Games and AI Techniques - Project 2 Report

By Sean Argall and Sparrsh Nagdda

# Introduction

This report seeks to determine how to most effectively train an agent given a specific game through altering various rewards and observations. The game we are using for this report is a standard dungeon delver gamer. The player starts with 100 food points, which are consumed as they explore the dungeon. Along the way, the player can collect food or soda to increase the points they have for exploring, and must avoid monsters that directly attack this pool of points. Each level is eight by eight tiles and has anywhere from one to five food and five to nine inner walls. These walls don’t block player movement permanently, but must be destroyed by moving into them before the player can progress.

To simplify the training process, the agents we are training are unable to stand still, and must always take an action that will (or at least attempt to) change their position. Outside of extremely niche circumstances, it is never beneficial for the agent to stand still, and thus keeping it as an option for them is much more harmful than it is helpful.

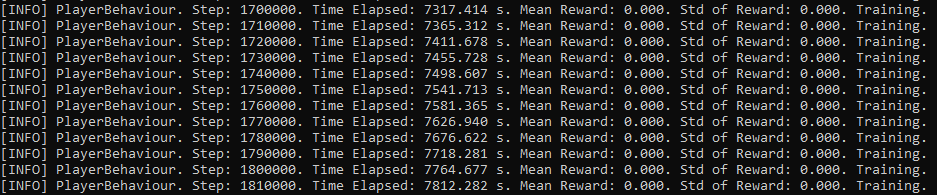
Reinforcement learning is a machine learning model which rewards desired behaviours and punishes undesired ones and is what we use to train the agents in this report. In general, a reinforcement learning agent can perceive and interpret its environment, take actions and learn through trial and error. The tools it has to explore this environment are the observations that we opt to pass to it, and are one of two focuses of this report.

# Establishing a baseline

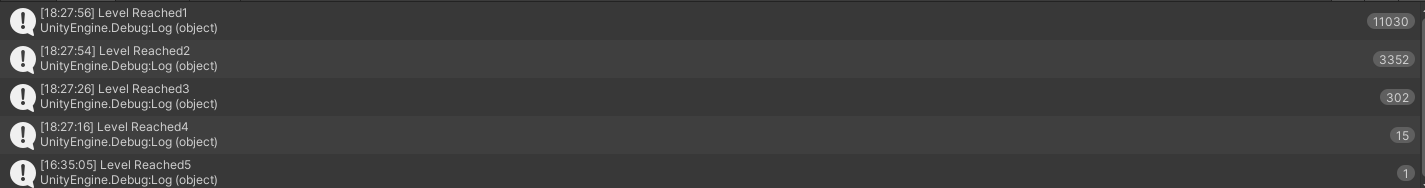
To understand the efficacy of our final agents, we first need to establish how well the agent performs without any training. The agent reached the fifth level exactly once after running for 1.81 million steps, or 14,700 games (1). 15 (0.1%) of games finished on level four, 302 (2.05%) finished on level three, and finally, 3,352 (22.80%) finished on level two. The agent finished on the first level 11,030 times. It’s also useful to understand how often the agent makes it to a specific level: 1.1% for the fourth level, 2.16% for the third level, and 24.88% for the second level.

From those statistics, the agent reaching level 5 once was very clearly an outlier and not something that can be expected from running almost 15 thousand games.

*(1) The level reached is logged when an agent dies, whether through a death caused by game logic, or by being terminated. Adding the total amount of “level reached” logs together gives us our number of games run.*



*Fig 1. Last 110k steps for the baseline run for a total step count of 1.81 million.*



*Fig 2. Total record of the level reached by the agents during the baseline run. Recorded on agent death.*

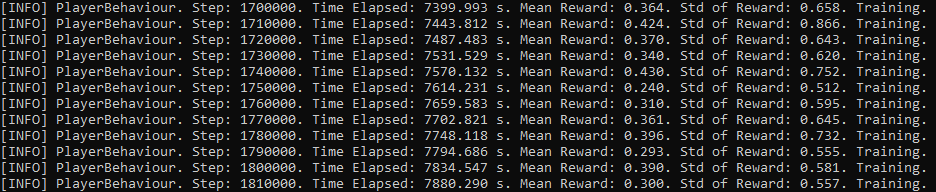
# Agent’s First Steps

The baseline agent has no understanding at all that progressing through levels is a good thing, let alone any idea that there are strategies that could be taken to do so more efficiently. For this project, we are measuring success as the highest level an agent can reliably reach. The baseline reaches level 2 with a 30.57% reliability, which is both a very poor amount of success and also too unreliable. A core of Artificial Intelligence is that generalised agents are “better” than specific agents. To reframe that, the more tasks an agent is capable of performing, the better it is. An agent that is hyper-specialised to perform a specific task is generally not well-designed. To keep our agent as general as we can, we will only add information to the agent as required - the bare minimum it requires is an understanding of what success is. As such, our first step is to give the agent a reward of 1 whenever it finishes a level.

1.81 million steps or 17,997 games later, there are multiple important observations to make. Despite that the agent managed to make it all the way to level 7 this time, the agent didn’t get past level one 73.91% of the time (compared to 75.03% of the time during the baseline run). For at least the last 100,000 steps, the average reward wasn’t improving - in fact, the average of the mean rewards is 0.343, which shows the average result as *worse* than the first of those 100,000 steps.



*Fig.3. Total records of the level recorded by the post-baseline run where we give the agent a reward of 1 when it finishes a level.*



*Fig.4. Last 110k steps of the post-baseline run where we give the agent a reward of 1 when it finishes a level.*

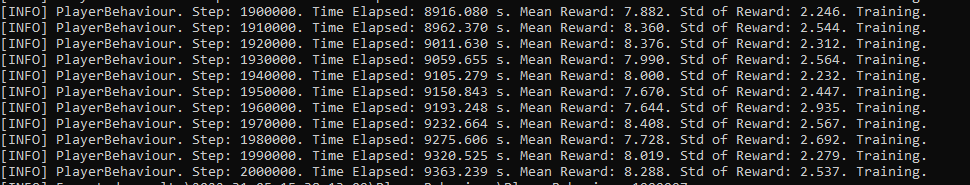
It is very much worth noting at this point that we had not yet taught the agent *why* it was getting specific rewards, and are instead just feeding it rewards in a way that the AI will see as totally arbitrary. This explains why there were no improvements with the above run compared to the previous run; although it did have notably improved success in reaching levels five to seven. How it managed this is unclear.

At this point, we added 48 observations to the agent, so that it could see the environment around itself. These observations were composed of 24 2d vector objects, each providing 2 observations. The first vectors are the player’s location and the location of the exit. The next 5 are the food locations, as a maximum of five food can spawn per level (if less is spawned, extra vectors that are zeroed out are provided). The same follows for the inner walls list, which is responsible for 9 vectors (18 observations). The remaining 8 vectors (16 observations) are allocated to enemy locations; while there is no maximum number of enemies that can spawn, the agents would have to make it to level 257 before the enemies start to outnumber the observations allocated to them. In this event, the observations will be allocated to the closest enemies - more on this later.

As we can see, these agents were significantly more successful! Indeed, of the 20,033 games completed in the 2 million steps taken, only 276 (1.85%) died on the first level, with only 2,187 (10.92%) dying in the first three levels, or 26.68% in the first five levels. That means that almost 75% of agents fared better than any agent did in the baseline runs! Additionally, 0.38% of agents made it to level 15 and beyond. Increasing that percentage is our main focus for future training.



*Fig. 5. Total record of levels recorded by the 48 observations run which include the player’s location, exit location, food location, walls location, and enemy locations.*



*Fig. 6. The last 110k steps of the 48 observations run as described above.*

# 

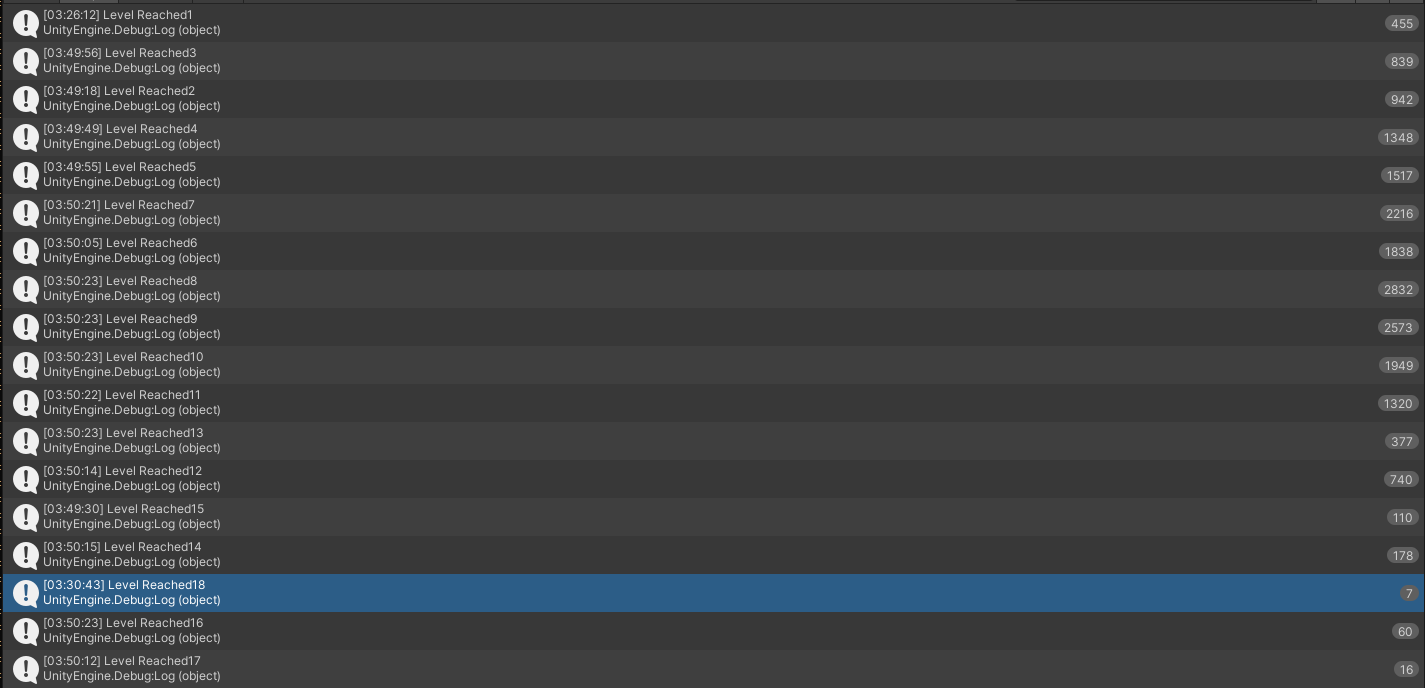
*Fig. 7. Graph of mean reward vs the step count for the current model. In this case, the reward is equivalent to the level reached.*

# Further Education

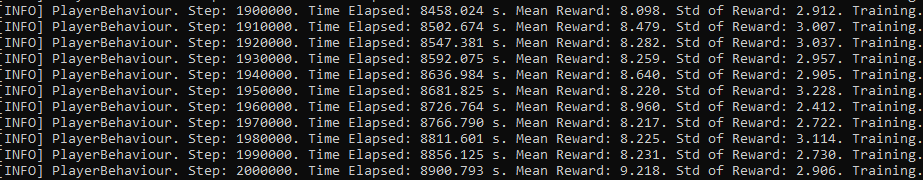
As discussed before, the agent should be generalising as much as possible. Sadly, technology is not at the stage where you can just give a problem to an agent, and it will figure out appropriate solutions, let alone rewards or how to define success. While a generalised agent is wonderful, it’s also usually unrealistic. Our next test gives the agents rewards when they collect food or soda, and punishes them for making contact with enemies. It’s worth noting that while the punishment is relative to the damage a specific enemy does, the agent isn’t able to observe differences in enemy type. However, food and soda are organised within the observations, so the earlier of the observations within that category are more likely to be food, with lower items being more likely to be soda.

It is important to recognise that adding rewards without thought is unlikely to yield valuable results. While for the sake of completeness we eventually run tests with a negative reward for steps taken, we don’t expect it to be particularly useful for training the agent. With a map space of 8x8 or 64 tiles, an agent that explores *every single tile*, while not desirable behaviour, is only going to expend 64 food in doing so. Ignoring enemy and inner wall positions, the agent needs an average of 8, or a maximum of 16, steps to reach the exit from any starting location. Whether exploring an entire level aimlessly or travelling efficiently, the collection of food is more significant to survival, in addition to avoiding enemies. An average of 3 food/soda spawns each level, with 15 points of food value for each item. As long as the agent avoids enemies and collects one to two items of food, it should come out of each level with a net positive gain in food. As such, the importance of a step is not that it has been taken, but rather where it puts the agent. Therefore, training it according to which locations are important is more significant than training it to take fewer steps.

Compared to the 0.38% from the previous run, 1% of the agents reached level 15 or further, although none made it past level 18. This is an improvement over the previous test, but not a drastic one.

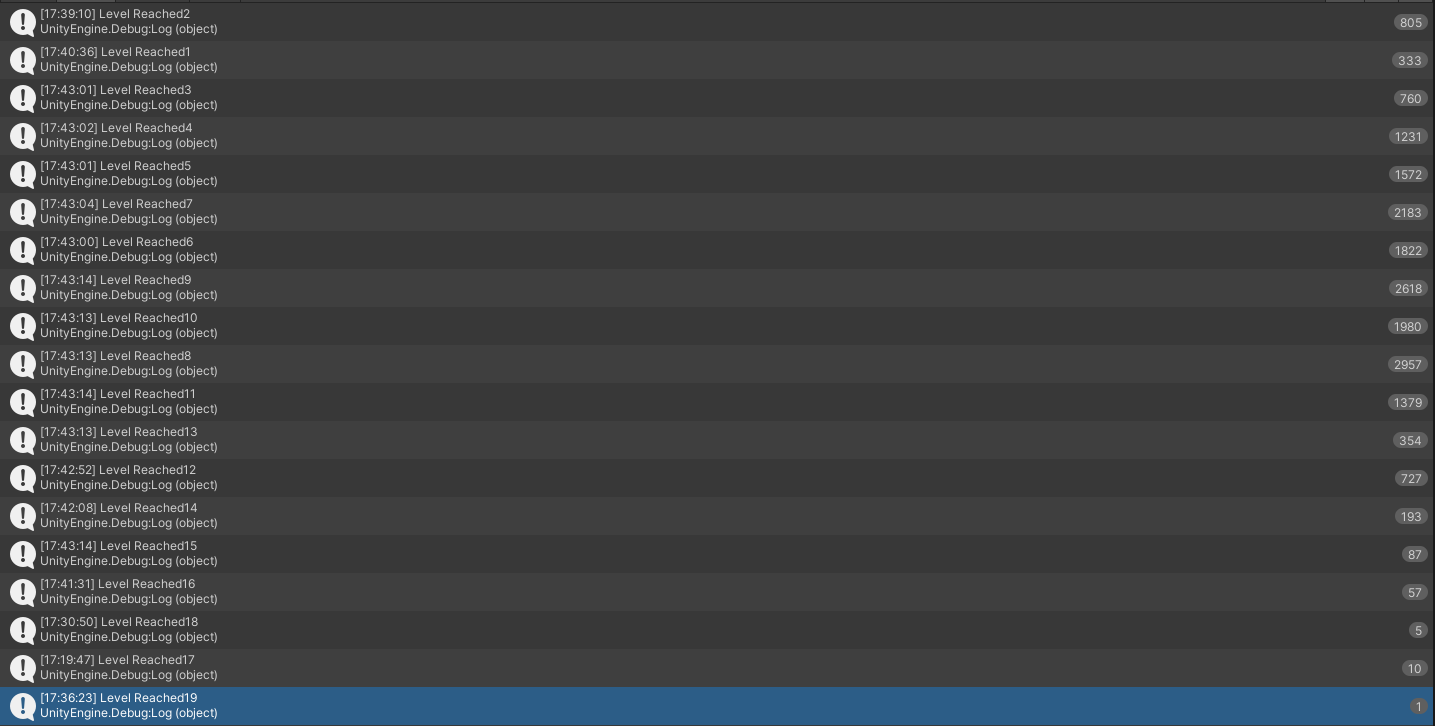


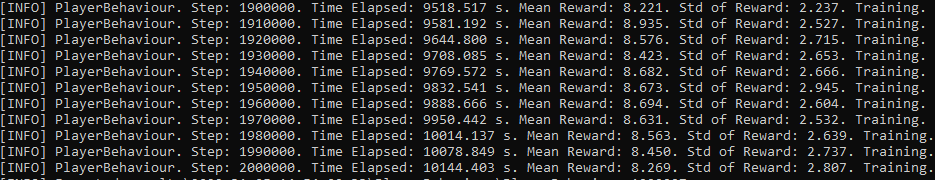
*Fig. 8. Level record for the model which rewards food and punishes contact with enemies.*



*Fig. 9. The last 100k steps of the model described in fig. 8.*

While this next one was mostly an intermediary to better compare changes and make changes more gradually, there’s still some information worth covering here. First, this run added 3 observations - the mean distance to enemies (float, one observation) and the position of the closest enemy (vector, two observations). Secondly, while this run was worse than the previous one for levels reached (only 0.83% reached level 15 or better), it did reach level 19. This juxtaposition of improving in some regards and being worse in others sets an interesting position to explore from. It’s very easy for models to get trapped in local minima, so a change like this suggests an escape from one of these areas (note: the term ‘local minima’ is typically used to refer to the learning of the agent in a specific model. We are co-opting it here to refer to the development of the agent).



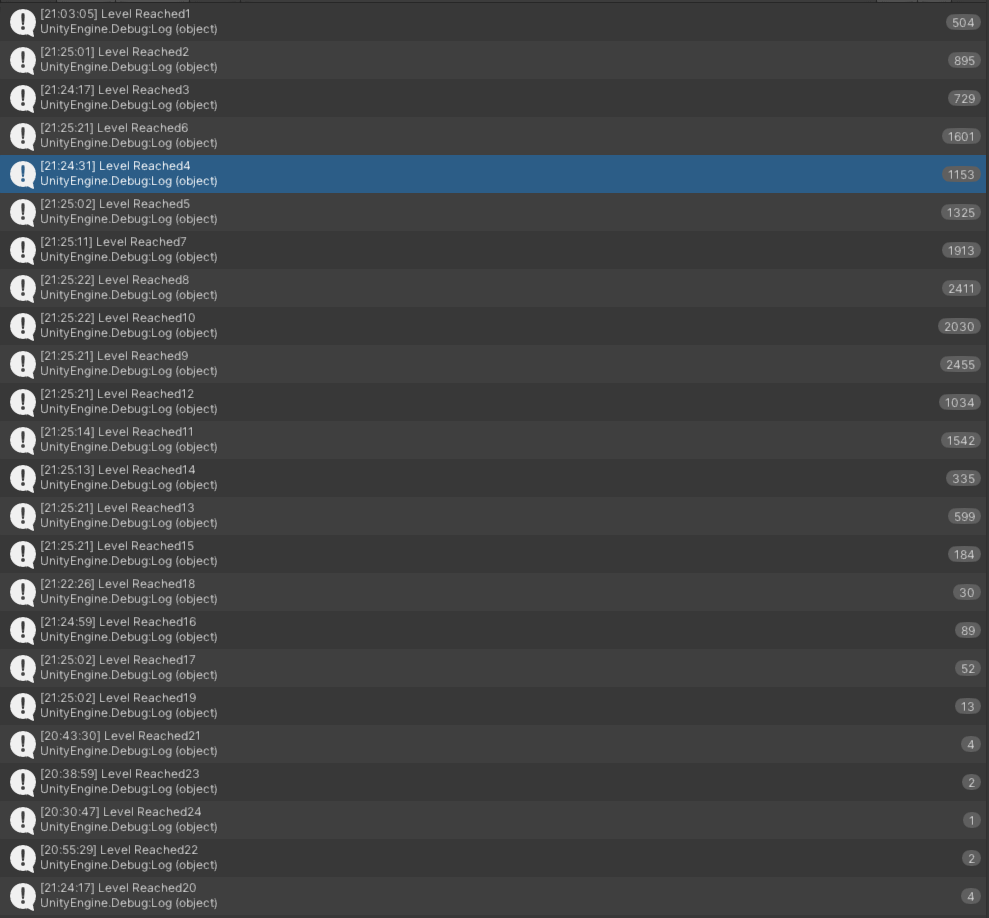
*Fig. 10. Level record for the model where 51 observations were added. These include the player’s location, exit location, food locations, wall locations, and enemy locations from the earlier model. Additionally, it also includes the mean distance to the enemies and the position of the nearest enemy.* 

*Fig. 11. Last 100k steps for the model in fig. 10.*

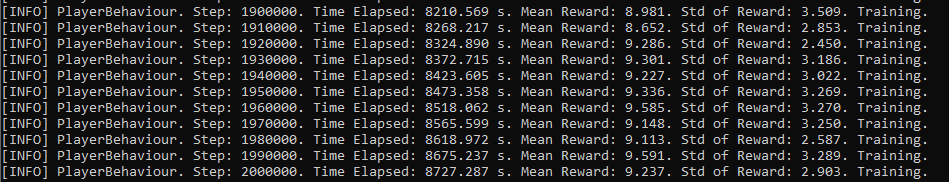
The improvements between the second and the third (and fourth) runs, while notable, are not as significant as they ideally would be. As such, we had to make a change in our approach. Further analysis of the results we had received to this point showed that the agent had no issue identifying that the ultimate success was progressing to the next levels. Taking the average of 8 steps to complete a level, and a starting food balance of 100, agents would at most reach level 13 without collecting any food. As our agents can reach level 18 (with varying amounts of success), they’ve learned that collecting food is a useful trait, even before a reward was associated with it. Giving the agents a reward tied to food seemed to increase how quickly the agents learned to grab it but didn’t teach them anything they wouldn’t eventually learn.

Rather than teach the agents something they were already learning, we decided to remove the rewards tied to food gain, as well as the penalties for making contact with an enemy. Additionally, we removed observations tied to providing the agent with perfect information for the game, except for inner walls. We replaced those with observations for the position of only the closest food and enemy and kept the average enemy distance observation. While knowing the position of all enemies is useful, it’s much more important that the agent learns to avoid the enemies close to it. Enemies that aren’t in the immediate vicinity aren’t an immediate concern. Of course, if several enemies are close to the agent and only one position is given, the agent might be lacking vital information. Providing the agent with the average enemy distance allows it to learn that being close to multiple enemies is indeed not a desired behaviour, and should encourage it to rush for an exit rather than collect food. As discussed earlier, one or two food items depending on the level are sufficient to cover the costs of succeeding at a specific level. While on early levels with few enemies, it is very likely to be worthwhile to procrastinate and collect all the food on a level, the frequency of enemies at the higher levels makes food collection a much riskier prospect. Therefore, informing agents of only the food closest to them should decrease the instances of agents hunting for all food, instead only seeking food along their path to the exit.

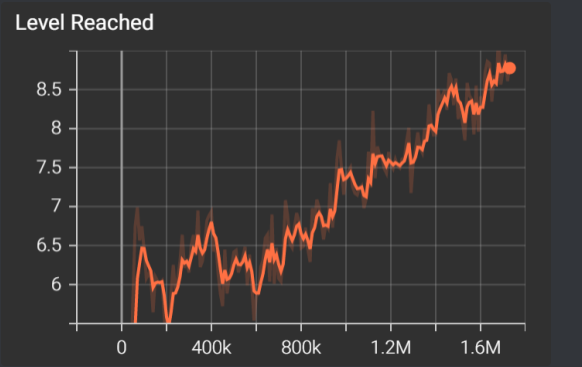
This run achieved exactly what we were hoping for; by forcing the agents to focus only on the items close to them and not otherwise distracting them, the agents made another drastic improvement, and reach their highest level of 24! 2.02% of agents reached level 15 or better, which doubled the previous record. The mean reward also increased, despite the reduction in the number of rewards provided (although penalties were also removed) by a full point. At its peak performance, this agent is one we are proud of. While the number of agents that survived to level 20 is limited, this demonstrates that even without rewards associated with specific behaviours, the agent can learn to collect food and avoid enemies. To reach level 24 assuming an average of 8 steps to complete a level, the agent would have to collect at least 100 points of food more than it loses by contacting the enemy. Levels 16 to 32 all have 4 enemies; assuming an even distribution of them across the map, each enemy can control a 4x4 space. With inner walls mixed into that, the agent is very close to an enemy at all times, and staying out of their reach is not as simple as it seems.



*Fig. 12. Level record for the model where we removed the rewards for food and the penalty for engaging with enemies. These left us with 27 observations which include the position of the closest food as well as the enemy, and the average distance from the enemies.*



*Fig. 13. Last 100k steps for the model in fig. 12.*



*Fig. 14. Average mean score vs Step Count for the model in fig. 12.*

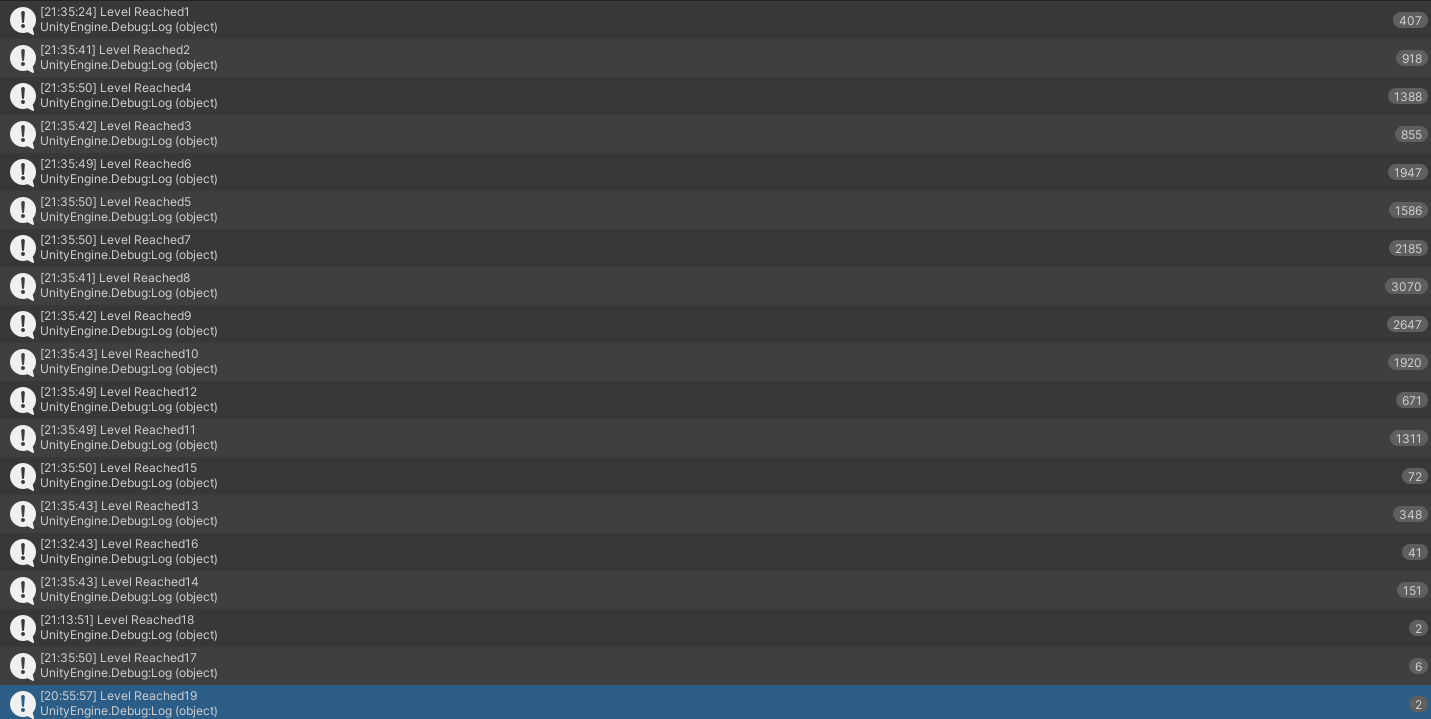
# Confirming Our Learnings

In reinforcement learning, since a small change in observations can drastically change the result, we tend to make minimal differences between runs to better understand what produced the difference in results. However, the previous run excluded two things simultaneously - rewards for food items, as well as the number of food observations (just the closest compared to a full list).

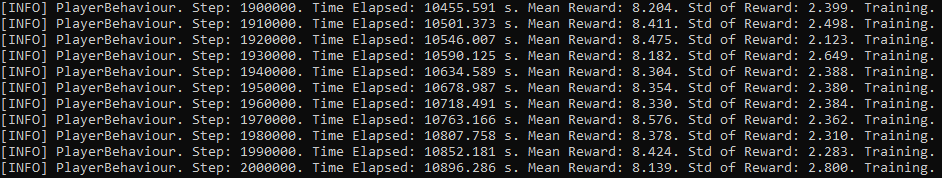
With the success of the last test, the purpose of this run was to confirm our hypothesis and our findings from earlier in this report. This run is not expected to outperform the previous run, based on the reasons provided for performing that run in the first place.

We observe that out of the 19,527 agents, only 51 agents, or, 0.26% agents cross level 15.

Of all the runs that have had observations, this is the worst result thus far. Comparatively, the majority of agents (3,070) here die at level 8. This accounts for 15.72% of the total agents.

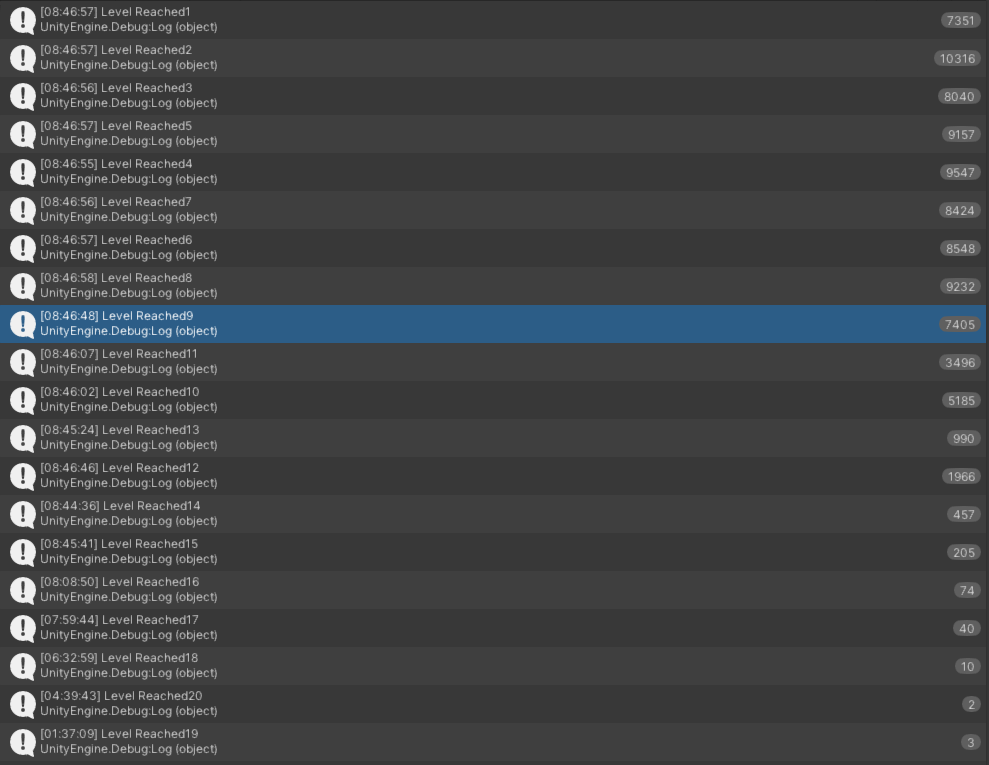


*Fig. 15. Level record for the model with no food rewards. The only relevant observations are the reward added for level completion, the starting position, the closest enemy distance, and the food locations. Importantly, we did not provide the closest food observation.*

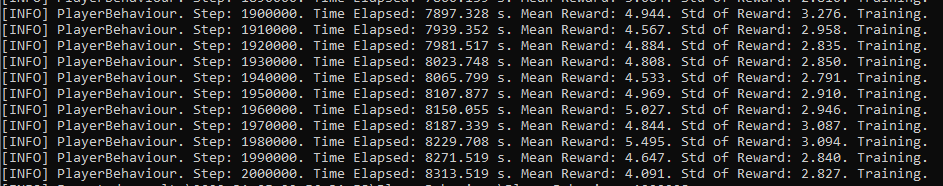


*Fig. 16. Last 100k steps for the model in fig. 15.*

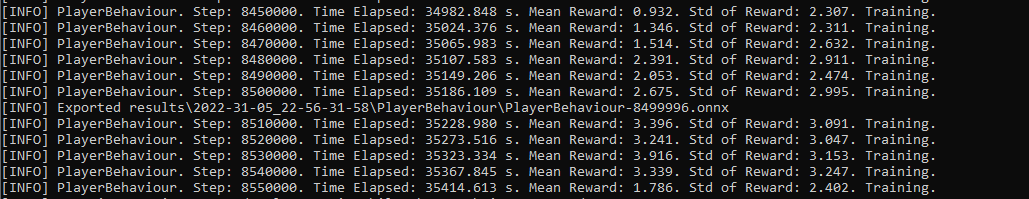
While based on logic, we’ve also stated that a negative reward for steps is unlikely to aid the training of the agent, due to its insignificance compared to other aspects. This run maintains the same list of observations as the previous two. However, we now add a move penalty so the agent will attempt to clear the level or find the exit in minimal steps to attract the lowest penalty. We notice that the maximum level it reached was 20 but this is an outlier as only 2 out of 90,448 made it to that level. Of all the agents, only 0.14% progressed to level 15 or further, which is abysmally low. Indeed, it is an even worse result than the previous result which was a record low. Comparatively, the agents which made it to level 5 were 10.12% of the total agents. The most common level for the agents to die on was level 2, at 11.41% of deaths. The baseline model had 24.88% of agents die at level 2, that was only 3,552 agents. As such, while this model didn’t perform as poorly as the baseline, it suffered extraordinarily badly at training. The next section goes further into this.



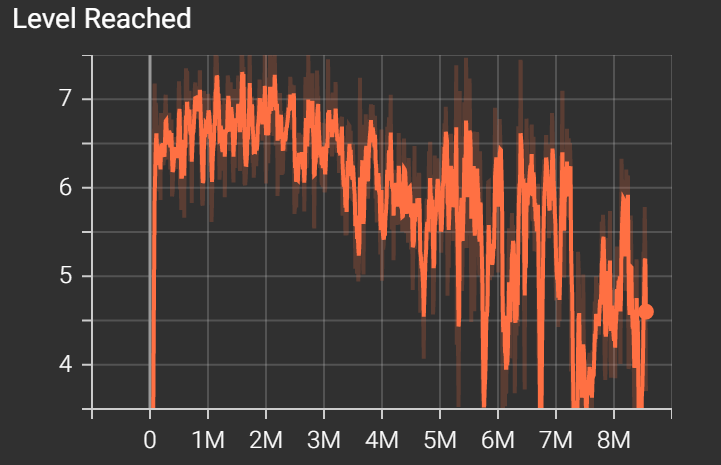
*Fig. 17. Level record for the model with move penalty*



*Fig. 18. Mid 100k steps for the model with move penalty.*

****

*Fig. 19. Last 100k steps of the model in fig. 17.*

****

*Fig. 20. Graph of average score vs step count for the 8M step count model.*

A natural assumption would be that if we ran the models long enough or longer than the earlier iterations, then they would have increased performance. This model was run to test that hypothesis.

When we ran the version overnight, we noticed that the performance drastically decreased and even went below its predecessors. Initially, we chose 2M as a step count due to time constraints, and this run confirmed it to be an ideal configuration.

Coming back to why our model performed thrice as bad as our baseline, given by the graph in fig. 20, our model hit the average mean peak around 2M and then starts slowly dropping until 5M after which it absolutely plummets. Once the value started plummeting, the agents barely made it to level 5 and hence, we have such a large number of agents dying between level 1 and level 4. The performance of this model is unique in the way that it hits level 20 but at the same time, once the agents start getting overtrained (after 2M step count), they barely survive any higher levels. In the earlier models, the agents of low step counts would only not be able to survive the higher levels but in this particular model, the agents of low step counts, as well as very high step counts, don’t survive much.

We notice that as per fig 18, the mean reward wasn’t very high, degrading from 4.944 to 4.091. The step count at this point was 2M which was the fixed step count for all the other models. Comparatively, as per fig 19, by the time we reach step count 8.55M, the average mean score is now meandering in the 2’s.

Hence, contrary to intuition, in reinforcement learning, the results are not directly proportional to the training time of the agents.

(Run with all observations and rewards (except attempt move) enabled)

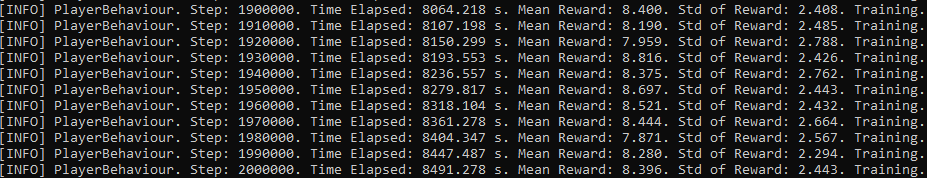
For the final run, we enabled all observations and rewards, excluding movement penalties. These include start position, exit position, inner wall positions, as well as all enemy and food locations. This also included the observations of the closest enemy and food position.

Out of 18,923 games, only 54 agents made it past level 15. This is an abysmally low 0.29%, not even comparable to our early tests after the baseline. While this run had an agent reach an impressive level 21, it was one of only 4 (0.02%) to survive past level 17.

Comparatively, the majority of the agents made it to level 9 (15.72%). Moreover, the agents which made it to levels 7, 8, or 9 comprise 41.35% of all the agents. Compared to previous agents, this one reached the second-highest mean reward. Combining all of that information, these agents were very successful at collecting food and avoiding enemies, but mysteriously just did not complete levels with any sort of efficiency.



*Fig. 21. Level record for the model with all observations and rewards enabled, except without a movement penalty.*



*Fig. 22. The last 100k steps for the model with all observations and rewards enabled, except for the movement penalty.*

# 

# Conclusion

In our given problem set, we had a lot of possibilities for the selection of our observations and rewards.

As established earlier, to deduce the findings in reinforcement learning, it is very necessary to slowly introduce changes or variability and record the results accordingly.

Hence, just concocting a model with as many observations as possible is not a good idea. Running models with small changes integrated one at a time is a much more effective method to produce a model that meets the required quality. Not only does this assist with developing your understanding of how the agent learns to play the game, but you should use these learnings to deduce which combinations of rewards and observations to test. Machine learning is not fast, so testing every possible combination is unrealistic. Instead, your development should also be strongly influenced by what steps are required for success. Training an agent to minimise its movement is counterproductive when encouraging it to move along productive paths, not just direct paths.

It is also important to recognise that the size of the game we used had a massive impact on how we developed our models and how our agents were trained. Because they were repeatedly able to find the exit on each level without any observations or incentives to do so, we did not have to encourage our agents to move closer to the exit in an effort to help them find it and start receiving rewards. Instead, they managed that all on their own. If the board were significantly larger, rewards tied to the agent’s distance would be necessary, or to perhaps re-think how the agent achieves its primary objective.