Assignment3 Writeup

Yusu Weng(yw706) Zhaoxiang Liu(zl355)

A Stationary Target

Q1

Assume N is the false negative rate, Bj is the P(Targetj|Observationt) P(Target in Celli|Observationt ^ Failure in Cellj) = P(Targeti， Observationt,Failurej)/P(Observationt,Failurej)

P(Observationt,Failurej) = P(Observationt) * P(Failurej|Observationt)

P(Failurej|Observationt) = 1-P(Success in Cellj|Observationt)

= 1-P(Target in Cellj|Observationt)*P(Query return True in Cellj|Target in Cellj) = 1-Bj(1-N)

P(Observationt,Failurej) = P(Observationt) *[1-Bj(1-N)]

P(Targeti, Observationt, Failurej) = P(Observationt) * P(Targeti|Observationt) * P(Failurej|Targeti, Observationt)

P(Failurej|Targeti, Observationt)= {

1 (if i!=j)

N (if i=j)

}

So P(Targeti, Observationt, Failurej) = {

P(Observation) * Bj (if i!=j)

P(Observation) * Bj * N (if i=j)

}

P(Target in Cell i| Observationt, Failure in Cellj) = P(Target in Celli,Observationt, Failure in Cellj)/P(Observationt, Failurej)=

Bj/1-Bj(1-N) (if i!=j)

BjN/1-Bj(1-N) (if i=j)

Q2

P(Target Found in Celli|Observationt) = P(Target in Celli Observationt)*P(Target Found in Celli|Target in Celli) = BjN

Q3

```
Agent.py
```

Rule1 : Search Pmax(Target in Celli|Observationt) = max(Bj)

Rule2: Search Pmax(Target Found in Celli|Observationt) = max(NBj)

We run each rule for 500 times:

```
Rule1: 5810 steps

Rule2: 4680 steps
```

The result shows that rule2 perfromes better than rule1, we think the reason may be that when there are many cells with same highest possibilties the agent will pick the cell with easier finding type

Q4

```
MotionAgent.py
```

> Suppose in a scenario that we find a high possible cell with lon distance to dig and a near one with low possible. How can we modify the post-possibility, If we believe that 10 steps=1 searc time, which means that for the cell with longer distance we dig totalDistance/10 more times to get the target. So we can establish the following formula
>
> 1-(1-$P_{modifiedPossibility}$)^(distance/10+1)=$P_{currentBelief}$
>
> The $P_{modifiedPossibility}$ is the new belief, which is reasonable because for the cell with longer distance we need flip the cell distance/10 more time to get the target.We can deduce the new belief reversely $P_{modifiedPossibility}$=1-(1-$P_{currentBelief}$)^(distance/10+1)
>
> We will replace old belief $P_{currentBelief}$ no matter calculating from either of rules with new belief $P_{modifiedPossibility}$
>
> $P_{modifiedPossibility}$ is smaller than $P_{currentBelief}$, So we add the distance factor to the final decision probability.
>
> > runing rule 1 with dim 50 and for 10 times .. average search action is 43050.3 runing rule 2 with dim 50 and for 10 times .. average search action is 36776.4

Below is the original algorithm using rule1 anf rule2 to randomly select high possible cell to dig

```
runing rule 1 with dim 50 and for 100 times ..

average search action is 107792.4

runing rule 2 with dim 50 and for 100 times ..

average search action is 77893.6
```

```
runing rule 1 with dim 50 and for 100 times ..

average search action is  43050.3

runing rule 2 with dim 50 and for 100 times ..

average search action is  36776.4
```

We can see the significant reduction by using improved algorithm

Q5

For rule2, the agent will more prefer to search cells with low false negative rate since having higher possibilities to find the target. However, if the target locates at the cell with difficult terrian, the agent may waste a lot of time stuck in the wrong cells.

Moving Target

```
BonusAgent.py
```

After the target moved, we need to update the Bj. Since the target can only move to one of its neighbor with the same probability, so we set the new belief:

$$\text{Bj' = } \sum^{i \in neighbors} Bi/numofi'sneighbor$$

Then, based on the infomation we receive from the tracker, we set cells whose types are the same as tracker's report to be ZERO.

Finally we go in the same process as Stationary Target part to generate P(Bj|Observation t+1)

We run rule 1 & rule 2 each for 500 times:

```
Rule1: 5320 steps
Rule2: 3483 steps
```

Q4

Naive Algorithm:

```
Rule1: 5562 steps + 122758 motions = 128320 actions

Rule2: 3738 steps + 94075 motions = 97831 actions
```

Improved Algorithm:

```
Rule1: 12553 steps + 4349 motions = 16902 actions

Rule2: 10553 steps + 3456 motions = 14009 actions
```

The result shows that the Improved Algorithm will cost more search times but can efficently reduce motions