Bayes Filter

Project #1

Introduction

Consider the example from Section 2.4.2 of *Probabilistic Robotics* of a robot that can detect, with uncertainty, whether a door is open or closed and can use its end effector to attempt to open the door. The relevant details from this section are summarized below.

Bayes_Filter($bel(x_{t-1}), u_t, z_t$**):**
For all x_t *do:*
$\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1})\, bel(x_{t-1}) dx_{t-1}$
$bel(x_t) = \eta\, p(z_t|x_t)\, \overline{bel}(x_t)$
Return $bel(x_t)$

Table 1: Bayes Filter Algorithm in pseudocode.

You have a robot that has a camera and a manipulator arm. The camera and sensor suite are configured to detect if the door is open or not. Similarly, the manipulator arm is a simple piston which may push the door to open it. The robot is being used to monitor the state of the door. Mathematically, the system state (x_t) corresponds to the door being open or closed, the measurement (z_t) is also the door being open or closed, and the input to the system (u_t) is either nothing or a push on the door. Since this is a binary state, we can simply map this to zero or one for the sake of mathematical representation.

Table 2: State mappings to binary.

x_t	Closed	0
	Open	1
z_t	Closed	0
	Open	1
u_t	Do nothing	0
	Push	1

However, the robot's camera is a low quality black and white camera with a poorly tuned sensor model. Thus, it cannot always accurately detect when the door is open. The robot has a true positive rate (sense open when the door is open) of 0.6, and a true negative rate (sense closed when the door is closed) of 0.8. Being binary events, the false positive and false negative rates must in turn sum to zero. The sensor model probabilities are summarized below.

$$p(z_t = 1\,|x_t = 1) = 0.6$$
$$p(z_t = 1\,|x_t = 0) = 0.4$$
$$p(z_t = 0\,|x_t = 1) = 0.2$$
$$p(z_t = 0\,|x_t = 0) = 0.8$$

Equation 1: Measurement model probabilities.

The robot can also use its manipulator to affect the state of the door. Again, this manipulator is not perfect and will have varying success given the actual state of the door. If the robot takes no action ($u_t = 0$) the door's state will remain the same. Similarly, if the door is already open ($x_t = 1$) and the robot pushes ($u_t = 1$) the door will remain open. Conversely, if the door is closed, the robot will open the door with a probability of 0.8.

Tasks

Task 1

Analyze the sensor (measurement) model described in Equation 1. What does this tell us about the usefulness or reliability of this sensor? Using the style of notation demonstrated in the lecture notes and Equation 1, write out the mathematical representation of the propagation model: $p(x_t|u_t, x_{t-1})$ for all possible prior states and inputs. What does this model indicate about the system dynamics?

Task 2

Implement the Bayes Filter algorithm in the programming language of your choice. Again, I highly recommend using Python. This implementation should take as input a sequence of measurements and a sequence of inputs, and optionally an initial probability that the door is open. I recommend breaking the problem down into two separate sub-methods: one to implement the prediction step (line 3) and one to implement the measurement update step (line 4). The necessity for breaking it up in this manner will become more apparent later in the course. Ultimately, your implementation should return the probability that the door is open (i.e. $bel(x_t = 1)$).

Please remember to incorporate good programming habits into your code. Strive for modularity and a clean separation of concerns in your code structure by breaking down complex tasks into smaller functions. Consider how errors may arise in your program and consider implementing appropriate error handling techniques (try-except blocks, assert statements, et cetera).

All code should be well-documented and include inline comments explaining the purpose of functions, methods, and complex logic. Document the input parameters, output values, and a brief description of the function's purpose for each function or method. Use descriptive variable and function names that convey their purpose or functionality. Comment sections of code that may be non-intuitive or involve complex algorithms.

Use your code to answer the following questions:

1. If the robot always takes the action "do nothing" and always receives the measurement "door open" how many iterations will it take before the robot is at least 99.99% certain the door is open?
2. If the robot always takes the action "push" and always receives the measurement "door open" how many iterations will it take before the robot is at least 99.99% certain the door is open?
3. If the robot always takes the action "push" and always receives the measurement "door closed" what is the steady state belief about the door? Include both the state and the certainty.

Grading Rubric

Task 1	**Excellent – 5 pts** The student has correctly answered the question and provided an explanation detailing how that answer was reached.	**Acceptable – 4 pts** The student has correctly answered the question.	**Marginal – 2 pts** The student has attempted to answer the question, but the answer is incorrect due to minor errors.	**Unacceptable – 0 pts** No answer was given, the response was lacking sufficient detail, or the answer was completely incorrect.
Task 2	**Excellent – 5 pts** The student has correctly answered the question and provided an explanation detailing how that answer was reached.	**Acceptable – 4 pts** The student has correctly answered the question.	**Marginal – 2 pts** The student has attempted to answer the question, but the answer is incorrect due to minor errors.	**Unacceptable – 0 pts** No answer was given, the response was lacking sufficient detail, or the answer was completely incorrect.
Code Quality	**Excellent – 5 pts** The student has intuitive, concise, well written code that follows a logical and organized structure. Comments are provided and document the functionality of the code.	**Acceptable – 4 pts** The student's code makes sense given the context of the problem. Some comments or documentation is provided.	**Marginal – 2 pts** The student's code is poorly structured and not intuitive. Little to no comments or documentation is provided.	**Unacceptable – 0 pts** The student's code fails to run, runs incorrectly, or otherwise fails to address the problem.