TBMI26 – Computer Assignment Reports Reinforcement Learning

Deadline – March 15 2019

Author/-s:
Martin Smelik (marsm914)
Stefano Toffol (steto820)

In order to pass the assignment you will need to answer the following questions and upload the document to LISAM. **You will also need to upload all code in .m-file format**. We will correct the reports continuously so feel free to send them as soon as possible. If you meet the deadline you will have the lab part of the course reported in LADOK together with the exam. If not, you'll get the lab part reported during the re-exam period.

1. **Define the V- and Q-function given an optimal policy. Use equations <u>and</u> describe what they represent. (See lectures/classes)**

$$\hat{V}(s_k) \leftarrow (1-\eta)\,\hat{V}(s_k) + \eta\left(r_k + \gamma\hat{V}(s_{k+1})\right)$$

$$\bar{Q}(s_k, a_j) \leftarrow (1-\eta)\,\bar{Q}(s_k, a_j) + \eta\left(r + \gamma\bar{V}(s_{k+1})\right) =$$

$$(1-\eta)\bar{Q}(s_k, a_j) + \eta\left(r + \gamma \max_a \bar{Q}(s_{k+1}, a)\right)$$

   The both formulas look very similar, however the main difference in these approaches is that V function is computed for a given policy, meaning that if we are given the optimal policy, with V function we can compute the values for each position by taking the combination of actual value and value from the next step. On the other hand Q function changes the values according to the combination of the value corresponding to the optimal move and maximum value in the following step as Q function has corresponding values for each action.

2. **Define a learning rule (equation) for the Q-function <u>and</u> describe how it works. (Theory, see lectures/classes)**

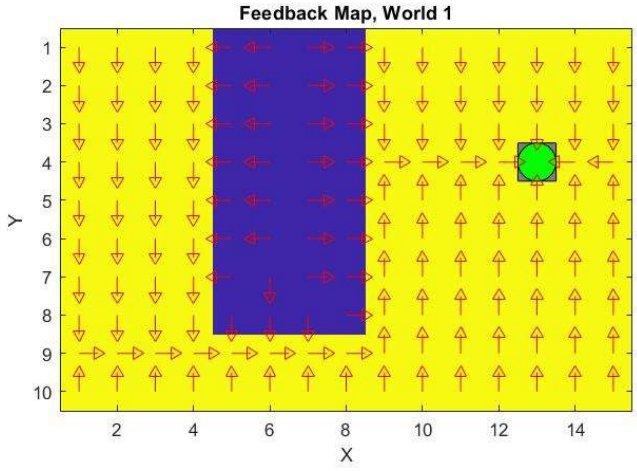
$$\bar{Q}(s_k, a_j) \leftarrow (1-\eta)\,\bar{Q}(s_k, a_j) + \eta\left(r + \gamma\bar{V}(s_{k+1})\right) =$$

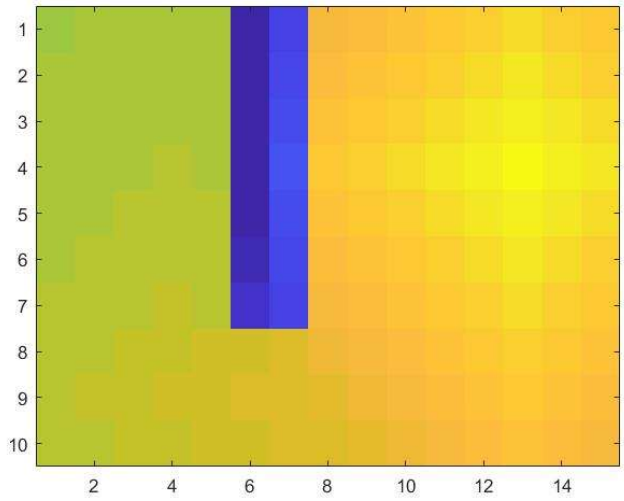
$$(1-\eta)\bar{Q}(s_k, a_j) + \eta\left(r + \gamma \max_a \bar{Q}(s_{k+1}, a)\right)$$

At each position s_k and for each action a_j we can recompute the Q value by using the formula above. The main idea of this formula is that we combine the actual value corresponding to the position and action with the maximum possible value we could obtain for the position we would get to by using action a_j. This is influenced by γ, the discount factor, a constant that we need to set in the beginning and which influence is explain in later excercises.

3. **Briefly describe your implementation, especially how you hinder the robot from exiting through the borders of a world.**
   Initially we set all values in Q table equal to 0 except of those corresponding to the "end of the world" where we choose -infinity for the action leading to exiting the world. Afterwards we trained the robot according to the pseudocode from the lecture notes, maximized the reward in the get policy function to get the best policy.

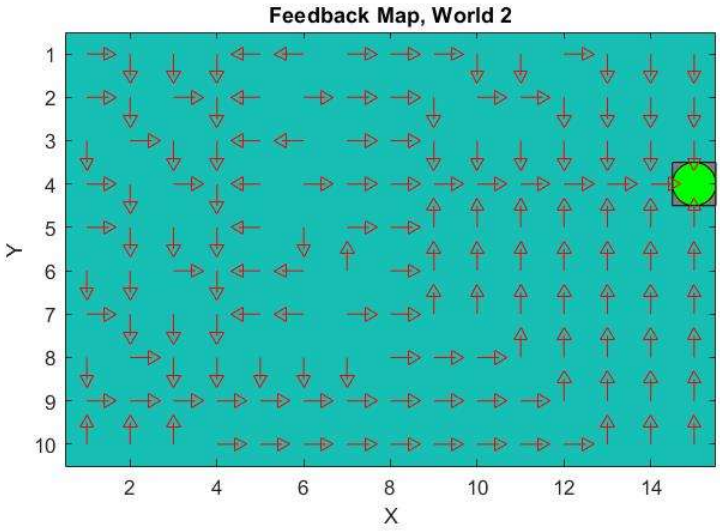
4. **Describe World 1. What is the goal of the reinforcement learning in this world? What parameters did you use to solve this world? Plot the policy and the V-function.**

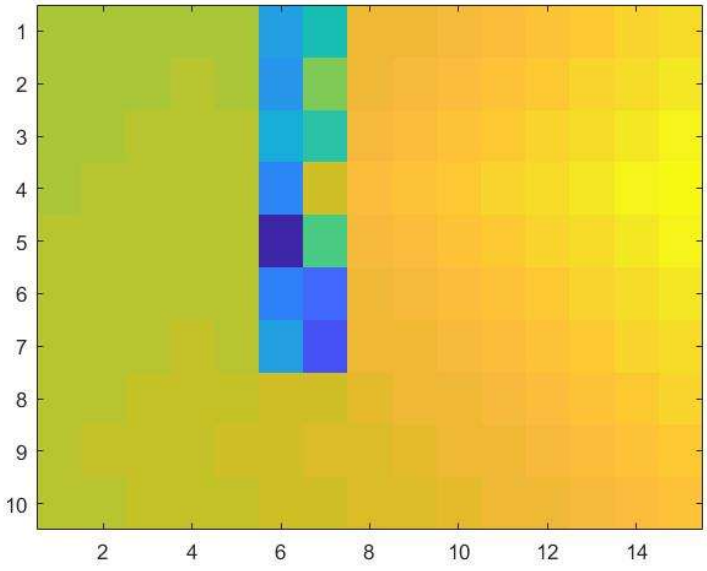




This world has two major parts – yellow, which is “cheap” to move in and blue which is “expensive” to move in and therefore we want to learn our robot to avoid this area. To train the robot, we used 1000 iterations, learning rate = 1, epsilon = 1 and gamma = 0.9. We can see that the V function is increasing if we approach the ending point and it is the highest at the place where we have to move through “expensive” area. Also we can see that the policy found is the best we could get.

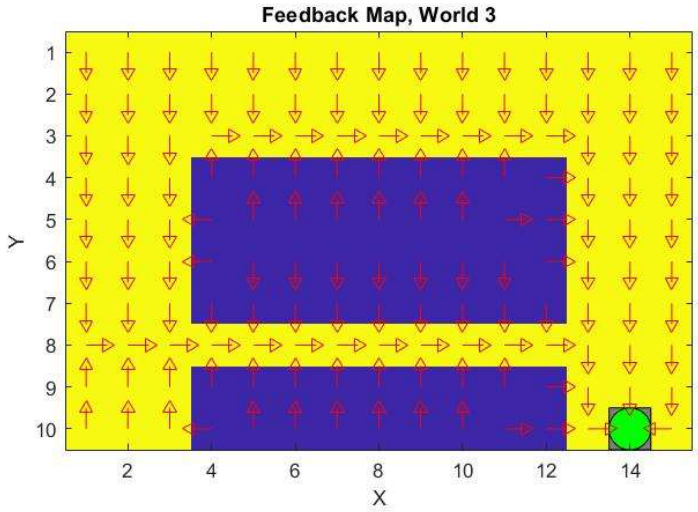
5. **Describe World 2. What is the goal of the reinforcement learning in this world? What parameters did you use to solve this world? Plot the policy and the V-function.**

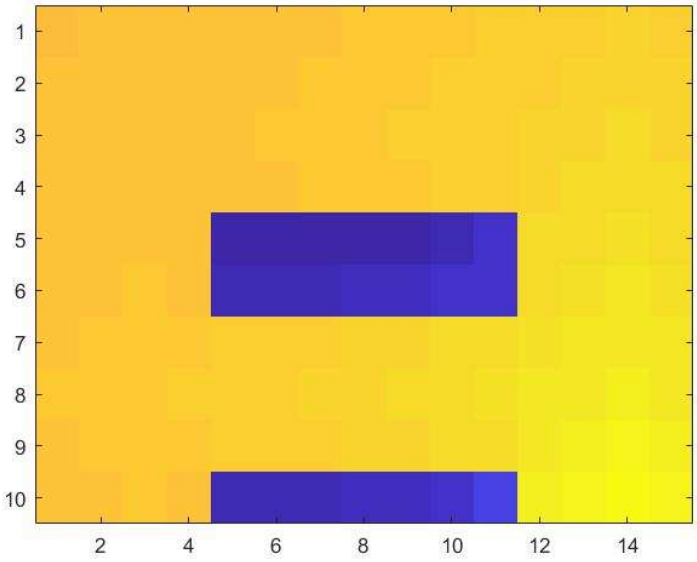




This world has again two major part with the same coordinates as world one, the main difference is that the blue area in the world 1 is “expensive” to move in world 2 just with probability 20%, therefore just in every fifth case (on average). Therefore we can see that the policy is very similar to the policy in world 1. We can see again that the V function increases in the “expensive” area. To train the robot, we used 2000 iterations, learning rate = 0.1, epsilon = 0.9 and gamma = 0.9.

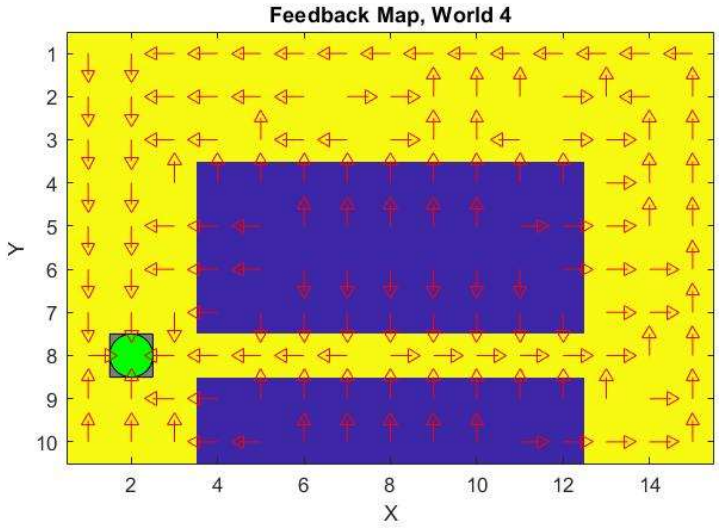
6. **Describe World 3. What is the goal of the reinforcement learning in this world? What parameters did you use to solve this world? Plot the policy and the V-function.**

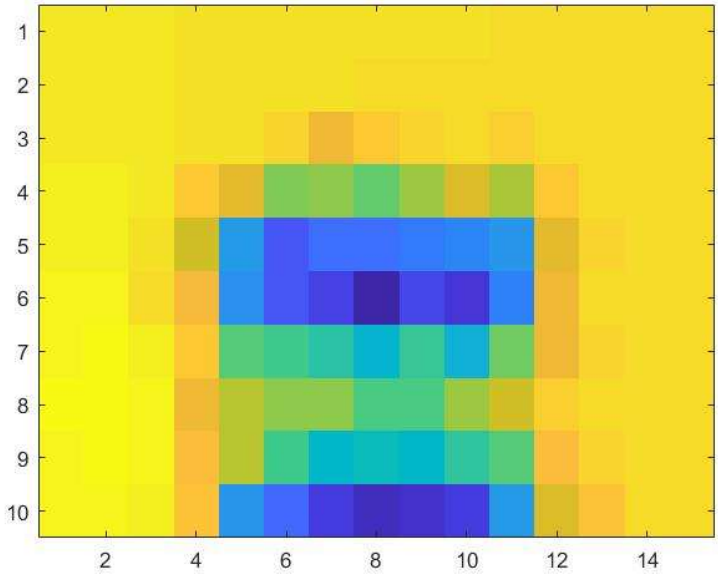




This world is very similar to world 1, with the difference that there are two areas and they are placed differently. We can see quite similar behavior as in world 1 (avoiding blue and finding the shortest path for yellow). The V function is increasing again when getting to the ending point and we used 2000 iterations, learning rate = 1, epsilon = 0.9 and gamma = 0.9.

7. **Describe World 4. What is the goal of the reinforcement learning in this world? How is this world different from world 3, and why can this be solved using reinforcement learning? What parameters did you use to solve this world? Plot the policy and the V-function.**



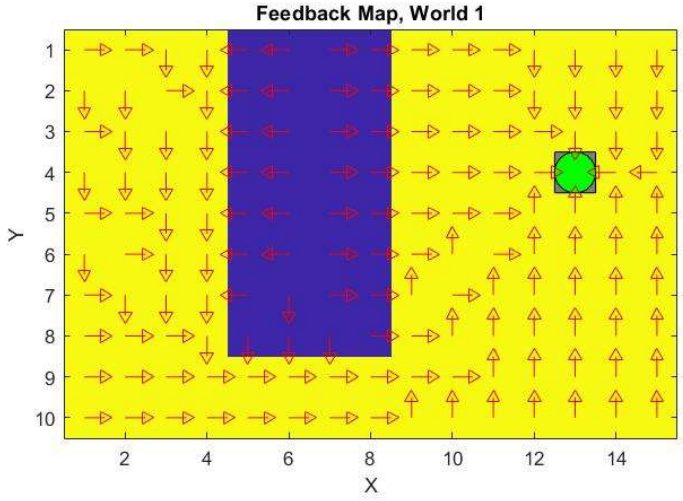


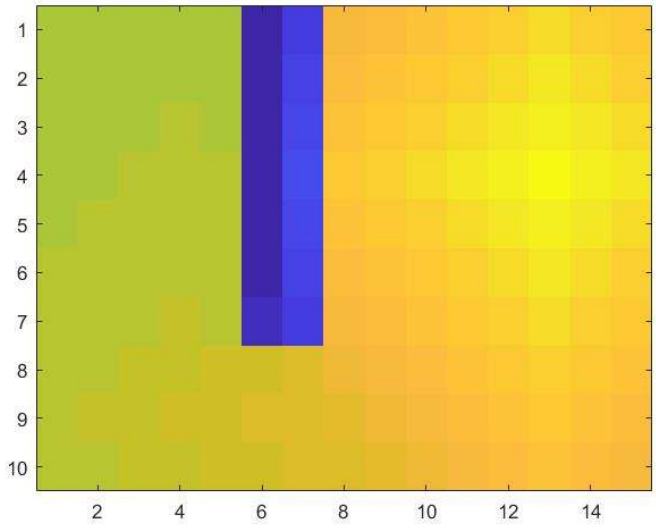
World 4 is the same as the world 3 with one exception, which is that with probability of 25% we are forced to move random direction (except of the one leading to the "end" of the world) That is why our robot was trained to avoid the shortcut between two blue areas as there would be very high probability of being forced to go to the blue area. We can see that also in the V-function that the "price" for going through the shortcut would be too high. For this experiment we used 5000 iterations, learning rate = 0.1, epsilon = 0.9 and gamma = 0.5.

8. **Explain how the learning rate α influences the policy and V-function in each world. Use figures to make your point.**

   Notice that we used slightly different notation and that is why we will talk about eta as a learning rate. For the following experiments we used the same setting as in the experiments above with the difference of decreasing eta to 0.1 for worlds 1 and 3 and increasing eta to 0.3 for worlds 2 and 4

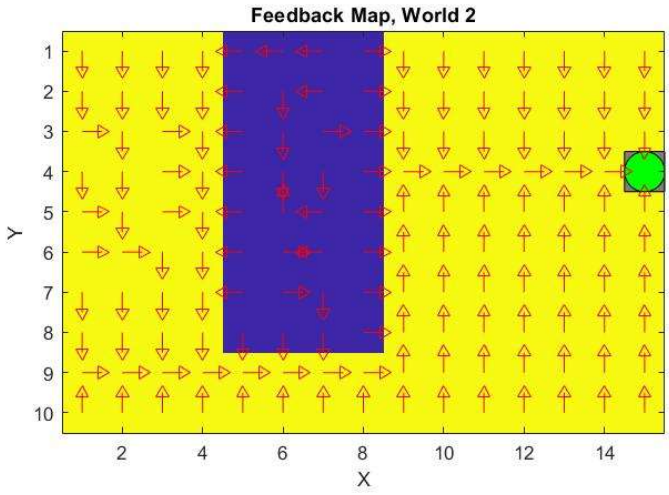
   World1:

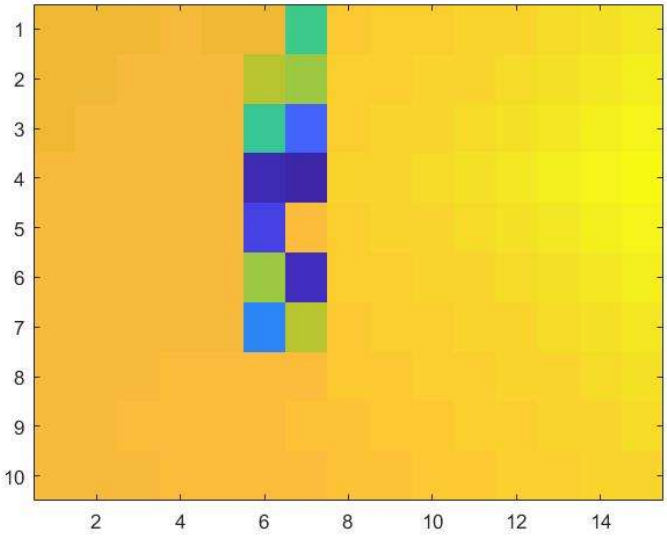




As the world is static lowering learning rate actually just slows down the convergency but otherwise doesn't make any major difference.

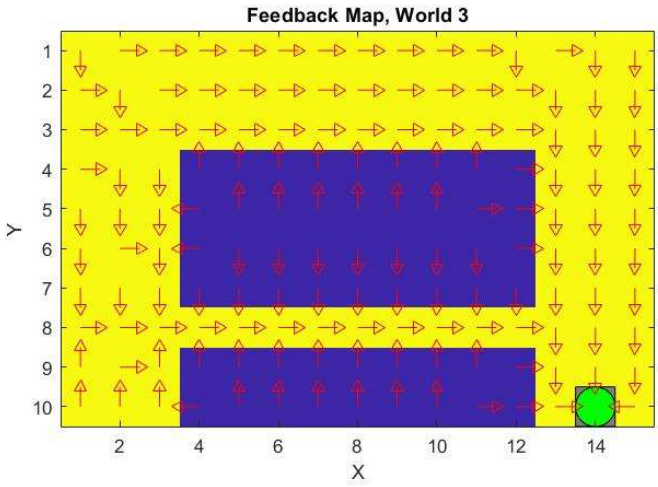
World 2:

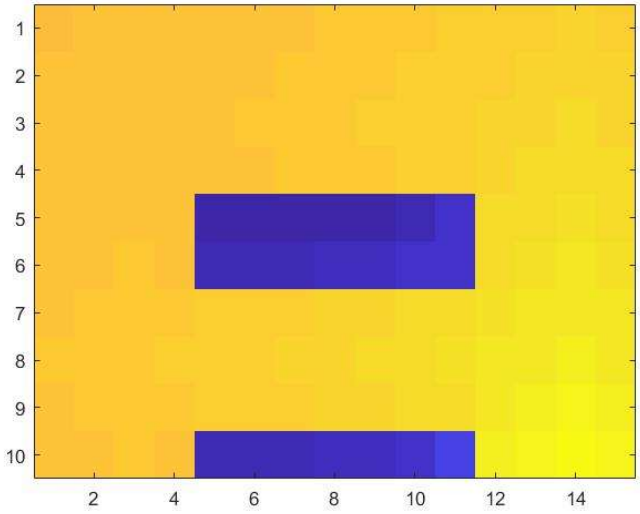




In this case the learning rate makes quite a difference as the world is not static and it might have a big influence in the "expensive" part of the world as for example at the point (6,1) we were probably quite "lucky" in some iterations in the end and therefore we learned the V value is quite high despite expecting much lower value

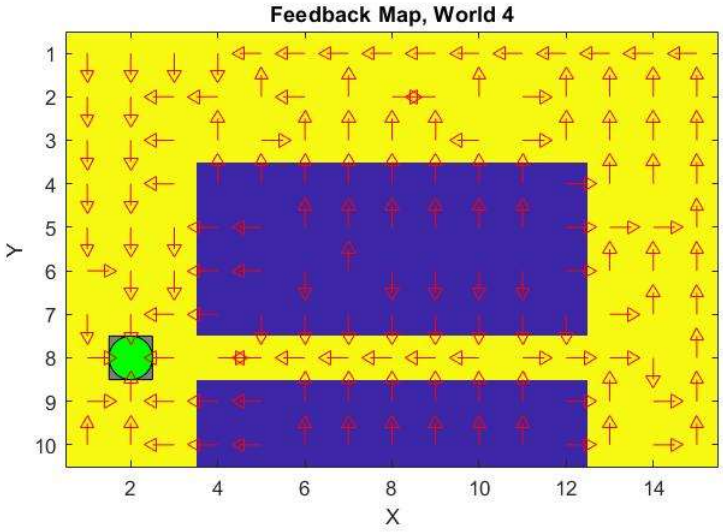
World 3:

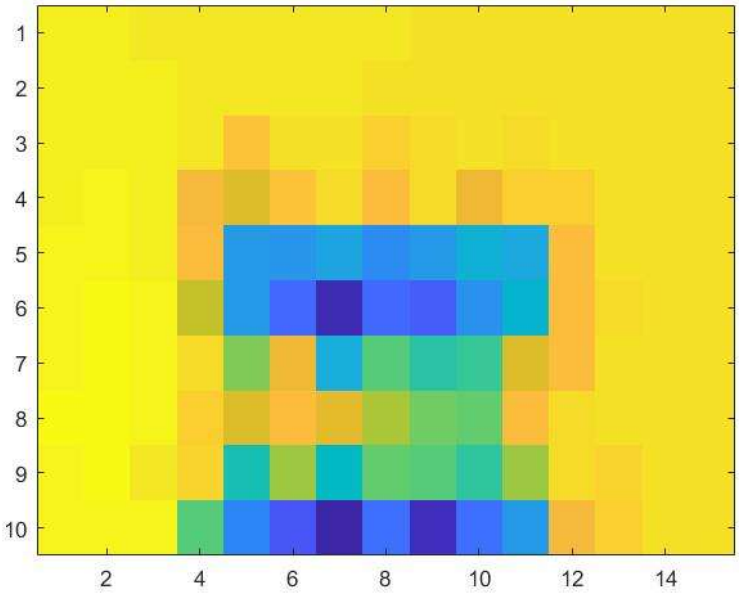




Again, for the same reasons as in world 1 there is not such a difference with lower learning rate.

World 4:



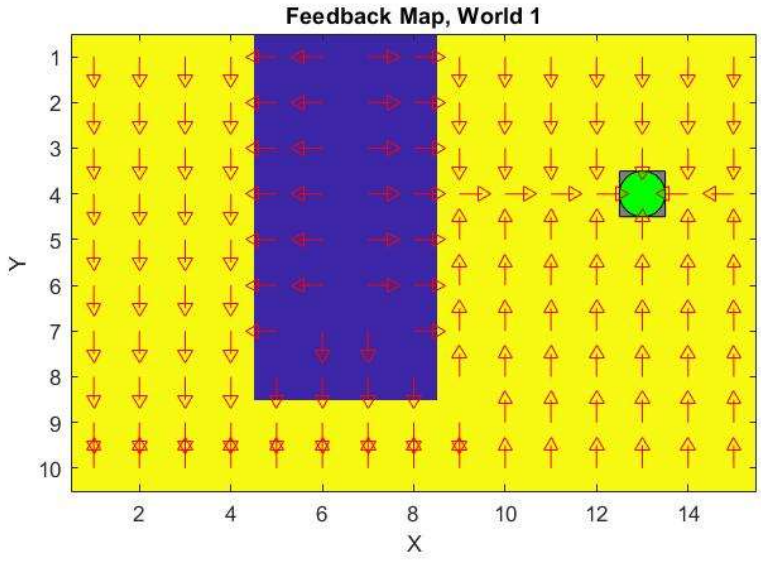


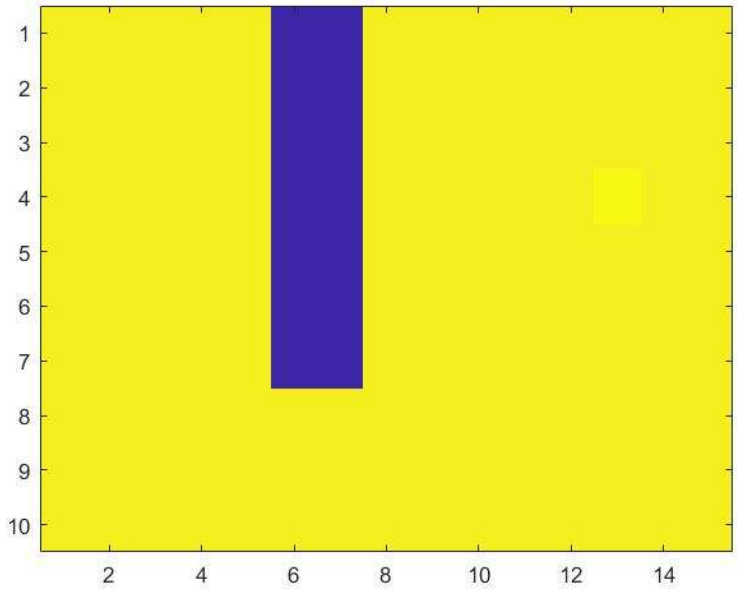
Again as in world 2, we don't have some random factor in our world and that is why increasing the learning rate might be slightly dangerous because then the policy might be very much influenced by the random movement.
Generally, we can say that high learning rate can often increase the speed of convergence but also increase the probability of oscillating around the best solution especially for more complicated cases as for example non static worlds.

9. **Explain how the discount factor γ influences the policy and V-function in each world. Use figures to make your point.**

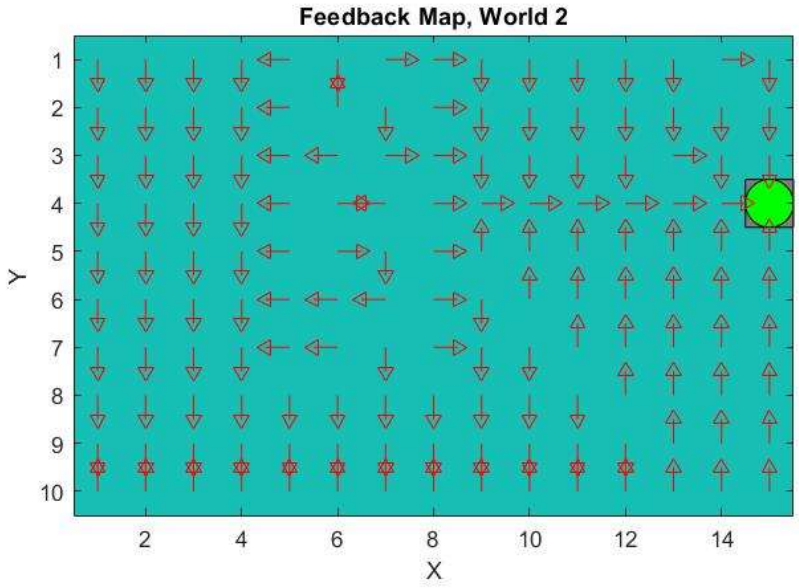
   For the following experiments we used the same setting as in the experiments in excercises 4-7 with the difference of gamma being reduced to 0.01 for worlds 1-3 and reduced to 0.1 in world 4.

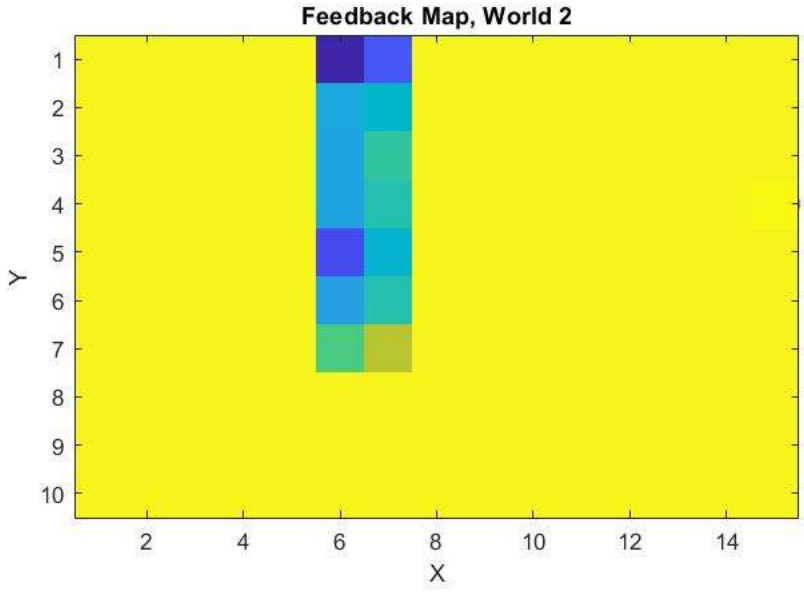
   World 1:



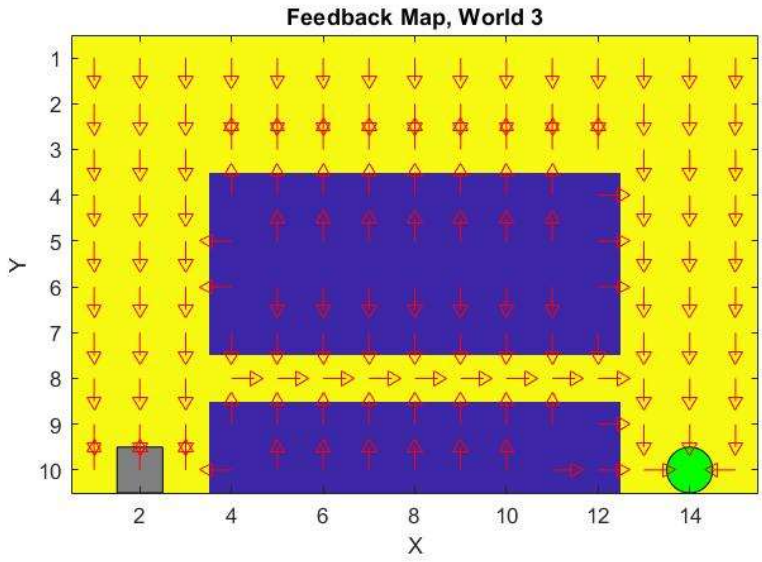


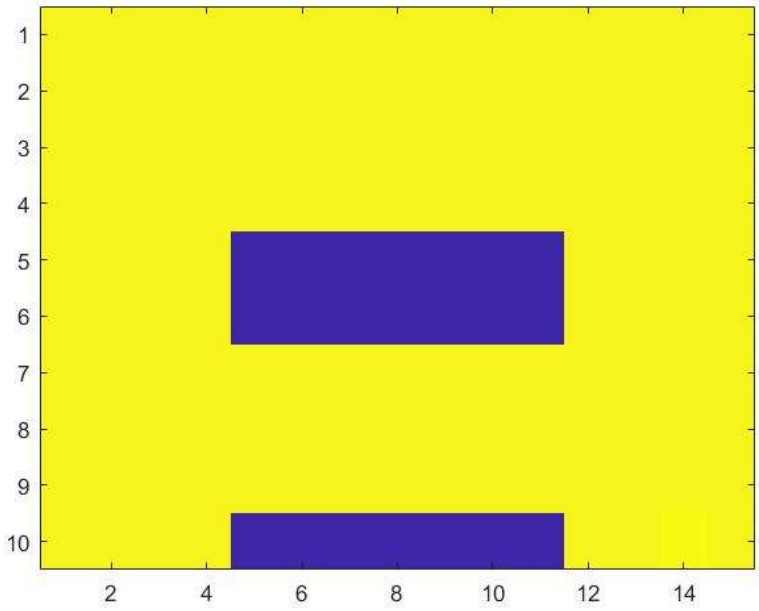
World 2:


Feedback Map, World 2
Y
X
1
2
3
4
5
6
7
8
9
10
2
4
6
8
10
12
14


Feedback Map, World 2
Y
X
1
2
3
4
5
6
7
8
9
10
2
4
6
8
10
12
14

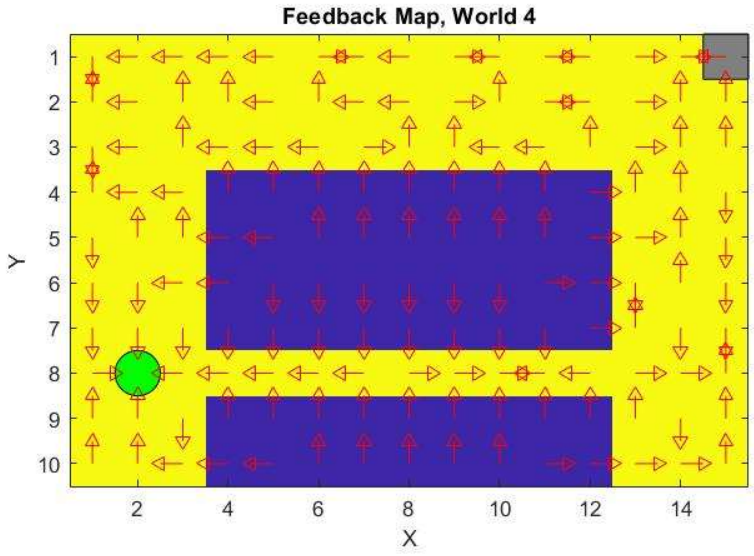
World 3:

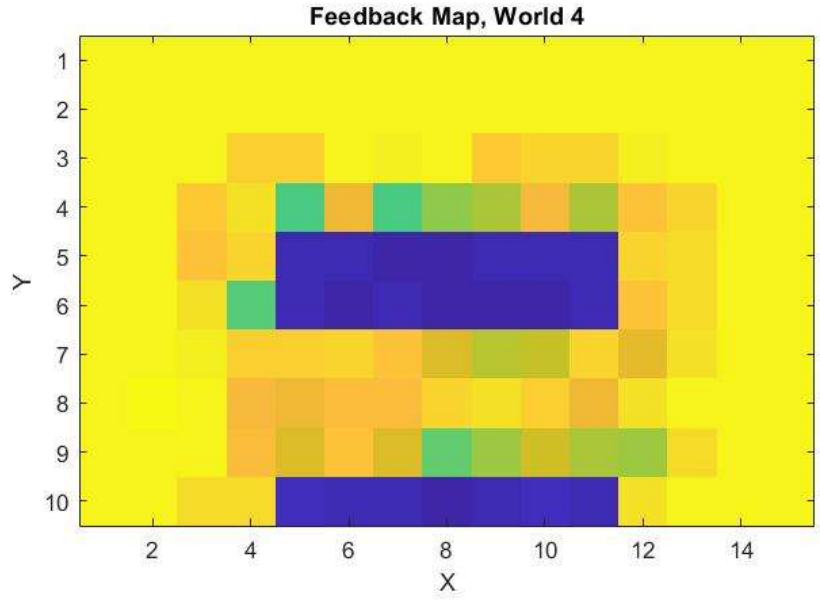




We can see the same phenomenon in all 3 worlds and that is because when being focused more on short term reward instead of long term, we might get troubles in the border areas as the price for all the action is very close to each other and that is why we can get infinite loops there. Also notice that we had to use very low gamma to get such a phenomenon

World 4:



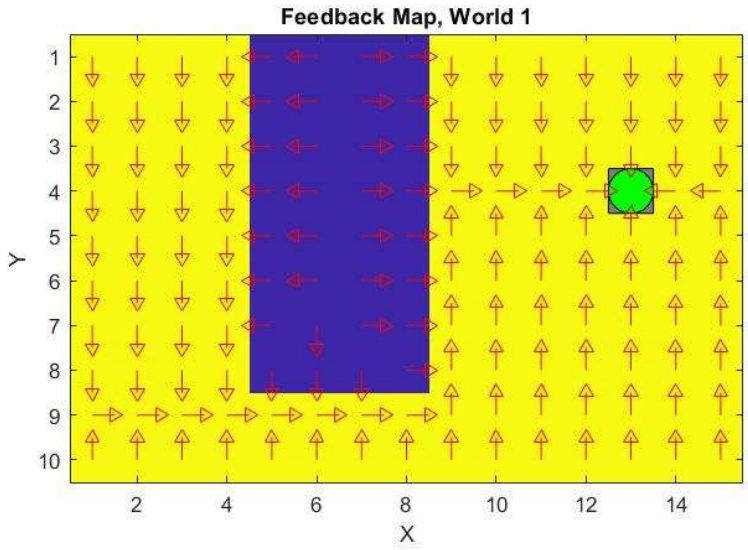


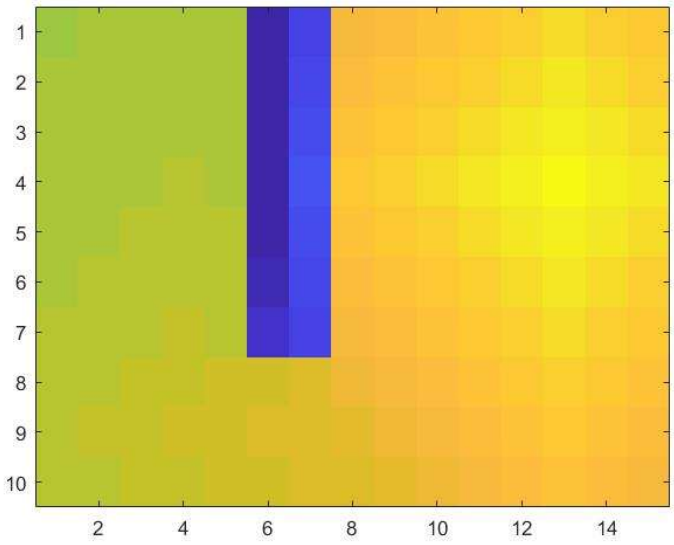
We can see again infinite loops which happen because of the same reason as in the worlds above, the only difference is that because of random movement we only need gamma to be 0.1 to see this phenomenon.
Generally it makes more sense to be focused on the long term reward in our case as our aim is to get to the goal.

10. Explain how the exploration rate ε influences the policy and V-function in each world. Use figures to make your point. Did you use any strategy for changing ε during training?

For the following experiments we used the same setting as in the experiments in excercises 4-7 with the difference of epsilon being 0 in worlds 1-3 and epsilon being 0.1 in world 4.

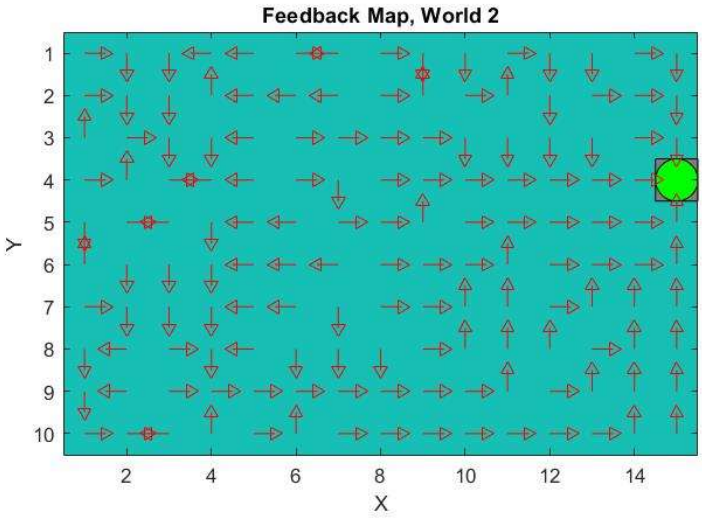
World 1:

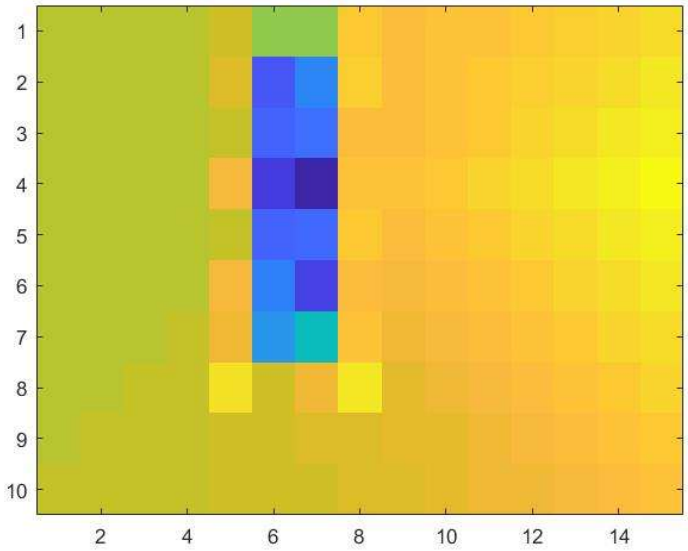




We can see that in this world setting exploration rate to 0 does not really make the difference as we always start from different point and that makes enough exploration for us

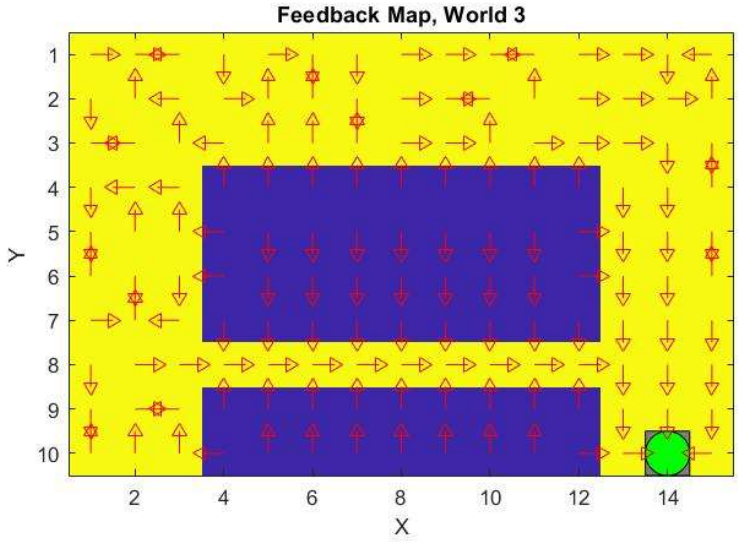
World 2:

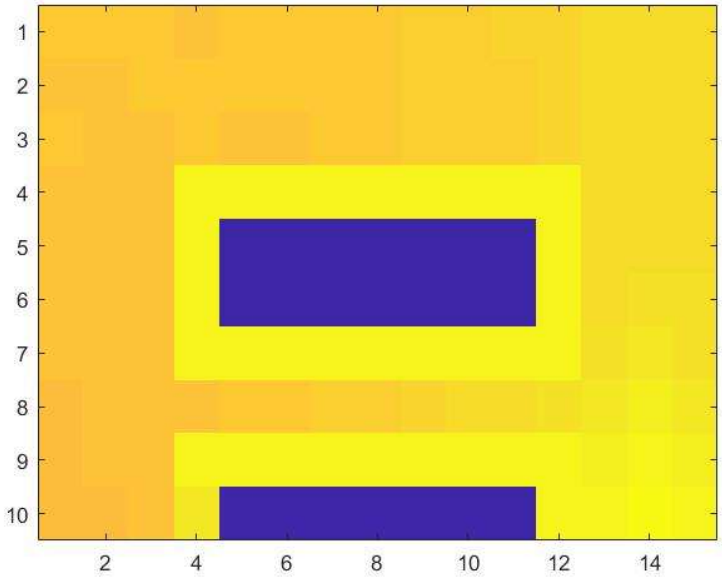




In this case the exploration factor seems to be missing as without it we hardly move away from the infinite loop once we get one which happens at the positions far away from the ending point. That is why having exploration rate higher is helpful.

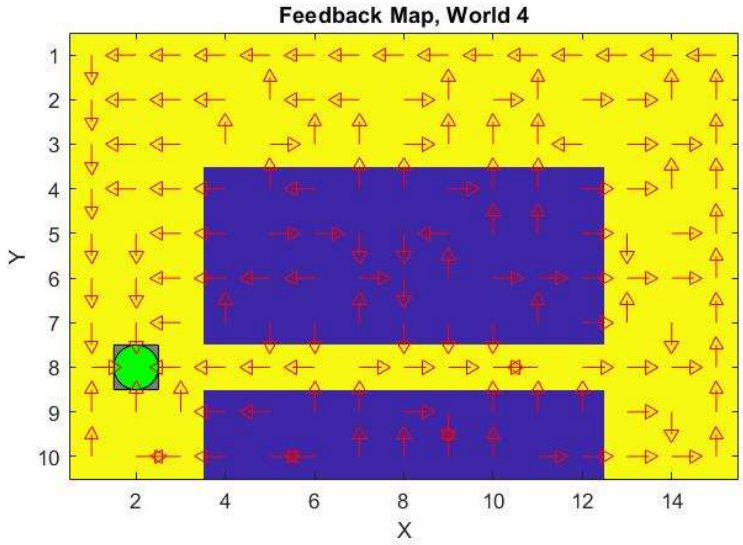
World 3:

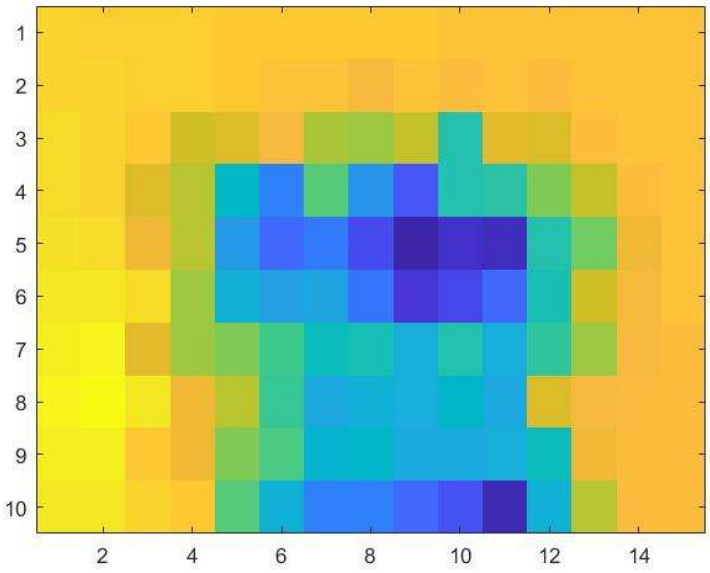




In this case we can see that the exploration rate is very important as we very often use “the shortcut” to get to the ending point and that is why we didn’t explore enough the areas in the top of the image.

World 4:





In this case the “random movement” is actually the same thing the exploration rate does and that is why even with quite low exploration rate we might get quite reasonable result, however still not as accurate as with higher exploration rate as we can see (because of infinite loops).

11. **What would happen if we instead of reinforcement learning were to use Dijkstra's cheapest path finding algorithm in the ''Suddenly Irritating blob'' world? What about in the static ''Irritating blob'' world?**

    We think that for the static “Irritating blob” world the result of the Dijkstra algorithm is actually something we are trying to converge to and using Dijkstra in this case should not make the difference. However for “Suddenly Irritating blob” it would depend on what “map” we get as in 80% we can get something different to what we get in the rest 20% of cases. Therefore if we were lucky in the train initialization, we would get 80% accuracy, otherwise just 20%.

12. **Can you think of any application where reinforcement learning could be of practical use? A hint is to use the Internet.**

    As we have seen on the lectures, reinforcement learning could be used for learning robots to play some complicated game where we can’t really decide what the good strategy is. Also reinforcement learning is used for manufacturing in companies, where robots must be trained to do some relatively easy movement like move some part from one place to another. Also it seems that this type of learning is used in evaluating the trading strategies.

13. **(Optional) Try your implementation in the other available worlds 5-12. Does it work in all of them, or did you encounter any problems, and in that case how would you solve them?**