Bayes Filter
RBE 595: Advanced Robot Navigation

Miheer Diwan
Robotics Engineering
Worcester Polytechnic Institute
msdiwan@wpi.edu

***Abstract*—This report presents an implementation of the Bayes filter algorithm for monitoring the state of a door using a robotic platform equipped with sensors and a manipulator arm. The sensor and propagation models are analyzed, and the Bayes filter algorithm is implemented in Python. The implementation is then used to answer specific questions regarding the behavior of the robot in different scenarios.**

I. Introduction

In this report, we discuss the implementation of the Bayes filter algorithm for monitoring the state of a door using a robotic platform. The robot is equipped with sensors and a manipulator arm, allowing it to detect the state of the door and take actions to manipulate it if necessary.

II. Sensor and Propagation Models

A. Sensor Model Analysis

The sensor model provides the probabilities of sensor measurements given the actual state of the door. Specifically:

$$\begin{aligned} p(z_t = 1 | x_t = 1) &= 0.6 \\ p(z_t = 0 | x_t = 1) &= 0.4 \\ p(z_t = 1 | x_t = 0) &= 0.2 \\ p(z_t = 0 | x_t = 0) &= 0.8 \end{aligned}$$

This indicates that the sensor is more reliable at detecting the closed state of the door compared to the open state. Additionally, the sensor has a higher chance of false negatives compared to false positives.

B. Propagation Model

The propagation model represents the dynamics of the system, indicating the probability of the current state given the previous state and the control input. Since the control input only affects the state when it is a push, the propagation model can be represented as follows:

$$p(x_t | u_t, x_{t-1}) = \begin{cases} 1 & \text{if } u_t = 0 \text{ and } x_t = x_{t-1} \\ 0.8 & \text{if } u_t = 1 \text{ and } x_t = 1 \\ 1 & \text{if } u_t = 1 \text{ and } x_t = 0 \\ 0 & \text{otherwise} \end{cases}$$

This indicates that the state remains unchanged when the robot does nothing. If the robot pushes the door, the door remains open if it was already open, and opens with a probability of 0.8 if it was closed.

$$\begin{aligned} \text{bel}(x_1) &= \int p(x_1 | u_1, x_0) \text{bel}(x_0)\, dx_0 \\ &= \sum_{x_0} p(x_1 | u_1, x_0) \text{bel}(x_0) \end{aligned}$$

$$\begin{aligned} \text{Task 1} = {} & p(x_1 | U_1 = \text{do nothing}; X_0 = \text{is open}) \text{bel}(X_0 = \text{is open}) \\ & + p(x_1 | U_1 = \text{do nothing}; X_0 = \text{is closed}) \text{bel}(X_0 = \text{is closed}) \end{aligned}$$

$$\begin{aligned} \text{Task 2} = {} & p(x_1 | U_1 = \text{push}; X_0 = \text{is open}) \text{bel}(X_0 = \text{is open}) \\ & + p(x_1 | U_1 = \text{push}; X_0 = \text{is closed}) \text{bel}(X_0 = \text{is closed}) \end{aligned}$$

III. Implementation of Bayes Filter Algorithm

The Bayes filter algorithm is implemented in Python, with separate functions for the prediction and measurement update steps. The implementation takes sequences of measurements and control inputs as input, along with an optional initial probability that the door is open. The initital probabilities were assumed to be 50 %.

Algorithm 1 Bayes Filter Algorithm

```
1: Input: Previous belief state $bel(x_{t-1})$, control input $u_t$, measurement $z_t$
2: for each possible state $x_t$ do
3:     Compute predicted belief state $\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) \cdot bel(x_{t-1})\, dx_{t-1}$
4:     Update belief state $bel(x_t) = \eta \cdot p(z_t|x_t) \cdot \overline{bel}(x_t)$
5: end for
6: Output: Updated belief state $bel(x_t)$
```

A **probability threshold parameter** of 0.9999 was also introduced to check the certainty.
A parameter called **'max_iterations'** was also introduced so that the algorithm does end up in an infinite loop in case of errors.

IV. Experimental Results

A. Question 1

If the robot always takes the action "do nothing" and always receives the measurement "door open" **— The number of iterations required are 9.**

B. Question 2

If the robot always takes the action “push” and always receives the measurement “door open” **— The number of iterations required are 4.**

C. Question 3

If the robot always takes the action “push” and always receives the measurement “door closed” **— The steady state belief about the door’s state after 10 iterations is that the door is open and has a probability of 100 % (after rounding off).**

V. Conclusion

In this report, we have presented an implementation of the Bayes filter algorithm for monitoring the state of a door using a robotic platform. We have analyzed the sensor and propagation models and provided experimental results to answer specific questions about the behavior of the robot in different scenarios. We also observed the limitations of the Bayes filter through this example. The primary limitations of Bayes filters lie in their computational complexity and reliance on discrete state assumptions. Firstly, Bayes filters can become computationally burdensome, particularly in systems with high-dimensional state spaces, as they require maintaining and updating probability distributions over all possible states. This computational overhead can hinder real-time applications and scalability. Secondly, the assumption of discrete states may not accurately capture the continuous nature of real-world systems, leading to discretization errors and potential loss of information. While Bayes filters offer a principled framework for state estimation, addressing these limitations remains crucial for their effective application in complex and dynamic environments.

References

[1] Thrun, S., Burgard, W., & Fox, D. (2005). Probabilistic robotics. MIT press.