

**Introduction to Robotics**  
**Professor Balaraman Ravindran**  
**Department of Computer Science**  
**Indian Institute of Technology, Madras**  
**Lecture - 33**  
**Kalman Filter**

(Refer Slide Time: 00:20)

Recursive State Estimation



The Bayes Filter algorithm:

```
1: Algorithm Bayes_filter( $bel(x_{t-1}), u_t, z_t$ ):
2:   for all  $x_t$  do
3:      $\bar{bel}(x_t) = \int p(x_t | u_t, x_{t-1}) bel(x_{t-1}) dx$ 
4:      $bel(x_t) = \eta p(z_t | x_t) \bar{bel}(x_t)$ 
5:   endfor
6:   return  $bel(x_t)$ 
```

We made **no** assumptions about the transition model, the measurement model nor the belief distributions.



Introduction to Robotics

Prof. Balaraman Ravindran

Hello, everyone. Welcome to the first module in Week 10. So this week, we are going to continue looking at recursive state estimation equation. So the last week, we looked at the base filter algorithm.

And so, essentially, so this is how it operated. So we were given the belief in the previous state. And then we had the control action  $u_t$ , and also the measurement or observation  $z_t$  that we made. And based on this, we ended up computing the revised belief for the state  $x_t$ , for the time step  $t$ .

And the way we did this is we first computed  $\bar{bel}(x_t)$ , which is essentially applying the motion model, which is given by  $p(x_t | u_t, x_{t-1})$ . Applying the motion model on the belief state, on the prior belief state. And then making a correction update or a measurement update, based on the actual sensory inputs that we get or the measurements that we need. And there we used the measurement model, which is  $p(z_t | x_t)$ , and the  $\bar{bel}(x_t)$  that we computed in between.

So this is essentially the structure of the base filter algorithm. This is the same structure we will be following for all the algorithms that we will be looking at this week as well. So one of the

important things I want to remind you is that when we looked at the base filter algorithm, it is more of a conceptual algorithm. So I did say that, for practical implementation, we will run into problems computing the eta, so we had to look at simpler versions where we make assumptions about the model. So in the base filter algorithm that we looked at, we made no assumptions about the transition model. So it could be of any form.

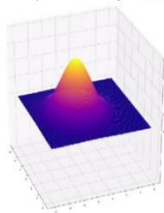
And similarly, we made no assumptions about the belief distribution nor about the measurement model itself. So all of these were assumed to be arbitrary probability distributions or mass functions, which we could then manipulate.

And the very simple example that we looked at, they were all multinomial distributions. In fact, they were all binomial because it just only two outcomes that we were looking at. And that is the setup that we had last week. So this week, we will first start looking at a family of algorithms called the Gaussian Filter algorithms.



(Refer Slide Time: 02:37)

### Gaussian Filters

- ▶ Gaussian Filters are a family of recursive state estimators for continuous spaces, that all share the idea that the belief is represented by a multivariate normal distribution,  $\mathcal{N}(\mu, \Sigma)$ .


$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

- ▶ This choice to represent beliefs by a Gaussian distribution creates a model where there is a small margin of uncertainty around a single true state.
- ▶ Gaussian distributions are a poor choice for estimation problems where many distinct hypothesis exist.

Introduction to RoboticsProf. Balaraman Ravindran

So here, the basic assumption with Gaussian filter algorithms is that the, the family of belief distributions that we are going to use. The belief distributions are going to come from a multivariate normal distribution. So the multivariate normal distribution here, which we will denote by the symbol  $n$ , of  $\mu$  comma  $\Sigma$ , where  $\mu$  is the mean of this distribution. And  $\Sigma$  is the covariance matrix. So you should be familiar with this, if not earlier, at least after looking at the revision videos from last week.

So the density itself is given, the probability of  $x$  for under the normal distribution,  $n \mu \Sigma$ , it is actually given by this expression, as you know. So it is  $\frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2\pi}}$ , nearly  $\frac{1}{\sqrt{2\pi}}$   $\Sigma$ , and  $e^{-\frac{x^2}{2\pi}}$   $\Sigma$ . So this is what we know from the univariate and the multivariate version is this.

And the idea behind the using this is that my belief state is going to be such that that is 1 true state and I have some margin of uncertainty around the true state. So if you look at the figure here, so the true state will be somewhere there. The true state will be somewhere there and I have some kind of uncertainty around what the true state is and that is represented by the Gaussian distribution.

So that would mean that at every point of time, my belief is such that there is some known true state and uncertainty around that. So this is fine. So if you think about how we were doing it earlier, so earlier, let us say that there were like 5 different states that the robot could be in; I basically had 5 numbers.

So which, which looked at the probability that the robot is in each 1 of those states, if you remember the door open door closed, so the 2 states where the door is open, or the door is closed. And basically, I had two numbers that represented the probability that I was in a state with the door open, and the probability that I was in a state with the door closed.

There is really, there is no need to have any notion of similarity between these states because they were just numbers. There are just two independent numbers. Actually, I should not say independent numbers, were just two numbers that I were using, that I was using for representing this.

I could have many more such state, not just 2. I could even have like 100 different states, which the robot could be in. And I would end up using 100 different numbers for representing the probability.

But once I moved to this kind of a Gaussian setting, so what is the most important thing, apart from the fact that I am looking at a single true state and the margin of uncertainty around that; the other important thing is that I am also looking at a notion of distance in this state space.

So the earlier, we were doing the other belief updates, I really did not have a notion of a distance. The, none of the probability distributions that I was dealing with had a notion of distance as an integral part of it. But now, when I get into a Gaussian family of distributions, so the notion of distance becomes very important because I have to compute what is the distance that  $x$  is from the mean. So very simply, there itself.

So when I say, there is a region of uncertainty around a true state, that means, there is a notion of distance that allows me to model this region of uncertainty. So far, I, in the basic filter algorithm, I did not quite need to have a notion of distance. It is kind of moot if you think about it. In almost any robotic problem that we are looking at, we would have, we would have a notion of distance, very natural notion of distance.

So it is not a very, very critical thing to worry about. But I just wanted to point out that from now onwards, we will have to worry about this notion of distance between the in the state space, so we actually have to worry about a place that has a distance defined on it that would limit the kind of representations that we can use for the statement represent the state space.

And another thing to note here is that in the earlier case, so when we had like 5 or 6, let us say we have 5 or 6 states in which the robot could be in, and I was using 5 or 6 numbers to represent the probability distribution, that could have been multiple states, which the robot considered as equally good candidates. So my belief distribution could have been multimodal.

But when I actually look at simple Gaussian distributions, when I start looking at a single Gaussian distribution, it actually turns out to be a poor choice, when these kinds of multiple hypotheses could exist. If this belief distribution could be multimodal, then the Gaussian cannot capture it.

So all of this is great. But the Gaussian assumption is a very powerful assumption because it makes computation rather simple. And it is not often the case that we end up in situations where there are multiple distinct hypotheses and that for Gaussian filters are very popular family of filters that are used very widely in practice.

(Refer Slide Time: 08:03)

## Gaussian Filters



### The Kalman Filter

Besides the Markov assumption, to ensure that the belief distribution estimated by the Kalman filter algorithm will be a Gaussian distribution, we assume that the following three properties hold:

1. The system must follow linear dynamics, with randomness in state transition modelled as Gaussian noise.

$$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t.$$

$$\text{Here, } \epsilon_t \sim \mathcal{N}(0, R_t)$$

The state transition probability  $p(x_t | u_t, x_{t-1})$  is thus given by:

$$\mathcal{N}(A_t x_{t-1} + B_t u_t, R_t), \text{ i.e.,}$$

$$\frac{1}{(2\pi)^{n/2} |R_t|^{1/2}} \exp\left(-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1} (x_t - A_t x_{t-1} - B_t u_t)\right)$$

So the first Gaussian filter that we are going to look at is called the Kalman Filter. So apart from assuming that the belief is, a belief is a Gaussian, the Kalman Filter makes the following assumptions about the Gaussian distribution.

So we already know that the dynamics, both the movement model, the motion model, as well as the measurement models, both of these are satisfying the Markov assumption. But now, we will addition, we are going to assume the following for the motion model.

The first thing we will assume is the system follows a linear motion model. So what do I mean by that? So the state transition expression is given by that. So if I am given  $x_{t-1}$  and a given  $u_t$ , so the new  $x$  is determined by a linear combination of  $A_t x_{t-1}$ , and that  $A_t$  is just a set of coefficients for  $x_{t-1}$  and  $B_t u_t$ , again, the set of coefficients for  $u_t$ , plus some kind of noise to account for the stochasticity.

Remember, we had earlier, we had the probability of the state transition, we had this probability expression. What is the probability of  $x_t$  given  $u_t$  and  $x_{t-1}$ ? To account for this noise, account for the probability. So we are going to assume that there is a noise term which is given by  $\epsilon_t$ . Let me repeat it again.

So the dynamics is linear, in the sense that  $x_t$  is given by  $x_{t-1}$  times some coefficients, at without basically constant plus  $u_t$ , which is the action that you take at time  $t$ , times some other set of coefficients. So  $x_t$ , this whole expression is linear in both  $x_{t-1}$  and in  $u_t$ , and to take

care of the stochasticity, we actually assumed there is an additive noise denoted by  $\epsilon_t$ , and this additive noise is assumed to be 0 mean and with the variance  $R_t$ . It is assumed to be a Gaussian with 0 mean and the variance of  $R_t$ .

So this is what this notation is. So  $\epsilon_t$  is assumed to be a Gaussian random variable, a Gaussian noise random variable with 0 mean and a covariance of  $R_t$ . So this essentially means that if you keep on repeating this transition many, many times, I may go to from  $x_{t-1}$ , I apply  $u_t$  and I end up at  $x_t$ . So the noise would average out.

So I will basically have 0 noise and then the linear model will be the correct prediction, just the linear part alone. So without the just this part alone would make the correct prediction in expectation. So that is essentially what we are assuming.

Now, the total state transition property, remember, that is what we are interested in. The total state transition probability  $p$  of  $x_t$  given  $u_t$  and  $x_{t-1}$  can now be written as a Gaussian variable, whose mean is given by the linear part of the dynamics  $A_t x_{t-1} + B_t u_t$ , and the covariance is given by  $R_t$ , which is the covariance of the noise.

So this is fairly straightforward. So I take a Gaussian random variable and I add a deterministic fixed quantity to it. So, therefore, the overall distribution of the entire expression is now has a mean of that fixed quantity that I added because originally, it was 0 mean and now, and the covariance stays the same because there is no noise in the term here. Make sense?

And of course, I can, I can write it out in in a little bit more scary form, but this is the actually the multivariate Gaussian distribution written out with the mean,  $A_t x_{t-1}$ , and plus  $B_t u_t$  and the covariance of  $R_t$ . It is rather straightforward. So do not get scared with this expression.

So just to recap, we are assuming that the motion model follows a linear dynamics. So there is  $x_{t-1}$  and there is  $u_t$ . And the expression for  $x_t$  is basically linear in both,  $x_{t-1}$  and in  $u_t$  plus an additive Gaussian noise, which is 0 mean and with a covariance of  $R_t$ . So I hope, I hope that is clear to people.

(Refer Slide Time: 12:33)

Gaussian Filters

2. The measurement probability must also be linear in its arguments, with added Gaussian noise:

$$z_t = C_t x_t + \delta_t$$

Here,  $\delta_t \sim \mathcal{N}(0, Q_t)$


The measurement probability  $p(z_t|x_t)$  is thus given by:

$$\mathcal{N}(C_t x_t, Q_t), \text{ i.e.,}$$
$$\frac{1}{(2\pi)^{n/2} |Q_t|^{1/2}} \exp\left(-\frac{1}{2}(x_t - C_t^{-1} z_t)^T Q_t^{-1} (x_t - C_t^{-1} z_t)\right)$$

3. The initial belief  $bel(x_0)$  must be normal distributed with some mean  $\mu_0$  and covariance  $\Sigma_0$ . i.e  $\mathcal{N}(\mu_0, \Sigma_0)$

$$bel(x_0) = \frac{1}{(2\pi)^{n/2} |\Sigma_0|^{1/2}} \exp\left(-\frac{1}{2}(x_0 - \mu_0)^T \Sigma_0^{-1} (x_0 - \mu_0)\right)$$

Introduction to Robotics Prof. Balaraman Ravindran



Gaussian Filters

### The Kalman Filter

Besides the Markov assumption, to ensure that the belief distribution estimated by the Kalman filter algorithm will be a Gaussian distribution, we assume that the following three properties hold:

1. The system must follow linear dynamics, with randomness in state transition modelled as Gaussian noise.

$$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t$$

Here,  $\epsilon_t \sim \mathcal{N}(0, R_t)$

The state transition probability  $p(x_t|u_t, x_{t-1})$  is thus given by:

$$\mathcal{N}(A_t x_{t-1} + B_t u_t, R_t), \text{ i.e.,}$$
$$\frac{1}{(2\pi)^{n/2} |R_t|^{1/2}} \exp\left(-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1} (x_t - A_t x_{t-1} - B_t u_t)\right)$$

Introduction to Robotics Prof. Balaraman Ravindran

And so, what are the other assumptions we make? The second assumption we make is that the measurement probability, the measurement model should also be linear in its arguments. So what is the argument of the measurement model? Remember, the measurement model, we are looking at the probability of  $z_t$ , which is the measurement at time  $t$ , given the state is  $x_t$ .

So that translates to, in a linear model that translates to this expression. So  $z_t$  equal some set of coefficients  $C_t$  times  $x_t$  plus a noise model  $\delta_t$ . We had like  $\epsilon_t$  earlier. Now, we will use a noise model that is given by  $\delta_t$ . And so,  $\delta_t$  is again, a 0 mean, Gaussian, with the

covariance denoted by  $Q_t$ . So earlier, when I was specifying the motion model earlier, let us go back.

When I would have specified the motion model, I would have needed to give you this distribution. So and how did we do it when we did the example? So for every value that  $u_t$  and  $x_t$  minus, minus 1 could take, we specified the probability for every value that  $x_t$  could take. If you remember the sample example that we looked at, for every value that  $x_t$  minus 1, and  $u_t$  could take, and for every value that  $x_t$  could take, we specified a probability.

But now, our job is slightly simpler. So we do not have to run over all possible values of this. If I want to specify the state transition probability, all I need to specify are  $R_t$ ,  $A_t$ , and  $B_t$ . So I have to specify  $A_t$ , I have to specify  $B_t$ , and I have to specify  $R_t$  to specify the full transition model.

Likewise, for the measurement model, I have to specify  $C_t$ , which is the linear coefficients for the measurement model. As well as  $Q_t$ , which is a covariance matrix of the noise that I add in the measurement model.

So the overall measurement probability again is given by a Gaussian, where the mean is given by  $C_t x_t$ , and the covariance, which is given by  $Q_t$ , just similar arguments as we had in the previous slide.

So makes sense? Because I am adding a 0 mean Gaussian variable with  $Q_t$  as a standard deviation to fixed quantity, so the overall probability distribution has a mean of  $C_t x_t$ , and a covariance of  $Q_t$ . And again, I can write it out in the, the full multivariate Gaussian expression, which is basically this.

Then what is the third assumption that I am making? I said I am making 3 assumption, so the third assumption I am making is that the initial belief distribution I started with, which is  $bel x$  naught, if I remember. So that is something that I started with the initial distribution.

And earlier, so we had actually thought of it as something uniform. In the numeric example we looked at, we had  $bel x$  naught equal to open as 0.5  $bel x$  naught equal to close as 0.5. Again, the belief was just a set of numbers. But here, to make sure that our entire computation is tractable, we assume that the belief distribution is also given by a normal distribution, just like the noise



that we had in the two models. And we assume that the mean is  $\mu_0$ , the covariance matrix is  $\Sigma_0$ .

And so, basically, that is the expression. So we have  $\mu_0$   $\Sigma_0$ , and therefore,  $\text{bel } x_0$ , which was the distribution, belief distribution at time 0 is essentially given by a covariance matrix with mean  $\mu_0$  and variance, covariance  $\Sigma_0$ . And therefore, we can see what is going to happen. So  $\text{bel } x_0$  has mean  $\mu_0$  and  $\Sigma_0$ . So  $\text{bel } x_1$  would have a mean of  $\mu_1$  and  $\Sigma_1$ .

Remember that the measurement model and the motion model, the transition models do not change, they are given to you a priori. So you are using that and then, you are only refining your belief over time. What will change are the actions of the measurements you take, but the model themselves would stay fixed, or the distributions themselves would stay fixed.

So your,  $A_t$ ,  $B_t$ , and  $C_t$ , and your  $R_t$  and  $Q_t$  do not change. But on the other hand, the belief keeps changing at every time step because that is what you are updating in order to estimate the state in order to localize yourself as we will see later.

So your belief distribution will keep getting updated. So  $\text{bel } x_0$  is given by  $\mu_0$  and  $\Sigma_0$  and  $\text{bel } x_1$  would be given by  $\mu_1$  and  $\Sigma_1$ ,  $\text{bel } x_2$  would be given by  $\mu_2$  and  $\Sigma_2$ , and so on, so forth. And your entire base filter algorithm will now reduce to an algorithm where you update  $\mu_t$  and  $\Sigma_t$  given  $\mu_{t-1}$  and  $\Sigma_{t-1}$  and  $z_t$ . Is that clear?

So your belief distribution now is the one that gets updated. Therefore, your  $\mu_0$  and  $\Sigma_0$  will keep getting updated. And so, you will maintain your belief just by updating these true values.

So just to recap, your system dynamics is defined by multiple quantities. You have  $A_t$ , you have  $B_t$ , you have  $C_t$ , then you have  $R_t$ , and you have  $Q_t$ . So these are the quantities that need to be defined for specifying your system model. That is both the transition probability and the measurement probability.


And for representing your belief, you need two quantities, which is  $\mu$  and  $\Sigma$ . So you start off with some basic assumption  $\mu_0$ , and  $\Sigma_0$ . It could be a very, very flat

Gaussian distribution if you will, without a very significant concentration at the mean. The Sigma, Sigma naught can actually spread out the probability distribution as much as you want but the more informed your prior distribution is, the more quickly you are going to converge to something meaningful.

And at every time step, you use your system parameters and your current action and measurements in order to update your belief distribution as to where you are in the state. And hopefully, as you keep moving around, your belief becomes more and more concentrated around the actual state. So, and we will see how this operates in a bit.

(Refer Slide Time: 19:15)

Gaussian Filters




- ▶ Similar to the **Bayes Filter** algorithm, the **Kalman Filter** algorithm recursively estimates  $bel(x_t)$  from  $\{bel(x_{t-1}), u_t, z_t\}$
- ▶  $bel(x_t)$  is parameterized by  $\mu_t, \Sigma_t$

```

1:  Algorithm Kalman_filter( $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$ ):
2:       $\hat{\mu}_t = A_t \mu_{t-1} + B_t u_t$ 
3:       $\hat{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$ 
4:       $K_t = \hat{\Sigma}_t C_t^T (C_t \hat{\Sigma}_t C_t^T + Q_t)^{-1}$ 
5:       $\mu_t = \hat{\mu}_t + K_t (z_t - C_t \hat{\mu}_t)$ 
6:       $\Sigma_t = (I - K_t C_t) \hat{\Sigma}_t$ 
7:      return  $\mu_t, \Sigma_t$ 

```



Introduction to Robotics
Prof. Balaraman Ravindran

So like we said, so we are going to estimate belief  $x_t$ , based on belief  $x_t$  minus 1,  $u_t$ , and  $z_t$ . So very similar to the base filter algorithm. So these are the steps of the Kalman Filter algorithm. And steps 2 and 3 are actually updating  $bel$  bar. Steps 2 and 3 are updating  $bel$  bar. Steps 4, 5, and 6 are updating  $bel$   $x_t$ . So 2 and 3 update  $bel$  bar  $x_t$ ; 4, 5, and 6 are for computing  $bel$   $x_t$ .

So I am not going to get into the complete derivation of the Kalman Filter here, it is a little cumbersome. And so, what we would do is give you some material to read up if you are interested in looking at the derivation. Alternatively, I mean, so this is something that you could just, you know, take on faith. And then say, yeah, this looks like the right expression, but I am not going to let you do that. So I will give you some material, so you can look it up yourself. But

so essentially, the point here is that this is more additional reading for you to look at how the derivation comes up.

So the idea here is that, so  $\mu_t$  and  $\Sigma_t$  are going to represent my belief. And if you look at it, it kind of makes sense. Intuitively, it makes sense. So what is my  $\mu_t$  that is? So I am going to take my old  $\mu_{t-1}$ , which is I have that because I know  $x_{t-1}$ . So  $x_{t-1}$  is represented by these two entities,  $\mu_{t-1}$  and  $\Sigma_{t-1}$ . So I know my  $\mu_{t-1}$ , which is essentially my previous position; my expected position in the previous time step.

So I am going to assume that, hey, look, I am going to take  $\mu_{t-1}$  and I am going to apply  $F_t$  to that. I am taking  $\mu_{t-1}$ , which is my previous position, I am applying  $F_t$  to that and this is the linear part of my model. This is the linear part of the model if you remember that. So you had  $A x_{t-1} + B u_t$ , gives you  $x_t$ ; plus there was this noise term  $\epsilon_t$ , that was giving you  $x_t$ .

So now, what we are going to do is that  $\epsilon_t$  is essentially given to you by  $R_t$ . So what is that  $\epsilon_t$  going to do? It is going to basically make you more confused about where you are. So your initial confusion was  $\Sigma_{t-1}$ , your initial confusion about where you were was  $\Sigma_{t-1}$  and  $R_t$  is the additional noise that you are adding.  $R_t$  is additional noise you are adding, that is a covariance matrix for the, the transition. So that is additional noise in the transition.

So your  $\Sigma_t$  basically is a function of your original noise where you were. Your original uncertainty about your belief that you are at  $\mu_{t-1}$  plus the noise that is introduced by the motion. So the  $\mu_t$  and  $\Sigma_t$  together gives you a belief of  $x_t$ .

Now, that is basically your motion model or the prediction. The belief, belief after you do the prediction. And then, that is your prediction update and now, this is your correction update or your measurement update. And essentially, so this quantity here is called the Kalman gain, as we will note in the next slide as well, this quantity is called the Kalman gain.

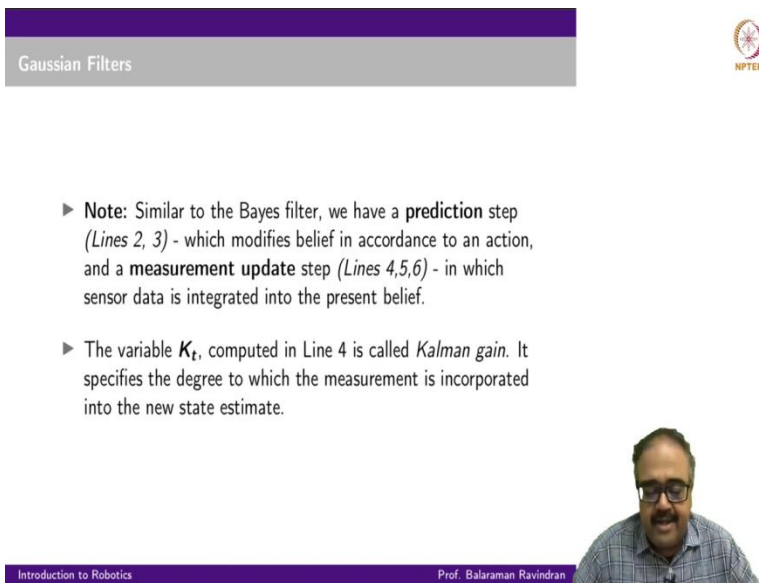
And so, your  $\mu_t$ , which is where you assume you are, is corrected by the measurement that you make. So the  $z_t$  is the actual measurement you make and  $C_t \mu_t$  is the prediction, the measurement, the linear part of the measurement model that you have, remember? So  $C_t \mu_t$

is okay if I was really at  $\mu_t$ . And so, this is the measurement I should have made if there was no noise, but  $z_t$  is actual measurement I make. So that difference is basically going to change what my mean is.

And likewise, I have a complex function here that also tells me how much noise I will have to add based on the actual noise that is expected in the measurements. So that gives me my  $\mu_t$   $\Sigma_t$ . So this part of the derivation is a little, not too hard; there is just a little involved. And I encourage you to just follow it offline.


So at the end of both these updates, so this is the prediction update, and that is the measurement update. So there is a prediction update and then, you have the measurement update. And after that, you get the new belief state, which is given to you by  $\mu_t$   $\Sigma_t$ .

(Refer Slide Time: 24:35)



Gaussian Filters

- ▶ **Note:** Similar to the Bayes filter, we have a **prediction** step (*Lines 2, 3*) - which modifies belief in accordance to an action, and a **measurement update** step (*Lines 4,5,6*) - in which sensor data is integrated into the present belief.
- ▶ The variable  $K_t$ , computed in Line 4 is called *Kalman gain*. It specifies the degree to which the measurement is incorporated into the new state estimate.



Introduction to Robotics Prof. Balaraman Ravindran

- ▶ Similar to the Bayes Filter algorithm, the Kalman Filter algorithm recursively estimates  $bel(x_t)$  from  $\{bel(x_{t-1}), u_t, z_t\}$
- ▶  $bel(x_t)$  is parameterized by  $\mu_t, \Sigma_t$

```

1: Algorithm Kalman_filter( $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$ ):
2:    $\hat{\mu}_t = A_t \mu_{t-1} + B_t u_t$ 
3:    $\hat{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$ 
4:    $K_t = \hat{\Sigma}_t C_t^T (C_t \hat{\Sigma}_t C_t^T + Q_t)^{-1}$ 
5:    $\mu_t = \hat{\mu}_t + K_t (z_t - C_t \hat{\mu}_t)$ 
6:    $\Sigma_t = (I - K_t C_t) \hat{\Sigma}_t$ 
7:   return  $\mu_t, \Sigma_t$ 

```



So this is exactly what we were saying earlier. So lines 2 to 3 are the prediction step. And lines 4, 5, and 6 are the measurement update. And  $K_t$  that we compute in line 4, as you saw earlier is called the Kalman gain. So the  $K_t$  is called the Kalman gain; 2 to 3 are the prediction updates, I am sorry, 2 to 3 are the prediction updates. And 4, 5, and 6 are the measurement updates. So I hope that that is clear.

So the Kalman Filter makes very strong assumptions. It assumes, A, that the dynamics are all linear, the main component of the dynamics is linear. The motion model is linear, the measurement model is linear. So this is a very strong assumption that it makes. And the second assumption it makes is that the noise or the probability distribution that you are seeing for both the motion model and the measurement model, the noise model is as Gaussian. And it also assumes that the belief is a Gaussian distribution.

So basically, there are 3 things. So the linearity assumption, and the Gaussian assumption on all the 3 models that you use. So this makes the life a little easier. So you could basically, I mean if you think about it, these updates are actually just simple matrix multiplications, you can very easily implement it on your favorite tool like MATLAB or whatever is it that you have.

This a simple matrix multiplication and inverse operation, so you can implement it very efficiently. And you do not have to worry about the computational difficulty that I was pointing out in looking at the normalizing factor. And it is also nice in a way because everything is

Gaussian, you are guaranteed that at every state your Bayesian, your, at every state, your belief distribution as well is Gaussian.

If you start off with a Gaussian belief, it will stay a Gaussian distribution because your dynamics and your measurement model both have Gaussian noise. So this is the reason we make this strong assumption because it makes computation so simple.

And the reason Kalman Filters are very popular, Kalman Filters and they are a family of filtering algorithms are very popular is because these assumptions do work in practice. And or slightly, slightly stronger versions of this work in practice. And so, let us look at how the whole Kalman Filter thing would look like.

(Refer Slide Time: 27:07)

The slide features a purple header with the text "Gaussian Filters" and the NPTEL logo in the top right corner. Below the header, the text "A simple one-dimensional localization scenario" is centered. A diagram shows a satellite icon at the top, a robot icon with a red arrow pointing to the right below it, and a list of two bullet points: "▶ The robot moves along the horizontal axis." and "▶ The robot queries it's GPS sensors on it's location". At the bottom, there is a purple footer with "Introduction to Robotics" on the left and "Prof. Balaraman Ravindran" on the right, next to a small video inset of the professor.

So let us take a simple one-dimensional localization scenario. So I have a robot that is moving on the horizontal axis, moving in x-coordinate. I am interested only in looking at what is the displacement of the robot along the x-coordinate. So I am just looking at this, I am not, I am going to assume that the robot does not move in any other direction. I am not worried about the effectors of the robot or anything.

So my state is only the displacement along this line just to make things a little easy and easier to visualize and to draw as well. And I also assume that the robot has a GPS sensor that can basically query its location and the sensor has some amount of noise obviously, especially, if the

robot is indoor, the sensor is going to have a lot of noise. But for the time being, let us assume that we have a GPS sensor that gives you a rough idea of what the location is.

And we are also going to assume whatever we had looked at earlier that the motion model is linear. So in terms of the velocity with which the robot moves and the original location, so we can write a linear model as to what its new location should be. And we are also going to assume that the measurement model is linear with Gaussian noise.

(Refer Slide Time: 28:26)

Introduction to Robotics Prof. Balaraman Ravindran

So this is how the Kalman Filter is going to look like. So I am going to start off with some initial beliefs. So let us look at one diagram at a time. So let us, let us look at this one; that is the first, this the initial belief. Let us look at the initial belief alone.

So I have, this is the belief that I am starting off with. So what does this mean? This means that that is my  $\mu_t$ , that is my  $\mu_t$  and I have some kind of a  $\Sigma_t$  that represents the current belief, current noise in the belief, or kind of uncertainty in the belief that I add this  $x \mu_t$ . So that would be my  $\mu_t$  and I will have a  $\Sigma_t$  here.

Now, what do I do is I actually make a measurement. So this is my first thing to do, I make a measurement. So the measurement tells me, hey, look, this is where you are likely to be. So the measurement's mean is here, the measurement's mean is here and the noise in the measurement is this much. So this is basically what my measurement model tells me.

Now, once I incorporate this uncertainty into my location, you can think of it as I do not move at all. So my belief does not change because I did not make any movement. So no, no, no, no-op; so my belief, my location does not change. So my belief bar is the same as my belief; belief  $x_t$  minus 1 is the same as belief bar  $x_t$  because I did no movement when and then now I do a measurement.

So given that this is the noise associated with the measurement update that I have made, now I basically, incorporate my original noise, which is this, which is my, remember, now this is now my belief bar of  $x_t$  plus my measurement. Now of these together, so give me that as my belief state. So that is my belief right after this.

Now, what I do is I move to the right. So basically I say that I am moving to the right here. Now, after I move to the right, I have to now apply my movement model. So I know what my action is, my action was move to the right, I apply my movement model. Now, my movement model also has some kind of noise in it.

So what happens is, even though I was so certain about where I started out, this is where I started out, this is my belief  $x_t$  now. This is my belief  $x_t$ . Now, my belief bar  $x_t$  plus 1 becomes a lot more noisier but at least you can see that the mean has shifted to where it should be, so this was where the original mean was. This is where the original mean was, now that it has shifted here and that is the effect of the linear part of the model and the fact that the variance is now increased, the variance is increased is because of the noise in the movement model.

Now, at once I have this, this is my belief bar, I make a measurement and the measurement gives me a mean that is somewhere here. That is what the measurement model tells me and that is the noise around the measurement model. So putting these together and that is my final belief at  $x_t$  plus 1.

So I started at  $t$  minus 1 with the belief of that, then that was also because I did not move that was my belief, belief bar of  $x_t$ ,  $x_t$ . And then, I make a measurement, I integrate that and I get my belief  $x_t$ . Then I make a movement here, which a movement was go to the right, that kind of shifts my  $\mu_{t-1}$  to  $\mu_t$  bar and also kind of spreads out the probability mass because of the uncertainty in the movement.

And then, I make a new measurement, so that is a new measurement that I make and that tells me this is the more likely location. So I combine the belief bar and the measurement and finally, get my



bel  $x_t$  plus 1. So that is how the Kalman Filter will look, will work. So every, every point of time, you can see that all the intermediate computations that we do are all Gaussians or all normal distributions. So that makes it easier for us to do the computation.

So far, we looked at the simple Kalman Filter algorithm, which assumes everything is Gaussian, but as we know that, you know, the fact that the movement model and the measurement model are both linear is a very strong assumption. So we look at ways of getting around that, one, by looking at a slightly different version of the Kalman Filter and then, by looking at other more complex filters as well in the next few lectures.