[PA-6] Hidden Markov Model - Benjamin Cape

Description

> How do your implemented algorithms work? What design decisions did you make? How you laid out the problems? In particular, for this specific assignment, explain your model precisely, and explain exactly how you compute each new distribution of possible states.

- How do your implemented algorithms work?

My algorithm(Problem), takes a maze and a list of observations to determine where we are on the grid. It loops over each observation(reading) and updates the state based on that observation. We first create a new state, with empty expectations, and the current reading. We use the reading to print out the next process nicely in the console. We then loop over each location, find all of it's neighbors, and sum the probabilities of all the neighbors into the current location - this accounts for the **conjunction over the possible moves from all the neighbors**. The next step is to consider the **disjunction over the sensor reading**, this is accomplished by multiplying the transition model by the sensor model for each location given the proper reading correctness. The final step is **normalizing** all the probabilities such that we retain constant 100% likelihood for all outcomes.

- What design decisions did you make?

I decided to take, once again, and object oriented approach to split out functionality into respective entities. The `markov_model.py` files is generic, over any board with data at each location. The `maze.py` file contains all the maze related specific functionality, and the `main.py` file specifies the specific problem with the colors. I chose this to make it as generic as possible. I considered splitting the `markov_chain` even to allow for any generic markov chain, rather than board specific, but didn't seem necessary.

- How did you lay out the problems?

See above.

- In particular, for this specific assignment, explain your model precisely, and explain how you compute each new distribution of possible states.

The model is a hashmap of expectations for locations on a board. Each location is given a probability (expectation) that the robot is in that location. Therefore, updating probabilities is very fast and efficient, and clear. We can update probabilities in O(n) time by looping over each location. See Implementation for information on how we compute the new distribution of possible states.

Evaluation

> Do your implemented algorithms actually work? How well? If it doesn't work, can you tell why not? What partial successes did you have that deserve partial credit? Include a comparison of running time results using different heuristics and inference.

- Do your implemented algorithms actually work? How well?

Yes, in fact they seem to work very well. After running a various randomly generated mazes, the robot is generally very good at localizing itself.

- What partial successes did you have that deserve partial credit?

I believe that I deserve full credit.

- Include a comparison of running time results using different heuristics and inference.

There are no heuristics for this assignment, nor is there any inference. But here is an evaluation of state determination based on some randomly generated maps:

See bottom…

This one shows very well that with a proper path from (1,1) -> (1,0) -> (2,0) that is G, R, B and no other such path, we are placed INCREDIBLY likely at location (2,0) and very unlikely at the remaining locations of the board.

Discussion

1. What is the state transition model?

The state transition model is adding up all of the neighboring state's probabilities, because we assume uniform provability that the robot moves in any direction, so summing up is essentially saying, we were either in neighbor 1, 2 3 or 4 (max being 4 for 4 neighbors).

2. What is the sensor model?

The sensor model is multiplying the retrieved probability from summing the neighboring state probabilities with the probability that our reading was correct given the actual color of the square. I.e. if we just read a RED, and the square we are on is in fact red, then we multiply by 0.88, whereas for all other colors we would multiply by 0.04

3. How do you implement the filtering algorithm, given that there are several possible values for the state variable (the state variable is not boolean).

The filtering algorithm is simply keeping track of the previous state, and rather than re-computing everything on each transition, using the previous state to calculate the new state. That is described above, but simply put we create a new board, fill in initial probabilities given the transition model, and then update those probabilities with the sensor model.

Extra Credit

I implemented a viterbi algorithm that backtracks from the final state to find the best path.

This is visible with the following output from a random generated path and map. The `*` indicates where the robot thinks it is.

```
-------- State (Read: None) ---------
        0               1               2               3               4
4:      G0.05           #               G0.05           R0.05           Y0.05
3:      #               R0.05           B0.05           R0.05           #
2:      #               Y0.05           G0.05           G0.05           Y0.05
1:      Y0.05           G0.05           B0.05           B0.05           #
0:      Y0.05           R0.05           G0.05           G0.05           B0.05
-------- State (Read: B) ---------
        0               1               2               3               4
4:      G0.008          #               G0.008          R0.008          Y0.008
3:      #               R0.008          B0.187          R0.008          #
2:      #               Y0.008          *0.008         G0.008          Y0.008
1:      Y0.008          G0.008          B0.187          B0.187          #
0:      Y0.008          R0.008          G0.008          G0.008          B0.187
-------- State (Read: Y) ---------
        0               1               2               3               4
4:      G0.004          #               G0.025          R0.004          Y0.088
3:      #               R0.025          B0.004          R0.025          #
2:      #               *0.088         G0.046          G0.025          Y0.088
1:      Y0.088          G0.025          B0.025          B0.046          #
0:      Y0.088          R0.004          G0.025          G0.046          B0.067
-------- State (Read: G) ---------
        0               1               2               3               4
4:      G0.015          #               G0.055          R0.006          Y0.012
3:      #               R0.006          B0.005          R0.002          #
2:      #               Y0.008          G0.134          G0.194          Y0.012
1:      Y0.012          *0.194         B0.006          B0.006          #
0:      Y0.012          R0.006          G0.094          G0.174          B0.011
-------- State (Read: Y) ---------
        0               1               2               3               4
4:      G0.003          #               G0.005          R0.003          Y0.038
3:      #               R0.001          B0.008          R0.009          #
2:      #               *0.32          G0.009          G0.007          Y0.216
1:      Y0.216          G0.001          B0.018          B0.016          #
0:      Y0.039          R0.013          G0.012          G0.012          B0.009
```

```
-------- State (Read: G) ---------
        0               1               2               3               4
4:      G0.007          #               G0.015          R0.002          Y0.004
3:      #               R0.01           B0.001          R0.001          #
2:      #               Y0.01           G0.24           G0.17           Y0.02
1:      Y0.015          *0.385         B0.001          B0.002          #
0:      Y0.009          R0.002          G0.038          G0.033          B0.001

-------- State (Read: None) ---------
        0               1               2               3               4
4:      G0.05           Y0.05           R0.05           B0.05           R0.05
3:      #               #               #               Y0.05           B0.05
2:      R0.05           B0.05           G0.05           G0.05           #
1:      B0.05           G0.05           Y0.05           R0.05           B0.05
0:      G0.05           R0.05           Y0.05           #               Y0.05
-------- State (Read: G) ---------
        0               1               2               3               4
4:      G0.172          Y0.008          R0.008          B0.008          R0.008
3:      #               #               #               Y0.008          B0.008
2:      R0.008          B0.008          *0.172         G0.172          #
1:      B0.008          G0.172          Y0.008          R0.008          B0.008
0:      G0.172          R0.008          Y0.008          #               Y0.008
-------- State (Read: B) ---------
        0               1               2               3               4
4:      G0.018          Y0.007          R0.001          B0.023          R0.001
3:      #               #               #               Y0.007          B0.023
2:      R0.001          *0.267         G0.012          G0.012          #
1:      B0.267          G0.001          Y0.012          R0.007          B0.023
0:      G0.012          R0.012          Y0.001          #               Y0.001
-------- State (Read: G) ---------
        0               1               2               3               4
4:      G0.044          Y0.001          R0.001          B0.001          R0.002
3:      #               #               #               Y0.002          B0.002
2:      R0.018          B0.009          G0.221          G0.027          #
1:      B0.009          *0.407         Y0.001          R0.002          B0.002
0:      G0.221          R0.001          Y0.001          #               Y0.001
-------- State (Read: Y) ---------
        0               1               2               3               4
4:      G0.006          Y0.05           R0.0            B0.0            R0.0
3:      #               #               #               Y0.034          B0.0
2:      R0.003          B0.032          G0.012          G0.012          #
1:      B0.032          G0.001          *0.67          R0.002          B0.0
0:      G0.022          R0.03           Y0.004          #               Y0.005
-------- State (Read: Y) ---------
        0               1               2               3               4
4:      G0.003          Y0.086          R0.002          B0.001          R0.0
```

3:	#	#	#	Y0.038	B0.001
2:	R0.003	B0.002	G0.027	G0.002	#
1:	B0.002	G0.028	Y0.015	R0.025	B0.0
0:	G0.004	R0.002	*0.577	#	Y0.012