PRP Project

August 2021

Probability and Random Processes - Project

Topic: Applications of Probability and Random Processes in Robotics

Subject: Bayesian Filters and Belief distributions

Team 19:
Aakash Reddy Gorla (2020102034)
Chinmay Deshpande (2020102069)
Aditya Seghal (2020112013)
Aravapalli Dheeraj Murugan Sai (2020102020)

Abstract

Robotic systems are commonly seen these days, built to perform even the simplest of tasks. However, when encoding the system responses to every changing situation, one must find a way to ensure that there is no erratic behaviour, as much as they can. So every response must have some probability associated with it. This is but one basic example. There are multiple instances where probability and the notion of random variables are involved in the field of robotics. This project delves into the applications of probability and random processes in various different topics in the field of Robotics. Specifically, we will be looking at Bayesian filters and will attempt to make and interpret a working model as code.

Introduction

The major goal of any robotic system is to develop robust software that enables robots to withstand the numerous challenges arising in unstructured and dynamic environment. This lead us to focus on one of the key area in robotics: Uncertainty. Hence, uncertainty arises if the robot lacks critical information for carrying out its task.

"A robot that carries a notion of its own uncertainty and that acts accordingly, will do better than one that does not" - Sebastian Thrun

Factors that cause uncertainty:

- **Environment:** Uncertainty is common in physical environment. Dynamic environments tend to have more uncertainty than static environments, due to the fact that something is always changing. Moreover, the less well-defined an environment is, the more uncertainty will be associated with it.
- **Sensors:** A sensor's uncertainty depends on two primary factors. First, range and resolution of a sensor are bound by physical laws as a certain sensor can only percieve a certain type of data with a finite resolution. Second, sensors are subject to noise, which perturbs sensor measurements in unpredictable ways. Hence, corrupting the required data.
- **Actuators:** Actuators and motors are subject to movement inaccuracies and resolution. These can be affected by control noise, damage and the amount of precision with which the robot can inherently operate, determined during manufacture.
- **Models:** Models are abstractions of the real world. Abstractions tend to leave out certain minor details. As such, it follows that this model will describe the objects, conditions and ongoing processes in the environment with some degree of uncertainty. Although this facet of uncertainty is overlooked, it does affect the output.
- **Computation:** Robots processes real-time data which limits the amount of computation that can be carried out. Even in the most cutting-edge robotic systems, there is always going to be a trade-off between accuracy and computational efficiency, due to the large amounts of dynamically changing data that they may have to process at any given point of time.

Hence, to overcome such uncertainity, probability and probabilistic algorithms are used for robotics. Probabilistic robotics is an approach to robotics that deals in the uncertainty in robot perception and action. In this approach, probabilistic algorithms take in a multitude of possibilities, and by operating upon the individual probabilities of all of them, returns a single output that represents the outcome or case having the highest probability. It is quite similar to finding the line of best fit when approximating a graph from a set of points, except this time, we have a 'Best Fit Case', from a set of possibilities.

In this project, we have looked at the topic of the Bayesian Filter and the Belief Distribution, which are used extensively in robot sensors and and help determine system movement patterns.

Background

Basic Concepts in Probability:

$\sum_x p(X = x) = 1$

$p(x) = \left(2\pi\sigma^2\right)^{-1/2} exp\left\{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right\}$(Normal variables have a major role in probabilistic robotics and are denoted by $(x; \mu, \sigma^2)$)

$p(x|y) = \frac{p(x,y)}{p(y)}$

$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\sum_{x'} p(y|x')p(x')}$(Bayes rule: discrete)

$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\int p(y|x')p(x')dx'}$(Bayes rule: continuous)

$p(x|y) = \eta * p(y|x)p(x)$ (where η is $p(y)^{-1}$and is called the normalizer variable)

The above equations in effect help calculating $p(x|y)$. Their advantage lies in the fact that using ηsimplifies many formulas which would be very long otherwise. Using ηindicates that the final result must be normalized to zero.

$p(x, y|z) = p(x|z)p(y|z)$

(If x, y, zrepresent independent events)

$p(x|z) = p(x|z, y)$

Markov Assumption:

The Markov assumption/complete state assumption states that given the current state $x_t$all future and past states are independent. The following conditions will induce violations of the Markov assumption:

- Innacuracies in $p(z_t|x_t)$and $p(x_t|u_t, x_{t-1})$
- approximation errors in the calculation of belief function

We can include these many such variables in the state representation but doing so increases the computational complexity of the filter greatly. Practically Bayesian filters are resistant to any violations of the Markov assumption.

Important Terms:

1 State:

The collection of measurements of all factors in the environment and the robot is called the state. State can be static(like walls) or dynamic(velocity, position,etc). State variables are generally denoted by $x$and at time t :x_t. Some of the more commonly used state variables are :

- Robot position: Contains 6 variables(3 Cartesian and 3 angular) to determine its location and orientation on a global frame
- Configuration of robot actuators
- Velocity(both the robot and its joints)
- Environmental Features(static)(walls,trees,etc)
- Environmental Features(dynamic)(people,cars,etc)

A state $x_t$is said to be complete when it can accurately predict the future. This however does not mean that the future state variable is deterministic of the past , as $x_t$can be independent of the past. Processes that meet these conditions are known as markov chains. In reality complete state variables are difficult to obtain as many of the measurement required(memory, brain dumps,etc) are hard to obtain. State variables can be both continuous and discrete. Discrete time is generally taken in examples.

2 Interactions:

Interactions can ocur both in the form of influence on the environment through actuators and measurement through sensors.

- Sensor Measurements: Perception is the process in which the robot obtains information about the environment through sensors. Generally we would call this continuous process a *measurement* but it can be called an *observation* or *percept.* Also sensors will have some sort of delay.
- Control Actions: These are the robots ouputs and change the state of the world. Motion and object manipulation are considered as actions. For consistency's sake we consider that the robot is always executing a control action.

3 Measurement vs Control:

Both measurement and control are fundamentally different,which is essential later and hence understanding these divergences are imperative.To increase the amount of information available to the robot, data is delivered from the environment through perception. Generally control actions tend to decrease the data on hand due to the variable surroundings and noise ,but in certain situations can identify a situation. This can be shown by an example of when a robot attempts to open a door, without being sure of its nature. If the door does not open when pushed then it can be identified as locked and the opposite is also true. We do not intend to imply that actions and perceptions are temporally spaced but actually that both control and perception take place simultaneously and both cause a change in the amount of data available.

4 Probabilistic Generative Laws:

Probabilistic principles govern the evolution of the state and its measurements. The state at time $x_t$is created stochastically in most cases. As a result, specifying the probability distribution from which $x_t$is derived is a good idea. At first appearance, the emergence of the emergence of the emergence of the emergence of the emergence. The evolution of state variables can be displayed with the probability distribution:

$p(x_t|x_{0:t-1}, z_{1:t-1}, u_{1:t})$

Methods

Belief distributions and Bayesian filters

A belief distribution determines a robot's understanding of the environment in which it is in. Every action done carried out will depend in some way on past statistics, be it previous measurements, probabilities of previous events or any environmental factors. As a result, a belief distribution is typically a conditional probability distribution, conditioned over the past statistics. The belief distribution, denoted by $bel(x_t)$, is given as:

$bel(x_t) = p(x_t|z_{1:t}, u_{1:t})$

where $x_t$is the data variable, $z_{1:t}$and $u_{1:t}$are respectively the past measurements and past controls (controls are directives given to the robot to execute a task with a fixed preset measurement associated with it, such as go 5 steps forward) taken at discrete time intervals from 1 to t, hence they are given with the index $1 : t$. In other words, the robot is, in a way asking itself this question - "If I am currently at time t, and I have all measurements and controls taken at integer time from $t = 1$until now, then based on this, what do I do next?". This question can be partly an-

swered using the belief distribution of each state variable. At the same time, we can use another distribution $\overline{bel}(x_t)$which represents the probability conditioned over all controls from $1 : t$and all measurements from $1 : (t-1)$. This is basically a probability distribution made assuming that the very last measurement has not been taken. Therefore,

$\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t})$

Why is such a distribution important? $\overline{bel}(x_t)$enables us to predict $bel(x_t)$, and this process is called correction or measurement update.

The Bayesian Filter is an algorithm used to calculate a belief distribution from the given data. This algorithm consists of two main steps:

1. Processing the control u_t. The belief $\overline{bel}(x_t)$that the robot assigns to state $x_t$is obtained by the sum or integral of the product of two distributions: the probability assigned to x_{t-1}, and the probability that controls u_t. One can think of this using the idea that these two events can be chained together. $x_t$is obtained when $x_{t-1}$is processed AND the event $u_t$is executed. So we can get the required probability by multiplying the previous belief distribution with $u_t$and summing over all possible types of instructions that $u_t$can be.

2. This step is called the measurement update. What we have computed so far in the first step is $\overline{bel}(x_t)$, but this isn't what we want yet. We want $bel(x_t)$, and this will require us to chain the previous two events in step 1 with one more event. By the definition, $\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t})$.

   We can see here that in order to obtain $bel(x_t)$, we will have to take into account the measurement z_t, which is made to reach the state x_t. This means that we have to chain the event associated with x_{t-1}, then the execution of the instruction $u_t$and then event involving the taking of the measurement z_t. So we again need to multiply and sum over all possible cases again. However, to obtain the final $bel(x_t)$, we will need to normalize the distribution using a normalization constant η.

Thus we have the final algorithm (Taken directly from source text for accuracy's sake):

Algorithm Bayes_Filter($bel(x_{t-1})$, u_t, z_t):

2: for all x_t,do:

3: $\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) \cdot dx$

4: $bel(x_t) = \eta p(z_t|x_t) \cdot \overline{bel}(x_t)$

5: for end

6: return bel(xt)

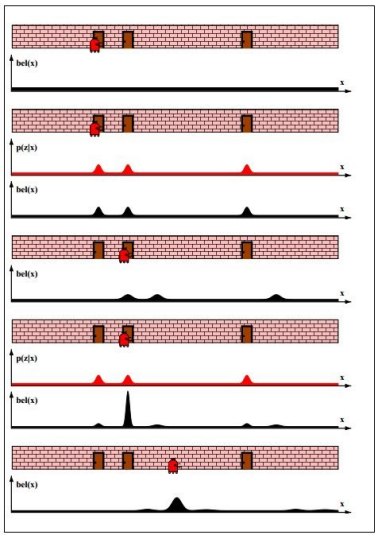

Figure 1: The above diagram is given for reference

Results and Discussion

Implementing and Interpreting the working model

We tried implementing the Bayes Filter in MATLAB, following the above stated algorithm, but with a few slight changes. In the given algorithm, only one sensor seems to have been considered. We tried implementing this method for any $n$sensors. This is applied on multiple sensors, and since the number of sensors had been specified using an interval with a step size, the output had to be scaled.

In a nutshell, this is a sensing algorithm using a Bayesian Filter. This enables us to calculate the belief distribution of only the very next next state for all n sensors taken.[Refer to Fig.2 on next page for results plot]

In our implementation we hardcoded the number of sensors to be 4 and their input to be normal random variables.This can be manipulated to suit the situation at hand. Each input indicates the possibility of an objective being present. The black line represents the belief distribution. The belief distribution represents the highest probability of where the robot believes the objective to be. The expectation of the belief distribution is 25. To explain this, we can use this scenario - if the sensor is trying to detect the location of a traffic cone on a one-way road from a particular starting point, then the expectation tells us that the most likely location of the cone, according to the belief distribution is at exactly 25m.

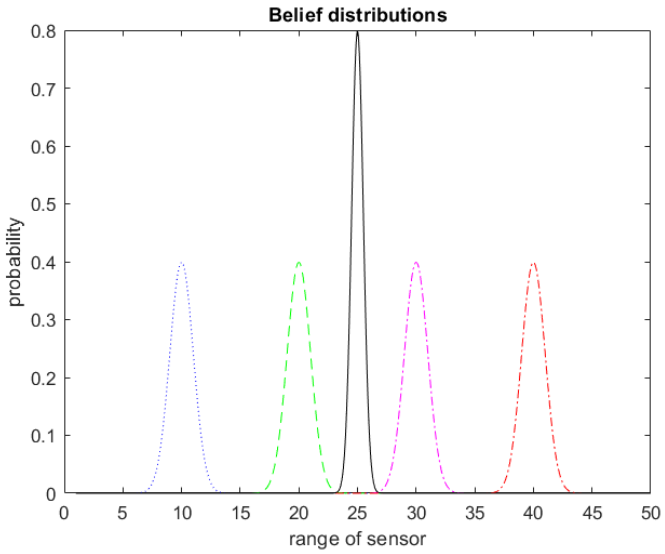

Figure 2: Results plot for the belief distribution

Conclusion:

To sum up, a massive chunk of the field of robotics depends upon the Bayesian filter. This filter helps facilitate the working of sensors in robotics, making use of conditional probability to determine the most likely location of any object in the immediate environment of the robotic system, which is especially useful for a moving robotic system. These calculations are done constantly, and determine every step the robot may take. There are also multiple other types of sensor filters such as Gaussian filters and Kalman filters, each of which make rely on probability quite heavily. Since all of these filters play an integral part in the functioning of robotic systems, we can clearly see that the importance of probability in this topic, as well as the field of robotics as a whole, is enormous.

Citations:

Thrun,Sebastian, et al. $Probabilistic\ Robotics$, MIT Press, 2005 (Source Text for Algorithm)

Thrun, Sebastian $Probabilistic\ Algorithms\ in\ Robotics$, 2000

https://github.com/AakashR13/PRP$_{p}roject_{1}9.git$

Contributions:

Aakash Reddy Gorla - Code, Background
Chinmay Deshpande - Belief Distribtuions and Bayesian Filters, Implementation and Interpretation
Aditya Sehgal - Abstract, Introduction