A study of paths taken via Hippocampally dependent Navigation to Multiple Reward Centers, using the Temporal Difference learning rule with the Actor-Critic Model

Zachary DeStefano

***Abstract*—For this project, I attempted to see what paths a rat would take if the Morris Water Maze contained multiple rewards. The initial inspiration for this project was the layout of cities and the transportation networks used to connect them. As it turns out, the Steiner Tree problem captures the issue of how to connect multiple points with lines of minimum total length. I hoped to observe if a trained rat would use paths that look similar to what is contained in the Steiner Tree. In order to do this, I trained the rat via temporal difference learning in the actor-critic model. After training, I started the rat at random locations and recorded the paths it took to the reward centers. The diagrams of all the paths had high densities around the areas where the Steiner Tree is likely contained. This result was not conclusive enough to be used in approximating ideal paths but it did provide some insight into how the rat behaves with multiple rewards.**

I. Introduction

The Morris Water Maze is one where a rat is released into a pool of water and searches for a platform which will provide relief from having to swim. The platform is thus considered a reward in the brain of the rat. Due to the constant location of these platforms, it is optimal for the rat to learn this location. By observing the rat over multiple trials we can extrapolate information about its learning abilities.

If we wish to approximate what would happen if we did a physical experiment, we can use a model for its neuronal activity. For the purposes of this experiment, I used the neuron model presented in Foster et al [5]. There are a series of place cells in its hippocampus that each have their own actor and critic cells. The place cells fire at rates dependent on proximity to their preferred locations and send signals to the actor and critic cells. When the rewards are reached, the actor and critic values are updated using the Temporal Difference learning rule. Output from the actor and critic cells is used to determined the rat's next move.

In Foster [5], there was a single reward center for the rat. For this experiment, I wanted to observe what would happen if you used the same neuron model but had multiple reward centers. More specifically, I wanted to observe the paths that a rat would take to travel between all the reward centers. The hope is that something similar to a Steiner Tree [2], [3] would emerge.

II. Related Work

A. Steiner Tree Problem

The issue of finding the most efficient network connecting points has been formulated as the Euclidean Steiner Tree Problem which is specifically as follows: Given n points in a $2D$ plane, connect them via line segments such that the sum of the lengths of the line segments is minimized and every point can travel to every other point. Currently this problem is NP-hard meaning that it is not known whether or not there is an efficient algorithm to solve it. There is a polynomial time approximation scheme that has been found for it [2], [3].

B. Slime Mode Interstates

There was an attempt at solving something similar to the Steiner Tree problem using the behavior of biological organisms. Food was placed at locations imitating the 20 largest cities in the United States and then slime mold was released [1]. The paths sketched out by the slime mold ended up being similar to the interstate highway system and is thus likely similar to what the Steiner Tree would look like. This is a great example of how the aggregate of simple actions taken by a biological organism can yield useful results. My goal with the rat experiment was to see if we would get useful results taking the aggregate of paths a rat traverses after learning.

C. DA-STDP for foraging tasks

Besides the Actor-Critic Model, another commonly proposed model to simulate reinforcement learning is Dopamine-modulated Spike-time dependent plasticity (DA-STDP). This is a possible model for my purposes as there are two papers in particular where they used it to model an actor foraging for rewards just like in our rat experiments.

In a paper by Skorkheim et al [6], they model an actor foraging for food using DA-STDP. As it turns out, DA-STDP can be used to model the behavior effectively, however you have to impose additional constraints related to maintaining synaptic homeostasis in order for it to be effective. A similar experiment was done by Evans [4], where he modeled the

movement of a robot foraging for food in an environment.

These papers indicate that it may be possible to use DA-STDP to model the rat's behavior in my environment. However, there are two key differences between my set-up and those papers. First, in my experiment, the input is place cells thus the rat knows its global location. In those papers, the input is the immediate environment around the actor at a given point. The second major difference is that both experiments have many food items spread around the environment and it does not matter which food items the actor receives. For my experiment, I only had a few reward centers and I was aiming for the actor to visit all of them.

In addition to the key differences which hurt the applicability of those models to my work, there is another issue with DA-STDP. The fact that there are relatively few reward centers in my experiment would mean that the spiking would occur much less than in other papers that used DA-STDP. For this reason, training the rat to move around my maze would likely be impractical. I thus decided to stick with the Actor-Critic Model for my purposes.

III. Experiment Design

A. Rat Movements

For the real experiments there would be a rat moving around a circular pool. Since it was impractical to use a physical rat, I had a "rat" variable with an x-y location moving around a circle. In the model, the rat has place cells in its hippocampus and each of the place cells has a preferred location in the circle. Each place cell feeds into a critic cell as well as 8 actor cells. The place cell location, responses by the actor and critic cells, and final direction chosen by the rat was determined using the model specified in Foster et al [5].

There were two important things to consider when modeling the rat's movements: how much it changes direction and what to do if it hits the wall. For simplification, I assumed that a rat "bounces off" a wall if it hits it. This means that I add π radians to its direction of movement in order to reverse it. Unfortunately, it is possible that if the rat hits the wall from a certain angle at the right location, then applying this rule would cause it to bounce indefinitely. To overcome this problem, if the rat failed to be at a proper location after "bouncing" then I change its direction by $\frac{\pi}{2}$ radians instead of π radians.

The other thing to consider is how much the rat's direction is allowed to change in each time step. Here I deviated from the design in Foster [5] and chose my own method. I assumed that in each time step, the rat has only three choices: continue traveling in the same direction, move slightly to the left, or move slightly to right. Traveling in the same direction means that the angle of movement stays constant. Moving slightly to the left means that the angle increases by $\frac{\pi}{4}$ radians. Moving slightly to the right means the angle decreases by $\frac{\pi}{4}$ radians. In order to determine the rat's next move, I shifted the coordinate system so that the previous angle was the positive x-direction. If the vector representing the next angle had a positive y-value then I had the rat move slightly to the left. If the vector had a negative y-value then the rat moved slightly to the right.

While I always used the bounce rule, I did not always apply the restriction in how much the rat could change direction. When I was training the rat to obtain the actor-critic values, the rule was applied in order to ensure that realistic paths were used for learning. However when testing the rat and doing a mapping of the paths used, I wanted to ensure that we were observing what the actor and critic were instructing the rat to do. I thus did not apply the direction rule and only used the direction that the actor-critic values had dictated.

B. Rat Training and Rewards

Training of the rat heavily followed the procedure used in Foster et al [5] and many parts of the morris_water_maze.m code were employed. In each trial, the rat was released in one of 4 locations and the it moved until it found a reward center or 250 time steps had passed, whichever came first. Once the reward center was found, the actor critic values were updated. For my purposes, I had multiple reward centers which initially all gave a reward of 1.

In order to make sure a rat does not get too attached to one reward, their reward value decreased each time the rat reached it. I used two methods for this: decrease by constant amount and decrease by constant factor. The constant amount and constant factor varied for each trial. The reward center was not used once the value fell below a certain threshold. The rat was trained until a certain number of trials occurred or all the rewards had been spent, whichever came first.

For later trials, the initial reward center values varied. This was done in order to see if there would be an effect if we modeled the fact that cities have different populations. I took the population values and did a rescale and shift so that the initial reward values fell between 0.5 and 1.

C. Reward Center Locations

For the reward center locations, they were inspired by cities in the United States and Europe. For my United States map, I used New York, Chicago, Los Angeles, Seattle, and Atlanta. For the Europe map, I used Warsaw, Berlin, and Vienna. I found their latitude and longitude coordinates in signed degrees format and plotted them on a 2D plane, rescaling and shifting so they fit in the water maze. I had two trials with the American cities and two trials with the European cities;

D. Rat Testing

Once the rat was trained, I recorded the paths it takes when released back into the maze. The rat movement was exactly

the same as during training except that the actor and critic values were left unchanged and there were no restrictions on its direction. The rat also started at random locations in the maze. In order to model a random location, I used polar coordinates and chose random θ and r values that are within the realm of the circle and then converted the (r, θ) pair to its corresponding (x, y) pair. The random starting locations and lack of restriction on direction were employed in order to observe what paths the actor and critic values would tell the rat to traverse.

IV. Results

A. Initial Runs

For this experiment, it was critical to learn the actor gradients at each of the place cell. This would give a good indication as to the preferred direction of the rat at each point which indicates the direction it is most likely to travel. When there was only one reward, the gradients tended to point toward that reward after learning. For my experiment, I was hoping they would all point to the nearest reward out of multiple rewards. Unfortunately, this did not always happen.

A major issue that emerged occurred when the rat reached different reward centers a roughly equal amount of times and there was a distinct region between the reward centers where the gradient direction does not favor one reward center too strongly. When I observed the rat after this had occurred, instead of it randomly choosing one of the regions as I had hoped, it would instead become stuck between regions and move in a zig-zag manner near the border.

When the paths were mapped, there were mixed results overall. In some cases, the highest density of path lines occurs in the area between reward centers which is exactly what we want. In other cases, the density of path lines is vaguely close to that but the movement of the rat was too erratic to conclude anything.

B. European cities, constant initial rewards

For Figure 1, the reward center locations reflect the European cities mentioned earlier. For this run, I depreciated the rewards by a constant factor in each time step. The reward values were all uniform in the beginning. After training, the actor gradients tend to favor the rightmost reward center but still point to the other reward centers. When the testing was performed the rat moved toward each of the cities but mostly stayed in the path between the cities. An interesting result was that the area between the cities had the most dense concentration of path lines. Unfortunately, one reward center had a much more dense concentration than the others. For the critic weights, as can be observed in Figure 2, they were higher near the reward centers as expected, but areas nearest the rightmost reward had much higher critic values than the other areas.

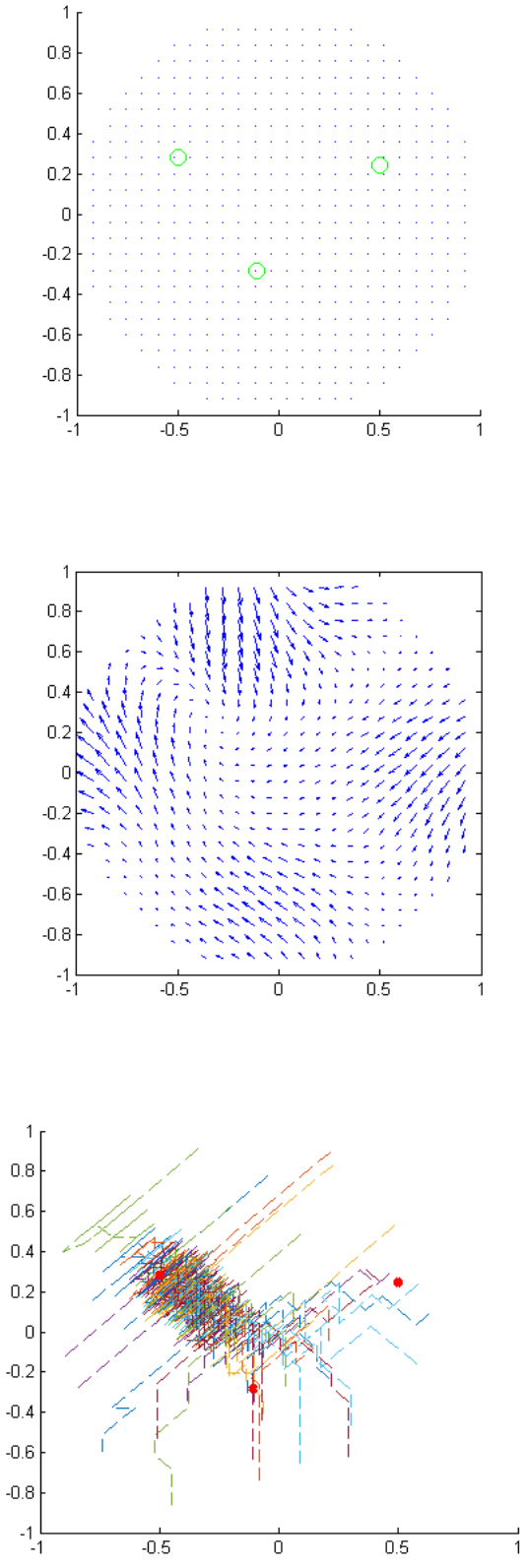

Fig. 1. Place Cells as blue dots and Reward Center Locations as green circles (top). Actor Gradient for each place in maze (middle). Paths traversed by rat with reward centers in red (bottom)

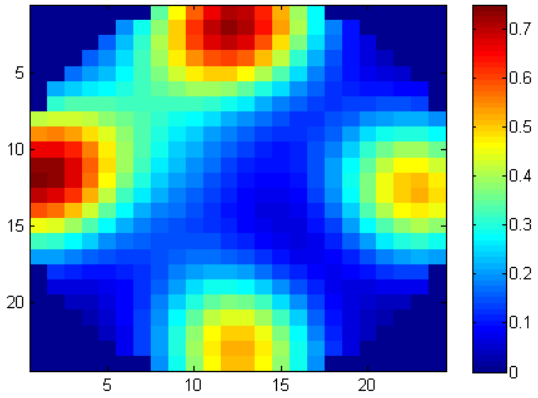

Fig. 2. Critic Weights for Maze from Figure 1

C. American cities, constant initial rewards

For Figure 3, the reward center locations reflect the American cities mentioned earlier. The initial rewards are constant but they decrement by a constant amount instead of constant factor each time the rat receives it. The results ended up quite similar to the case of European cities. There ended up being a distinct area between the reward centers heavily favored by the actor gradient so during testing that area is densest in terms of path lines. Instead of heavily favoring one reward center over another, the rat seemed confused in this area. In Figure 4, the critic values are highest on the right or left side and low in the middle which probably helps explain the fact that the rat had a hard time choosing there.

D. European cities, varying initial rewards

For Figure 5, I used the European cities but changed their initial reward values based on their population. Just like the previous trial with European cities, the reward values decreased by a constant factor each time the rat receives it. As can be observed, the results were not too different from before in that a strong preference exists for the area between the reward centers. The critic was similarly biased in Figure 6 toward one of the reward centers over the others. The paths lines are much more erratic though perhaps suggesting that not enough training is done when you vary the initial rewards.

E. American cities, varying initial rewards

For Figure 7, I used the American cities but changed their initial reward values based on their population. Just like previously with American cities, the reward values decreased by a constant amount. Again, the results were not too different from the previous trial with American cities. There is a heavy amount of activity in the area between reward centers. In Figure 8, the critic heavily favored the right side over the left side of the maze. Additionally, similar to the other trial with varying reward values, the paths lines ended up more erratic possibly due to lack of training.

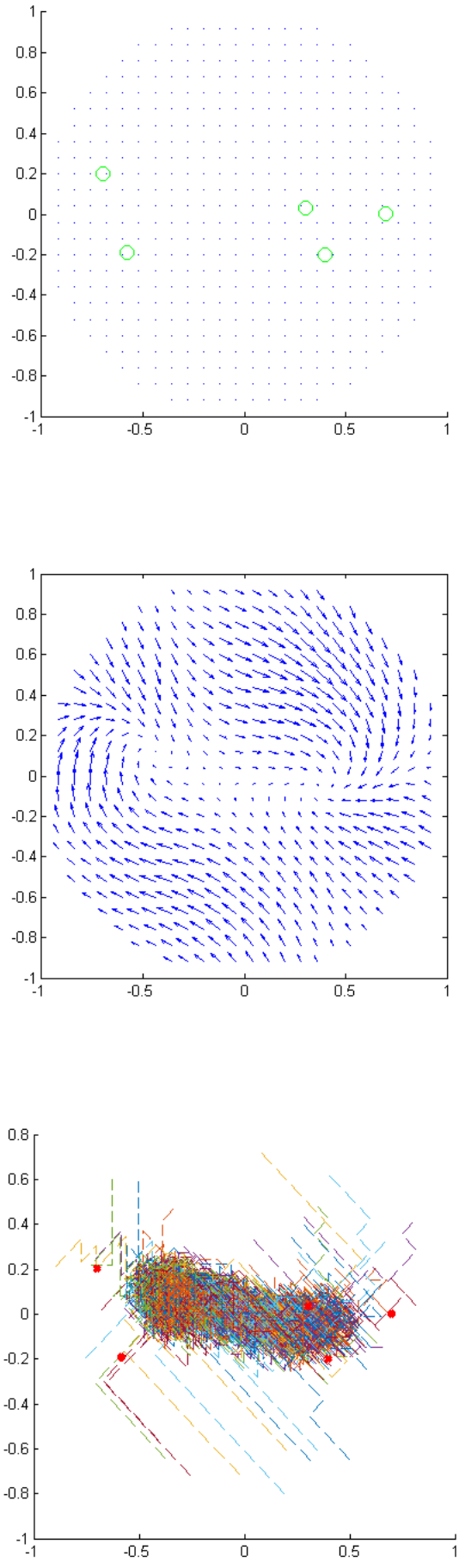

Fig. 3. Place Cells as blue dots and Reward Center Locations as green circles (top). Actor Gradient for each place in maze (middle). Paths traversed by rat with reward centers in red (bottom)

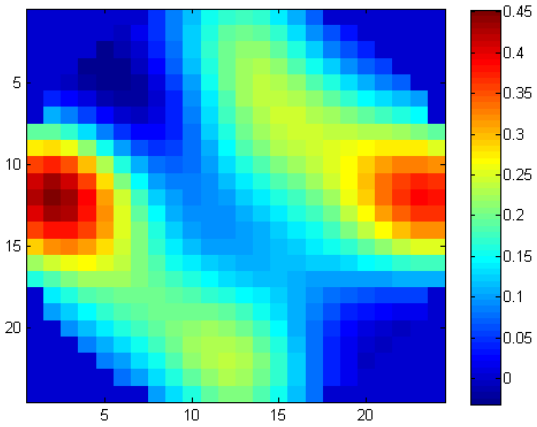

Fig. 4. Critic Weights for Maze from Figure 3

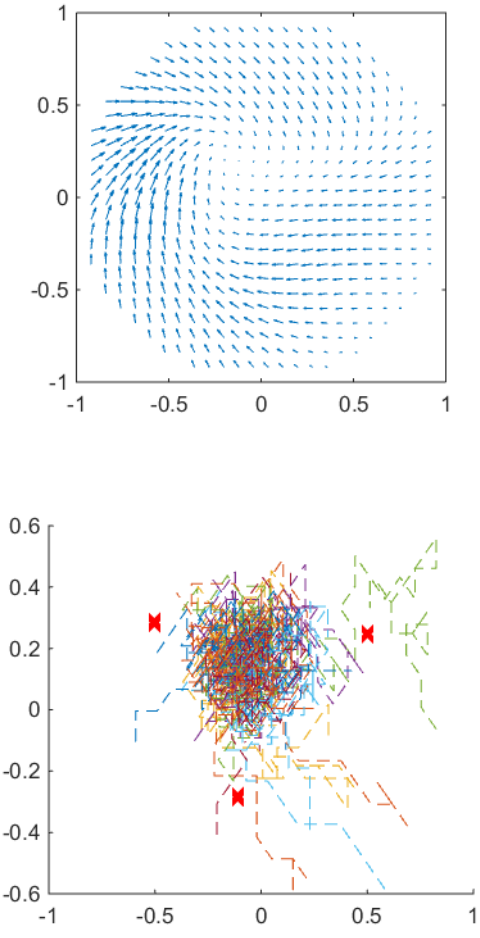

Fig. 5. Actor Gradient for each place in maze (top). Paths traversed by rat with reward centers in red (bottom)

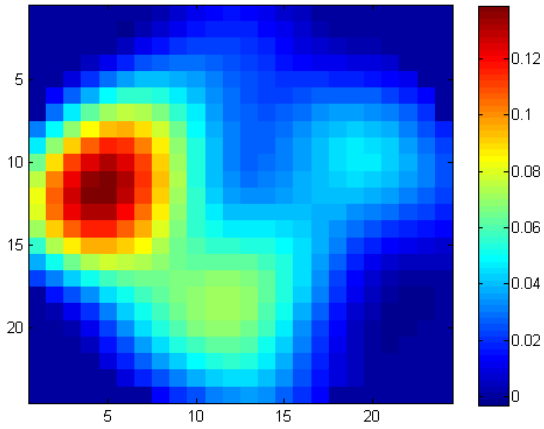

Fig. 6. Critic Weights for Maze from Figure 5

V. Conclusion and Future Work

Rat should not be used for path planning or anything related to planning networks or other instances of the Steiner Tree problem. The rat ended up with too much of a preference toward a particular reward center or it spent too much time in between reward centers unable to decide. Even after doing my best to overcome these adverse behaviors, the paths taken were not terribly informative. The best conclusion that could be drawn from the results is that the region with the densest concentration of path lines should contain a line segment in the Steiner Tree. Computing this line segment from the set of paths would be impractial though and the Steiner Tree approximation algorithm [2], [3] is definitely much more efficient than simulating the rat.

My hope with this project was that the actor cells would point toward their nearest reward center if quite close to one of them and in other cases, compromise directions would be mapped out. After that if you stopped the training and released the rat in random locations, it would take a variety of paths to the various reward centers. After mapping these paths, you would get a very dense concentration in a region that resembled a Steiner Tree or other optimal network. Unfortunately, while there were concentrations in areas that are likely part of the Steiner Tree, I would be hesitant to conclude anything about what the Steiner Tree should look like from these rat paths.

While there was no insight into the Steiner Tree problem through this project, I did gain insight into using the Actor-Critic Model with multiple rewards. The most ideal results occurred with uniform starting values and decreasing rewards when the rat reached a reward center. The rat does learn paths toward reward centers but further effort is needed to ensure that a rat does not stay stuck between reward centers. We also do not want the rat to have too strong a preference for one reward center over another.

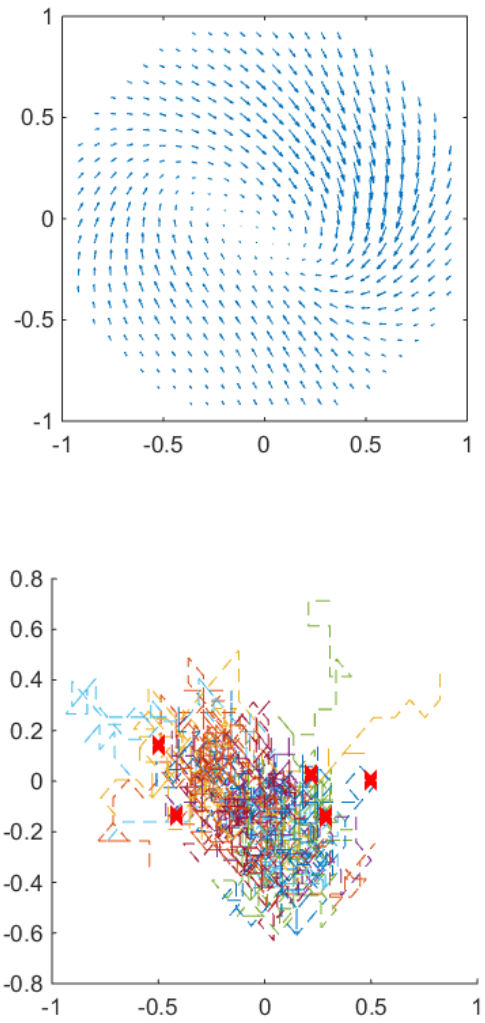

Fig. 7. Actor Gradient for each place in maze (top). Paths traversed by rat with reward centers in red (bottom)

The other possible extension of the multiple reward center work is using DA-STDP for the rat's behavior. The network would most resemble the one in Skorheim et al [6] however the input would be the place cell activity. The excitatory and inhibitory neurons as well as the output would stay the same for the most part with modifications to account for the number of place cell input neurons chosen.

REFERENCES

[1] Adamatzky, Andrew, and Andrew Ilachinski. "Slime mold imitates the United States interstate system." Complex Systems 21.1 (2012): 1.

[2] Arora, Sanjeev. "Polynomial time approximation schemes for Euclidean TSP and other geometric problems." Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on. IEEE, 1996.

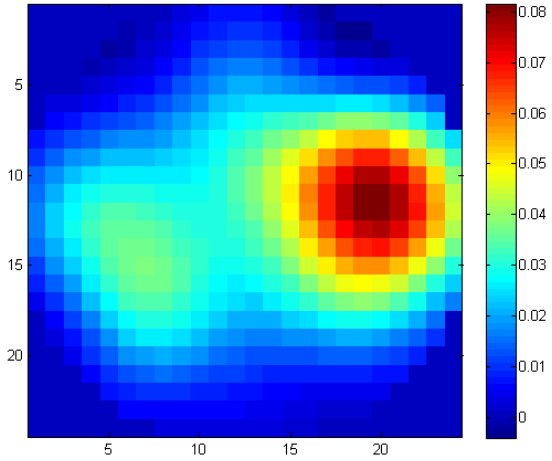

Fig. 8. Critic Weights for Maze from Figure 7

[3] Arora, S., "Nearly linear time approximation schemes for Euclidean TSP and other geometric problems," in Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on , vol., no., pp.554-563, 20-22 Oct 1997 doi: 10.1109/SFCS.1997.646145

[4] Evans, Richard. "Reinforcement Learning in a Neurally Controlled Robot Using Dopamine Modulated STDP." arXiv preprint arXiv:1502.06096 (2015).

[5] Foster, D. J., R. G. M. Morris, and Peter Dayan. "A model of hippocampally dependent navigation, using the temporal difference learning rule." Hippocampus 10.1 (2000): 1-16.

[6] Skorheim S, Lonjers P, Bazhenov M (2014) A Spiking Network Model of Decision Making Employing Rewarded STDP. PLoS ONE 9(3): e90821. doi: 10.1371/journal.pone.0090821