VARIN KALA
2022561

AI ASSIGNMENT 3

Date ..../..../........

THEORY

Q1) a) Direct Sampling is simple and accurate, but requires complete distribution knowledge. Rejection Sampling is flexible with an approx. proposal distribution but is inefficient for constrained distribution (60% prefer stressed air travel).
Gibbs sampling can handle complex dependencies but takes a lot of time to converge & requires conditional probabilities.

In this case, Direct Sampling can be challenging because of lack of explicit joint distribution. Rejection Sampling works, but because of the constraints, the acceptance rate would be low ~~leading~~ making it computationally expensive.
Gibbs sampling can estimate joint probabilities and can effectively model dependencies given the conditional probabilities.

b) $P(\text{leisure} \mid \text{train}) = 0.400$
$n(\text{train}) = 30$

$\therefore\ n(\text{leisure}) = P(\text{leisure} \mid \text{train}) \cdot n(\text{train})$
$= 0.4 \times 30 = \boxed{12}$

c) $P(\text{air}) = 0.8$ $\quad P(\text{business} \mid \text{air}) = 0.2$
$\therefore\ P(\text{air} \wedge \text{business}) = P(\text{business} \mid \text{air}) \cdot P(\text{air})$
$= 0.2 \times 0.8 = \boxed{0.160}$

d) ~~According~~ As per the law of Large Numbers, with an increase in size of sample, the estimates converge to true probabilities, which results in increase in Accuracy.
Also, large sample size results in a reduced variance of the estimates and Precision improves

Q2) a) Let ~~Journal~~ J be RV representing if a person accesses academic journals.

Let B be RV representing if a person reads books.

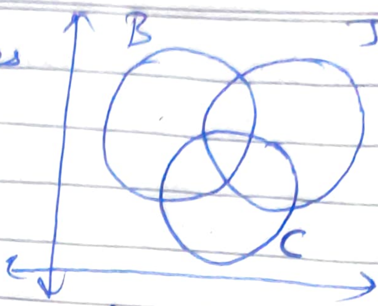
Let C be RV representing if a person participates in book clubs.

The statements are as follows:

- $P(b \vee j) = 0.91$
- $P(j \mid b) = 0.4$ and $P(\neg j \mid b) = 0.6$
- ~~$P(c \mid b) = 0.32$~~ $P(c \mid b, j) = 0.32$ & $P(c \mid b, \neg j) = 0.32$
- $P(j \wedge \neg b) = 0.227$
- $P(\neg j \wedge \neg b) = 0.09$
- $P(j \mid \neg b) = 0.716$
- $P(c \wedge j) = 0.088$
- ~~$P(\neg c \vee j)$~~ $= P(c \vee j) = 0.631$
- $P(j \mid c) = 0.4$
- $P(j) = 0.5$
- ~~$P(c \mid \neg b) = 0.0044$~~
- $P(c \mid \neg b, j) = 0.0044$ & $P(c \mid \neg b, \neg j) = 0.0044$

Date ..../..../........

After solving these probabilities we get the following:



~~P(j) = 0.5, P(b) = 0.683, P(c) =~~

c)

	b		¬b	
	c	¬c	c	¬c
j	0.0874	0.1858	0.0010	0.0004
¬j	0.1311	0.2787	0.2260	0.0896

b) All probabilities are +ve - ①

All probabilities sum up to 1 - ②

Additivity property is justified - ③

d) $P(b) = 0.683$, $P(j) = 0.5$, $P(c) = 0.4455$

After calculating for all possible cases,

B & J are independent given C &
C & J are independent given B.

Date ..../..../.....

Q3) Let Misclassification Alarm be M
Let Adversarial Perturbations be A
Let Backdoor Attacks be B
Let New Evidence be E.

a) $P(A|M,E) = \dfrac{P(M|A,E) \cdot P(A|E)}{P(M|E)}$

b) Priors: $P(A), P(B)$
Probability of Adversarial Perturbations & Backdoor Attacks.

Likelihood: $P(M|A), P(M|B), P(B|E), P(M|E)$
Likelihood of Alarm given A & B.

$P(B|E)$: Likelihood of Backdoor attacks given evidence also influences the posterior.

Posterior: $P(A|M,E)$
Updated belief about A given likelihood probabilities.

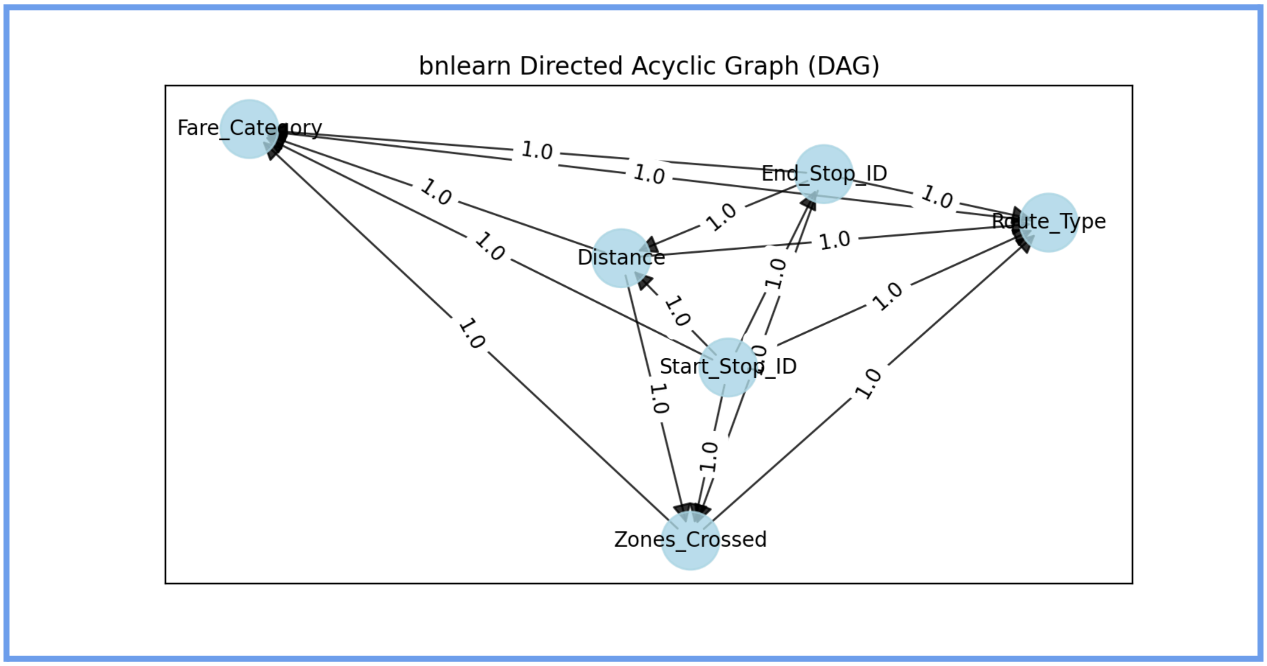
c) Since B is more likely given the evidence $P(B|E)$ increases.

And $P(A|E) + P(B|E) = 1$, $P(A|M,E) \propto P(A|E)$
$\therefore P(A|E)$ decreases

$\therefore$ Conditioning on the prevalence of Backdoor attacks lowers likelihood of adversarial perturbations being cause of the alarm.

Spiral

Q4) For the initial Bayesian Network, this was the DAG of the Network:



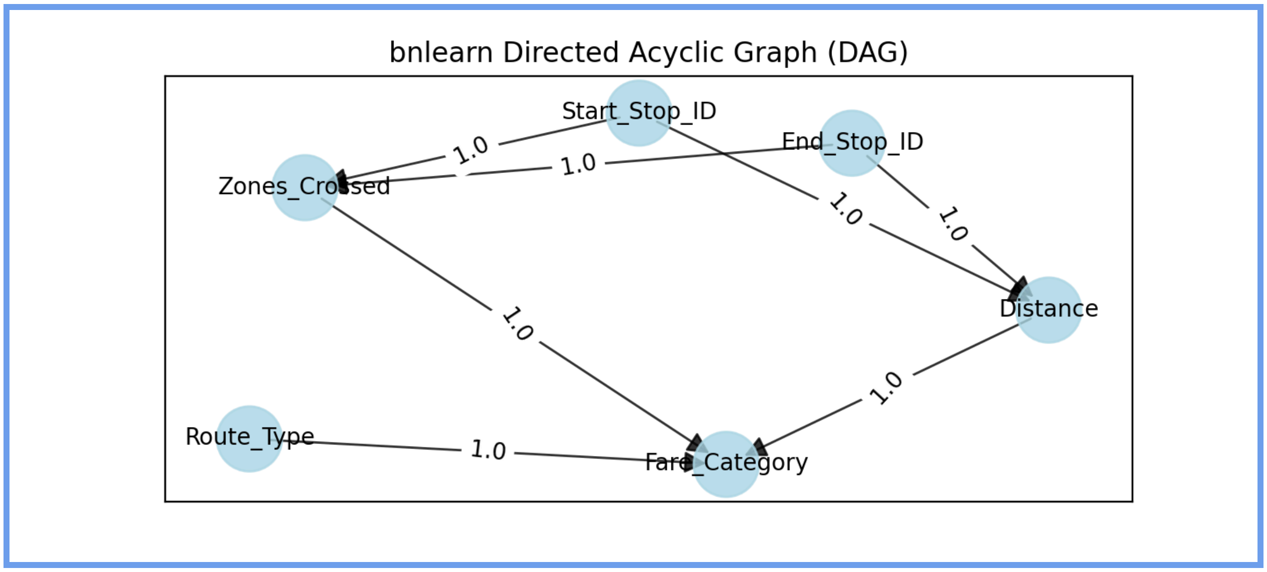
Accuracy obtained was 100%.

For the pruned Network, I used these edges:

```
edges = [
    ("Start_Stop_ID", "Zones_Crossed"),
    ("Start_Stop_ID", "Distance"),
    ("End_Stop_ID", "Zones_Crossed"),
    ("End_Stop_ID", "Distance"),
    ("Distance", "Fare_Category"),
    ("Zones_Crossed", "Fare_Category"),
    ("Route_Type", "Fare_Category"),
]
```

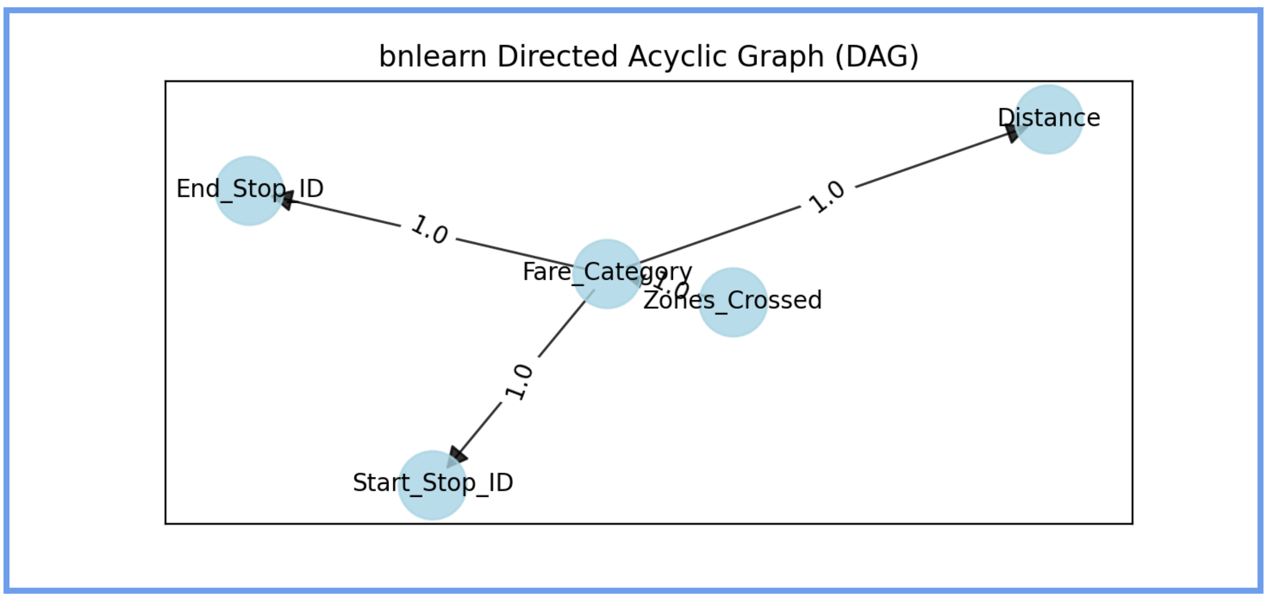
Start and End Stop IDs determine the number of zones for the trip and the total Distance Travelled as well. Distance, Zones Crossed and Route Type were all important for Fare Category to be ’Low’, ‘Medium’, or ‘High’. One thing also noticeable was that Route Type had only one value for all entries in the dataset. So even though semantically it may have significance, in this case we could have pruned the “Route_Type’ Node

This was the DAG of the pruned network:



Accuracy obtained was 100%

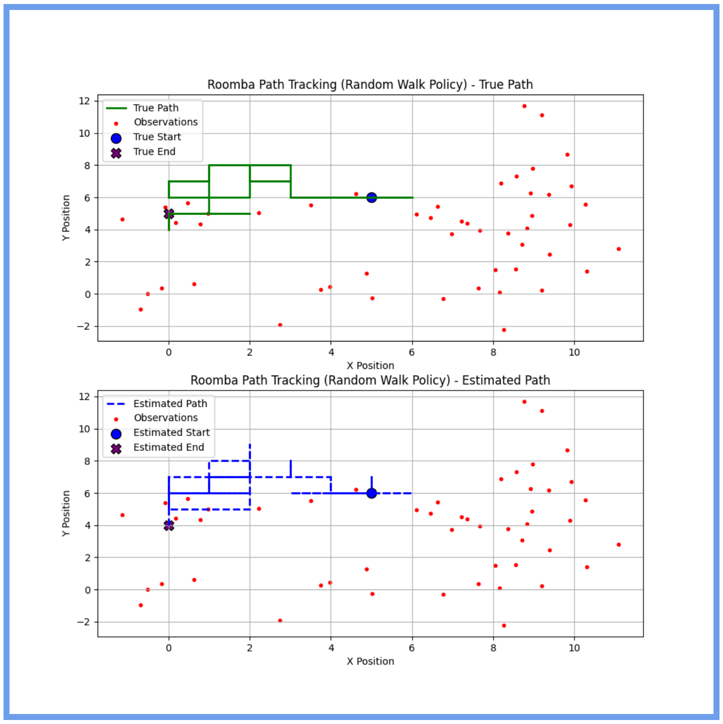
For the Optimized Bayesian Network, I used ‘Hill Climbing’ algorithm as the optimization technique and obtained this DAG:



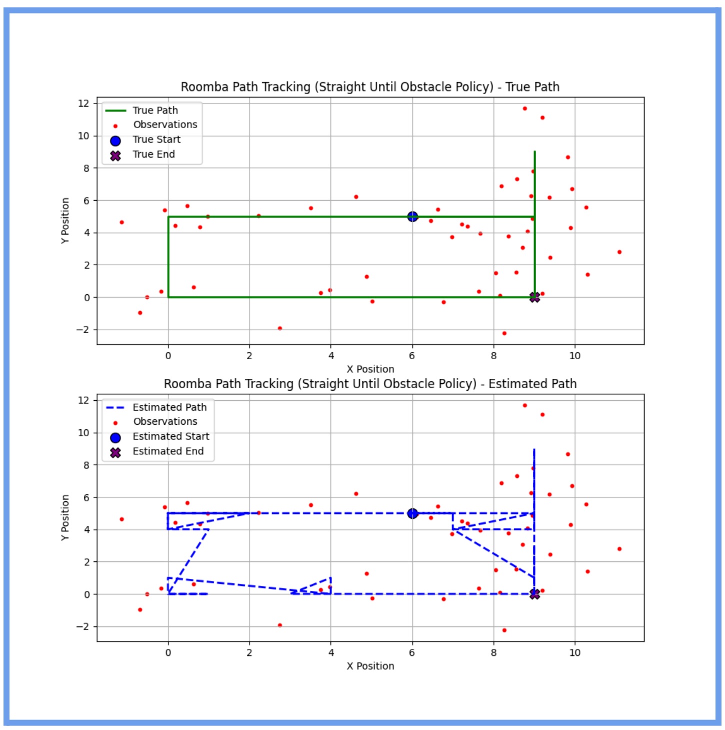
The Optimized and Pruned Bayesian Network was much faster than the original Network, since the original Network had complete dependencies on each other, resulting in the number of parameters to be considered being exponentially more than the other two.

Q5) Seed values chosen for this question were 111, 222, and 333.

For **seed 111**, this was the observation:



Roomba Path Tracking (Random Walk Policy) - True Path
True Path
Observations
True Start
True End
Y Position
X Position
Roomba Path Tracking (Random Walk Policy) - Estimated Path
Estimated Path
Observations
Estimated Start
Estimated End
Y Position
X Position



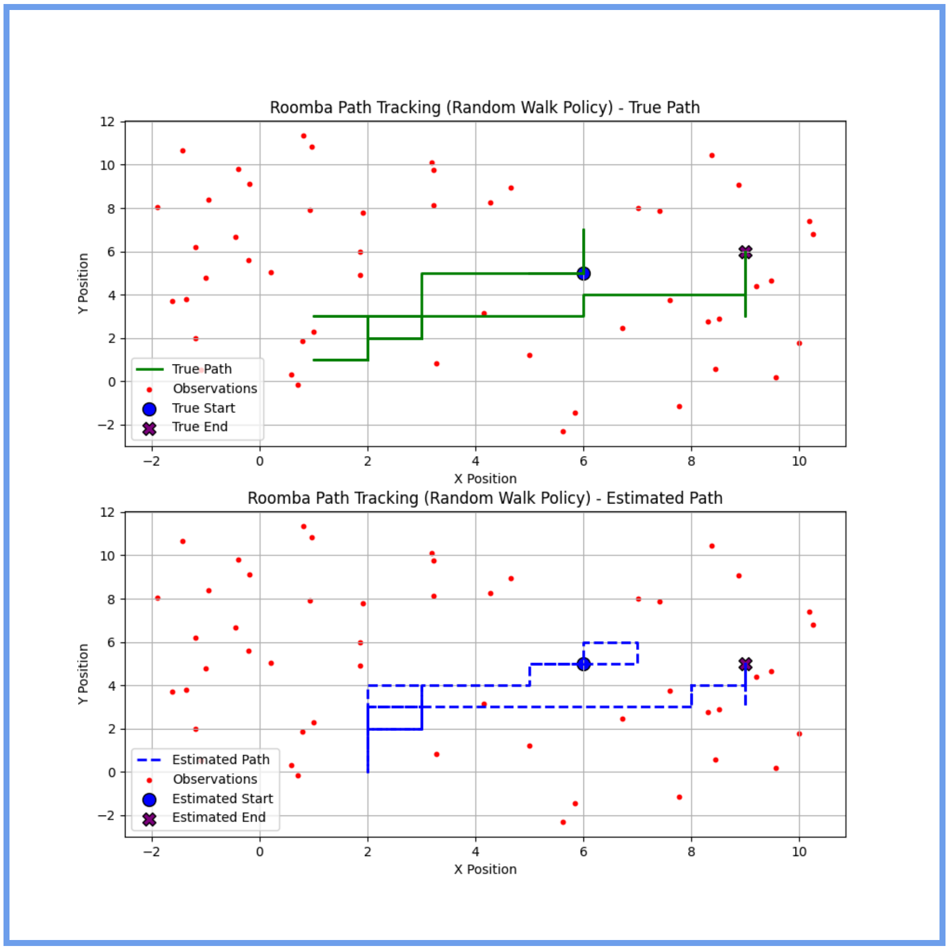
Roomba Path Tracking (Straight Until Obstacle Policy) - True Path
True Path
Observations
True Start
True End
Y Position
X Position
Roomba Path Tracking (Straight Until Obstacle Policy) - Estimated Path
Estimated Path
Observations
Estimated Start
Estimated End
Y Position
X Position

```
manojk@Manojs-MacBook-Air-New HMM_Question % python3 HMM_Question.py
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Movement: 100%|
Simulating Roomba movement for policy: straight_until_obstacle
Simulating Movement: 100%|

Processing policy: random_walk
Tracking accuracy for random walk policy: 34.00%
2024-11-24 17:46:33.354 Python[77049:14072945] ApplePersistenceIgnoreSt
12bl4qxv3qlbd9j440000gn/T/org.python.python.savedState

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle policy: 56.00%
```

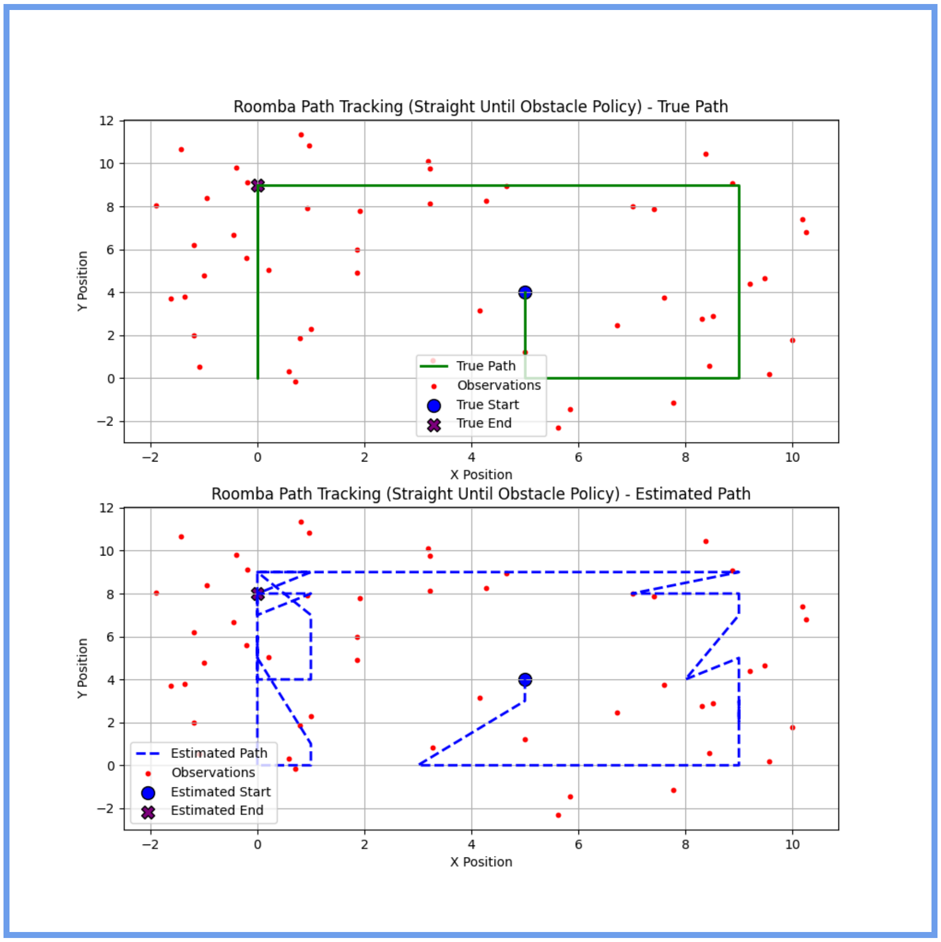
For seed 222, here’s the observation:




```
manojk@Manojs-MacBook-Air-New HMM_Question % python3 HMM_Question.py
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Movement: 100%|
Simulating Roomba movement for policy: straight_until_obstacle
Simulating Movement: 100%|

Processing policy: random_walk
Tracking accuracy for random walk policy: 64.00%
2024-11-24 18:05:02.623 Python[77389:14086920] ApplePersistenceIgnoreState: Ex
12bl4qxv3qlbd9j440000gn/T/org.python.python.savedState

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle policy: 42.00%
```

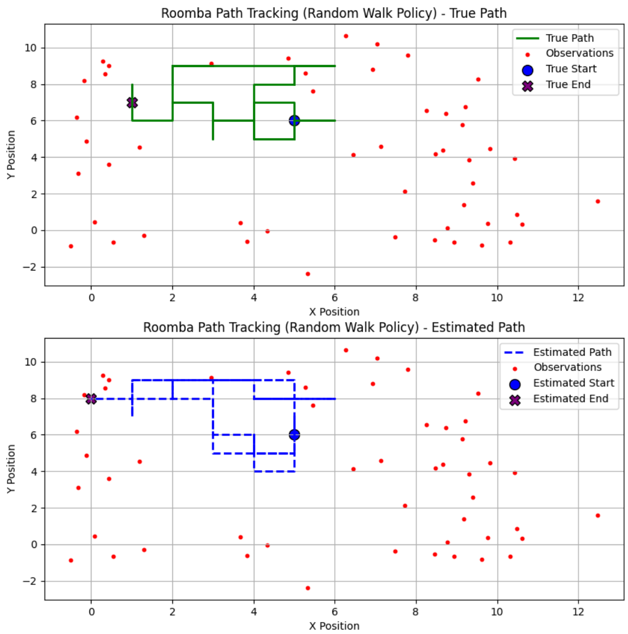


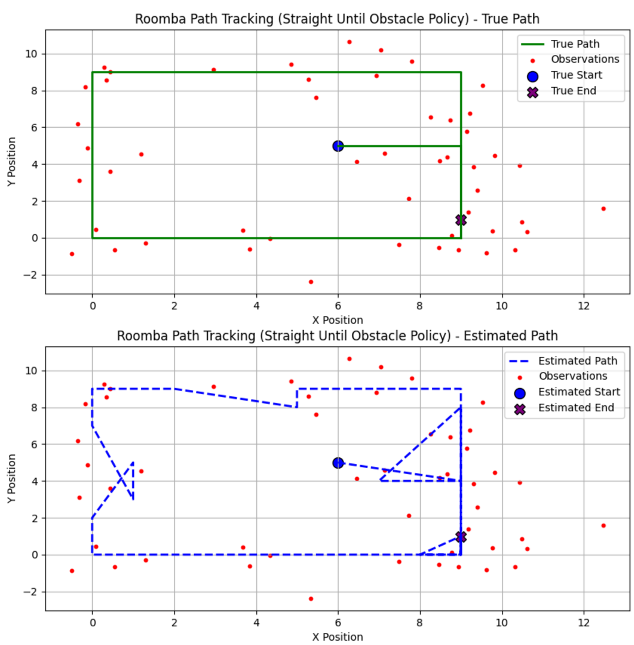

```
manojk@Manojs-MacBook-Air-New HMM_Question % python3 HMM_Question.py
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Movement: 100%|
Simulating Roomba movement for policy: straight_until_obstacle
Simulating Movement: 100%|

Processing policy: random_walk
Tracking accuracy for random walk policy: 64.00%
2024-11-24 18:05:02.623 Python[77389:14086920] ApplePersistenceIgnoreState: Ex
12bl4qxv3qlbd9j440000gn/T/org.python.python.savedState

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle policy: 42.00%
```

For **seed 333**, I observed this:


Roomba Path Tracking (Random Walk Policy) - True Path
True Path
Observations
True Start
True End
Y Position
X Position
Roomba Path Tracking (Random Walk Policy) - Estimated Path
Estimated Path
Observations
Estimated Start
Estimated End
Y Position
X Position


Roomba Path Tracking (Straight Until Obstacle Policy) - True Path
True Path
Observations
True Start
True End
Y Position
X Position
Roomba Path Tracking (Straight Until Obstacle Policy) - Estimated Path
Estimated Path
Observations
Estimated Start
Estimated End
Y Position
X Position

```
manojk@Manojs-MacBook-Air-New HMM_Question % python3 HMM_Question.py
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Movement: 100%|
Simulating Roomba movement for policy: straight_until_obstacle
Simulating Movement: 100%|

Processing policy: random_walk
Tracking accuracy for random walk policy: 58.00%
2024-11-24 18:06:56.541 Python[77500:14090060] ApplePersistenceIgnoreState: Exis
12bl4qxv3qlbd9j440000gn/T/org.python.python.savedState

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle_policy: 58.00%
```

On an average, the ‘straight_until_obstacle’ algorithm gives much better results than ‘random_walk’ algorithm.