AA274A: Principles of Robot Autonomy I
Course Notes

Oct 29, 2019

14 Localization and Filtering

Navigation is one of the most challenging competences required of a mobile robot. Success in navigation requires success at the four building blocks of navigation: *perception* (the robot must interpret its sensors to extract meaningful data); *localization* (the robot must determine its position in the environment); *cognition* (the robot must decide how to act to achieve its goals); and *motion control* (the robot must modulate its motor outputs to achieve the desired trajectory).

Of these four components, localization has received the greatest research attention in the past decade, and as a result, significant advances have been made on this front. In this lecture, we explore the successful localization methodologies of recent years. First, we discuss motivation based map-based localization. (Direct excerpt from [SNS11].) After brushing up on basic probability concepts, we will introduce Bayes Filter algorithm that lays the foundation of map-based localization.

14.1 Motivation

Localization is a strategy to solve a navigation problem. In this section, we discuss why map-based localization is extremely popular in mobile robots.

Figure 1 depicts a standard indoor environment that a mobile robot navigates. Suppose that the mobile robot in question must deliver messages between two specific rooms in this environment: rooms A and B. In creating a navigation system, it is clear that the mobile robot will need sensors and a motion control system. Sensors are absolutely required to avoid hitting moving obstacles such as humans, and some motion control system is required so that the robot can deliberately move.

How do we solve the problem? One might suggest designing sets of behaviors that together result in the desired robot motion. Fundamentally, this approach avoids explicit reasoning about localization and position, and thus generally avoids explicit path planning as well. This technique is based on a belief that there exists a procedural solution to the particular navigation problem at hand. For example, in figure 1, the behavioralist approach

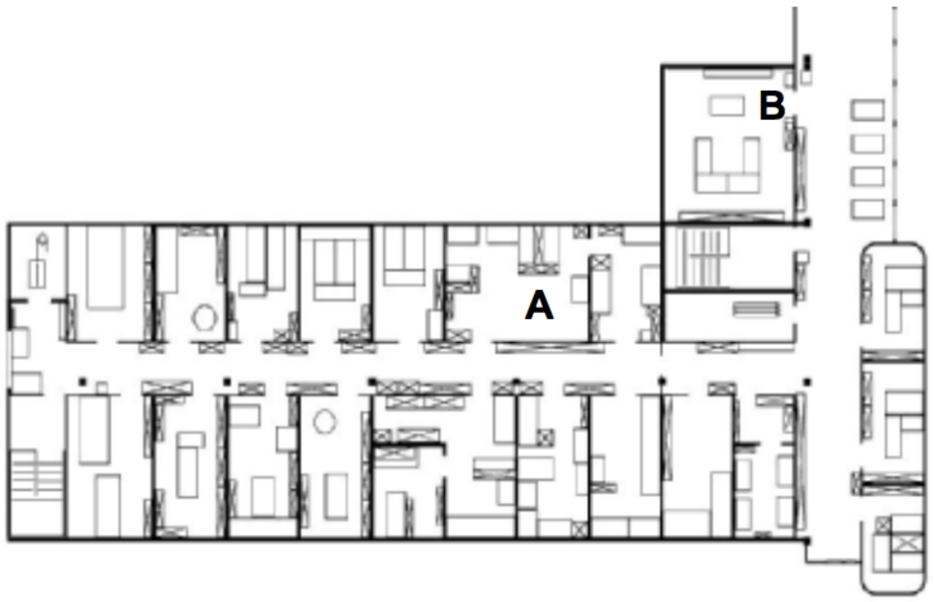

Figure 1: A sample environment with initial position A and the goal position B.[SNS11]

to navigating from room A to room B might be to design a left-wall following behavior and a detector for room B that is triggered by some unique queue in room B, such as the color of the carpet. Then the robot can reach room B by engaging the left-wall follower with the room B detector as the termination condition for the program.

The key advantage of this method is that, when possible, it may be implemented very quickly for a single environment with a small number of goal positions. It suffers from some disadvantages, however. First, the method does not directly scale to other environments or to larger environments. Often, the navigation code is location-specific, and the same degree of coding and debugging is required to move the robot to a new environment. Second, the underlying procedures, such as *left-wall-follow*, must be carefully designed to produce the desired behavior. This task may be time-consuming and is heavily dependent on the specific robot hardware and environmental characteristics. Third, a behavior-based system may have multiple active behaviors at any one time. Even when individual behaviors are tuned to optimize performance, this fusion and rapid switching between multiple behaviors can negate that fine-tuning. Often, the addition of each new incremental behavior forces the robot designer to retune all of the existing behaviors again to ensure that the new interactions with the freshly introduced behavior are all stable.

Map-based approach is a popular alternative that counters such disadvantages. In contrast to the behavior-based approach, the map-based approach includes both localization and cognition modules. In map-based navigation, the robot explicitly attempts to localize by collecting sensor data, then updating some belief about its position with respect to a map of the environment. The key advantages of the map-based approach for navigation are as follows:

- The explicit, map-based concept of position makes the system's belief about position transparently available to the human operators.
- The existence of the map itself represents a medium for communication between human and robot: the human can simply give the robot a new map if the robot goes to a new environment.
- The map, if created by the robot, can be used by humans as well, achieving two uses.

The map-based approach will require more up-front development effort to create a navigating mobile robot. The hope is that the development effort results in an architecture that can successfully map and navigate a variety of environments, thereby amortizing the upfront design cost over time.

Of course the key risk of the map-based approach is that an internal representation, rather than the real world itself, is being constructed and trusted by the robot. If that model diverges from reality (i.e., if the map is wrong), then the robot's behavior may be undesirable, even if the raw sensor values of the robot are only transiently incorrect. Therefore, the key challenges to map-based navigation is (1) to represent the map and (2) to model the belief regarding the position within the map.

How do we represent robot's belief of position? Most common approach to model robot's belief is to use probabilistic model. Localization techniques that use probabilistic model to represent robot's belief on its position is called *probabilistic map-based localization*.

14.2 Probabilistic Map-based Localization

The fundamental issue that differentiates various map-based localization systems is the issue of *representation*. There are two specific concepts that the robot must represent, and each has its own unique possible solutions. The robot must have a representation (a model) of the environment, or a map. What aspects of the environment are contained in this map? At what level of fidelity does the map represent the environment? These are the design questions for map representation.

The robot must also have a representation of its belief regarding its position on the map. Does the robot identify a single unique position as its current position, or does it describe its position in terms of a set of possible positions? If multiple possible positions are expressed in a single belief, how are those multiple positions ranked, if at all? These are the design questions for *belief representation*. [SNS11]

Decisions along these two design axes can result in varying levels of architectural complexity, computational complexity, and overall localization accuracy. The first major branch in a taxonomy of belief representation systems differentiates between *single-hypothesis* and *multiple-hypothesis* belief systems. The former covers solutions in which the robot postulates its unique position, whereas the latter enables a mobile robot to describe the degree to which it is uncertain about its position. A sampling of different belief and map representations is shown in figure 2.

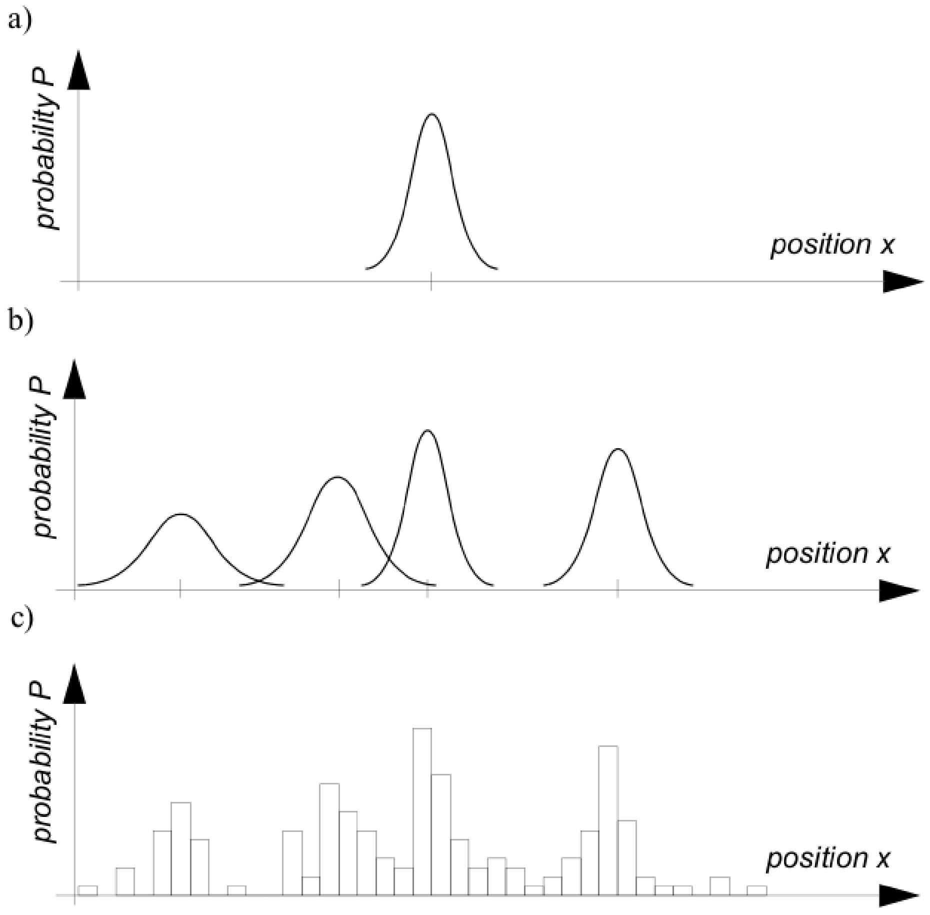

Figure 2: Belief representation regarding the robot position (1D) in continuous and discretized (tessellated) maps. (a) Continuous map with single-hypothesis belief, e.g., single Gaussian centered at a single continuous value. (b) Continuous map with multiple-hypothesis belief, e.g;. multiple Gaussians centered at multiple continuous values. (c) Discretized (decomposed) grid map with probability values for all possible robot positions, e.g., Markov approach. (d) Discretized topological map with probability value for all possible nodes (topological robot positions), e.g., Markov approach. [SNS11]

14.3 Basic concepts in probability

Probabilistic localization algorithms are variants of the Bayes filter. The straightforward application of Bayes filters to the localization problem is called Markov localization.[TBF05] Before we dive in any further, we will review basic probability.

14.3.1 Random Variables

Quantities such as sensor measurements, states of a robot and its environment are modeled as *random variables*. A random variable is a variable whose possible values are outcomes of a random phenomenon. In the case of sensor measurements, the source of randomness could be sensor noise. Random variables fall under one of two categories: discrete or continuous.

Discrete Random Variable. Discrete random variables can take on only a countable number of values. For example, if we flip a coin twice and the random variable X represents the number of coin flips resulting in "heads", X can only take on values of 0, 1 or 2 (and similarly for the case of "tails"). A discrete random variable is characterized by a probability mass function (PMF) $p(X = x)$ (or $p(x)$) where

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

Continuous Random Variable. A continuous random variable can take on an infinite number of values. A continuous random variable is characterized by a probability density function (PDF), $p(x)$, where the probability of a random variable being contained within the interval $[a, b]$ is

$$P(a \leq X \leq b) = \int_a^b p(x)dx$$

where

$$\int_{-\infty}^{+\infty} p(x)dx = 1.$$

Joint Distributions. An event can depend on more than one unknowns. We can parameterize our uncertainty over multiple variables using *joint distribution.* Joint probablility of two random variables X and Y is denoted as

$$P(x, y) := P(X = x \text{ and } Y = y).$$

Independence. Two events are called independent if and only if

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

or equivalently,

$$P(x|y) = P(x) \tag{1}$$

or

$$P(y|x) = P(y). \tag{2}$$

Therefore, independence is equivalent to saying that observing X does not have any effect on the probability of Y.

14.3.2 Conditional Probability

Conditional probability is the probability of an event happening given that another event has occurred. Suppose that we know that the event $Y = y$ occurs with probability $p(y) > 0$. The probability of the event $X = x$ occurring given that $Y = y$ has occurred, or alternatively stated as the conditional probability of X given Y, is given by

$$P(x|y) := \frac{P(x,y)}{P(y)}.$$

In order to better understand the concept of conditional probability, consider Fig. 3. In this figure, the sample space Ω contains the set of all possible outcomes for a random variable. Consider the case where there are two possible outcomes, B_1 and B_2 which may occur with an equal probability of 0.5. These events correspond to the orange and blue regions of Ω, respectively. An event A that occurs with probability 0.5 can also be defined, which corresponds to the centered shaded region in Fig. 3.

Suppose that we wish to determine the probability of A occurring and that a sample from Ω is observed to result in the event B_1. Given that B_1 has occurred, we restrict our attention to the region B_1. In this restricted region, the area corresponding to A occurring is the intersection of A and $B1$. Since region A overlaps with half of the region B_1, we have the result that $P(A|B_1) = 0.5$. A slightly different but equivalent interpretation of this figure, as given during lecture, is that we rescale the probability of event A occurring by the probability of the event that has happened. Mathematically, this is expressed as

$$P(A|B_1) = \frac{P(A \cap B1)}{P(B_1)} = \frac{0.25}{0.5} = 0.5.$$

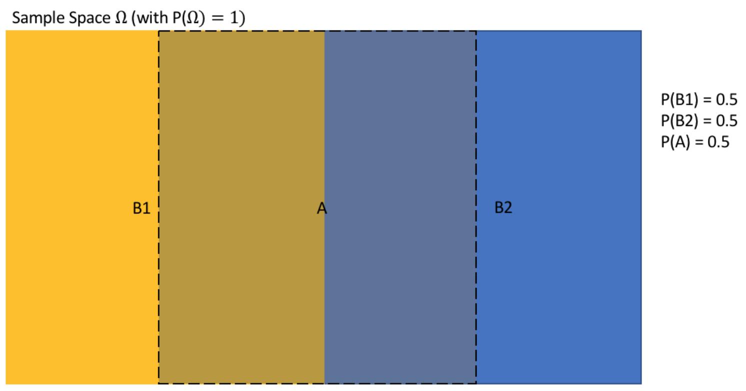

Figure 3: Sample space representation. Note that region A is centered an overlaps B_1 and B_2 equally.

14.3.3 Conditional Independence

If X, Y and Z are random variables such that,

$$P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z)$$

then X and Y are said to be *conditionally independent* given Z.

Note, however, that independence of two random variables does not imply conditional independence, and the converse is generally not true. In other words,

$$P(X, Y|Z) = P(X|Z)P(Y|Z)] not \Rightarrow P(X, Y) = P(X)P(Y), \tag{3}$$

and

$$P(X, Y) = P(X)P(Y) \not\Rightarrow P(X, Y|Z) = P(X|Z)P(Y|Z) \tag{4}$$

14.3.4 Law of Total Probability

Once again consider Fig. 3. If we wished to find the probability of A without any assumptions regarding conditioning, we can do so by calculating

$$P(A) = P(A \cap B1) + P(A \cap B2).$$

This idea can be generalized for discrete random variables into what is known as the theorem (or law) of total probability as

$$P(X) = \sum_{y \in Y} P(X, Y) = \sum_{y \in Y} P(X|Y)P(Y) \tag{5}$$

where the definition of conditional probability was used to obtain the last expression. For continuous random variables, the summations simply turn into integrals.

14.3.5 Bayes Rule

To facilitate the explanation of Bayes' Rule please note: a sample space is a set containing all the possible outcomes of an experiment and an event is a subset of elements in a sample space. Imagine that there is an event A whose set of outcomes is contained in the sets of events B_i where $i = 0, 1, ..., n$ of a sample space. For example in Fig. 3 event A's set is contained in the sets of events B_1 and B_2. Then,

$$P(B_i|A) = \tfrac{P(B_i \cap A)}{P(A)}.$$

Using the formula for conditional probability $P(B_i \cap A)$ and $P(A)$ can be expressed as:

$$P(B_i \cap A) = P(A|B_i)P(B_i).$$
$$P(A) = \sum_{i=0}^{n} P(B_i \cap A) = \sum_{i=0}^{n} P(A|B_i)P(B_i).$$

Combining these two expressions to derive Bayes' Rule for discrete sample spaces:

$$P(B_i|A) = \frac{P(A|B_i)P(B_i)}{\sum\limits_{i=0}^{n} P(A|B_i)P(B_i)}.$$

For a continuous sample space Bayes' Rule is:

$$P(B|A) = \frac{P(A|B)P(B)}{\int P(A|B=b')P(B=b')db'}.$$

Note that $P(A)$ is independent of which event Bi is selected. So the denominator of Bayes' formula can be considered a constant for all events Bi with event A. Furthermore, by knowing the probabilities $P(A|Bi)$ and $P(Bi)$, we can determine $P(Bi|A)$. As an example, a brewery buys equal amounts of hops from two farmers, $B1$ and $B2$. The brewery has a strict policy of using only one type of hops in each batch of beer it brews. Furthermore, the brewery uses half of all its hops from both sources to produce Pilsner beer. Bayes' Rule can be applied to determine, given that a batch of Pilsner is produced, what is the probability that the batch was brewed using hops from farmer $B1$. Note this example uses the sample space described in Fig. 3

$$P(B_1|A) = \frac{P(A|B_1)P(B_1)}{\sum\limits_{i=0}^{1} P(A|B_i)P(B_i)} = \frac{P(A|B_1)P(B_1)}{P(A|B_1)P(B_1)+P(A|B_2)P(B_2)} = \frac{0.5 \cdot 0.5}{(0.5 \cdot 0.5 + 0.5 \cdot 0.5)} = \frac{0.25}{0.5} = 0.5.$$

Bayes' Rule can be extended with the use of additional random variables. In the three variable case, given $X = x$, $Y = y$ Bayes' Rule can be conditioned on an additional variable $Z = z$ as follows:

$$P(x|y,z) = \frac{P(y|x,z)P(x|z)}{P(y|z)}.$$

This is the probability that event $Z = z$, and event $X = x|Y = y$ occur, which describes $Z = z$ influence on joint event $X = x|Y = y$.

14.3.6 Expectation and Covariance

The expectation value of a random variable $E(X)$ is the average result of an experiment over an infinite number of trials. It is also known as the first moment of the distribution. For the discrete case:

$$E(X) = \sum_{i=0}^{n} x_i P(X = x_i).$$

For the continuous case:

$$E(X) = \int\limits_{-\infty}^{\infty} x' P(X = x')dx'.$$

The expectation value of a constant is a constant and its calculation is a linear operation.

$$E(aX + b) = aE(X) + b.$$

In the case of a vector of random variables. The expectation of the vector is the vector of each variables expectation value.

Covariance is a calculation used to measure the relationship between random variables. For two random variable vectors $X = x, Y = y$, the covariance is:

$$Cov(X,Y) = E[(X - E(X))(Y - E(Y))] = E(XY^T) - E(X)E(Y^T)$$

If the covariance is positive this implies that X tends to increase at the same time as Y. If the covariance is negative then this implies that X tends to decrease as Y increases.If changes in X and Y have a random relationship, then the covariance will be close to zero. If two variables have a covariance of zero they are called uncorrelated. If two variables are independent they will be uncorrelated. However, two variables which are uncorrelated are not always independent.

14.3.7 Markov model of robot states

Mobile robot localization problems are two-folds. First, a robot needs to model its environment based on the sensor measurements. At the second time, the robot actuates and changes the environment, which results in different sensor data. Given error is present both in sensors and actuators, we need to define the *state*, *measurement* and *control* to capture robot's interaction with the environment.

14.3.8 State

We define the *state* as the collection of all aspects of the robot and its environment that can impact the future. x_t denote the state at time t. To simplify our equations, we adapt notation $x_{t_1:t_n} := x_{t_1}, x_{t_2}, ..., x_{t_n}$.

Any quantities that captures the robot's current condition can be a state. Common state variables in context of localization are:

- Robot pose (e.g., robot location and orientation relative to a global coordinate frame)
- Robot velocity
- Location and features of surrounding objects in the environment

A state x_t is called *complete* if x_t is the best predictor of the future. Very often in robot application, we can impose *Markov Property* in robot states and measurements. Markov property of robot state means robot's current state is only affected by the previous state. In other words, $P(x_t|x_{0:t-1} = P(x_t|x_{t-1})$. Markov assumption is extremely useful and lays the foundation of filtering algorithms that we will discuss later.

14.3.9 Measurement and control data

Exteroceptive sensors such as cameras, lasers, or ultrasonic rangefinders allow the robot to perceive the environment. The measurement data at time t will be denoted z_t. If we assume that the robot takes exactly one measurement at each point in time, the sequence of measurements is given as:

$$z_{t_1:t_n} := z_{t_1}, z_{t_1+1}, ..., z_{t_n} \tag{6}$$

using the shorthand notation introduced above.

Control data carry information about the change of state in the environment. In mobile robotics, a typical example of control data is the velocity of a robot.Control data will be denoted u_t. The variable u_t will always correspond to the change of state in the time interval (t-1;t].We define notation: $u_{t_1:t_2} := u_{t_1}, u_{t_1+1}, ..., u_{t_2}$, denotes the sequences of control data from time t_1 to time t_2, for $t_1 \leq t_2$.

Environment perception provides information about the environment's state, hence it tends to increase the robot's knowledge. Motion, on the other hand, tends to induce a loss of knowledge due to the inherent noise in robot actuation and the stochasticity of robot environments.

14.3.10 State equation

The evolution of state and measurements is governed by probabilistic laws. the probabilistic law characterizing the evolution of state might be given by a probability distribution of the following form: $p(x_t \mid x_{1:t-1}, z_{1:t-1}, u_{1:t})$. Notice that through no particular motivation we assume here that the robot executes a control action u_1 first, and then takes a measurement z_1.

If the state x is complete then it is a sufficient summary of all that happened in previous time steps. In particular,x_{t-1} is a sufficient statistic of all previous controls and measurements up to this point in time, that is, $u_{1:t-1}$ and $z_{1:t-1}$. From all the variables in the expression above, only the control u_t matters if we know the state x_{t-1}. That is :

$$p(x_t \mid x_{1:t-1}, z_{1:t-1}, u_{1:t}) = p(x_t \mid x_{t-1}, u_t)$$

14.3.11 Measurement equation

The probability of getting a measurement z_t is also given by a probabilistic law. Again, if we assume that the state x is complete, then we can say that state x_t already encodes the effect of all previous states $x_{0:t-1}$, previous measurements, $z_{1:t-1}$, and previous controls $u_{1:t}$. This allows us to arrive at the equation:

$$p(z_t \mid x_{0:t}, z_{1:t-1}, u_{1:t}) = p(z_t \mid x_t)$$

14.3.12 Overall stochastic model

It is important to keep the order of occurrence of different events in the model consistent. In our notation, we assume that the control effort u_t is determined after taking into account the state x_{t-1} and measurement z_{t-1}. The control effort u_t causes the robot to transition to a new state x_t where a new measurement z_t is taken, which in turns allows the determination of a new control effort u_{t+1}. This network model is a stochastic model known as a Markov process because the new state x_t only depends on the previous state x_{t-1} and the previous control u_t. In addition, since we cannot directly observe the current state, but instead estimate it through measurements, this model is further classified as a hidden Markov model.

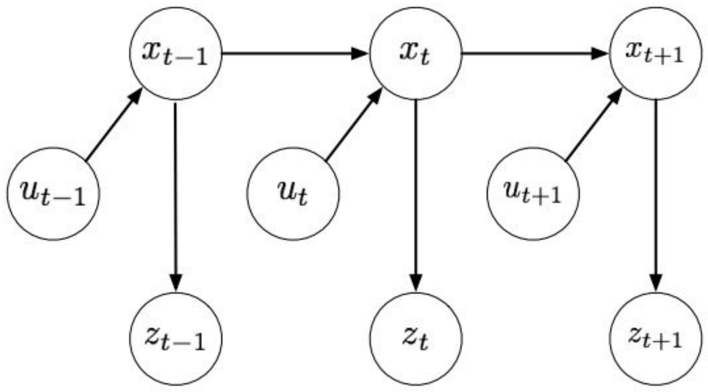

Figure 4: Overall dynamic Bayes network model (Hidden Markov Model)

14.3.13 Belief distribution

We can encode our internal knowledge of the state in a belief distribution, which is a probability distribution that has a value for each possible hypothesis of the true state. Formally, our belief distribution at a particular state is the probability of being at that state given all the measurements and control efforts we've seen so far.

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

Note that this probability distribution can be unimodal or multimodal. A useful tool to calculate our belief distribution is an intermediate belief distribution known as the prediction distribution. The prediction distribution is the probability of being at a state given all the past measurements (except the one we took at this new state) and the previous control efforts.

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

Therefore, the first step to calculate our final belief distribution is to calculate the prediction distribution. Next, we add the latest measurement to sharpen the belief. This second step of calculating the true belief from the prediction belief is called the correction step or the measurement update. This leads us to the Bayes filter algorithm.

14.4 Bayes Filter

The Bayes' filter algorithm is the most general algorithm for calculating state beliefs. The key assumption made in this algorithm is that the state x_t is complete, i.e. only depends on the previous state and control effort, x_{t-1} and u_t. Note that the Bayes' filter algorithm is the most general algorithm but is not necessarily achievable. The algorithm assumes a continuous state space, continuous time, and requires that we be able to compute all the beliefs in an arbitrarily dimensionally large space. Though not practical, the Bayes' filter algorithm is still a great starting point to derive more practical algorithms that can be used in different scenarios.

14.4.1 Overview of algorithm

This is a recursive algorithm. It is initialized with some guess of the initial state, i.e. an initial belief $bel(x_0)$. This can be a uniform distribution of density or point masses if we have no prior information.
The first step is to carry out the prediction step to compute $\overline{bel}(x_t)$ using the previous control effort u_t used to get to state x_t. This is simply the integration over all possible previous states x_{t-1}, of the product of the prior belief $belx_{t-1}$ (where we think we were before) and the transition model given that we know our last control effort $p(x_t \mid u_t, x_{t-1})$. This prediction belief is an encoding of how likely it is that we're now at x_t.

$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$$

The second step is the measurement update where the most recent measurement is incorporated to sharpen our belief. This involves taking our prediction belief and multiplying it by the probability of seeing the measurement that we did z_t if we were actually at our predicted state x_t. This allows us to correct our prediction belief and find our true belief.

$$bel(x_t) := \eta p(z_t \mid x_t)\overline{bel}(x_t)$$

After these two steps, a control effort u_{t+1} is taken, we transition to a new state x_{t+1}, and the process of prediction and measurement update is repeated. This is what makes the Bayes' filter algorithm a recursive algorithm.

Derivations The second step of the Bayes' Filter Algorithm involves the measurement update of our belief model. Here we will go into further detail regarding the derivation of the measurement update.

We begin with our belief

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

We then expand this using Bayes' Rule to get:

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

Where the denominator is the normalizor and is equal to η^{-1}. Next, we use the Markov Property to get:

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

Where the right term equates to the prediction distribution $\overline{bel}(x_t)$. Next we derive the Correction Update that we perform on that prediction distribution

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

Using the law of total probability, we expand to get:

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

Next we use the Markov Property to simplify the left term:

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

For general output feedback policies, u_t does not provide feedback additional information on x_{t-1} and so:

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

The right term then simplifies so that we get our result:

$$\overline{bel}(x_t) = \int p(x_t|x_{t-1}, u_t)bel(x_{t-1})dx_{t-1}$$

14.5 Discrete Bayes' filter algorithm

For problems with finite state spaces, a Discrete Bayes' Filter can be implemented. Initially, we represent the belief state, $bel(x_t)$, as a probability mass function $\{p_{k,t}\}$. The pseudo code for the algorithm is shown below:

```
Data: $\{p_{k,t-1}\}, u_t, z_t$
Result: $\{p_{k,t}\}$
initialization;
foreach $k$ do
    $\overline{p}_{k,t} = \Sigma_i p(X_t = x_t|u_t, X_{t-1} = x_i)p_{i,t-1}$;
    $p_{k,t} = \eta p(z_t|X_t = x_k)\overline{p}_{k,t}$;
end
return $p_{k,t}$
```

Algorithm 1: Discrete Bayes' Filter Algorithm

References

[SNS11] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[TBF05] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.