Constructive Artificial Intelligence
6COM1035

PART 2

Student ID: 19000687

Name: Mohamed Mostafa Ahmed Shaheen

MDP Explanatio

State/action	Rest	Find Energy	Consume Energy	Find Health	Consume Health	Explore	Run away
LLE	0.142	0.142	0.142	0.142	0.142	0.142	0.142
LLH	0.142	0.142	0.142	0.142	0.142	0.142	0.142
LLT	0.142	0.142	0.142	0.142	0.142	0.142	0.142
LLN	0.142	0.142	0.142	0.142	0.142	0.142	0.142
LHE	0.142	0.142	0.142	0.142	0.142	0.142	0.142
LHH	0.142	0.142	0.142	0.142	0.142	0.142	0.142
LHT	0.142	0.142	0.142	0.142	0.142	0.142	0.142
LHN	0.142	0.142	0.142	0.142	0.142	0.142	0.142
HLE	0.142	0.142	0.142	0.142	0.142	0.142	0.142
HLH	0.142	0.142	0.142	0.142	0.142	0.142	0.142
HLT	0.142	0.142	0.142	0.142	0.142	0.142	0.142
HLN	0.142	0.142	0.142	0.142	0.142	0.142	0.142
HHE	0.142	0.142	0.142	0.142	0.142	0.142	0.142
HHH	0.142	0.142	0.142	0.142	0.142	0.142	0.142
HHT	0.142	0.142	0.142	0.142	0.142	0.142	0.142
HHN	0.142	0.142	0.142	0.142	0.142	0.142	0.142
Dead	N/A	N/A	N/A	N/A	N/A	N/A	N/A

State Space Explanation

First letter is for the energy psychological variables and can be one of 2 states: L (Low) or (High)

Second letter is for the health psychological variables and can be one of 2 states: L (Low) or (High)

Third letter is for Stimuli state and can be one of 4 states: E (Energy Stimuli), H (Health Stimuli), T (Threat in form of red patch), N (No Stimuli).

Low is from above lower bound to optimal level of psychological variable (optimal level not included in range).

High is from optimal level to upper bound of psychological variable (optimal level included in range)

Action Space

7 actions are defined which are:

- Rest energy consummatory behavior
- Find Energy Appetitive behavior
- Consume energy consummatory behavior
- Find Health Appetitive behavior
- Consume health consummatory behavior
- Explore Appetitive behavior
- Runaway Appetitive behavior

All actions are initiated with 0.142 which refers to a 14.2% possibility of occurrence.

Reward function

The goal of the model is to reach HH state and stay in it as long as possible. Therefore, the reward function provides a positive reward of 5 for transferring from L to H in any of the letters of a given state and gives a reward of -5 in case of transferring from H to L, and for no change in letters no reward is given (Reward of 0).

Furthermore, if the state doesn't change with any of the letters being L a negative reward of -2 is given, and a reward of +2 is given if there was no L in the state.

Moreover, A negative reward of -4 is given for implementing explore action with L in the state otherwise a reward of 0 is given. A reward of +5 is given when there is a threat stimulus, and the action is runaway, as runaway action doesn't affect the PVs of the robot. A reward of -5 is given to the rest action when there is a threat stimulus present to prevent it from dying early.

Policy

The algorithm used for the controller is the SARSA algorithm. Firstly, the state space and the action space are initiated. The state of the robot is retrieved, and action is chosen randomly. Each state action has its own probability which is initiated by 14.2%, a random probability is generated, and an action is chosen based on it. For example, a random probability of 0.72 means the 5^{th} action is chosen (0.72 – (5 * 0.142) = 0.01) according to initial states. The action is then implemented, and the next state is determined. A reward is determined according to the observed new state. Additionally, another action is chosen for the new state and then the update stage happens in which the new state is considered to be the current state transition to the previous state and the new state observed as a result of the first action is assigned to the current state. Moreover, the Q function of the state action gets updated by the following formula Q (S, A) ← Q (S, A) + Alpha (0.8) * (Reward + (gamma (0.5) * Q (S', A')) – Q (S, A)). Furthermore, the difference between the previous Q function and the new Q function affects the probability of the state action (a difference of 0.06 for each 1-unit difference), and the increase or decrease in the probability of the state action is taken with equal proportion from the rest of the state action probabilities. Measures are implemented in the code to prevent percentages from going under 0.0 and ensure that maximum probability is maintained at a total of 1.

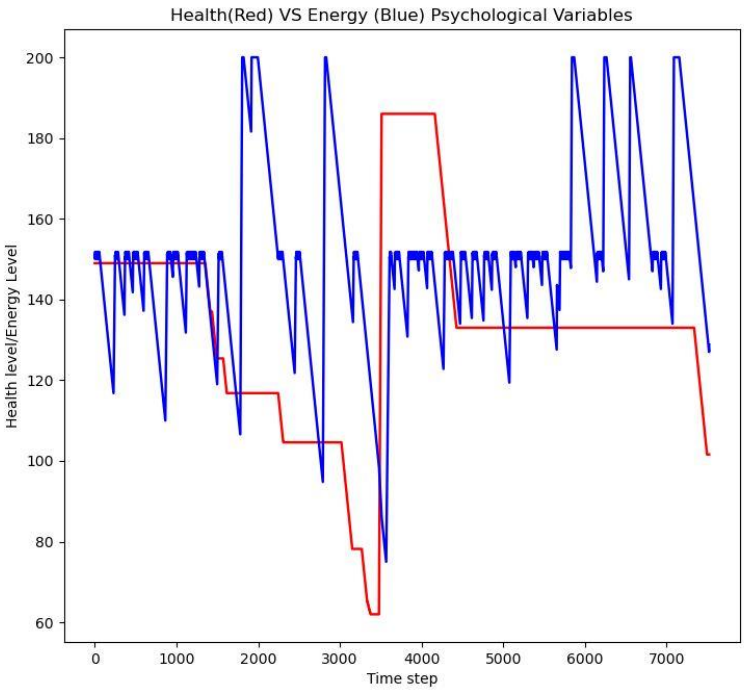
Qualitative analysis

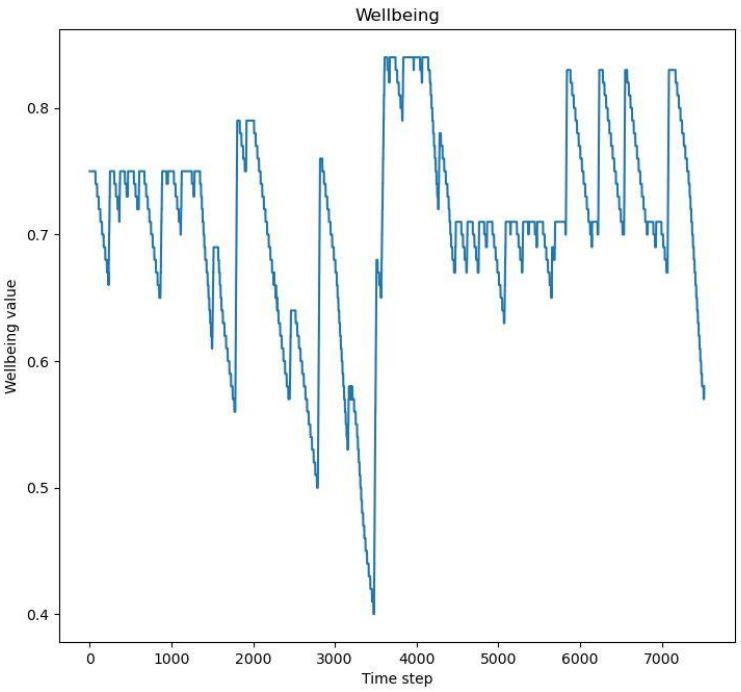
It was clear through the complex and simple environment runs that the robot explores at most half of the environment. In a complex environment, the robot covers much less of the environment than the simple one due to the presence of obstacles. Furthermore, it was noticed throughout all the runs that the robot eventually prioritize the resting behavior and it is represented in the simulation of the learned behavior, and shortly after the beginning of a random start simulation. It was also noticed that at random start simulations in both environments the robot tends to be more active at beginning of the simulation and then tends to prioritize certain behaviors. New behavior in the robot was observed around obstacles as it performs a 360-degree spin approaching walls. Reaction to red patches seems to vary in learned behaviors, some tend to run away immediately, and sometimes other behaviors are performed inside the red patch. Random start simulations take time to recognize running away from a red patch is the correct choice. The 2 appetitive behaviors find the energy and find health don't get executed much in all simulations as a result of having no immediate benefit to the robot like other consummatory behaviors. Learned behaviors simulations have varying rates of response to different patches. For example, learned behavior one turns out to be the best at avoiding red patches in both environments, while learned behavior 2 seemed to be best at maintaining its energy.

Quantitative analysis

```
Vitality = 0.7085807738332865
Vitality to the nearest two decimals = 0.71
Survival Time = 1 (never died)
```

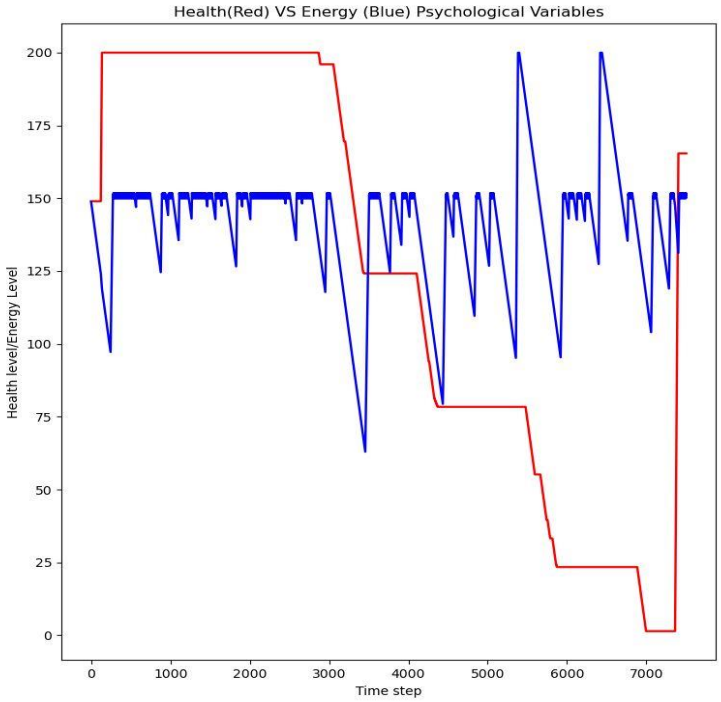
1. Complex Env. Random start

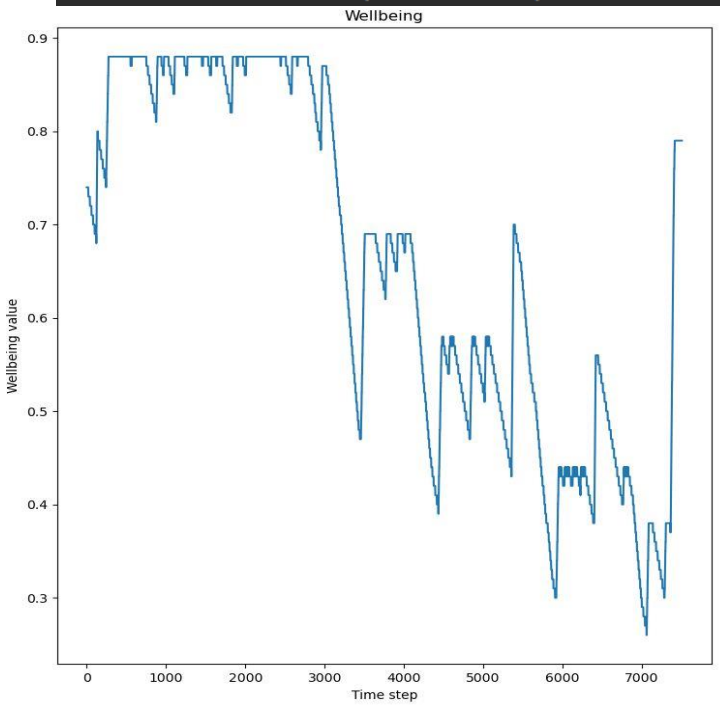




2. Complex Env. learned behavior 1

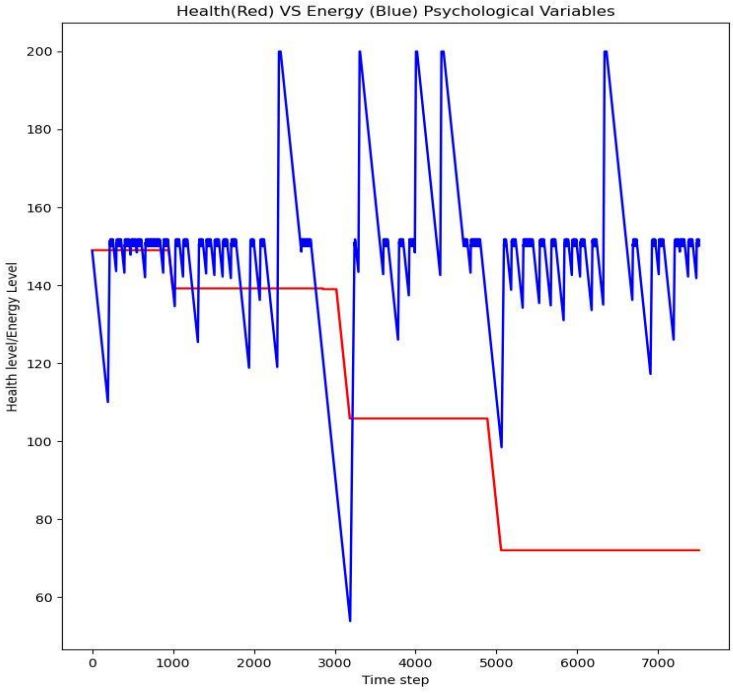
```
Vitality = 0.6603358875267119
Vitality to the nearest two decimals = 0.66
Survival Time = 1 (never died)
```

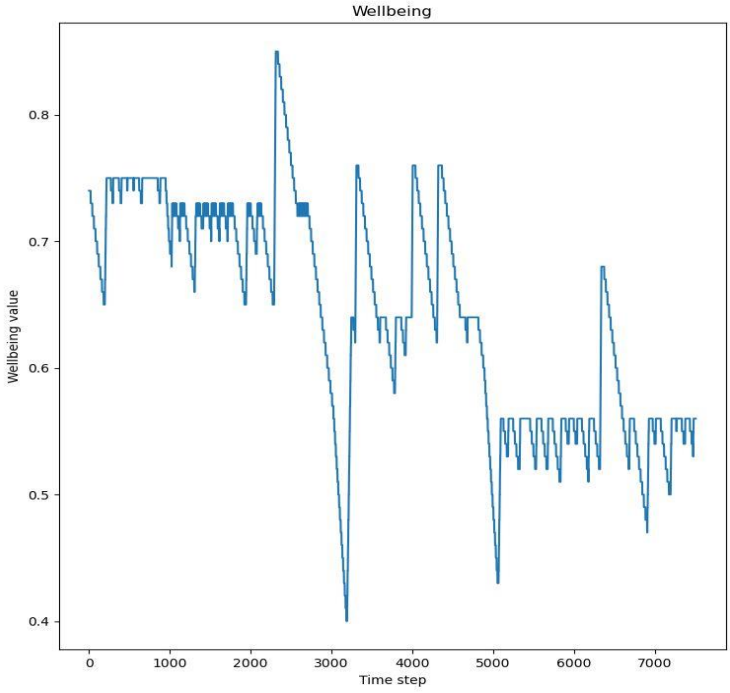




3. Complex Env. learned behavior 2

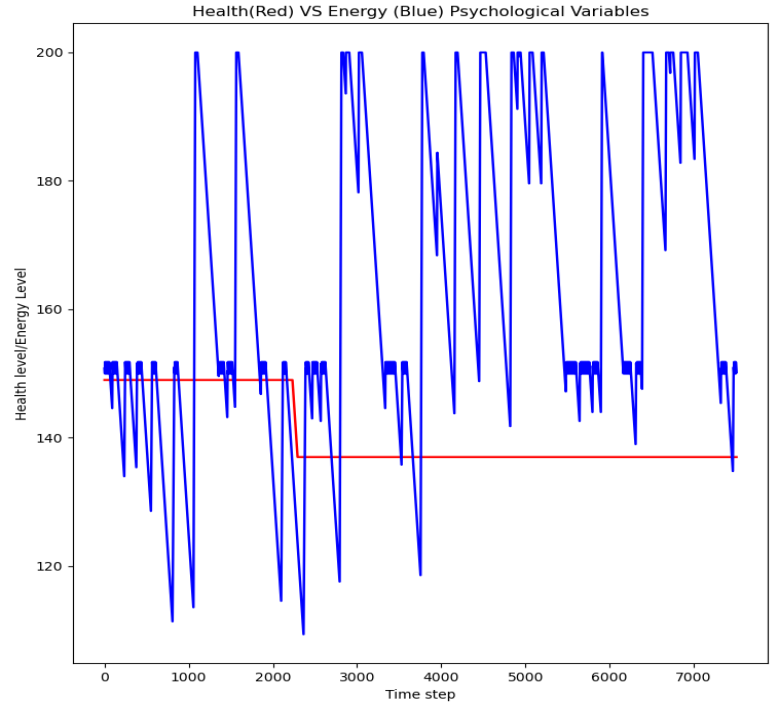
```
Vitality = 0.6418034905409399
Vitality to the nearest two decimals = 0.64
Survival Time = 1 (never died)
```

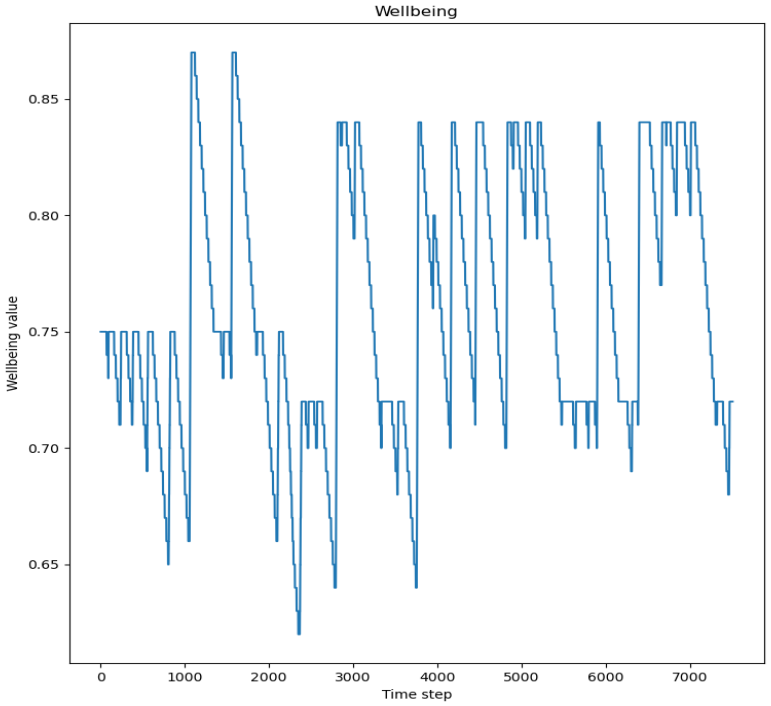




4. Complex Env. learned behavior 3

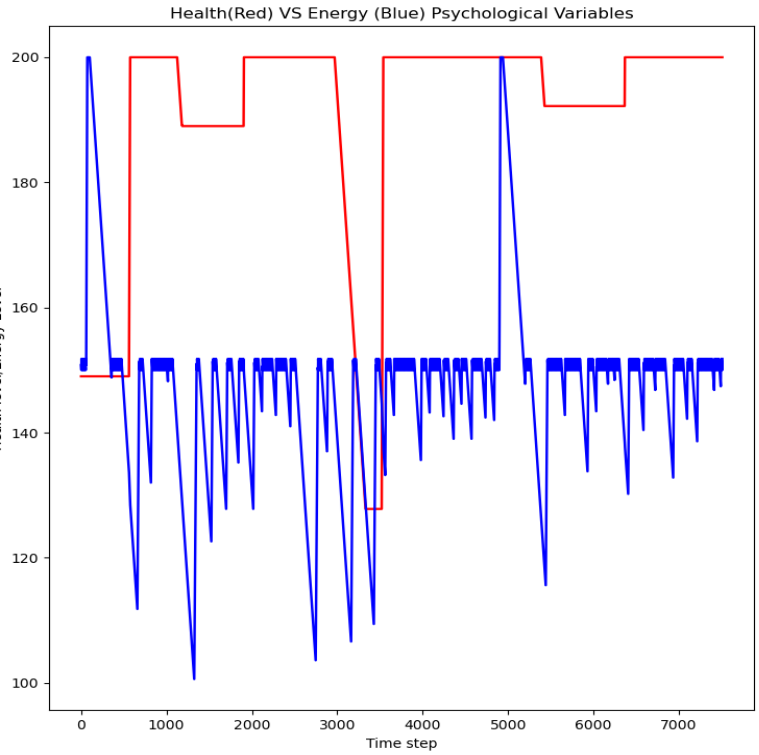
```
Vitality = 0.7599010388918492
Vitality to the nearest two decimals = 0.76
Survival Time = 1 (never died)
```

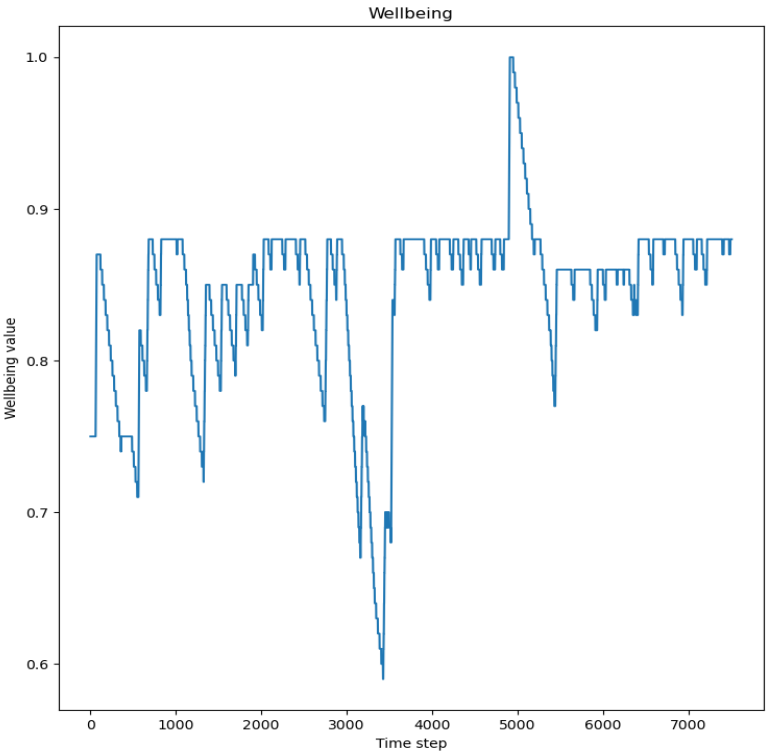




5. Simple Env. Random start

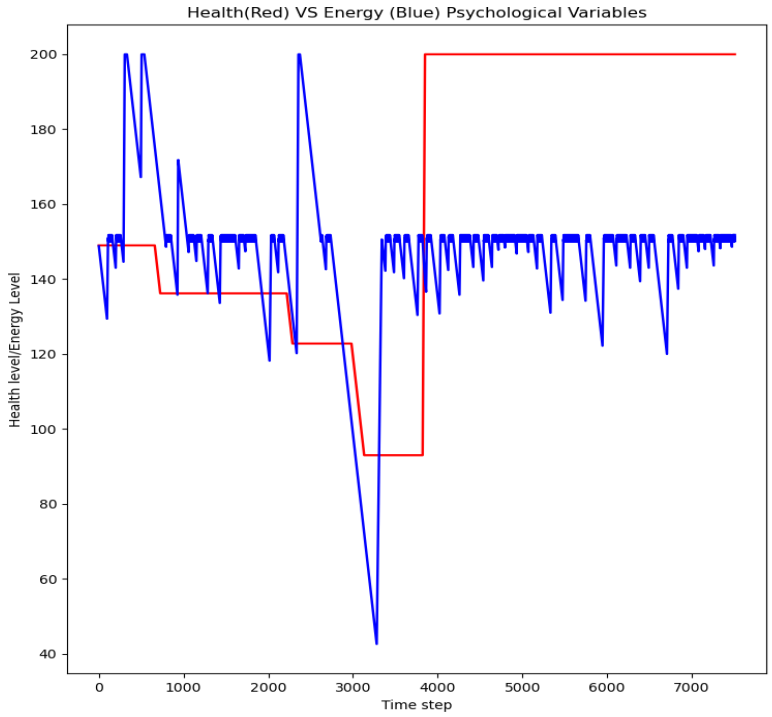
```
Vitality = 0.8429119722999555
Vitality to the nearest two decimals = 0.84
Survival Time = 1 (never died)
```

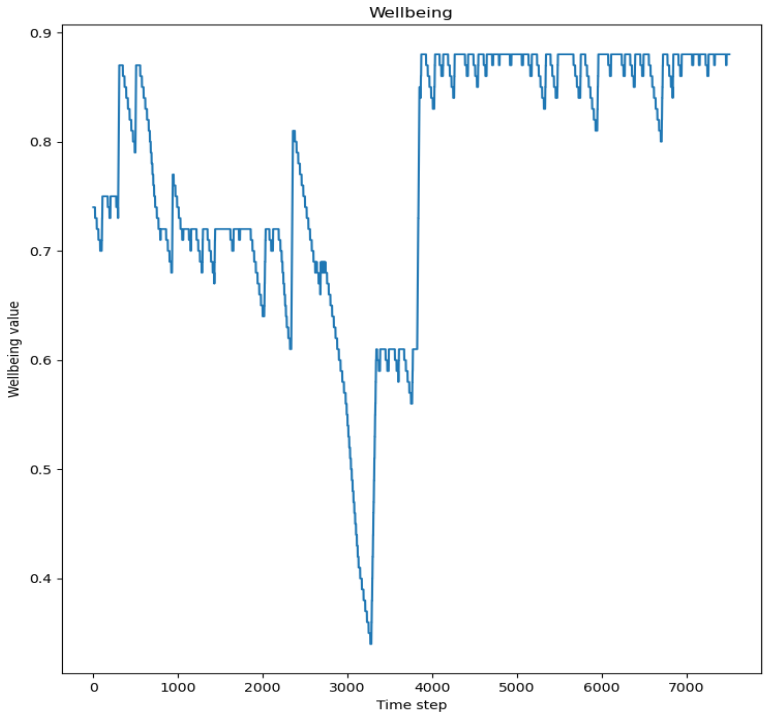




6. Simple Env. learned behavior 1

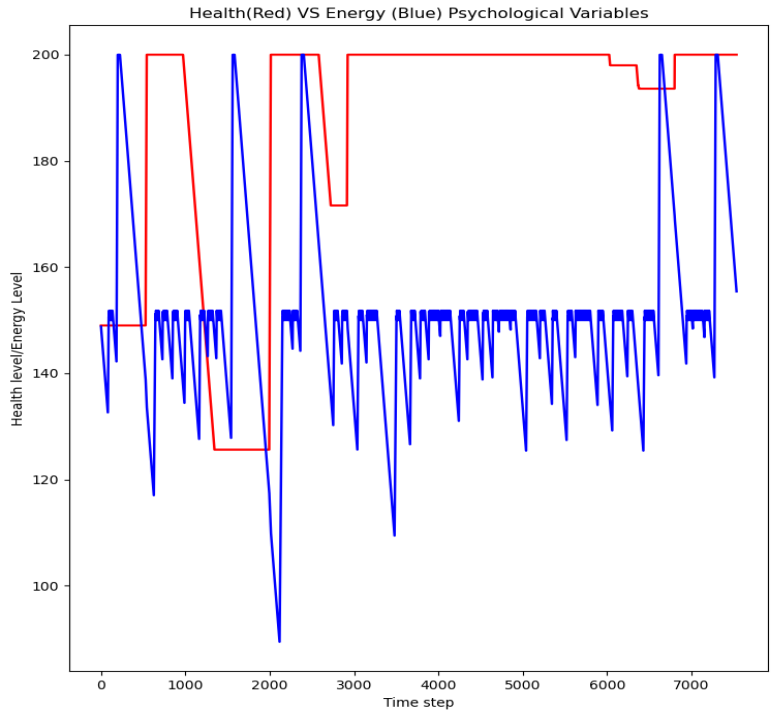
```
Vitality = 0.7726822722429855
Vitality to the nearest two decimals = 0.77
Survival Time = 1 (never died)
```

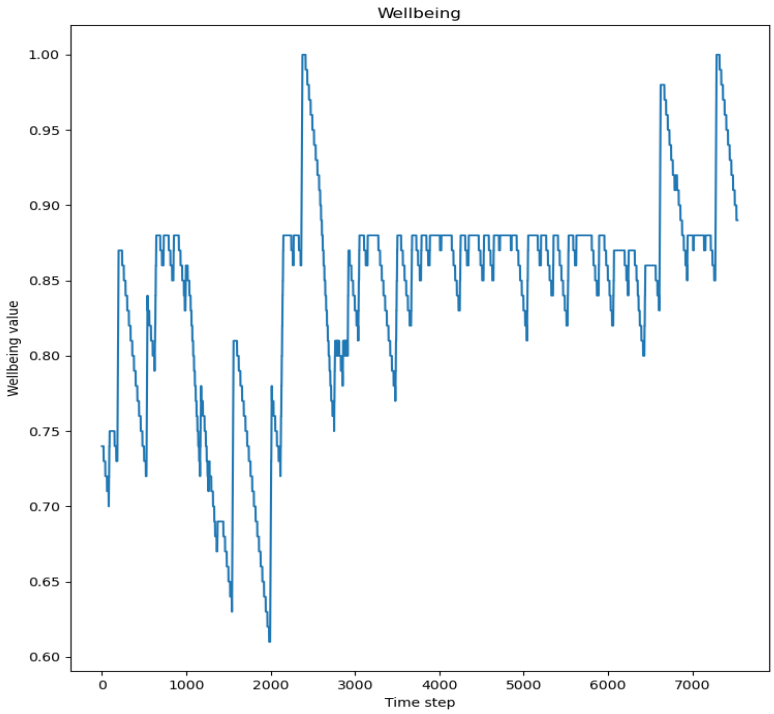




7. Simple Env. learned behavior 2

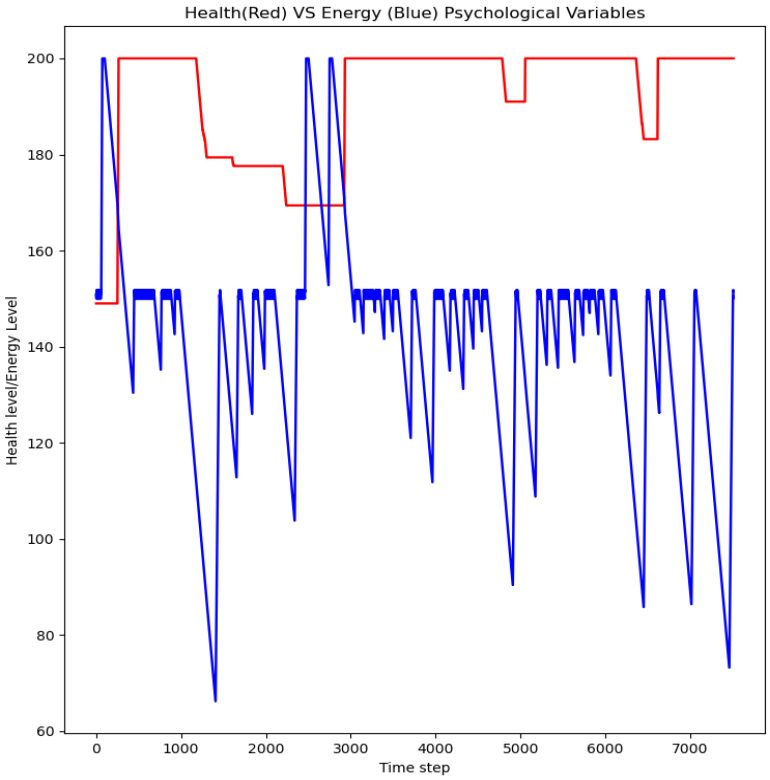
```
Vitality = 0.8425851910828419
Vitality to the nearest two decimals = 0.84
Survival Time = 1 (never died)
```

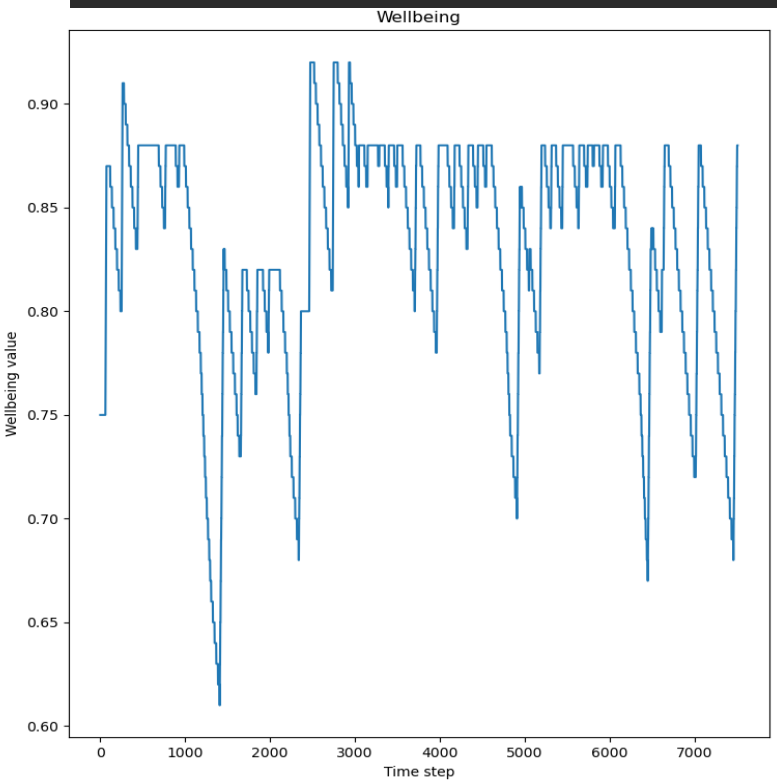




8. Simple Env. learned behavior 3

```
Vitality = 0.8295765849760687
Vitality to the nearest two decimals = 0.83
Survival Time = 1 (never died)
```





In the graphs, the resting behavior is clearly indicated by the spikes average spikes that only reach the optimum level, and the largest spikes for energy are due to the consummatory behavior consume energy. The complexity of the world is clearly represented in the vitality values as vitality reached 0.84 max with an average of 0.82 on the simple environment while it topped at 76 with an average of 0.69. Slow learning is demonstrated in random start graphs as an energy variable huge drop decreased in severity as time passed. Learning behavior 2 is considered the best in energy maintenance especially if we ignore the huge energy drops resulted from the negative reward given to the robot if it rests while on a red patch making the robot avoid resting on red patches. Moreover, learning behavior 2 seems to be the best out of all the learned behaviors as demonstrated by the well-being graph and vitality score. Learned behavior one is the best in avoiding red patches as demonstrated by the slight small drops in health shown in both environment graphs as it tries to run away from the red patch.

Demonstration videos

At the random start video, the robot is seen at the beginning learning that resting is an efficient action then it starts to repeat the resting action more frequently. The robot also was seen avoiding a red patch and soon after it went and kept roaming inside a red patch which shows the learning process is still undergoing and the inconsistency previously mentioned in qualitative analysis. Find health action is demonstrated by the robot circling around the green patch. Moreover, explore action is frequently implemented making the robot roam around, especially because the simulation was in a simple environment and the health level was properly maintained due to the presence of many green patches as seen in quantitative analysis. As mentioned in qualitative analysis the activeness of the robot in exploring decreases gradually as it learns the efficiency of rest behavior. The robot is also seen rotating around 360 degrees approaching some walls.

In the learned behaviors video, the robot is relatively much less active than in the random start executing rest behavior multiple times consecutively making it seem like it stopped working. The robot has various reactions to red patches avoiding them as soon as entering at times, roaming around inside a red patch then running away at times, and walking through them at other times. The walls provide an even bigger challenge now that the robot rotates around 180 to 360 degrees making it harder to reach the green patch in the center or explore the map. Find energy action is demonstrated when the robot keeps circling inside a blue patch however, the reward function doesn't teach the robot to execute consume energy action afterward so finding energy doesn't necessarily help the robot maintain its psychological variables. In the simple world, finding health was more noticed as the robot circles inside green patches.

Comparison of behaviors

It is clear that the robot performs poorly when it comes to exploring the map in part 2 than in part 1. The robot in part 1 tended to almost explore the whole map in the simple environment and around half in the complex environment, However, in Part 2 the robot almost covers half of the map in the simple and a quarter of the map in the complex. However, when it comes to vitality and well-being part 2 robot seems to be much better reaching a max of 0.84 while part 1 reached a max of 0.73 in simple environments. The robot in part 1 is more consistent in behavior, however, the robot in part 2 has less consistency. The runaway behavior in part 2 is better as the robot frequently runs from the red patch as soon as it steps in it unlike in part 1 where it sometimes performed rest behavior inside a red patch. In part 2 it was rare to see the robot resting in a red patch. Moreover, find energy and Find health behaviors are rarely executed in part 2 unlike in part 1 where they were frequently used. Furthermore, the robot no longer revolves in a semi-circle around red patches in part 2 as was seen in part 1.