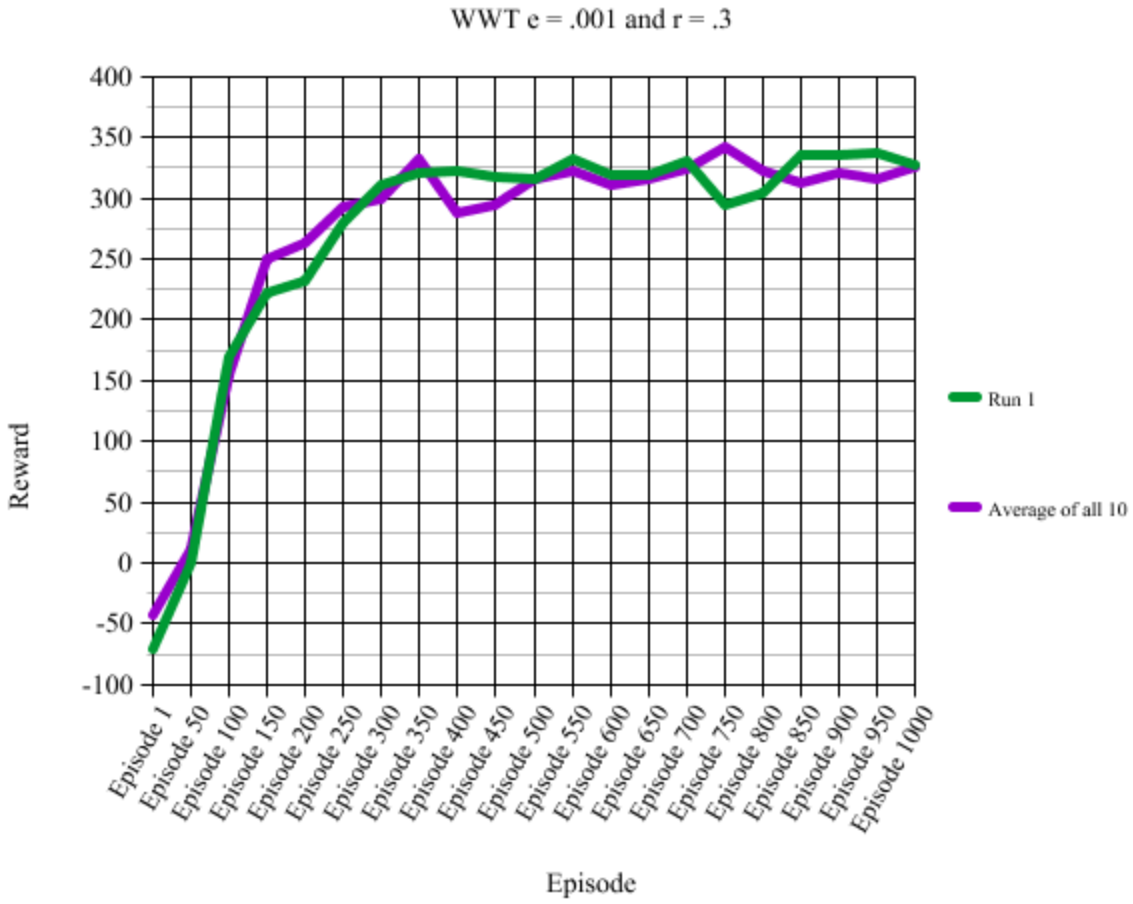
Manav Ramesh CS 440

MP1 Part 2

1. WorldWithoutThief
    a. We observe that throughout all 10,000 episodes the reward simply remains at 0. As we do not actually have any random actions occurring thus the AI simply chooses to do nothing other than keep the 0 reward and not explore anything, in colloquial terms the robot is playing it safe and making sure there is no negative reward as it does not need to make any random decisions.
    b. When the epsilon is set to .1 the actions result in a change in reward as the epsilon forces the robot to take risks and look for better opportunities than 0 reward. Due to this the first 20 or so episodes result in negative or around 0 reward, but after initial struggles the robot understands what needs to be done to gain maximum reward thus resulting in a consistent stream of rewards which end up fluctuating between 200 and 100 which is a net success considering what 0 epsilon did for our robot.
    c. We observe that when we raise epsilon to .5 we get a random fluctuating reward that ends up settling in the negatives. The reasoning behind this is with epsilon set so high we make 50% of our actions randomly which leads to lots of failures and negative rewards which cannot be counteracted by the other 50% that is attempting to learn from its mistakes. Thus showing us that randomness in required to generate a reward yet too much of it yields exactly what one would expect, random results which is not a positive thing.
2. WorldWithThief
    a. When we run the simulation without knowing where the thief we get a negative reward that simply fluctuates between -50 and -8. This result is explained by the fact that we cannot predict where the thief will be resulting in an almost impossible set of actions to avoid the thief to provide positive reward which cannot be consistently duplicated, thus resulting in a mostly negative reward scheme.
    b. When the thief is known is selected the rewards after the first 5-10 episodes of negative rewards the rewards start increasing at a consistent rate settling around the high 200's mark. This is a result of knowing where the thief is and after initial struggles and learning that the thief provides negative rewards the robot simply avoids the thief and gains the highest reward that has been reached yet.
    c. I started by editing my epsilon rating, i realized the lower i put the better the results seemed while not actually getting to zero. Thus my final value for epsilon was .001. for learning rate the .1 original value seemed low and as i raised it i found better results until it stagnated around .4 and lowered around .5/.6. Thus my final value for the learning rate was .3.



3. The graph above shows the first run in green starting negative and rapidly increasing until stabilizing around the 320 reward mark. The average run confirmed that this was not a one off instance instead was a consistent trend that could be established after 10 independent runs. The values given to epsilon and the rate yields a slow but increasing reward function that usually reaches between 300 and 350. The reason that the final result is not the exact same is due to the possibility of taking a random step due to epsilon. With my low epsilon the randomness effect is extremely diminished, yet we see that the final results are not the same. As explained before this is a result of the way the action is chosen and has a chance of being random instead of what the robot expected to do, resulting in lower or higher rewards in the next episode.