Unit 4: Localization Part 2
Introduction to Markov Localization

Computer Science 4766/6912

Department of Computer Science
Memorial University of Newfoundland

June 13, 2018

Introduction to Markov Localization

- In this section we introduce *Bayes filter* which is the basis of most robot localization algorithms
- We will refer to this style of localization as **Markov localization** because Bayes filter utilizes the Markov assumption (to be described)
- Three different Markov localization algorithms will be presented:
  - Grid localization
  - Monte Carlo localization
  - Kalman filter localization

Different Localization Problems

Local localization (a.k.a position tracking): The initial pose of the robot is known

Global localization: The initial pose of the robot is *unknown*; More difficult than local localization; Requires a multiple hypothesis belief representation

The kidnapped robot problem: This is a particular form of global localization where the robot is transported from its current position to a new position—without telling the robot that this transportation has taken place. A robot which solves the kidnapped robot problem will be able to determine its new position. A robot which cannot solve the problem will still believe it is at the old position.

A robot that can solve the kidnapped robot problem is able to recover from errors much more readily than one that cannot.

Notation

Note: The notation will now change to that of [Thrun et al., 2005]

x_t The robot's state at time t

This is a vector giving everything we need to know about the robot's state at time t; It will often just give the robot's pose, $x_t = [x, y, \theta]^T$, but sometimes additional information about the world, such as the position of landmarks, will be added to x_t

u_t Control input at time t; This can either be a measured velocity, or a proprioceptive (e.g. wheel encoder) measurement of the robot's movement from time $t-1$ to time t

z_t Sensor input at time t

We assume the robot is in some state x_{t-1} (which happens to be unknown). It then executes some action u_t, which moves it into state x_t. The exact state is uncertain, so the robot reads its sensors to obtain z_t. The task of the localization algorithm is to estimate the probability distribution of x_t, given u_t, z_t, and the distribution of x_{t-1}.The figure below illustrates this development,

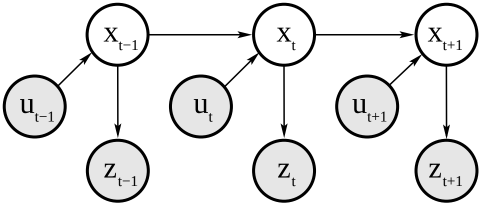


Figure 2.2 The dynamic Bayes network that characterizes the evolution of controls, states, and measurements.

The robot's belief that it is in some state x_t will be denoted $bel(x_t)$,

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

where the subscript $1:t$ indicates all current and past measurements or odometry readings.

If we know the robot's start position, then $bel(x_0)$ would be initialized to 1 at $bel(x_0^{true})$ and 0 everywhere else (minimum uncertainty). If the start position is totally unknown then $bel(x_0)$ would be set to a uniform distribution (maximum uncertainty).

Assuming we begin with the robot's previous belief $bel(x_{t-1})$, the new belief $bel(x_t)$ will be obtained through Bayes filter in a two stage process. In the first stage the new odometry reading u_t is incorporated. This gives us the following **predicted** belief

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

The Markov Assumption: Knowing the previous state vector x_{t-1} and the system's current inputs u_t and z_t gives everything we need to know to compute x_t.That is, we don't need the past history of states $x_{t-2}, x_{t-3}, ...,$ or the past history of movements or sensor inputs in order to probabilistically determine x_{t+1}. Another way of saying this is that the state vector x is considered complete **complete**.

For this assumption to hold, the state vector must include a *complete* description of all objects within the environment (inluding a complete map; a description of all people/robots/animals in the environment and their complete state vectors; etc...).

Thus, this assumption is generally untrue in practise, but we utilize it nonetheless as it renders the localization problem tractable.

All of the main probabilistic localization algorithms can be derived from from Bayes Filter,

Bayes Filter

Inputs: $bel(x_{t-1}), u_t, z_t$
For all x_t,

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

$$bel(x_t) = \eta p(z_t|x_t)\overline{bel}(x_t) \quad (2)$$

Output: $bel(x_t)$

Equation (1) updates the belief to account for the robot's motion. This generates the **prediction** $\overline{bel}(x_t)$. Equation (2) achieves the **measurement update**. It incorporates the sensor values and combines this information with the prediction to update the belief.

The prediction: equation (1)

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

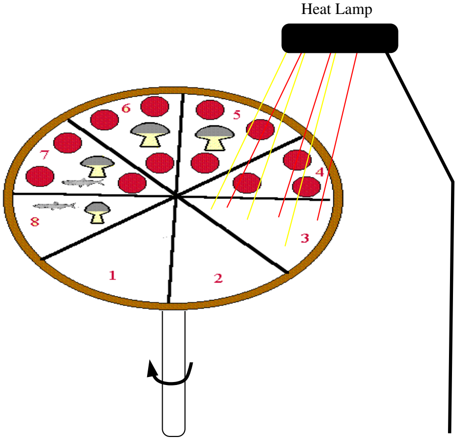
$bel(x_{t-1})$ gives the probability of each possible previous state x_{t-1}. The prediction is obtained by summing the probability of each possible previous state times the probability that we have just made the trip from that previous state to the current state:

$$p(x_t | u_t, x_{t-1})$$

This is known as the **motion model**.

Example: A Pizza-Turning Robot

Mrs. Vanelli's has purchased a pizza-turning robot which employs a rather inaccurate motor. The robot is designed such that at any discrete time step t, only one of the pizza's eight slices will be underneath the heat lamp



We wish to determine the probability that slice i of the 8-slice pizza is under the lamp.

Our belief representation will be discrete; It is an array of length 8 storing the probabilities of each slice being under the lamp

At every time step the action, u_t, is the same—the pizza rotates by one noisy step; This fact, plus the discrete representation allow equation (1) of Bayes filter to be simplified:

For all x_t,

$$\overline{bel}(x_t) = \sum p(x_t | x_{t-1}) bel(x_{t-1})$$

This robot has no sensing capability. Thus, $bel(x_t) = \overline{bel}(x_t)$. There is therefore no need to apply equation (2).

$$bel(x_t) = \sum p(x_t|x_{t-1}) bel(x_{t-1})$$

What is the robot's belief that slice *i* is under the heat lamp?

$$bel(x_t = i) = \sum_{j=1}^{8} p(x_t = i|x_{t-1} = j) bel(x_{t-1} = j)$$

We need a *motion model* for $p(x_t = i|x_{t-1} = j)$. Lets say that the turning mechanism has a 0.5 probability of turning the pizza by one slice; a 0.25 probability of turning by two slices; and a 0.25 probability of not turning it at all. We assume all turns are in the positive direction.

$$p(x_t = i|x_{t-1} = j) = \begin{cases} 0.25 & \text{for } i = j \\ 0.5 & \text{for } i = j \oplus 1 \\ 0.25 & \text{for } i = j \oplus 2 \\ 0 & \text{otherwise} \end{cases}$$

where $\oplus$ indicates addition with wrap-around (e.g. $7 \oplus 2 = 1$)

Assume that $bel(x_0)$ is 1 for $x_0 = 1$ and 0 otherwise. The application of equation (1) yields:

	i							
t	1	2	3	4	5	6	7	8
0	1	0	0	0	0	0	0	0
1	0.25	0.5	0.25	0	0	0	0	0
2	0.0625	0.25	0.375	0.25	0.0625	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
60	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125

A peak in $bel(x_t)$ at the most likely position persists for some time, but our uncertainty about which slice is under the lamp only grows with time—eventually leading to global uncertainty (i.e. a uniform distribution).

Back to the Bayes filter...

The measurement update: equation (2)

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

Recall that $\overline{bel}(x_t)$ is our new belief in x_t after incorporating u_t. We remain confident in x_t only if our sensory observation z_t is consistent with x_t. This confidence is expressed by the **measurement model**,

$$p(z_t|x_t)$$

η is a normalizing factor applied after all $p(z_t|x_t)\overline{bel}(x_t)$ have been computed,

$$\eta = \frac{1}{\int p(z_t|x_t)\overline{bel}(x_t)dx_t}$$

Typically, the measurement model requires a map of the environment so that we can determine how likely it was to observe z_t at position x_t.

Back to the pizza...

To correct the problem of ever-increasing uncertainty, a mushroom detector is installed. In order for this sensor to be useful in determining which pizza slice is under the lamp, a map of the pizza is required.

The mushroom map:

slice i	1	2	3	4	5	6	7	8
Mu?	Yes	Yes	Yes	Yes	No	No	No	No
$p(z_t = Yes \mid x_t = i)$	0.9	0.9	0.9	0.9	0.1	0.1	0.1	0.1
$p(z_t = No \mid x_t = i)$	0.1	0.1	0.1	0.1	0.9	0.9	0.9	0.9

The bottom two rows give the measurement model—the mushroom detector is correct 90% of the time

Assume again that we know the heat lamp is initially on slice 1. The application of the prediction step yields,

t	$\overline{bel}(x_t)$							
1	0.25	0.5	0.25	0	0	0	0	0

To apply the measurement update, we need to know z_t. Assume $z_t = Yes$. $p(z_t = Yes|x_t = 1) = 0.9$, so

t	almost $bel(x_t)$							
1	0.225	0.45	0.225	0	0	0	0	0

We then have to normalize this ‘almost correct’ belief

t	$bel(x_t)$							
1	0.25	0.5	0.25	0	0	0	0	0

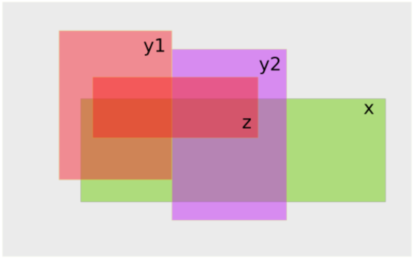
We continue to update our belief; Assume $z_2 = z_3 = Yes$,

t	$bel(x_t)$							
1	0.25	0.5	0.25	0	0	0	0	0
2	$\overline{bel}(x_t)$							
	0.0625	0.25	0.375	0.25	0.0625	0	0	0
	$bel(x_t)$							
	0.0662	0.2647	0.3971	0.2647	0.0074	0	0	0
3	$\overline{bel}(x_t)$							
	0.0165	0.0993	0.2482	0.3309	0.2335	0.0699	0.0018	0
	$bel(x_t)$							
	0.0227	0.1362	0.3405	0.4540	0.0356	0.0107	0.0003	0

Notice how the measurements now reduce uncertainty.

Derivation of Bayes Filter

We will derive Bayes filter below. First, we illustrate the probability of an event conditioned on multiple other events. Consider this Venn diagram,



The Law of Total Probability requires mutually exclusive and exhaustive events y_i. Hence:

$$
\begin{aligned}
p(x) &\neq \sum_i p(x|y_i)p(y_i) \\
p(x|z) &= \sum_i p(x|y_i, z)p(y_i|z)
\end{aligned}
$$

Bayes Rule is always applicable,

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

If an event is conditioned on multiple events, we can still apply Bayes rule as long as we take care which of these other events is expanded,

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

Bayes filter takes as input the belief from the last time step,

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

It requires the motion and measurement models to be supplied,

$$\begin{aligned} \text{Motion model:} \quad & p(x_t|u_t, x_{t-1}) \\ \text{Measurement model:} \quad & p(z_t|x_t) \end{aligned}$$

As its output, it produces the current belief,

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

We begin with the current belief and work backwards to determine how to compute it,

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

Apply Bayes rule,

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

Bayes rule is applied for all x_t. The denominator above does not depend upon x_t. Therefore, we do not need to evaluate it. We have the constraint that the probability over all x_t must sum to 1. Therefore, we treat the denominator as a normalizing constant which can be determined after all of the numerators have been found. We call this constant η,

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

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

Consider the first factor,

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

If x_t is given then we don't need to know anything about the past measurements or controls in order to compute the probability of z_t (Markov assumption). Thus,

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

Where $p(z_t|x_t)$ is the measurement model.

Now consider the second factor. This is equal to our definition of the prediction,

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

Therefore we have established equation (2) of Bayes filter,

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

But how do we obtain $\overline{bel}(x_t)$? We repeat again the definition:

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

The previous possible states x_{t-1} are assumed to be mutually exclusive and to represent all possibilities. Hence, we can apply the law of total probability:

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

Consider the first factor in the integral,

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

If x_{t-1} is given, knowledge of previous measurements $z_{1:t-1}$ and past controls $u_{1:t-1}$ tell us nothing (Markov). Hence,

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

This is the motion model.

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

Consider the second factor,

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

Knowledge of the most recent action u_t tells us nothing about x_{t-1}(unless we know that in certain previous states we would have taken certain actions—assume that we don't know this, meaning that actions are chosen randomly). Therefore, we can re-write the factor as follows:

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

We can recognize this as $bel(x_{t-1})$ which is the input to Bayes filter.Therefore the integral above can be rewritten as follows:

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

This is equation (1) of Bayes filter.

The above analysis shows that Bayes filter appropriately computes $bel(x_t)$ if the input $bel(x_{t-1})$ is correct. If we assume that the initial belief $bel(x_0)$ at time $t = 0$ was correct, then the correctness of Bayes filter follows by induction.

References

Thrun, S., Burgard, W., and Fox, D. (2005).
Probabilistic Robotics.
MIT Press.