Probabilistic Reasoning Over Time

Luke Runnels

October 30, 2023

1 Models used

1.1 Sleep Hidden Markov Model

The following Hidden Markov Model(HMM) was used for experimentation. Given a state variable denoted S, which represents if a student got enough sleep or not along with evidence variables RE and SE, which represents if the student has red eyes or is sleeping in class, the HMM is represented in the following tables. At time 0, there is the prior table.

Prior Table of the HMM	
S_0	$P(S_0)$
s	0.7
¬ s	0.3

Table 1: Prior Table

At time t > 0, there is the transition table between previous state at time step t-1 and the current state at time step t.

Transition Table of the HMM		
S_{t-1}	S_t	Prob$(S_t \mid S_{t-1})$
s	s	0.8
s	¬ s	0.2
¬ s	s	0.3
¬ s	¬ s	0.7

Table 2: Transition Table

Finally, there is the sensor table between the state variable at time step t and the observation variable, which combines RE and SE into a single tuple.

Observation table of the HMM		
S_t	O_t	Prob($O_t \| S_t$)
s	RE SE	0.02
s	RE ¬SE	0.21
s	¬RE SE	0.18
s	¬RE ¬SE	0.49
¬s	RE SE	0.08
¬s	RE ¬SE	0.09
¬s	¬RE SE	0.72
¬s	¬RE ¬SE	0.22

Table 3: Observation Table

For easier implementation in the inference methods, the tables are compressed into the matrices

$$\mathbf{S}_0 = \begin{bmatrix} 0.7 \\ 0.3 \end{bmatrix} \tag{1}$$

for the prior table and

$$\mathbf{T} = \begin{bmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \end{bmatrix} \tag{2}$$

for the transition table.

The observation matrix is a little more nuanced. It will be initialized as an SxS identity matrix and constructed on the fly every time step as evidence comes in, with probability entries from Table 3. For example, at time step t, if the evidence tuple (RE, SE) is observed, then the observation matrix will be formatted as

$$\mathbf{O}_t = \begin{bmatrix} 0.02 & 0 \\ 0 & 0.08 \end{bmatrix} \tag{3}$$

where $O_{t_{11}}$ formats state value s and $O_{t_{22}}$ formats state value $\neg s$.

1.2 Robot maze modelled as a Dynamic Bayesian Network

The structure of a dynamic bayesian network(DBN) is somewhat similar to the HMM. However, the entries to the prior table, transition table, and observation table are not known ahead of time. In addition, there are multiple observation and state variables that are embedded within the tables, so attempting to model the tables ahead of time is a challenging endeavor by itself.

To model this situation, a robot maze environment is used for experimentation.

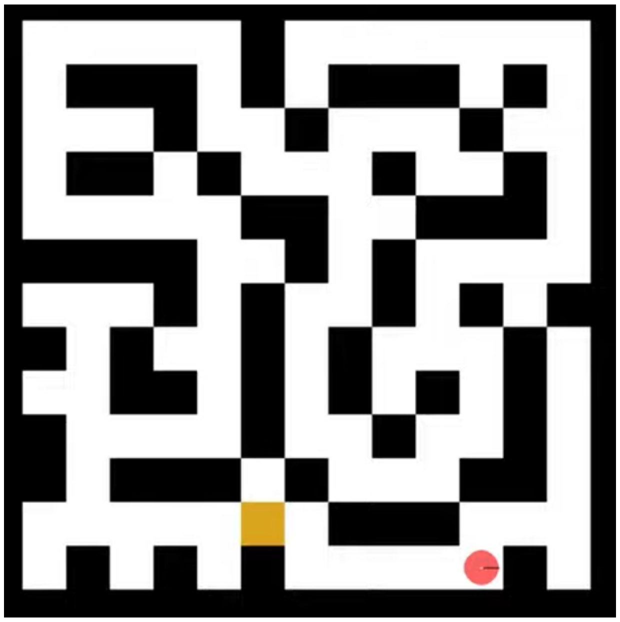


Figure 1: Screenshot of the robot maze environment that is used to model the DBN

It is known that the environment will maintain three state variables. These variables are X, Y, and robot heading. The prior table is an initial distribution of all three of the state variables, given that X and Y occupy a free cell that the robot can move onto. The transition and observation tables are similarly structured from the HMM, where there are probability distributions between the previous state and current state as well as between evidence and the current state. However, it is also known that the observations in the environment are sensors from the robot, which indicates if the north, south, east, and west adjacent cells are free or not. This is represent the observation variable in this environment.

This environment was originally developed by Robert Geraghty and James Hale of the TU Masters group. [1]

2 Algorithm Explanations

2.1 HMM Inference Techniques

2.1.1 Filtering & Smoothing using Country-Dance Algorithm

One of the most significant inference tasks that can be performed on a HMM is filtering, or state estimation. That is, given observations from time steps 1 to t where t is the current time, then filtering attempts to compute the probability distribution $P(\mathbf{S}_t|\mathbf{O}_{1:t})$ where $\mathbf{S}_t$ is the state variable at time t. To ensure that $P(\mathbf{S}_t|\mathbf{O}_{1:t})$ is computed with the most accuracy, it is a common step to apply smoothing for all of the distributions at the previous time steps, or compute the probability distributions $P(\mathbf{S}_k|\mathbf{O}_{1:t})$ for all $0 \leq k < t$.

Although these tasks can be computed separately, they can be computed in a forward and backward pass with constant space using the Country-Dance algorithm. The distributions $P(\mathbf{S}_t|\mathbf{O}_{1:t})$ and $P(\mathbf{S}_k|\mathbf{O}_{1:t})$, according to Russell el at. [2], can be computed as forward and backward passes using the matrix operations

$$P(\mathbf{S}_t|\mathbf{O}_{1:t}) = \alpha \mathbf{O}_t \mathbf{T}^T P(\mathbf{S}_t|\mathbf{O}_{1:t-1}) \tag{4}$$

and

$$P(\mathbf{S}_k|\mathbf{O}_{1:t}) = \mathbf{T}\mathbf{O}_{k+1} P(\mathbf{S}_{k+1}|\mathbf{O}_{1:t}) \tag{5}$$

respectively, where α is a normalization constant, $\mathbf{O}_t$ is an observation matrix, and $\mathbf{T}$ is the transition matrix.

It should be noted that $\mathbf{O}_t$ and $\mathbf{T}$ that formatted using the conventions laid out in section 1.1. Also, since 4 and 5 are recursive equations, the base case for $P(\mathbf{S}_t|\mathbf{O}_{1:t})$ is defined as the prior distribution of the HMM and the base case for $P(\mathbf{S}_k|\mathbf{O}_{1:t})$ is the last computed distribution of $P(\mathbf{S}_t|\mathbf{O}_{1:t})$ in the final step of the forward pass.

2.1.2 Online Smoothing using Fixed-Lag Smoothing

The smoothed distributions are computed along with the filtered distribution using the Country-Dance algorithm. However, there is an online formulation for computing the the smooth distribution d time steps back that does not require a forward and backward pass.

According to Russell el at. [2], the smooth distribution $P(\mathbf{S}_k|\mathbf{O}_{1:t})$ can be computed as

$$P(\mathbf{S}_{t-d}|\mathbf{O}_{1:t}) = \alpha P(\mathbf{S}_t|\mathbf{O}_{1:t}) \times \mathbf{B1} \tag{6}$$

as long as t > d+1, where α is a normalization constant, $\mathbf{B}$ is a maintained d-transformation matrix and $\mathbf{1}$ is a vector of ones. $\mathbf{B}$ is initialized as an SxS identity matrix, and updated as

$$\mathbf{B} = \mathbf{BTO}_t \tag{7}$$

where t $\leq$ d.

If t > d, then $\mathbf{B}$ is updated as

$$\mathbf{B} = \mathbf{O}_{t-d}^{-1}\mathbf{T}^{-1}\mathbf{BTO}_t \tag{8}$$

A separate vector of evidence is often maintained to calculate $\mathbf{O}_{t-d}$ as new evidence comes in.

2.1.3 Most likely explanation using Viterbi Algorithm

According to Russell el at. [2], the most likely sequence of state values from the state $\mathbf{S}_t$ aims to compute $argmax_{\mathbf{S}_{1:t}} P(\mathbf{S}_{1:t}|\mathbf{O}_{1:t})$ for each time step t seen so far. This can be accomplished using the Viterbi Algorithm. The Viterbi Algorithm is implemented by finding $max_{\mathbf{S}_{1:t-1}} P(\mathbf{S}_{1:t-1}, \mathbf{S}_t, \mathbf{O}_{1:t})$, where

$$m_{1:t} = P(O_t|\mathbf{S}_t) max_{\mathbf{S}_{t-1}} P(\mathbf{S}_t|\mathbf{S}_{t-1}) max_{\mathbf{S}_{1:t-2}} P(\mathbf{S}_{1:t-2}, \mathbf{S}_{t-1}, \mathbf{O}_{1:t-1}) \tag{9}$$

Since $m_{1:t}$ is in a recursive relationship with $max_{\mathbf{S}_{1:t-2}} P(\mathbf{S}_{1:t-2}, \mathbf{S}_{t-1}, \mathbf{O}_{1:t-1})$ or $m_{1:t-1}$, this algorithm is usually implemented through dynamic programming by maintaining a message sequence at every time step t.

This represents the forward pass of the Viterbi Algorithm. The backward pass simply finds the state value S_t that maximizes $m_{1:t}$ at every time step. The computed sequence of S_t will be the most likely sequence.

2.2 DBN Inference Techniques

2.2.1 Why Particle Filtering?

HMMs are considered a special case of DBNs, with just one hidden state variable and one observation variable. Similarly to HMMs, DBNs define a prior model, a transition model, and an observation model. Unlike HMMs though, these models can be distributed over multiple hidden and observation variables. Therefore, applying exact inference techniques like the ones defined in section 2.1 are much more computationally expensive. So, approximate inference is usually applied to DBNs through a series of algorithms known as particle filtering.

2.2.2 Inference using Particle Filtering

Although there are numerious interpretations for particle filtering, this paper will implement Russell and Norvig's explanation by sampling and weighting. Roughly speaking, a collections of state values called particles will first be sampled using the DBN's prior model. Then, the following series of steps will be applied at each time step.

1. Each particle is propagated forward using the DBN's transition model .
2. Each particle is assigned a weight based on the DBN's observation model.
3. The collection of particles are resampled with replacement using the weights.

3 Performance and Analysis

3.1 HMM Performance

In the following subsections, the performance comparison between a fixed evidence tuple and a random evidence tuple at every time step will be reported and analyzed using the algorithms described in section 2.1 and the HMM described in section 1.1. The fixed evidence tuple will be denoted as notRedEyes_notSleepClass, which corresponds to the entries $\neg RE \neg SE$ in table 3. All algorithm tests will be ran from time step 1 to time step 25.

3.1.1 HMM Filtering Results

The following two figures show the probability of the Country-Dance algorithm predicting the state variable value s at time t.

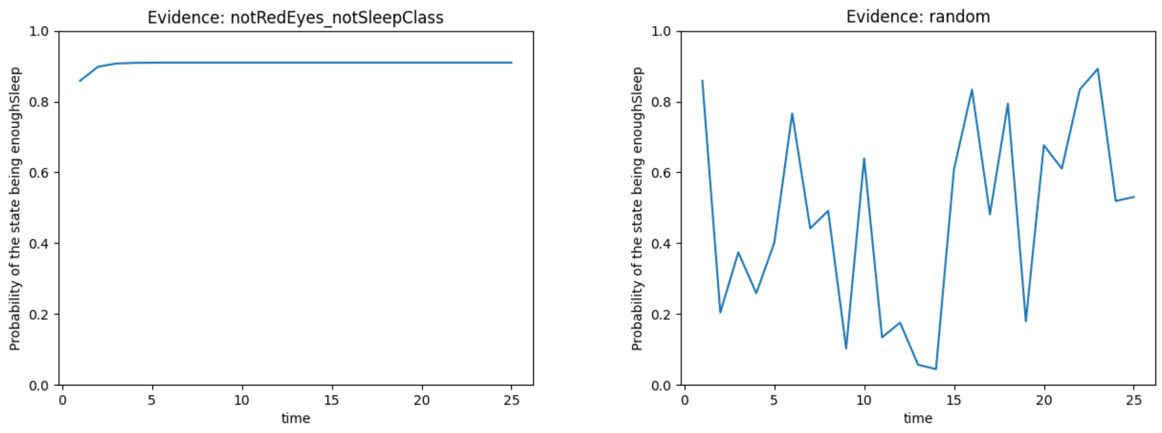


Figure 2: State estimation for time steps 1 to 25. Reports the probability of the most likely state being enough sleep

3.1.2 HMM Smoothing Results

The following show the comparison between the Country-Dance algorithm and the Fixed Lag Smoothing algorithm for predicting the probability of the state variable value s, for time slots t-2, t-3, t-4, and t-5 with fixed evidence and random evidence.

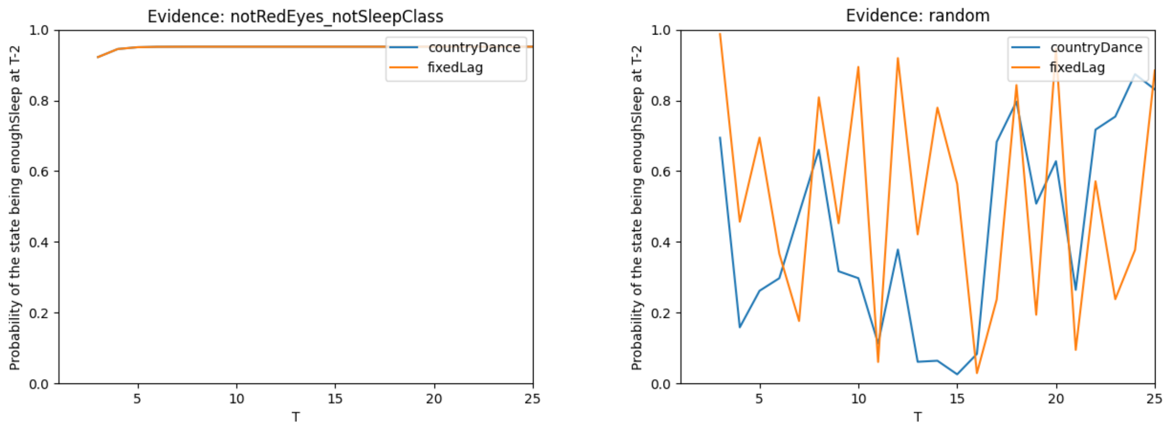
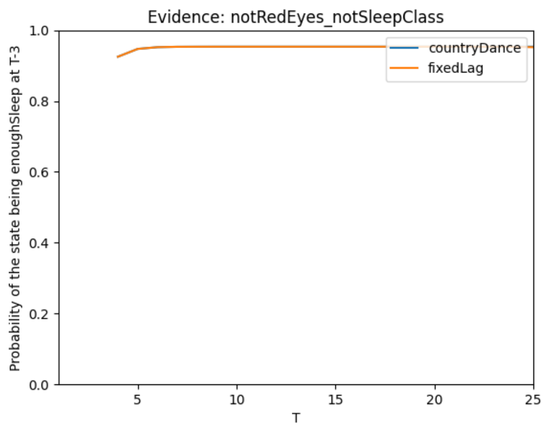


Figure 3: Fixed lag smoothing for time steps 1 to 25 with a lag value of 2(Probability of the state being enough sleep at T-2)



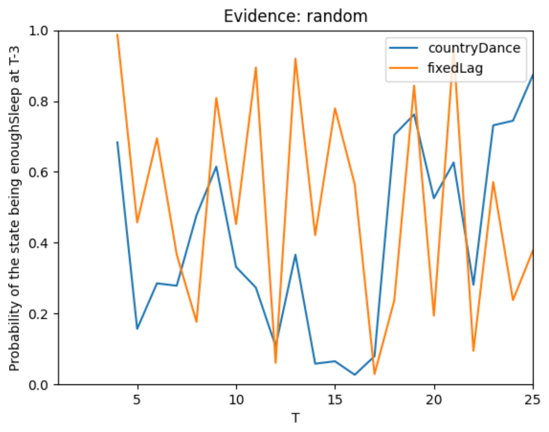
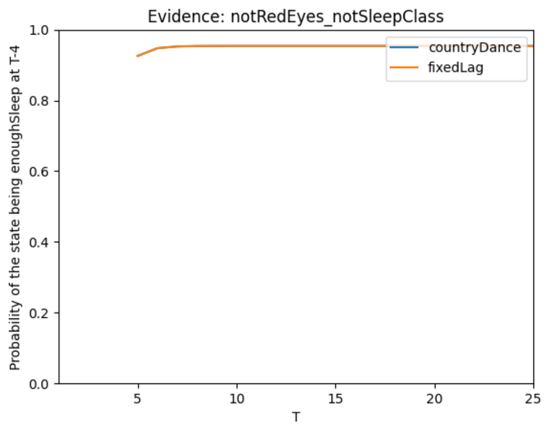


Figure 4: Fixed lag smoothing for time steps 1 to 25 with a lag value of 3(Probability of the state being enough sleep at T-3)



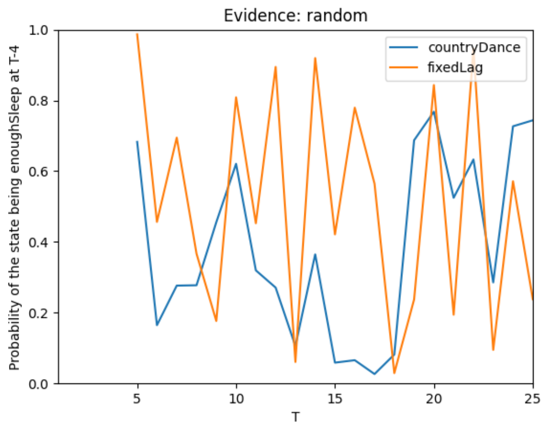
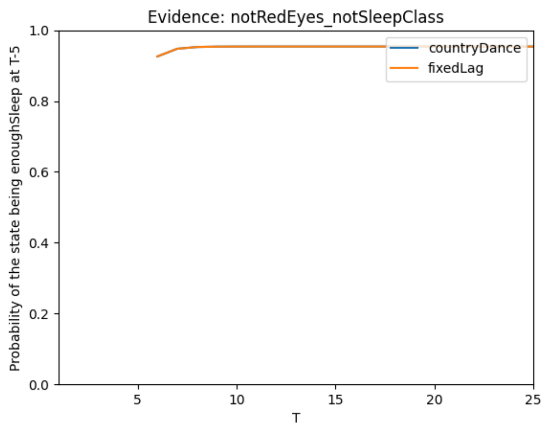


Figure 5: Fixed lag smoothing for time steps 1 to 25 with a lag value of 4(Probability of the state being enough sleep at T-4)



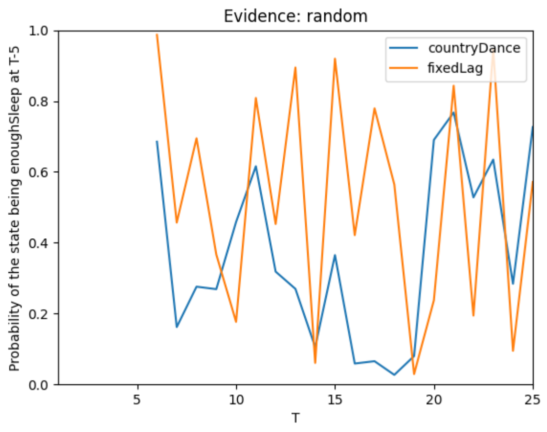


Figure 6: Fixed lag smoothing for time steps 1 to 25 with a lag value of 5(Probability of the state being enough sleep at T-5)

3.1.3 HMM Most Likely Explanation Results

The following two tables showcase the most likely path computed by the Viterbi algorithm at time step 25 for fixed evidence and random evidence. The first column reports the probability of the state variable being enoughSleep, which represents the state variable value s.

Viberti Algorithm at t=25		
Probability of enoughSleep	stateVariable	stateValue
0.9258	s_1	enoughSleep
0.9477	s_2	enoughSleep
0.9525	s_3	enoughSleep
0.9536	s_4	enoughSleep
0.9538	s_5	enoughSleep
0.9539	s_6	enoughSleep
0.9539	s_7	enoughSleep
0.9539	s_8	enoughSleep
0.9539	s_9	enoughSleep
0.9539	s_{10}	enoughSleep
0.9539	s_{11}	enoughSleep
0.9539	s_{12}	enoughSleep
0.9539	s_{13}	enoughSleep
0.9539	s_{14}	enoughSleep
0.9539	s_{15}	enoughSleep
0.9539	s_{16}	enoughSleep
0.9539	s_{17}	enoughSleep
0.9539	s_{18}	enoughSleep
0.9539	s_{19}	enoughSleep
0.9539	s_{20}	enoughSleep
0.9538	s_{21}	enoughSleep
0.9534	s_{22}	enoughSleep
0.9517	s_{23}	enoughSleep
0.9441	s_{24}	enoughSleep
0.9097	s_{25}	enoughSleep

Table 4: Most likely explanation at time step 25 given the evidence tuple (notRedEyes, notSleepInClass)

Viberti Algorithm at t=25		
Probability of enoughSleep	stateVariable	stateValue
0.6838	s_1	enoughSleep
0.1602	s_2	notEnoughSleep
0.2720	s_3	notEnoughSleep
0.2690	s_4	notEnoughSleep
0.4558	s_5	notEnoughSleep
0.6150	s_6	notEnoughSleep
0.3177	s_7	notEnoughSleep
0.2696	s_8	notEnoughSleep
0.1054	s_9	notEnoughSleep
0.3650	s_{10}	notEnoughSleep
0.0589	s_{11}	notEnoughSleep
0.0656	s_{12}	notEnoughSleep
0.0267	s_{13}	notEnoughSleep
0.0804	s_{14}	notEnoughSleep
0.6908	s_{15}	enoughSleep
0.7690	s_{16}	enoughSleep
0.5285	s_{17}	enoughSleep
0.6341	s_{18}	enoughSleep
0.2840	s_{19}	enoughSleep
0.7265	s_{20}	enoughSleep
0.7437	s_{21}	enoughSleep
0.8728	s_{22}	enoughSleep
0.8310	s_{23}	enoughSleep
0.5044	s_{24}	enoughSleep
0.5304	s_{25}	enoughSleep

Table 5: Most likely explanation at time step 25 given random evidence

3.2 HMM Analysis

Based on the tables and figures from these sections, it is clear that given fixed evidence, the accuracy of these algorithms are consistent, as the likelihood of predicting enoughSleep is consistently high in all three of the algorithms.

However, the accuracy of these algorithms given random evidence is interesting. As seen in these figures, neither the curve shapes from the country dance, fixed lag, or Viterbi algorithms match. Comparing country dance and fixed lag, the inconsistent curves can be attributed to the fact that fixed lag is not receiving all observations at every time step, unlike country dance. Comparing country dance and the Viterbi algorithm, the viterbi algorithm is selecting the maximum transition $P(\mathbf{S}_t|\mathbf{S}_{t-1})$ and the maximum states at every time step. This is not necessarily the case with the country dance algorithm, as it factors in all state possibilities at every time step. So, this mismatch with the country dance, fixed lag smoothing, and the Viterbi algorithm does make sense, as they are making slightly different calculations, along with random values from the observations.

3.3 DBN Performance

In this section, the performance of the particle filtering will be test on various parameter configurations of the robot maze from section 1.2. There are five parameters that are significant.

1. Dimension: The size of the maze in an NxN setting.
2. Number of particles: The number of samples that will be maintained in the particle filtering algorithm.
3. Action Noise: The probability that an action will move either direction perpendicular to the inteded direction.
4. Observation Noise: The probability that any given observation value will flip values erroneously.
5. Action Bias: Provides a bias for the robots actions. Positive values increase the likelihood of South and East movements, and negative favor North and West. [1]

To thoroughly test the performance of the environment, various runs will be executed with the following parameters values.

1. Dimension: 10, 20, 30
2. Number of particles: 10, 100, 500, 1000
3. Action Noise: 0.01, 0.5, 0.95
4. Observation Noise: 0.01, 0.5, 0.95
5. Action Bias: -0.5, 0, 0.5

These tests will construct particles in the form (x, y, heading), which are the state variables in the environment.

In the following subsections, the top three combinations from every maze dimension will be presented according to the number of correct locations that that particular particle filter instance predicted with respect to the actual robot location. It should be noted that location refers to (x, y) from the particle (x, y, heading), since these two state variables provide the best indication that the particle filter can localize onto the robot.

The full tables that detail the number of correct locations will be listed in the appendix.

3.3.1 Performance for a 10x10 maze

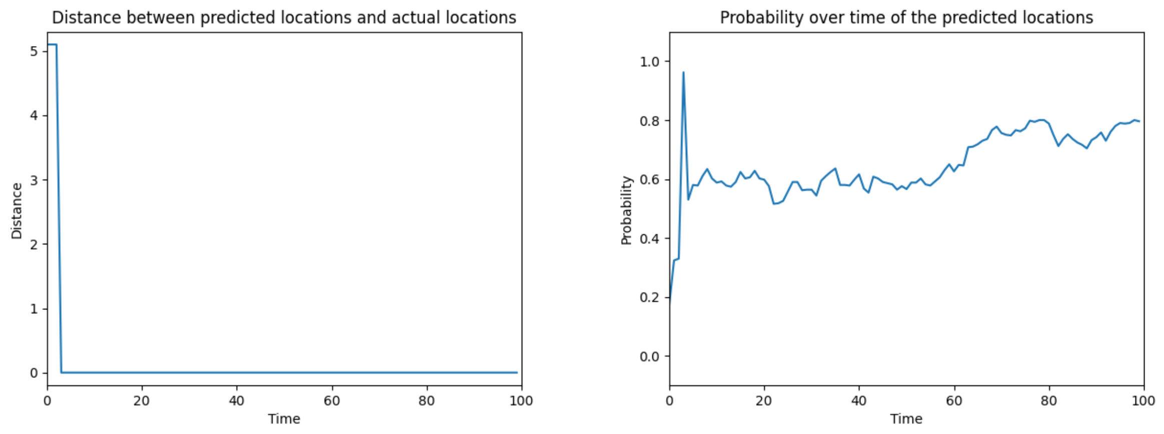


Figure 7: Number of particles = 500, action noise = 0.95, observation noise = 0.01, action bias = -0.5. Predicted 97/100 locations for 100 time steps.

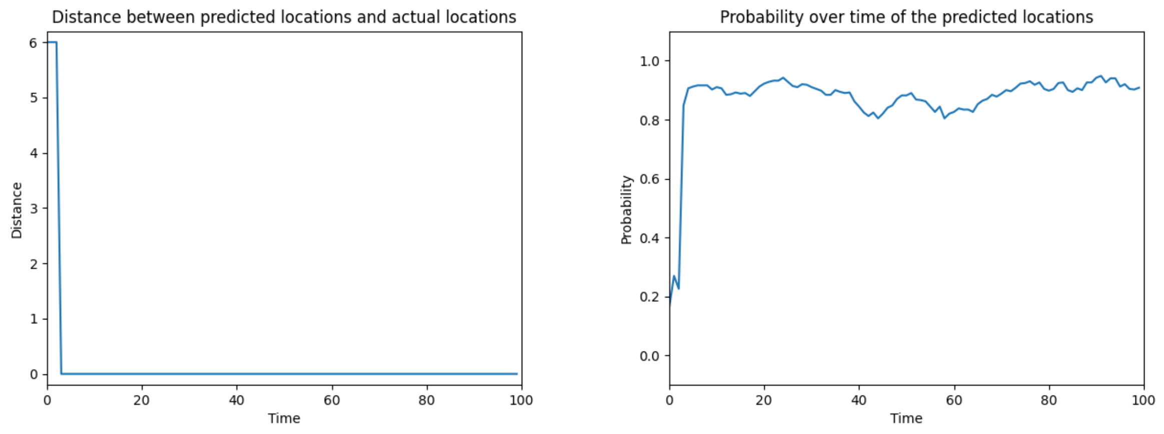


Figure 8: Number of particles = 500, action noise = 0.95, observation noise = 0.01, action bias = 0. Predicted 97/100 locations for 100 time steps.

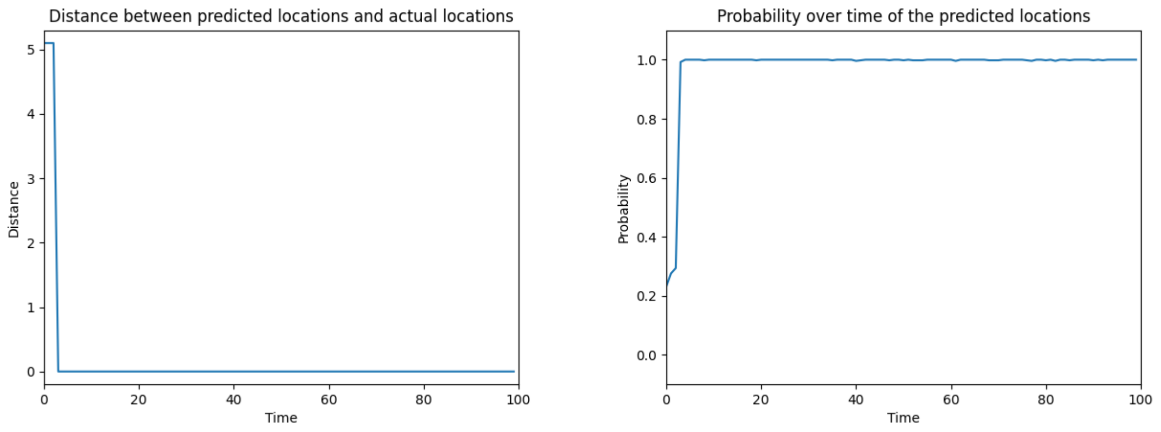


Figure 9: Number of particles = 500, action noise = 0.95, observation noise = 0.01, action bias = 0. Predicted 97/100 locations for 100 time steps.

3.3.2 Performance for a 20x20 maze

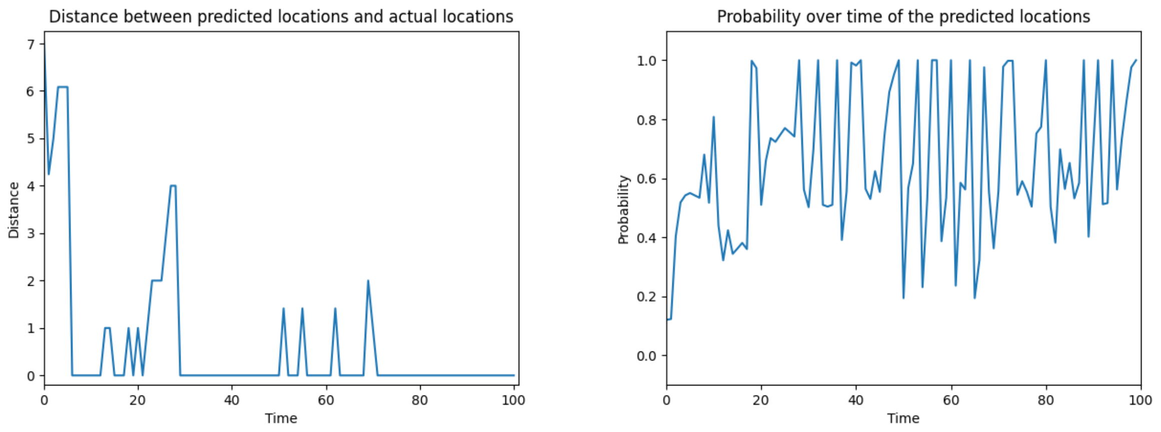


Figure 10: Number of particles = 500, action noise = 0.5, observation noise = 0.01, action bias = 0. Predicted 70/100 locations for 100 time steps.

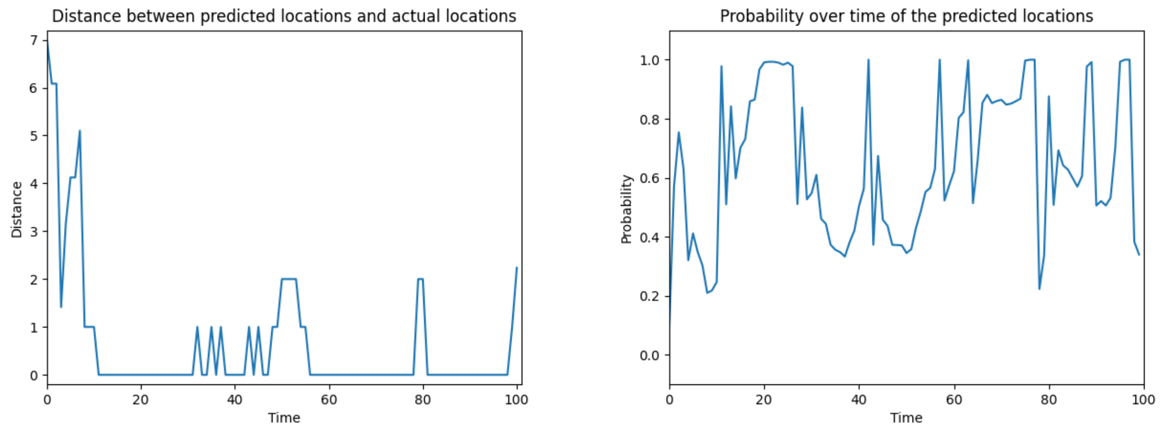


Figure 11: Number of particles = 1000, action noise = 0.5, observation noise = 0.01, action bias = 0. Predicted 73/100 locations for 100 time steps.

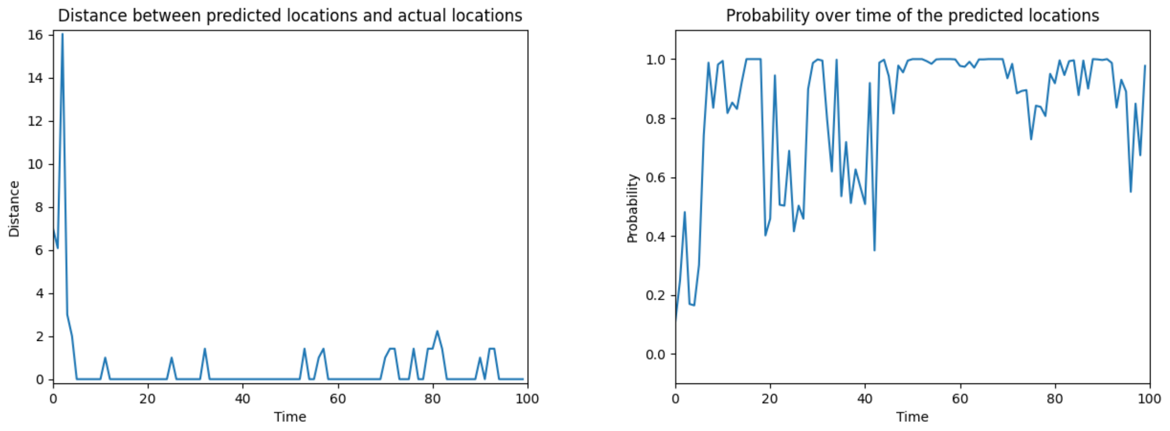


Figure 12: Number of particles = 1000, action noise = 0.5, observation noise = 0.95, action bias = 0.5. Predicted 78/100 locations for 100 time steps.

3.3.3 Performance for a 30x30 maze

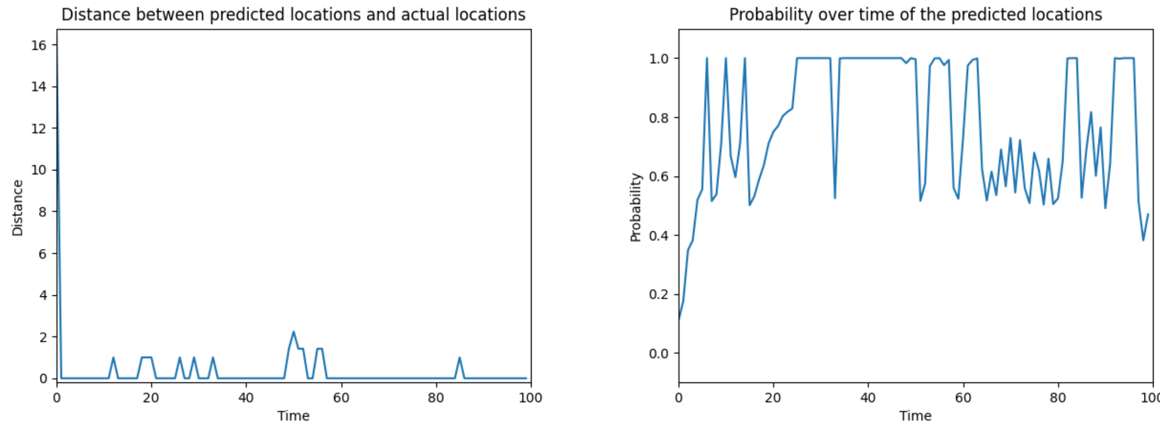


Figure 13: Number of particles = 1000, action noise = 0.5, observation noise = 0.01, action bias = -0.5. Predicted 85/100 locations for 100 time steps.

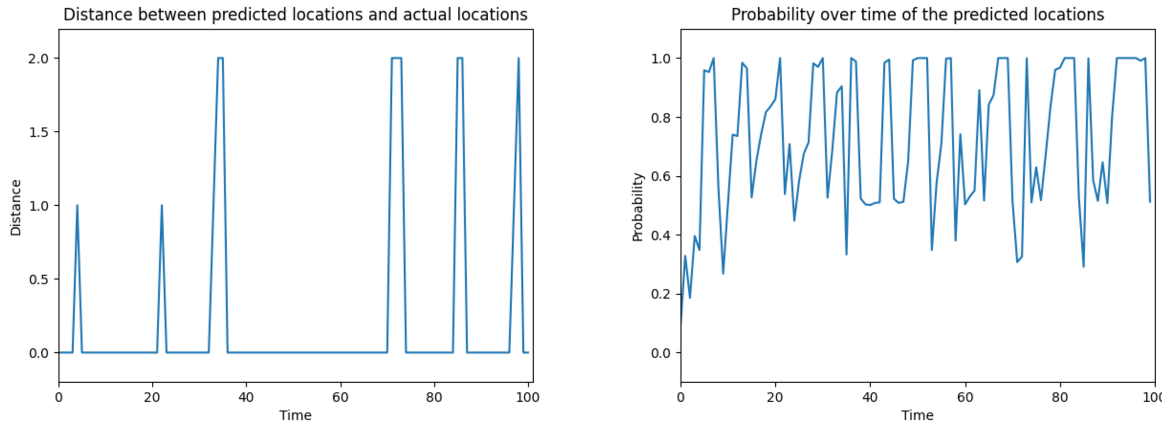


Figure 14: Number of particles = 1000, action noise = 0.5, observation noise = 0.01, action bias = 0. Predicted 89/100 locations for 100 time steps.

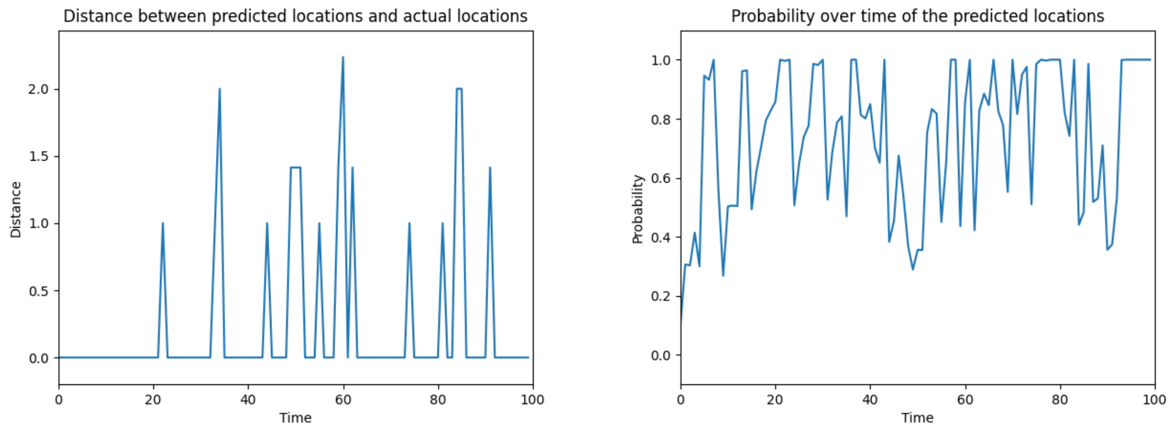


Figure 15: Number of particles = 1000, action noise = 0.5, observation noise = 0.01, action bias = 0.5. Predicted 84/100 locations for 100 time steps.

3.4 DBN Analysis

These results are interesting. As the dimensions of the maze increases, better localization is associated with more particles. It is also interesting that the action noise parameter stayed constant at 0.5 for 20x20 and 30x30 mazes, but it jumps to 0.95 for 10x10 mazes. These results most likely indicate the effect on that the maze has on the robot.

Since the robot can only see the cells immediately surrounding it, the observations can become indistinguishable from one another for massive mazes in terms of what location the robot location could be at. Therefore, there needs to be lots of exploration for the particle filter instance in order to localize on the robot. So, for the mazes 20x20 and 30x30, it does make sense that the best instances ran with 1000 particles, 0.5 for action noise, and 0.01 for observation noise. This parameter combination seems to favor exploration with 1000 parameters to filter out, with some sense of where the robot could be given low observation noise and some action noise.

However, for the 10x10 maze, the particle filter most likely does not need to explore as much, but it needs to quickly exploit the particles quickly, This probably explains why 500 particles and a high action noise combination performed the best.

4 Conclusion

Performing probabilistic reasoning overtime is a powerful technique for making inferences in a dynamic probabilistic model. As shown in this paper, it is possible to perform inference tasks through exact inference, given a simple Hidden Markov Model, as long as evidence is expected to remain static. However, if a lot of dynamic components to the model such as changing evidence, multiple hidden states and evidence variables, and as well as any other dynamic associations, then approximation inference techniques such as particle filtering is better suited.

5 Appendix

5.1 Number of correct locations on mazes of dimension 10x10

Dimension 10x10, Number of particles = 10			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	0
0.01	0.01	0	9
0.01	0.01	0.5	0
0.01	0.5	-0.5	37
0.01	0.5	0	11
0.01	0.5	0.5	17
0.01	0.95	-0.5	33
0.01	0.95	0	2
0.01	0.95	0.5	0
0.5	0.01	-0.5	0
0.5	0.01	0	21
0.5	0.01	0.5	0
0.5	0.5	-0.5	27
0.5	0.5	0	12
0.5	0.5	0.5	3
0.5	0.95	-0.5	0
0.5	0.95	0	0
0.5	0.95	0.5	0
0.95	0.01	-0.5	0
0.95	0.01	0	0
0.95	0.01	0.5	0
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	28
0.95	0.95	0	12
0.95	0.95	0.5	0

Table 6: Most likely explanation at time step 25 given random evidence

Dimension 10x10, Number of particles = 100			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	0
0.01	0.01	0	21
0.01	0.01	0.5	34
0.01	0.5	-0.5	39
0.01	0.5	0	7
0.01	0.5	0.5	8
0.01	0.95	-0.5	0
0.01	0.95	0	26
0.01	0.95	0.5	30
0.5	0.01	-0.5	71
0.5	0.01	0	52
0.5	0.01	0.5	72
0.5	0.5	-0.5	12
0.5	0.5	0	0
0.5	0.5	0.5	4
0.5	0.95	-0.5	81
0.5	0.95	0	81
0.5	0.95	0.5	76
0.95	0.01	-0.5	58
0.95	0.01	0	28
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	52
0.95	0.95	0	51
0.95	0.95	0.5	83

Table 7: Most likely explanation at time step 25 given random evidence

Dimension 10x10, Number of particles = 500			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	35
0.01	0.01	0	37
0.01	0.01	0.5	34
0.01	0.5	-0.5	28
0.01	0.5	0	8
0.01	0.5	0.5	21
0.01	0.95	-0.5	1
0.01	0.95	0	27
0.01	0.95	0.5	31
0.5	0.01	-0.5	75
0.5	0.01	0	84
0.5	0.01	0.5	81
0.5	0.5	-0.5	0
0.5	0.5	0	4
0.5	0.5	0.5	7
0.5	0.95	-0.5	75
0.5	0.95	0	83
0.5	0.95	0.5	81
0.95	0.01	-0.5	97
0.95	0.01	0	97
0.95	0.01	0.5	97
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	15
0.95	0.95	-0.5	97
0.95	0.95	0	94
0.95	0.95	0.5	94

Table 8: Most likely explanation at time step 25 given random evidence

Dimension 10x10, Number of particles = 1000			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	38
0.01	0.01	0	4
0.01	0.01	0.5	38
0.01	0.5	-0.5	33
0.01	0.5	0	14
0.01	0.5	0.5	13
0.01	0.95	-0.5	58
0.01	0.95	0	45
0.01	0.95	0.5	30
0.5	0.01	-0.5	80
0.5	0.01	0	77
0.5	0.01	0.5	33
0.5	0.5	-0.5	10
0.5	0.5	0	3
0.5	0.5	0.5	5
0.5	0.95	-0.5	70
0.5	0.95	0	84
0.5	0.95	0.5	34
0.95	0.01	-0.5	82
0.95	0.01	0	83
0.95	0.01	0.5	84
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	81
0.95	0.95	0	77
0.95	0.95	0.5	87

Table 9: Most likely explanation at time step 25 given random evidence

5.2 Number of correct locations on mazes of dimension 20x20

Dimension 20x20, Number of particles = 10			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	0
0.01	0.01	0	0
0.01	0.01	0.5	0
0.01	0.5	-0.5	0
0.01	0.5	0	0
0.01	0.5	0.5	0
0.01	0.95	-0.5	0
0.01	0.95	0	8
0.01	0.95	0.5	10
0.5	0.01	-0.5	0
0.5	0.01	0	0
0.5	0.01	0.5	0
0.5	0.5	-0.5	0
0.5	0.5	0	0
0.5	0.5	0.5	0
0.5	0.95	-0.5	21
0.5	0.95	0	0
0.5	0.95	0.5	34
0.95	0.01	-0.5	0
0.95	0.01	0	0
0.95	0.01	0.5	0
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	0
0.95	0.95	0	0
0.95	0.95	0.5	0

Table 10: Most likely explanation at time step 25 given random evidence

Dimension 20x20, Number of particles = 100			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	19
0.01	0.01	0	10
0.01	0.01	0.5	0
0.01	0.5	-0.5	2
0.01	0.5	0	0
0.01	0.5	0.5	0
0.01	0.95	-0.5	0
0.01	0.95	0	0
0.01	0.95	0.5	0
0.5	0.01	-0.5	34
0.5	0.01	0	0
0.5	0.01	0.5	0
0.5	0.5	-0.5	1
0.5	0.5	0	1
0.5	0.5	0.5	0
0.5	0.95	-0.5	52
0.5	0.95	0	65
0.5	0.95	0.5	0
0.95	0.01	-0.5	0
0.95	0.01	0	0
0.95	0.01	0.5	0
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	0
0.95	0.95	0	0
0.95	0.95	0.5	0

Table 11: Most likely explanation at time step 25 given random evidence

Dimension 20x20, Number of particles = 500			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	35
0.01	0.01	0	18
0.01	0.01	0.5	0
0.01	0.5	-0.5	1
0.01	0.5	0	0
0.01	0.5	0.5	0
0.01	0.95	-0.5	21
0.01	0.95	0	0
0.01	0.95	0.5	1
0.5	0.01	-0.5	43
0.5	0.01	0	79
0.5	0.01	0.5	71
0.5	0.5	-0.5	4
0.5	0.5	0	0
0.5	0.5	0.5	0
0.5	0.95	-0.5	39
0.5	0.95	0	0
0.5	0.95	0.5	0
0.95	0.01	-0.5	0
0.95	0.01	0	0
0.95	0.01	0.5	0
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	0
0.95	0.95	0	0
0.95	0.95	0.5	0

Table 12: Most likely explanation at time step 25 given random evidence

Dimension 20x20, Number of particles = 1000			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	26
0.01	0.01	0	24
0.01	0.01	0.5	0
0.01	0.5	-0.5	2
0.01	0.5	0	3
0.01	0.5	0.5	2
0.01	0.95	-0.5	24
0.01	0.95	0	0
0.01	0.95	0.5	0
0.5	0.01	-0.5	67
0.5	0.01	0	73
0.5	0.01	0.5	69
0.5	0.5	-0.5	5
0.5	0.5	0	1
0.5	0.5	0.5	0
0.5	0.95	-0.5	68
0.5	0.95	0	65
0.5	0.95	0.5	78
0.95	0.01	-0.5	0
0.95	0.01	0	0
0.95	0.01	0.5	0
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	0
0.95	0.95	0	1
0.95	0.95	0.5	0

Table 13: Most likely explanation at time step 25 given random evidence

5.3 Number of correct locations on mazes of dimension 30x30

Dimension 30x30, Number of particles = 10			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	0
0.01	0.01	0	0
0.01	0.01	0.5	0
0.01	0.5	-0.5	0
0.01	0.5	0	0
0.01	0.5	0.5	0
0.01	0.95	-0.5	0
0.01	0.95	0	0
0.01	0.95	0.5	0
0.5	0.01	-0.5	0
0.5	0.01	0	0
0.5	0.01	0.5	0
0.5	0.5	-0.5	0
0.5	0.5	0	0
0.5	0.5	0.5	0
0.5	0.95	-0.5	0
0.5	0.95	0	0
0.5	0.95	0.5	0
0.95	0.01	-0.5	0
0.95	0.01	0	0
0.95	0.01	0.5	0
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	0
0.95	0.95	0	0
0.95	0.95	0.5	0

Table 14: Most likely explanation at time step 25 given random evidence

Dimension 30x30, Number of particles = 100			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	0
0.01	0.01	0	0
0.01	0.01	0.5	14
0.01	0.5	-0.5	0
0.01	0.5	0	0
0.01	0.5	0.5	0
0.01	0.95	-0.5	0
0.01	0.95	0	0
0.01	0.95	0.5	0
0.5	0.01	-0.5	0
0.5	0.01	0	0
0.5	0.01	0.5	0
0.5	0.5	-0.5	0
0.5	0.5	0	0
0.5	0.5	0.5	0
0.5	0.95	-0.5	0
0.5	0.95	0	0
0.5	0.95	0.5	0
0.95	0.01	-0.5	0
0.95	0.01	0	0
0.95	0.01	0.5	0
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	0
0.95	0.95	0	0
0.95	0.95	0.5	0

Table 15: Most likely explanation at time step 25 given random evidence

Dimension 30x30, Number of particles = 500			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	0
0.01	0.01	0	0
0.01	0.01	0.5	25
0.01	0.5	-0.5	0
0.01	0.5	0	0
0.01	0.5	0.5	0
0.01	0.95	-0.5	0
0.01	0.95	0	0
0.01	0.95	0.5	21
0.5	0.01	-0.5	0
0.5	0.01	0	0
0.5	0.01	0.5	0
0.5	0.5	-0.5	0
0.5	0.5	0	0
0.5	0.5	0.5	0
0.5	0.95	-0.5	0
0.5	0.95	0	0
0.5	0.95	0.5	0
0.95	0.01	-0.5	80
0.95	0.01	0	47
0.95	0.01	0.5	0
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	79
0.95	0.95	0	60
0.95	0.95	0.5	0

Table 16: Most likely explanation at time step 25 given random evidence

Dimension 30x30, Number of particles = 1000			
actionNoise	observationNoise	actionBias	correctLocations
0.01	0.01	-0.5	0
0.01	0.01	0	2
0.01	0.01	0.5	35
0.01	0.5	-0.5	0
0.01	0.5	0	0
0.01	0.5	0.5	0
0.01	0.95	-0.5	1
0.01	0.95	0	1
0.01	0.95	0.5	32
0.5	0.01	-0.5	85
0.5	0.01	0	89
0.5	0.01	0.5	84
0.5	0.5	-0.5	0
0.5	0.5	0	0
0.5	0.5	0.5	0
0.5	0.95	-0.5	77
0.5	0.95	0	73
0.5	0.95	0.5	68
0.95	0.01	-0.5	0
0.95	0.01	0	0
0.95	0.01	0.5	0
0.95	0.5	-0.5	0
0.95	0.5	0	0
0.95	0.5	0.5	0
0.95	0.95	-0.5	16
0.95	0.95	0	0
0.95	0.95	0.5	62

Table 17: Most likely explanation at time step 25 given random evidence

References

[1] James Hale Robert Geraghty. Cs5313_localization_env. *https://github.com/robger98/CS5313_Localization_Env*, 2020.

[2] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach.* Pearsons, 4 edition, 2021.