CS 520: Assignment 3 - Probabilistic Hunting

Aditya Vyas
Vedant Choudhary
Nitin Reddy
Siddharth Sundararajan

April 19, 2019

The purpose of this assignment is to model your knowledge/belief about a system probabilistically, and use this belief state to efficiently direct future action.

1 Representation

1.1 Code

In our code, the map is represented as a numpy matrix. The values assigned to the numpy matrix are randomly assigned based on the probability of the terrain type (flat with probability 0.2, hilly with probability 0.3, forested with probability 0.3, and caves with probability 0.2). Our code has terminal based input variables through which we can change the map size (default $10 * 10$), choose if we want to visualize the map, change the decision rule (Rule 1 or Rule 2) etc.

1.2 Visualisation

Figure 1 displays the visualisation of the map. Note that the map is represented as a heat map. The four parts in the figure as explained below:

- The left top map shows the actual map of the terrains generated by the code. The red symbol, X shows the location of the target in the map. The color range of the actual map from lightest to darkest correspond to Flat, Hilly, Forested and Cave.
- The right top map shows the belief matrix, which basically gives the probability of target being in a particular cell
- The bottom left shows the confidence matrix (probability of the target being found in a cell) as our agent opens up cells based on observations made. The shade of the belief and confidence matrix determines the probability in that cell. In both these cases, as the probability increases the shade of color increases in the heat map.

- The bottom right map shows the cells that the agent checks. The more the cell has been opened, darker the shade.

NOTE: The number of iterations gets updated every iteration. This present on the top of the figure.

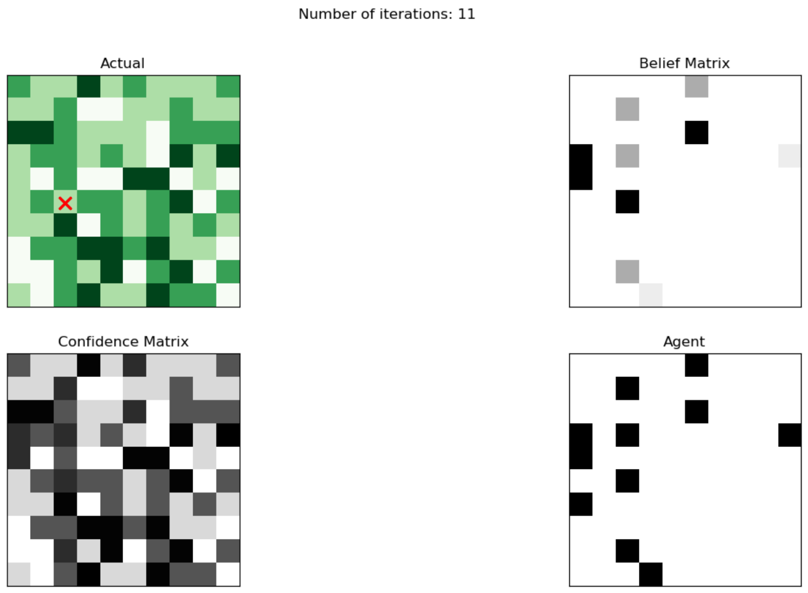

Figure 1: Probabilistic Hunting Visualisation

2 A Stationary Target

2.1 How to efficiently update the belief state?

The information at any given time t is the observations till time t and the failure of searching the current cell. Updating the belief state is the same as finding the probability of target in a new cell given that we know the observations till now and the failure in the previous cell. Using this information, the agent updates the belief state. The agent works in one of the three states - Initial, Update and Termination.

- **Initial state** : This state occurs only when $t = 0$. At this state the agent assigns equal probability to each of the cells. This is because at $t = 0$, there are no observations and the failure of searching the previous cell is not there. At this state, the agent opens the cell with maximum probability (which in this case is a random cell) and it moves to the update state.

- **Update state** : The agent stays in the update state until it is able to find the target. Note that this is the state where the agent spends the most time. In this state at any given time t two recurring steps occur.

Step 1 : The agent checks if the cell opened at time t has the target. If the target is present, then the agent moves to the termination state. If the target is not present, then the agent moves to Step 2.

Step 2 : The agent reduces the probability of the current cell having the target by some factor (which is the false negative rate of the current cell's terrain). Then, as the overall probability is not equal to one, the agent normalizes all the cells. For example, let us consider a map size of $3 * 3$ and the current cell is terrain *Flat* (Note that the false negative rate of the terrain flat is 0.1.), then,

$$\text{Initial probability of all cells} = \frac{1}{9}$$
$$\text{On opening current cell probability of current cell} = 0.1 * \frac{1}{9} = \frac{1}{90}$$

The total changes after the agent opens the current cell,

$$\text{Before opening cell Total probability} = 9 * \frac{1}{9} = 1$$
$$\text{After opening cell Total probability} = 1 * \frac{1}{90} + 8 * \frac{1}{9} = \frac{81}{90}$$

The agent then normalizes the values,

$$\text{Probability of current cell} = \frac{\frac{1}{90}}{\frac{81}{90}} = \frac{1}{81}$$
$$\text{Probability of remaining 8 cells} = \frac{\frac{1}{9}}{\frac{81}{90}} = \frac{10}{81}$$

Now, the total probability adds up to 1. Then, the agent goes back to Step 1 and it recursively runs.

- **Termination state** : The agent comes to this state from Step 1 of the update state. In this state, the agent generates a random number between 0 and 1. If the randomly generated number is greater than the false negative rate of the current cell (target), then the agent is sure that the target exists in the current cell and target is found. If not, then the agent goes back to Step 1 of the update state and continues.

2.2 How to efficiently update the confidence state?

The information at given time t is the observations till time t. The confidence state is updated only after the belief state is updated. A confidence state is a cell on the map. Each cell is of a certain terrain (flat, hilly, forest or cave). A confidence matrix comprises of the cells on the map. A cell in the confidence matrix is updated using the following formulae.

$$\text{Confidence State}[i, j] = (1 - \text{FNR(Terrain)}) * \text{Belief State}[i, j]$$

where, Confidence State$[i, j]$ is the confidence state of cell of i^{th} row and j^{th} column, FNR(Terrain) is the false negative rate of the terrain present in the cell and Belief State$[i, j]$ is the belief state of cell of i^{th} row and j^{th} column after the update.

2.3 Comparing the two decision rules

The two rules that are compared are :

- Rule 1: At any time, search the cell with the highest probability of containing the target.
- Rule 2: At any time, search the cell with the highest probability of finding the target.

In case of Rule 1, the decision to search the cell highest probability of containing the target is made based on the belief state after every iteration. Whereas, in case of rule 2, the decision to search the cell with the highest probability of finding the target is made based on the confidence state after every iteration. Note that the confidence state is updated only after the belief state is updated every iteration.

On average, Rule 2 requires lesser searches when the target is present in terrains other than cave and, hence performs better. But, when the target is present in the cave terrain, rule 1 on average performs better. This behavior is seen in Figure 2 when the map size is fixed at 10.

This behavior is seen because rule 2 in general, has more information than rule 1 at any given time. Rule 2 uses the information present in the belief state (after its update) along with the information of the terrain (False Negative Rate) before making a decision. Therefore, when the target is present in terrains beside a cave, the rule 2 is able to find the target sooner compared to rule 1. When the target is present in the cave terrain, rule 2 does not easily find the target because its False Negative Rate (FNR) is quite high (0.9). This high FNR causes only a small increase in confidence over time and therefore, rule 2 takes more time when compared to rule 1.

Yes, the behavior of rules 1 and 2 hold across multiple maps as seen in Figure 3. You can observe that the trend of iterations for each terrain follows the same pattern.

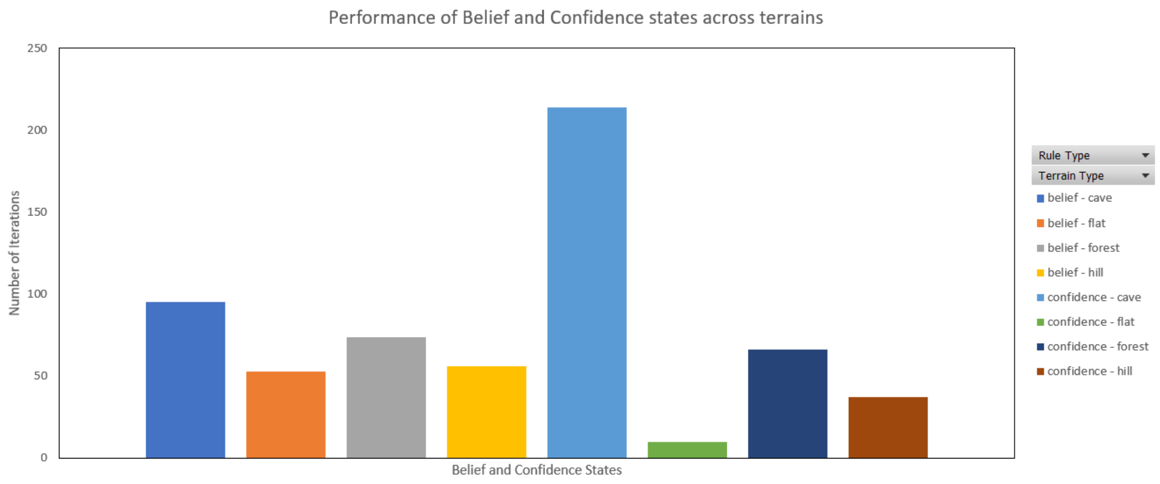

Figure 2: Comparison of the belief state and confidence state across terrains when map size = 10

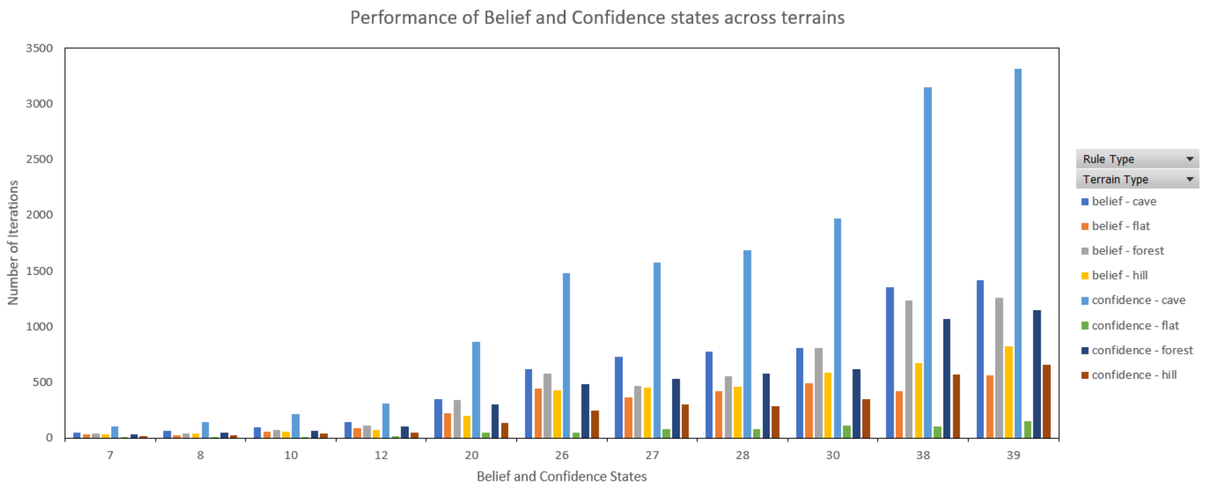

Figure 3: Comparison of the belief state and confidence state across terrains across different map sizes

2.4 What happens when distance is included?

2.4.1 How can you use the belief state and your current location to determine whether to search or move (and where to move), and minimize the total number of actions required?

The agent makes a decision based on two factors - current belief state and current location (where the agent is currently searching). Using this information, the agent can do one of two things. It can either search the current cell or move to another cell. Note that either task performed is counted as an action. The agent makes a decision in the following manner :

- **Initial State** : This state occurs only when $t = 0$. At this state the agent assigns equal probability to each of the cells. The agent opens a random cell and it moves to the update state.

- **Update State** : The agent stays in the update state until it is able to find the target. Note that this is the state where the agent spends the most time. In this state at any given time t three recurring steps occur.

  Step 1 : The agent checks if the cell opened at time t has the target. If the target is present, then the agent moves to the termination state. If the target is not present, then the agent moves to Step 2.

  Step 2 : The agent reduces the probability of the current cell having the target by some factor (which is the false negative rate of the current cell's terrain). Then, as the overall probability is not equal to one, the agent normalizes all the cells. In addition to this, the agent also computes the Manhattan distance from the current cell to all other cells. The probability computed in each cell is divided by its Manhattan distance from the current cell. Then the agent moves to Step 3.

Step 3 : The agent moves to the cell with the highest value. Note that this cell is the closest cell that is likely to have the target from the current cell. The agent moves to the chosen cell (the one with the highest value). The agent moves to Step 1.

- **Termination State** : The agent comes to this state from Step 1 of the update state. In this state, the agent generates a random number between 0 and 1. If the randomly generated number is greater than the false negative rate of the current cell (target), then the agent is sure that the target exists in the current cell and target is found. If not, then the agent goes back to Step 1 of the update state and continues.

2.4.2 Compare performance of Rule 1 and Rule 2 based on new decision rules

The two rules that are compared are :

- Rule 1: At any time, search the cell with the highest probability of containing the target.
- Rule 2: At any time, search the cell with the highest probability of finding the target.

In case of Rule 1, the decision to search the cell highest probability of containing the target is made based on the belief state after incorporating the distance factor (after every iteration). Whereas, in case of rule 2, the decision to search the cell with the highest probability of finding the target is made based on the confidence state after every iteration. Note that the confidence state is updated only after the belief state is updated every iteration.

Figure 4 shows the comparison of performance of rule 1 and 2 with/without distance. On average, it is seen that rule 1 does better with the additional information of current location and distance. But, in rule 1, if the target is present in the terrain - cave, the agent takes longer to find (due to the FNR). On the other hand, on average, the additional distance factor does not have a big impact on rule 2. Similar to rule 1, if the target is present in the cave terrain, the agent takes longer to find with the additional information.

This behavior is seen because the additional information of distance incorporated (with a low FNR) helps to narrow down on the target sooner. But, it is also seen that as the FNR increases the additional information slows down in finding the role. In rule 2, especially, there is no major difference because the majority role is played by FNR.

Note that the behavior of rules 1 and 2 hold across multiple maps as seen in Figure 5.

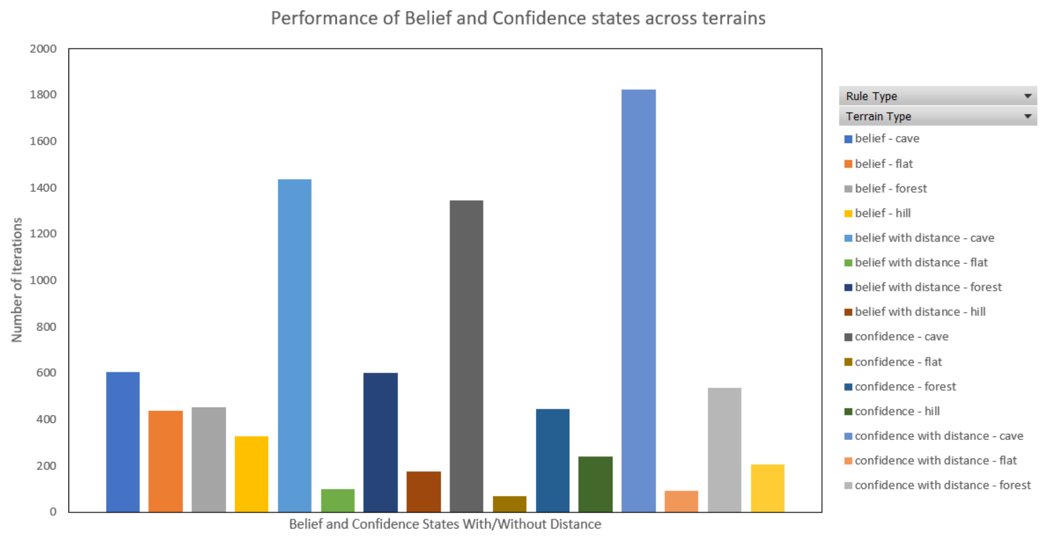

Figure 4: Comparison of the belief state and confidence state across terrains when map size = 25 with/without distance

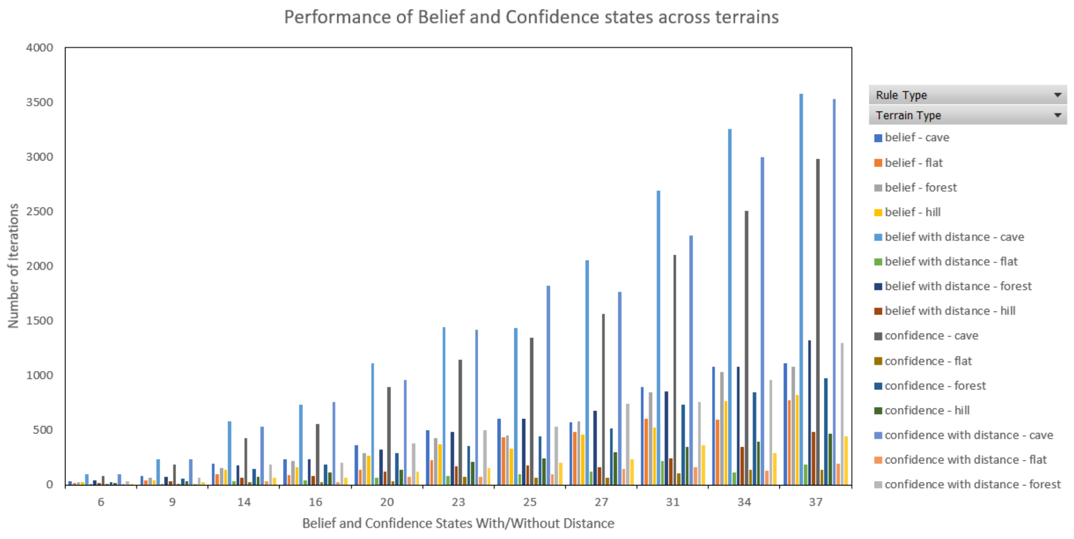

Figure 5: Comparison of the belief state and confidence state across terrains with/without distance across different map sizes

2.5 An old joke

An old joke goes something like the following:
"A policeman sees a drunk man searching for something under a streetlight and asks what the drunk has lost. He says he lost his keys and they both look under the streetlight together. After a few minutes the policeman asks if he is sure he lost them here, and the drunk replies, no, and that he lost them in the park. The policeman asks why he is searching here, and the drunk replies, “the light is better here”.

In light of the results of this project, discuss.

The joke is a good example of how our Bayesian model works. When the drunk man is searching for keys under the streetlight, it is because he believes that the keys can be found in a place which is well lit. This is analogous to finding the target in flat terrains because of high confidence since FNR is low (rule 2). However, he avoids searching for the keys in the park, even though he knows he lost it there. This is similar to having to search for the target in a cave terrain (belief). Due to its high negative rate, even if the target is there, it is difficult to find one.

According to the logic behind our code, a human's best chance of action depends on the true probability of his keys being in the park vs being under the street lamp and the success rate of finding the keys in the park and in the street map. If the probabilities of finding the keys be it street lamp or park is relatively equal, then it makes sense to search for the keys under the street lamp initially.

One thing which is different between our code and how the drunk man behaves is that, in our code when the target is not being found in easier terrains, it updates the belief after every iterations leading to decreased probability in those type of terrain cells, which allows the agent to move over to different terrains and cells and not get stuck to just searching in the flat or "well-lit" area like the drunk man.

3 A Moving Target

In this section, the target is no longer stationary, and can move between neighboring cells. Each time you perform a search and the agent fails to find the target, the target will move to a neighboring cell (with uniform probability). However, all is not lost - whenever the target moves, surveillance reports to you that the target was seen at a Type1 x Type2 border where Type1 and Type2 are the cell types the target is moving between (though it is not reported which one was the exit point and which one the entry point.

3.1 How can you update your search to make use of this extra information?

Even though the entry and exit points of the target are not known by the surveillance reports, the entry and the exit can be inferred once we have successive surveillance reports. For example, let us assume that the output of the first report is (Flat, Cave) and the output from the second report is (Forest, Flat), and as the target moves to neighbouring terrain (cell) we can infer that the target has moved from Cave to Flat in first step and Flat to Forest in second step. This is because Flat is the only terrain common between both the reports. This information is considered and incorporated into the calculation of beliefs and confidence. Belief and confidence values are distributed from that terrain to other terrains accordingly.

Even though the target is moving, the extra information from the surveillance report will make this problem identify the target sooner.

3.2 How to efficiently update the belief and confidence state?

- **Initial state** : This state occurs only when $t = 0$. The surveillance report code gives us (Type1, Type2) based on movement of the target without giving additional information of the entry and exit points. Therefore, the belief state is divided equally among the type1 and type2 terrains and, for the other two terrains it is set to zero. This is possible because the surveillance report gives us an additional knowledge which helps us change our belief about regions not in Type1 or Type2.

- **Update State** : Similar to the case of stationary target, the agent stays in the update state until it is able to find the target.

  Step 1 : The agent checks if the cell opened at time $t + 1$ has the target. If the target is present, then the agent moves to the termination state. If the target is not present, then the agent moves to Step 2.

  Step 2 : At any time t, the belief of the corresponding terrain at time t are interchanged and equally distributed among its neighbours of the other terrain type. For example, if the satellite information obtained is Type1 and Type2, the belief in terrain Type1 is distributed equally and added to its neighbours of Type2 terrain. Similarly, the belief values of Type2 terrain is distributed equally and added to beliefs of Type1 terrain that are neighbours of Type2.

  The confidence state of cell for moving target is calculated the same way as for the stationary target case.

- **Termination state** : This step works similar to the case of stationary target.

For example, let us assume the following terrain layout of the matrix.

$$\begin{bmatrix} Flat & Forest & Cave \\ Hill & Cave & Forest \\ Flat & Forest & Hill \end{bmatrix}$$

We assume that the target is present in cell $(2, 1)$ i.e., Forest. A step-by-step procedure followed by the agent to update the belief state is seen below. The initial belief state is,

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

At first, the agent opens a random cell. Let us assume the agent opens up cell $(0, 0)$. Then, as explained earlier, the agent checks and normalizes the matrix if the target is not found as follows,

$$\begin{bmatrix} \frac{1}{81} & \frac{10}{81} & \frac{10}{81} \\ \frac{10}{81} & \frac{10}{81} & \frac{10}{81} \\ \frac{10}{81} & \frac{10}{81} & \frac{10}{81} \end{bmatrix}$$

At this point, as the target is not found, the agent returns surveillance information of $(Type1, Type2)$. Let us assume the information returned is - $(Forest, Cave)$. With this information, the agent sets all terrains other than terrains Forest and Cave to 0. Note, we know that the target has moved from cell $(2, 1)$ to cell $(1, 1)$ but, the agent does not have this information. The updated matrix is,

$$\begin{bmatrix} 0 & \frac{10}{81} & \frac{10}{81} \\ 0 & \frac{10}{81} & \frac{10}{81} \\ 0 & \frac{10}{81} & 0 \end{bmatrix}$$

The agent now normalizes the matrix as it does not add up to 1. The matrix after normalizing is seen as,

$$\begin{bmatrix} 0 & \frac{1}{5} & \frac{1}{5} \\ 0 & \frac{1}{5} & \frac{1}{5} \\ 0 & \frac{1}{5} & 0 \end{bmatrix}$$

At time, the agent opens a random cell (as the five cells have the same probability of $\frac{1}{5}$. Let us assume the agent opens cell $(2, 1)$, the matrix after updation and normalization is seen as,

$$\begin{bmatrix} 0 & \frac{10}{47} & \frac{10}{47} \\ 0 & \frac{10}{47} & \frac{10}{47} \\ 0 & \frac{7}{47} & 0 \end{bmatrix}$$

At this time, the agent returns the information obtained from surveillance. Let us assume that the information received is $(Cave, Hill)$. With this information, the agent sets all terrains other than terrains Cave and Hill to 0. Note, we know that the target has moved from cell $(1, 1)$ to cell $(0, 1)$ but, the agent does not have this information. The updated matrix is,

$$\begin{bmatrix} 0 & 0 & \frac{10}{47} \\ 0 & \frac{10}{47} & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

With this additional information, the agent creates two matrices. In the first matrix, it transfers the probability information from Type1 to Type2 and in the second matrix, it transfers the probability information from Type 2 to Type 1. Finally, the agent adds the two matrices and normalizes.

The first matrix is,

$$\begin{bmatrix} 0 & 0 & 0 \\ \frac{10}{47} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The second matrix is,

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

By adding the two matrices, we get,

$$\begin{bmatrix} 0 & 0 & 0 \\ \frac{10}{47} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

By normalizing this matrix, we get,

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Then, the agent gets another surveillance report and works recursively. Note that at this point, the belief matrix tells the agent that the target most likely is in cell $(1, 0)$. This is correct as we know where the target is currently present in that cell.

3.3 Comparing the two decision rules. What happens when distance is included?

The two rules that are compared are :

- Rule 1: At any time, search the cell with the highest probability of containing the target.
- Rule 2: At any time, search the cell with the highest probability of finding the target.

When compared to the case of stationary object, the agent identifies the moving target to be faster due to the information obtained from surveillance reports. This makes the moving target problem easier than the stationary target. Also, as seen in Figure 6, the number of iterations required to identify the target is comparatively lower than the case of stationary target. Similar to the case of stationary target, on average, Rule 2 requires lesser searches when compared to Rule 1 to successfully identify the target.

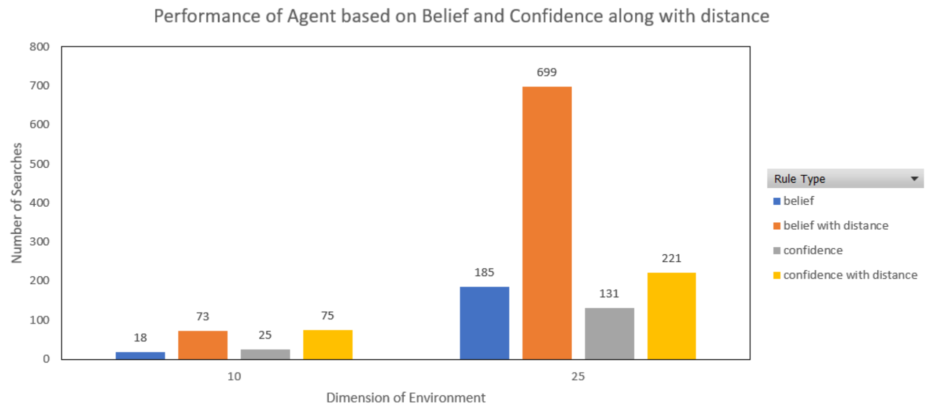

Figure 6: Comparison of the belief state and confidence with distance state across terrains when map size = 10 and 25

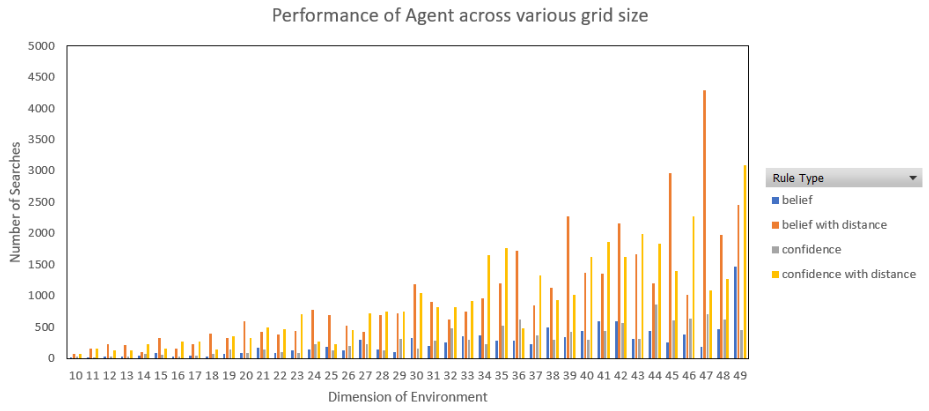

Figure 7: Comparison of the belief state and confidence state across terrains across different map sizes

As seen in Figure 6, in case of map size 25, the agent finds the target sooner using the confidence matrix (Rule 2) when compared to using the belief matrix (Rule 1).

When the agent considers distance before searching any terrain, it is seen that the number of searches required are slightly higher. This is because the agent takes the decision of searching closer to terrains than the farther ones. As we see in the Figure 7, belief with distance takes the maximum searches to identify the target across any map size. Rule 2 with distance takes the least number of searches to identify the target.