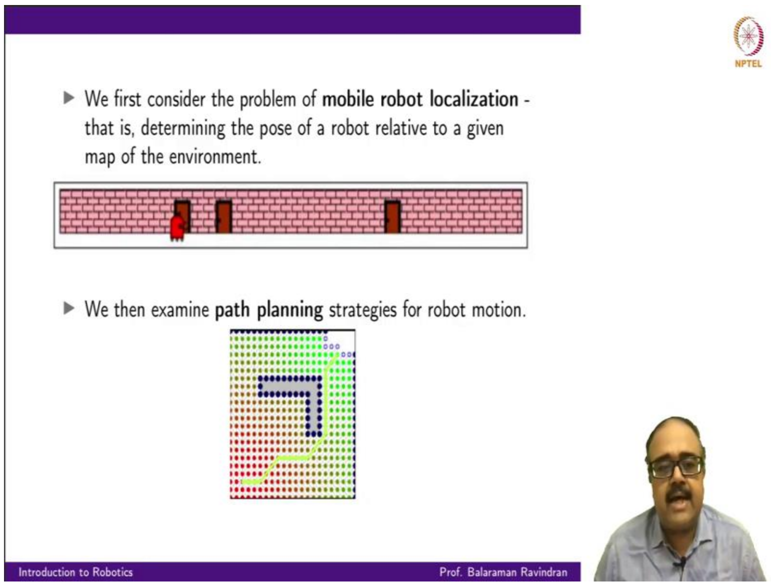
Introduction to Robotics
Professor Balaraman Ravindrian
Department of Computer Science
Indian Institute of Technology, Madras
Lecture 41
Localization Taxonomy

Hello, everyone and welcome to the final week of lectures in the intro to robotics course. And as before, we will continue looking at the algorithm and computer science aspects of it. So, we have looked at, you know, what constitutes the notion of state, and then we looked at recursive state estimation, looked at motion models, we also looked at mapping problems, measurement models, and also looked at how to estimate maps that far.

(Refer Slide Time: 00:43)



- We first consider the problem of **mobile robot localization** - that is, determining the pose of a robot relative to a given map of the environment.
- We then examine **path planning** strategies for robot motion.

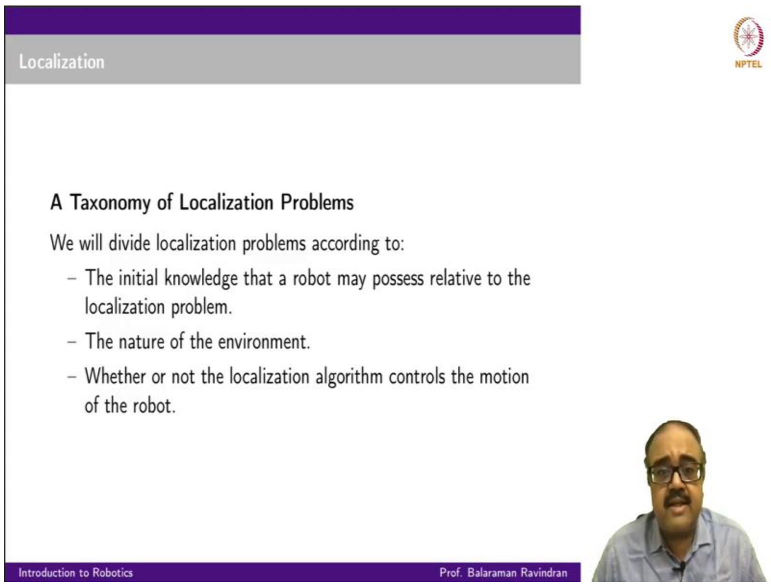
Introduction to Robotics   Prof. Balaraman Ravindran

So, this week, so we will first consider the problem of what is known as the mobile robot localization problem. That is, the question is, given that you have a map of the environment, a map, in this case, it looks like this. So, there is a lot of walls and a few doors in between, let us say you are given the map of the environment in some form, find what the pose of the robot is, relative to the map.

So, we already looked at this environment before in one of the examples, we are just looking at it again, in the context of localization, and then the second set of lectures, we will examine what are called path planning strategies for robot locomotion. So, the idea of path planning is you are given a map.

And you are given a desired location that you would like to reach, let us say is in the top corner. How would you find the efficient path at least how to find a feasible path to go from your start location to the end location? So, given the map, and your models, motion models, and measurement models, and so on, so forth, so under your localization strategy? How do you form a path from start to the road? So, that is the second question that we will look at in these sets of lectures.

(Refer Slide Time: 01:59)



Localization

A Taxonomy of Localization Problems

We will divide localization problems according to:

- The initial knowledge that a robot may possess relative to the localization problem.
- The nature of the environment.
- Whether or not the localization algorithm controls the motion of the robot.

Introduction to Robotics — Prof. Balaraman Ravindran

To start off, so a large fraction of the localization algorithms, a majority of the localization algorithms are essentially an extension of what we have seen as the recursive state estimation problem. In fact, many of the localization algorithms are a version of the base filter algorithm that we already seen, just like we looked at the map estimation algorithm was a version of binary base.

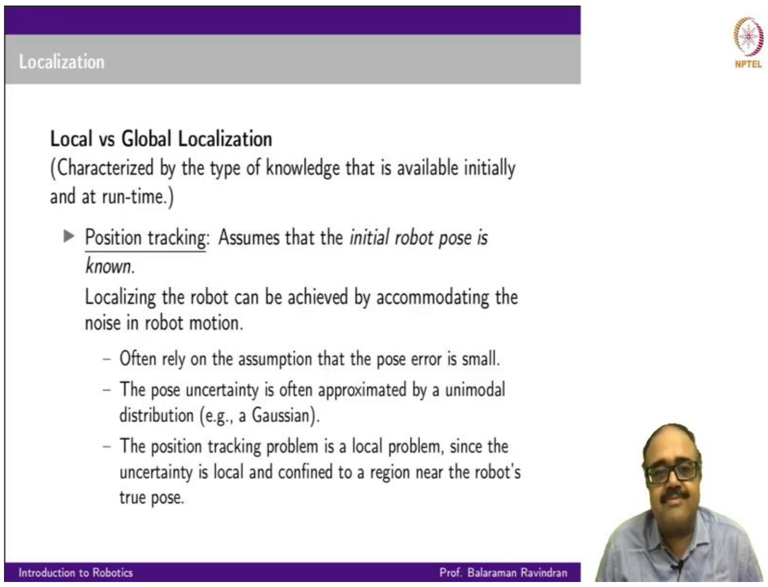
And many of these localization algorithms would also be versions of the base filter algorithm, we look at the very, very basic version of it. And for more, more complex versions of these algorithms, I leave you to study by yourself. That is not the goal of the course, the goal of the course is just to give you a very brief introduction to the variety of different problems that you will be solving algorithmically when you are working with robotics.

So, we will, we will start off by looking at a taxonomy of the localization problems, some kind of a classification of the localization problems, what are the various dimensions under which these localization problems can be classified. So, in fact, these are specific problems themselves. So, and each algorithm would address these problems in a variety of different ways. And we will also point out how the mark of localization algorithm handle some of these problems.

So, the first dimension is the initial knowledge that the robot may process relative to the localization problem. So, what does the robot know? And when it starts, so that that tells you what is it that you want to do? What are the, what are the challenges that you would face?

And the second is the nature of the environment in which the robot is operating. And the third is whether the localization algorithm actually controls the robot, the motion of the robot or is it just passively observing what is happening and so these are the three major dimensions. And finally, you could also have, whether you are working with one robot or multiple robots also, is gives rise to some interesting challenges, so, we will see in a bit.

(Refer Slide Time: 04:08)



Localization

Local vs Global Localization
(Characterized by the type of knowledge that is available initially and at run-time.)

- Position tracking: Assumes that the *initial robot pose is known.*
  Localizing the robot can be achieved by accommodating the noise in robot motion.
  - Often rely on the assumption that the pose error is small.
  - The pose uncertainty is often approximated by a unimodal distribution (e.g., a Gaussian).
  - The position tracking problem is a local problem, since the uncertainty is local and confined to a region near the robot's true pose.

Introduction to Robotics Prof. Balaraman Ravindran

So, the first one is what we will call local versus global localization. So, local localization, that means that I already have some idea of where the robot is. And I would like to be more precise in my localization in that locality, I would like to get a more precise estimate of the pose. And I have some idea that what the initial poses, so basically, as the robot moves starting from the initial pose, I would like to be able to track it with as little error as possible.

So, that is what the local localization essentially means. And it kind of manifests itself as the position tracking problem. So, in the position tracking problem, we are a little bit more aggressive. We assume that the initial position of the robot, the initial pose of the robot is completely known.

And then what you do is, you accommodate the small noise in the robot motion and the sensors that you have, and continue to localise the robot, continue to update the position of the robot assuming that the initial process known. So, this is the position tracking problem.

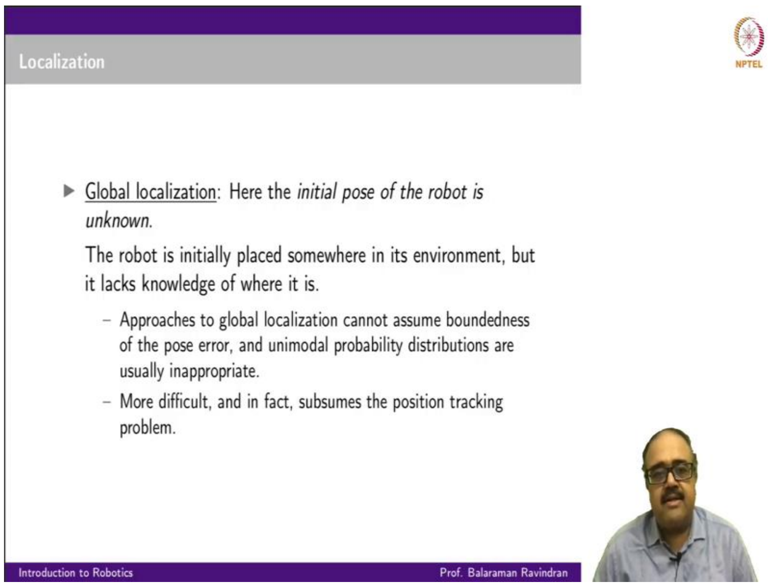
And as you can see, is very similar to the Gaussian filters and the Bayesian filters we looked at earlier.

And you start off with, in fact, many of the examples we looked at, we assume that we did not know the initial position, we were like, add a probability of 0.5 for both the initial positions a robot could be being, but you could start by knowing the initial position, and then tracking this. And we assuming that the error is usually small, otherwise, the error is very large.

Then you would have difficulty tracking the robot, you assume that the measurement error and the movement error are small. And, and since the uncertainty of the robot is essentially confined to a small region near the robots, true pose, because you assume that you know, the true pose, and the uncertainty is confined to the region near the robots true pose, you can in fact, these are cases where you can use a Kalman filter very effectively with a single Gaussian also, or even extended Kalman filter, you can use those efficiently with a single Gaussian, because you are not really interested in accommodating multiple hypotheses.

But you are only interested in tracking this one true pose with small amounts of error rounding. And because the uncertainty is confined to a small region, this is called a local localization problem, or local problem. That makes sense, this is exactly the kind of situations which the Gaussian filters were or more most appropriate for this kind of position tracking problem.

(Refer Slide Time: 07:00)



Localization

- Global localization: Here the *initial pose of the robot is unknown.*

  The robot is initially placed somewhere in its environment, but it lacks knowledge of where it is.
  - Approaches to global localization cannot assume boundedness of the pose error, and unimodal probability distributions are usually inappropriate.
  - More difficult, and in fact, subsumes the position tracking problem.

Introduction to Robotics Prof. Balaraman Ravindran

So, the second class of localization problems are called Global localization problem here, I am going to assume that the initial pose of the robot is completely unknown, I just know that the robot is initially placed somewhere in its environment. But it does not know where it is. Or even if it knows, it might have multiple positions where the robot could start at.

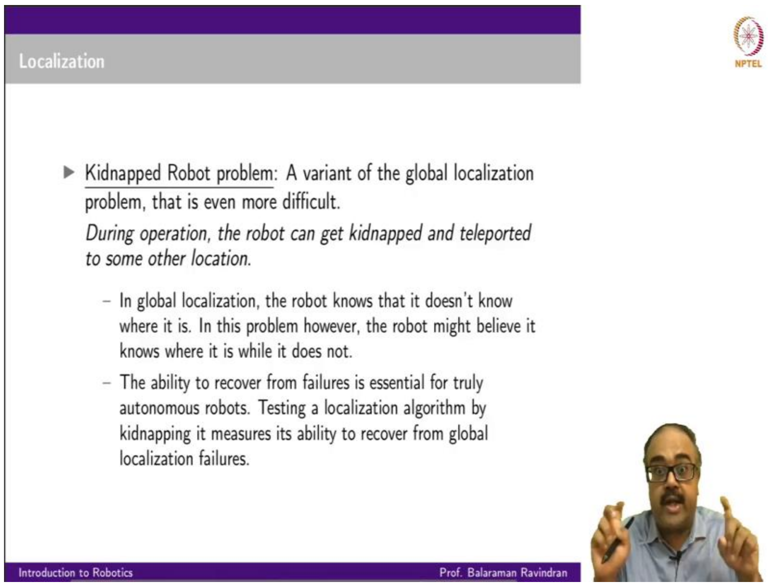
So, we do not know exactly where the robot is starting. And since this, a there could be multiple hypotheses that I would like to keep track of, and b the error could be anywhere, that my initial guess could actually be very off. So, we as we call this a global localization problem, because I do not have bound upper bound on what the error is, so I could be anywhere in this space.

And these are the problems for which Gaussian a single Gaussian based filter would have problem and things like the particle filter are more appropriate. And, in fact this is a much more difficult problem than the position tracking problem and includes the position tracking problem as a special case, as you can see.

So, the initial position of the robot is unknown. Yes, but it could be, you know, in one of few places, which could even be one, if it is in only one place to start off with, then that becomes a position tracking problem, but then, so this is the one way of handling this, of course, is to just set the prior probability that you have to something uniform across the entire space.

But then uni model probability distributions tend to have difficulty handling such a large prior. And we have to look at things like particle distributions, particle filters. So, that is the global localization problem.

(Refer Slide Time: 08:45)



Localization

- Kidnapped Robot problem: A variant of the global localization problem, that is even more difficult.

  During operation, the robot can get kidnapped and teleported to some other location.

  - In global localization, the robot knows that it doesn't know where it is. In this problem however, the robot might believe it knows where it is while it does not.
  - The ability to recover from failures is essential for truly autonomous robots. Testing a localization algorithm by kidnapping it measures its ability to recover from global localization failures.

Introduction to Robotics Prof. Balaraman Ravindran

And there is another interesting variant of the global localization problem which we call as a kidnapped robot problem. So, the kidnapped robot problem is actually a while variant. So, in some sense, it is it just makes the problem very difficult. What you are saying is a guy while the robot is operating, even if it has finally managed to localise itself somehow, and knows where the true position is it can suddenly get quote unquote, kidnapped, so basically you turn off the sensors of the robot pick it up, and place it somewhere else.

So, now what happens is the robot is thinking that it is somewhere else with a very, very high probability wherever it localise itself, because the robot was kidnapped and teleported to some other location. It does not know that that transition has happened. It is truly in a different part of the state space, but it continues to believe that it is where it was originally localising itself.

So, it makes it so to accommodate for this possibility of the kidnapping, you have to assume it is a global localization problem. You cannot continue to do position tracking. So, at every point of time you should be able to, you know, arbitrarily reposition the robot to another part of the state space.

Now, except in sci fi movies, why would anybody want to kidnap a robot? Kidnap robot problem is just a fancy way of saying that I could have, you know, arbitrary errors in a localization problem because of some unlucky sequence of noise measurements, or unlucky sequence of, you know, actuator failures that happen.

And therefore, the algorithm falsely thinks that you are in a different part of the environment, but it continues to get new measurements that kind of contradicts its original localization. And therefore, we would have to somehow recover from that earlier erroneous estimation. Is it clear so when we say kidnapped robot problem not really, that the robot is being you know, blindfolded and kidnapped or anything like that.

It is essentially a way of telling, testing whether the algorithm that you have, can recover from arbitrary errors, arbitrary global localization failures, instead of putting you in the one corridor in the third floor, it might put you in another corridor on the second floor of a building, or if you look at typical and office spaces, you might be thinking that you are next to somebody's cubicle, you might be next to a completely different part of the office.

Because all the cubicles look the same. And sometimes when I go to IT offices, I get lost, so the robot can certainly get kidnapped. In some sense, it can, it can localise itself catastrophically, in a different part of the state space. And so the question is, can you recover? It is different from global localization? Because in global localization problem, I know, the robot knows that it does not know where it is.

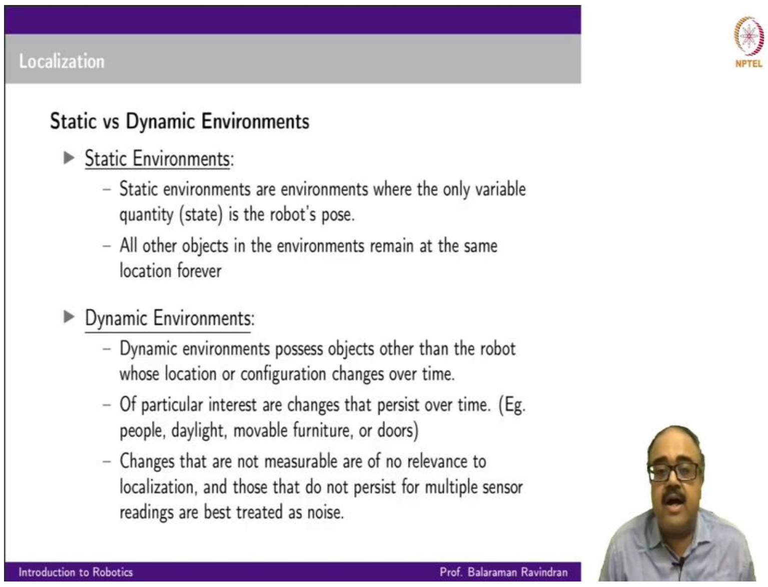
Because it is a belief distribution is fairly large. But this problem, the robot believes it knows where it is. The belief distribution could very well be a spike, could very well be a delta function in a particular location, the robot knows for sure, that is where it is. Now, if I put you in a different if I kidnap the robot, it is impossible for it to recover, because the belief has to have room for it to recover.

So, one way of handling these kinds of kidnaped robot problems should be to you know, periodically, if you are using something like a particle filter, so periodically to you know, put particles randomly all over the workspace and then try to see if you can recover the true pose by doing this kind of smoothing of the probability distribution, smoothing of the belief distribution.

So, that is something that you have to keep in mind that you should never have a if it is possible for you to have this kind of localization errors, global localization errors, then you should make sure that your belief estimation algorithm never becomes too certain. It always keeps the possibility open of having some unknown pose be the true one.

So that is always the, that is the challenge in designing algorithms to accommodate for the kidnapped robot problem. So, moving on. So, this is one dimension, so this is how, whether you think you are, whether you know how much, whether you are in which part of the state space or whether you have to assume for a global localization.

(Refer Slide Time: 13:46)



Localization

Static vs Dynamic Environments

- Static Environments:
  - Static environments are environments where the only variable quantity (state) is the robot's pose.
  - All other objects in the environments remain at the same location forever
- Dynamic Environments:
  - Dynamic environments possess objects other than the robot whose location or configuration changes over time.
  - Of particular interest are changes that persist over time. (Eg. people, daylight, movable furniture, or doors)
  - Changes that are not measurable are of no relevance to localization, and those that do not persist for multiple sensor readings are best treated as noise.

Introduction to Robotics

Prof. Balaraman Ravindran

The second direction is looking at static versus dynamic environment. So, what do I mean by that? Static environments are those where only the robot is moving, or none of the other quantities are changing, everything else in the environment is the same, the object is there, map is fixed, every object in the map is fixed, nothing moves.

And only the position of the robot changes, these are static environments. And these are the, again, the kind of situation that we have been looking at so far in the ball in the motion model, in the sensor model as well as the map, the state estimation algorithms you have seen so far.

On the other hand, you could have dynamic environments. So, dynamic environments are environments where objects other than the robot, could also have varying locations or varying configurations. Like these are configurations that change over time. And even here, there are two kinds of, there two kinds of changes that are that could happen.

So, one or changes that are transient, you know, like, like I was, I do not know, if you remember the example I was giving you in the sensor noise model, I said sometimes the paper could just fly in front of the sensor and cause it to make a short reading. So, these are very transient noises, transient motion, and they do not really, you know, they do not really affect your localization, they do not really affect your localization. And you could just treat them as noise, so this is basically what we are saying here.

So, changes they do not persist over time, they are very transient, they are there for a small time that is there and then again, it goes back on the floor. So, these are not of relevance to the localization problem, or of the to the path planning problem. And therefore, we can just treat these as noise, and then account for that in our model, account for that in our model.

So, that is exactly what we did in the sensor model, we accounted for these kinds of transient movement or transient changes in the variables, and then just decided to treat them as noise that is where the short noise came in to play in the sense of model that we had. So, and similarly any kind of this kind of temporarily blocking the path of the robot could also be modelled as noise in the motion model, like I am trying to move, but then there might be a small small probability that I fail, the not because, just because my you know, wheel sleep or something like that, it could also be because somebody was just standing in front of me and that one, one second, and after that, they moved away.

So, when it tried to move, there was a noise in the transition. What we are really interested in or, you know changes that persist over time. So, what do I mean by that? Suppose I move furniture, I take a table from somewhere and put it somewhere else. Now what happens, the table is something which will block the path of my robot. So in some sense, some pathway got opened up, because the table moved, but some other pathway got blocked, because the table went there.

So, if I was trying to put myself on the map and say, go next to the table, now where I have to go becomes very different, because I have to localise myself with respect to the table. And then I have to go there and if I move the table around, then my whole localization problem becomes complex, something that I have to redo all over again, not things like doors, If a door is closed, it is one thing if a door is open, it is a completely different situation.

And likewise, if there is a significant change in the lighting condition, or if people are there and people are standing, and then I have to either move around them or I have to localise with respect to people standing there, and then they might be, they might actually move to a different location in the environment, and so on, so forth.

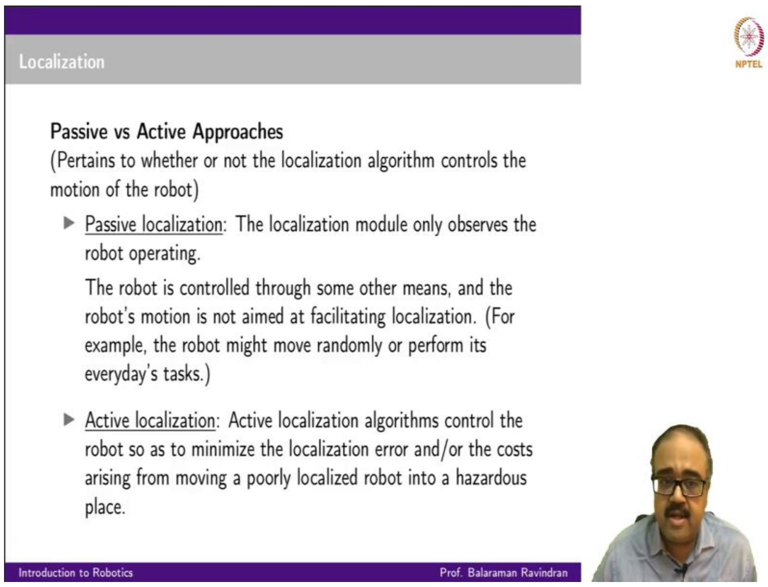
So, there are a few obstacles, few objects, whose positions are of interest to me, both in terms of the localization problem, and also in terms of planning a path. In such cases, we have to figure out a way to handle the handle these objects. So, there are multiple ways in which you can think of handling these objects. One thing is to actually make these objects locations, properties as part of the state, if you make this object, locations of properties are part of the state, then basically have to keep updating the state whenever these objects move. And every time when I want to estimate the new state, I also allow to check whether these objects have moved.

So, in effect, it becomes like a mapping algorithm also, where I am looking at a feature based map where the features are assigned to these objects. So, I am also estimating the map while I am estimating my location, so this becomes a mapping problem as well. But another way to do it is to assume that there is an independent mechanism that is updating the map.

So, at every point of time, when you look at the map, I have the position of these objects clearly marked in the map, even though when they are moving around. And now I have to just accommodate my next estimate will have to accommodate the current version of the map and not the previous version of the map.

So, I am going to assume that in this case, we can just assume that your localization can operate independent of the mapping problem and the updation of the map itself is done by an independent estimation process. So, there are multiple ways in which you can handle this dynamic environment.

(Refer Slide Time: 19:43)



Localization

Passive vs Active Approaches

(Pertains to whether or not the localization algorithm controls the motion of the robot)

- Passive localization: The localization module only observes the robot operating.

  The robot is controlled through some other means, and the robot's motion is not aimed at facilitating localization. (For example, the robot might move randomly or perform its everyday's tasks.)
- Active localization: Active localization algorithms control the robot so as to minimize the localization error and/or the costs arising from moving a poorly localized robot into a hazardous place.

Introduction to Robotics Prof. Balaraman Ravindran

So, the third dimension we wanted to talk about, it is what is called passive versus active. Passive versus active approach for localization, this essentially pertains to whether the localization algorithm itself controls the motion of the robot or not. So, in passive localization, the localization model only observes what the robot is doing. And based on that, just based on these passive observations, it tries to estimate where in the map the robot is.

So, the robot could be just moving around randomly waiting for some, some tasks to be assigned to it. Or it could be just going around trying to perform its everyday task, trying to maybe it is, it has to fetch some mail and deliver it or whatever it is supposed to be doing in office environment, or maybe in a factory shop environment, whatever is the task that is assigned to it, the robot is trying to execute that task.

So, the control itself, the movement of the robot itself, is not aimed at localization, it is not trying to localise itself actively. But the localization algorithm just has to run on the robot, observe all the actions the robot is taking and all the sensor information the robot is getting. And look at how to do the localization.

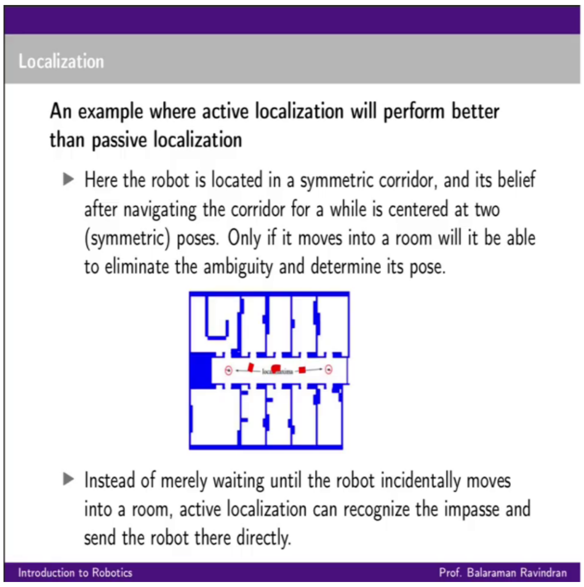
Again, in some sense, this is what we have been looking at so far, in the state estimation problem where the goal of the control was never to get the better state estimate. The goal of the control was something else, we do not know. We were always given the control. And we were passively observing, what was the sensor reading. And what was the control that was given in order to refine our belief state. So, that is what passive localization is.

Active localization, on the other hand, is very interesting, active localization says that, hey I am going to move my robot. So that I will find out where I am very quickly. I will not try to perform everyday tasks. Because if I am, if I do not know where I am, I could run into some kind of hazardous thing, I could run into an obstacle, or I could break the robot, I could fall off, you know, lead or something like that.

So, I really do not want to go about it, because the cost arising from the badly localised robot is very high. So, what I do is I first control my robot and actually move in such a way so that I will minimise the localization error. And again, this has to be this has to depend on the belief state that I am in and the sensor readings and getting, based on that, I am going to move in a way so that I minimise the localization error. And then I move to some kind of like position tracking, kind of a mode, and then try to execute my everyday task.

So, this is basically what we call it active localization, where at every point of time, you also try to make sure that your localization error is minimised. In some sense, this is how we operate, I mean, we just do not just start doing things without knowing where we are. If you are put in an unknown environment, the first thing we try to do is determine where we are before we decide what we are going to do next time. So, that is what basically the idea behind what active localization does.

(Refer Slide Time: 23:13)



Localization

An example where active localization will perform better than passive localization

- Here the robot is located in a symmetric corridor, and its belief after navigating the corridor for a while is centered at two (symmetric) poses. Only if it moves into a room will it be able to eliminate the ambiguity and determine its pose.

- Instead of merely waiting until the robot incidentally moves into a room, active localization can recognize the impasse and send the robot there directly.

Introduction to Robotics Prof. Balaraman Ravindran

So, here is an example where active localization will perform much better than a passive localization. So here is the, you know, a corridor. So, where most of the corridor looks symmetric, when I look from here, for example, I am going to see two doorways, and open space to the left and open space to the right. Correct and I am going to see two doorway. So, I really do not know whether I am here. Whether I am here or whether I am here, or whether I am here, because all of these places look similar.

Let me, let me take, lets I, over here, or here, or here, all these three places look similar, because there is a doorway to the top, there is a doorway to the bottom. And there is free space to the left, there is free space to the right. And whatever slight angles I look at, I will always see the same measurement.

So, if I, let us say I start in the middle of the corridor, so I do not know whether I am actually in the middle of the corridor or not, because I could either be here or here. So, what I should do is move to one of these two places, one of these two places marked by a circle, because if I come to this circle, I know for sure because there is an obstacle to the.

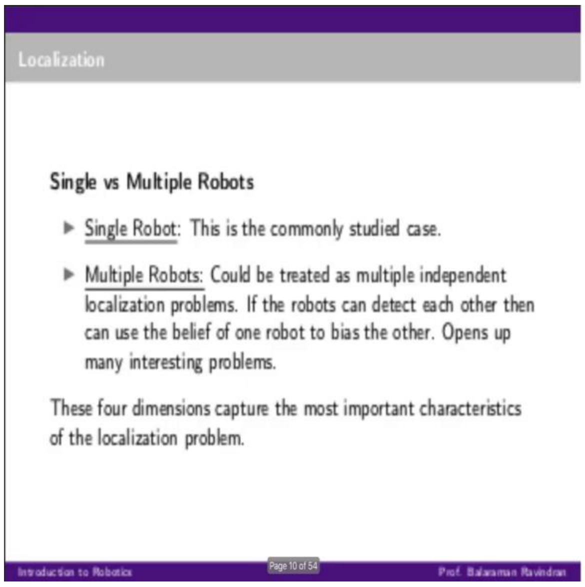
So, therefore I know that I am at the right end of the corridor, or if I come here, then I know I am at the left end of the corridor, and then from here onwards, I can just start, you know, moving to whichever room I want, I will know for sure, otherwise I will always have this ambiguity as to which room I am in.

So, this way this allows me these two locations allows me to disambiguate where I am in terms of the symmetric corridor. So, even though the rooms themselves are different, so once

I enter a room, I will probably know. But then that is, that is a waste, and assuming that there is some room one of these rooms is actually a dangerous room, I do not really want to go in there and I have my robot fry.

So, we would like to make sure that such things do not happen. So instead of just hoping that somehow the movement of the robot will allow me to localise myself active localization, that just basically tries to move the robot to one of these places, so that you can localise quickly, and then go ahead and do the rest of the job.

(Refer Slide Time: 25:51)



Localization

Single vs Multiple Robots

- Single Robot: This is the commonly studied case.
- Multiple Robots: Could be treated as multiple independent localization problems. If the robots can detect each other then can use the belief of one robot to bias the other. Opens up many interesting problems.

These four dimensions capture the most important characteristics of the localization problem.

Introduction to Robotics Page 10 of 54 Prof. Balaraman Ravindran

So, the last dimension that we would talk about is single versus multiple robots. So, the single robot case is the one that we have that is most commonly studied. And that thing that we have looked at also. But increasingly, it is becoming more common for people to consider using multiple robots system, because the space is large to cover and so you really like to have more than one robot.

In fact, there are some very nice use cases of robot, teams of robots, you know, taking visitors through museums where one robot hands off to another robot at various points. And therefore you get this robot guided tour of the museum. So, there are things like that that people have been working on. So, you could treat this multiple robot localization problem as multiple independent localization problems.

So, there is no need to actually consider this as a special case, just like if there are 10 robots, you have 10 single robot localization problems to solve. And you could do that. But what is interesting is, if I assume that the robots can detect one another, instead of just detecting the obstacles if the robot can detect another robot, there are a couple of things here, not only does the robot become another landmark, or another feature against which I can localise myself.

But that robot also has a belief state about where it is, I could potentially get the belief state from the other robot, say, hey come, not only so if you detect another robot, you basically communicate the belief states to one another. And therefore, now suddenly, you find that I have a huge update to my belief, because a, I know where the other robot is. And b, I know that what the belief of that other robot where that other robot believes it is.

Now, I know where I, this robot believes it is, and therefore I can combine both, and this opens up a lot of very interesting questions. So, what is the level at which the robots have to communicate to each other? You know, how do you accommodate the information of the robot? Can you ask the robot questions? Can you ask the robot question? Hey, I am trying to find this person, did you see him? A lot of interesting questions.

Opens up when I start talking about multiple robot localization, and again, multiple robots and active localization makes it even more interesting problem, so in some sense, these four dimensions, we talked about capture the most important characteristics of the localization problems, there are other ways in which you can think of, you know splitting the localization question, but these are the primary variations that we have to worry about.

Other kinds of variations could be depending on how noisy your sensors are, are they noise free, how many sensors do you have? How reliable are the sensors? What is the knowledge that you have off the map? What sort of a map do you have? So, all of these could potentially give rise to other interesting questions.