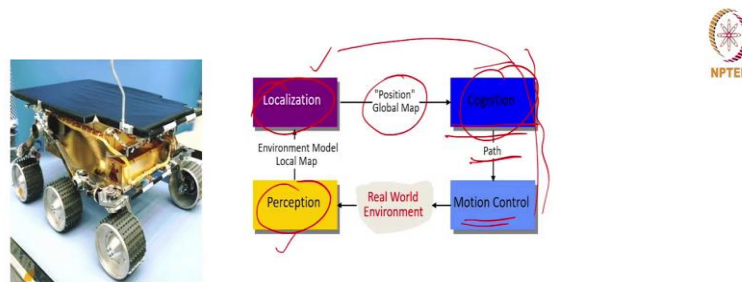


Wheeled Mobile Robots
Prof. Asokan Thondiyath
Department of Engineering design
Indian Institute of Technology, Madras

Lecture - 06
Mobile Robot Navigation

(Refer Slide Time: 00:13)



Chapter 6
Mobile Robot Navigation

Dr. T. Asokan
E-mail: asok@iitm.ac.in

1/11/2021



Hello everyone. Welcome back to the discussion on Mobile Robotics. Today we will start a new chapter which is basically on the Mobile Robot Navigation. So, in the last few lectures we talked about the localization. We discussed various challenges in the localization and with all those uncertainties and limitations, how the robot can actually localize itself in a given environment; whether the map is given or not the robot will be able to localize within the environment. So, that is what we discussed.

Now, as we discussed earlier in the autonomous navigation of robots, the four steps involved are basically the perception, localization and then comes the cognition and motion control. So, we actually completed this localization part, we talked about the perception how the robot will get the perception of its environment using sensors and various sensors used. Then, we talked about how the robot can do the localization, basically using odometry and the map given to the robot or map generated by the robot.

So, once the robot is clear about its location and it is already been given an instruction to go to a particular target. The next task is basically, within the known position with the

known position of the robot in the map, how the robot will actually plan its path to reach its target.

So, this stage is basically known as the cognition stage, where the robot will utilize all the information it has including the information given by the user as well as the information collected by the sensors or through the sensors, the robot will start using this information and then, decide on what is the next course of action or how it should actually execute or plan for its next course of action. So, that is basically using the information and then making decision is basically the cognition stage.

So, the cognition stage will help the robot to decide a path which it need to follow in order to reach its target. So, that is basically the path will be generated. So, this stage of generating a path and then following the path to reach its target is basically known as the mobile robot navigation.

So, the robot need to navigate from its current location to the target location using the information it collected and the information it has based about its location and then, use all this information and then plan a path and then reach the goal ok.

So, once you plan path is planned, then basically the motion control will help the robot to follow that path and then reach its target. So, this is the steps involved in autonomous version of a robot. So, today we will look into the mobile robot navigation and what are the various steps involved in navigating a robot from a known location to a target location.

(Refer Slide Time: 03:32)

Lecture 6.1



Navigation



So, the first part we will look at the navigation. Today, in the first lecture we will look at the navigation and various steps involved in navigation and as we progress we look into more aspects of navigation of the robot ok.

(Refer Slide Time: 03:45)

Navigation



Given partial knowledge about its environment and a goal position or series of positions, navigation encompasses the ability of the robot to act based on its knowledge and sensor values so as to reach its goal positions as efficiently and as reliably as possible.

Two Important Capabilities for Navigation are:

Path Planning: Identifying a trajectory that will cause the robot to reach the goal location when executed.

Obstacle Avoidance: Given real-time sensor readings, modulating the trajectory of the robot in order to avoid collisions.



So, now the navigation is defined as given a partial knowledge about its environment and a goal position or a series of positions, the navigation encompasses the ability of the robot to act based on its knowledge and sensor values so as to reach its goal position as efficiently and as reliably as possible. So, that is basically the navigation of a robot.

So, it knows the information about its current position and it has got sensors to collect information and with all these how the robot will actually move from its current location to and reach its goal position as efficiently and as reliably as possible is basically known as the navigation. And this involves two important capabilities for robots. The first one is we call this as the path planning and the second one is obstacle avoidance.

So, if the robot has to navigate first, it needs to plan a path from current location to the target location or to the target locations or via points and that is basically, the planning of the path. So, it will be identifying a trajectory that will cause the robot to reach the goal location when executed. So, that is basically the path planning.

The next one is as the robot moves on its path, it has to avoid obstacles also. So, that is basically the obstacle avoidance. Given a real-time sensor reading, how to modulate the trajectory of the robot in order to avoid collisions. That is basically the obstacle avoidance.

So, these are the two capabilities for the robots to be autonomous that is it needs to have a capability to plan the paths and then, avoid the path avoid the obstacles as the robot moves along the decide path. So, these are the two capabilities the robot need to have in order to do the navigation.

So, when we talk about the path planning, then there is a term trajectory. So, path planning and trajectory planning are slightly different. Basically, path planning will talk only about the paths the points x y θ for the robot; that is basically the path points. The trajectory basically talks about the time at which the robot reached this particular point.


So, trajectory is basically a path information along with its temporal information is basically known as the trajectory, because in the practical scenario you want the robot to reach a particular location at a particular time or in a particular duration of time. So, that is basically the trajectory, otherwise it is only there is no time information is not there, then we call it as a path otherwise, we called the if the time information we called it as the trajectory.

So, effectively the robot finally need to have the trajectory. So, that it will reach the target at a particular time or the decide time. And as it moves, there may be obstacles coming which was not anticipated or which was not there in the map already available.

In that case, the robot needs to re-plan its trajectory or modulate the trajectory and then avoid the obstacle and reach the target. So, that basically the obstacle avoidance capability.

So, these two capabilities need to be built into the robot in order to have an autonomous navigation of the robots ok. So, we will look at the path planning initially. And then, we talk about obstacle avoidance also.


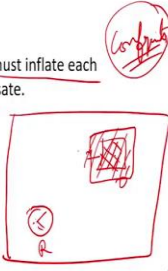
(Refer Slide Time: 07:24)



Path Planning: Assumptions

There are several assumptions made to simplify the process:

- **Robot is holonomic:**
This means the total number of controllable degrees of freedom of robot is equal to total number of degrees of freedom.
Differential-drive robots can rotate in place, and so a holonomic path can be easily mimicked if the rotational position of the robot is not critical.
- **Robot is simply a point:**
Because we have reduced the robot to a point, we must inflate each obstacle by the size of the robot's radius to compensate.



So, when we do this planning of path for a robot, we make few assumptions. Basically, to make our life easier or without too much of complications in the planning stage. We assume that the robot is basically a holonomic one. So, holonomic robot is that the robot can actually move the way it wants. So, it has got enough degrees of freedom enough controllable degrees of freedom as the number of degrees of freedom of the robots.

So, the total number of controllable degrees of freedom is equal to the total number of degrees of freedom. So, you know that any object will be having its own degrees of freedom, when the ground robot has got its own degrees of freedom, but an aerial robot will be having a different number of degrees of freedom.

If you can control all those degrees of freedom, then we call it as a holonomic robot. So, the robot will be considered as holonomic, as long as we can control all the degrees

of freedom that is nothing is independent I mean nothing is dependent on the other degrees of freedom. All the degrees of freedom are independent and we can control.

So, that kind of an assumption is basically the assumption that the robot is holonomic; that means, the robot can take a turn at its own location, it can move on sideways wherever whenever it needs to move in sideways or a lateral motion is needed the robot will be able to do that is basically the assumption of holonomic

Then another assumption is that it is simply a point. So, we assume that the robot is a point its size is not critical, we assume it is a point so that, we can actually plan the path without worrying about the size of the robots. This again a simplification of the actual scenario, but we assume that we the robot is a point.

And in order to take care of that assumption what we do is we will enlarge the size of the obstacles in its environment with the size of the robots; that is, we inflate each obstacle by the size of the robot's radius to compensate. So, what does it mean is that, if you have a scenario like this and suppose, there is a an obstacle here, and we have a robot of this size this is the robot; this is the robot and this is the obstacle.

Now, we assume that the robot size is 0; I mean there is no it is a point. So, there is no actual radius for the robot and to take care of that simplification what we do is, we will assume that the robot this obstacle size is bigger than the actual obstacle size. So, we will actually add the radius of the obstacle the robot to the size of the obstacle and we will assume that size of the obstacle is this.

So, we are actually inflating the size of the obstacle in order to take care of the size of the robot. So, effectively though we are assuming it is a point, we ensure that the robot will not really go and hit the obstacle, because of its size, because obstacle size itself is enlarged.

So, this one we call this as the configuration space where we configure the obstacles with the size of the robot. So, we called this the configuration space of the robot. So, actual space will be having a smaller obstacle, but in the configuration space we assume that the obstacle sizes are enlarged in order to take care of the robot size. So, that is basically the two assumptions that we make in planning for the path planning of the robots.

(Refer Slide Time: 11:26)

Path Planning: Assumptions



There are several assumptions made to simplify the process:

- **Robot is holonomic:**
This means the total number of controllable degrees of freedom of robot is equal to total number of degrees of freedom.
Differential-drive robots can rotate in place, and so a holonomic path can be easily mimicked if the rotational position of the robot is not critical.
- **Robot is simply a point:**
Because we have reduced the robot to a point, we must inflate each obstacle by the size of the robot's radius to compensate.

Path Planning Methods:

1. Graph Search Method (Road Maps)
2. Potential Field Planning



Now, as I mentioned; there are multiple methods for path planning and the two most important one ok. Basically, the classical methods there are many other new methods are being proposed by researchers, but we will talk only about the classical methods. So, the 1st one is known as the graph search methods. The 2nd one is the potential field planning. So, both are for path planning.

In graph search methods; what we do is, we will create a graph of the environment with our possible robot paths and then find out which is the shortest path. In potential field path planning, we do not go for a construction of a graph or anything. We assume that the obstacles and robot has got some properties and there is an attraction and repulsion.

And that actually represented using a potential field and then we look at this field and then decide which path will be having the lowest potential and then go along with that path. So, that is basically the potential field method. We will look into these two in detail. So, the first one is basically known as the graph search methods or sometimes it is known as road maps also.

Since primarily as you know. Now, if you use Google map or any other such algorithms, you can actually plan your route from one place to the other place in the map. So, if you want to go from your institute to a market or some other unknown location, you can search in the Google map and then Google map will show you multiple paths; that is basically it will show you all the possible ways in which you can reach.

So, that is basically it will create a roadmap to reach the location. There will be multiple you will see that it is not the single one there may be multiple of paths it will be giving you. So, that is the graph and then, within that multiple paths you can search or you can decide which one to take based on the distance to be covered or the time to be taken or there is a its a highway or not or there is a toll or not. So, there are different criteria you can use.

The same thing what we do here also. We try to find out all the possible paths and then see which path is the best for your application. So, there is there are two stages involved in graph search method. So, we will first look at the graph search method and then we will go to the potential field methods ok.

(Refer Slide Time: 14:02)



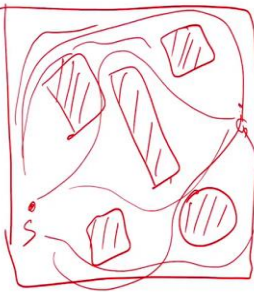
Graph Search

In graph search, a connectivity graph in free space is first constructed and then searched.

The graph construction process is often performed offline.

Involves two main steps:

- Graph construction**, where nodes are placed and connected via edges
- Graph search**, where the computation of an (optimal) solution is performed.



So, what we do in graph search is a connectivity graph in free space is first constructed ok. So, we will try to construct a connectivity graph in the free space and then, we search in that graph the best path ok. So, therefore, it will be normally done offline; that means, you have the all the information about the environment, the size of the obstacles and the location of the obstacles all are given and you know the initial position and the target position, then what we do?

We will try to find out all the connectivity between the starting point and the goal point. And then once you have this connectivity graph, you search within the graph. So, that is the procedure using graph search; that is if you have a map already given, because it is

an offline method and then, this is your start point and this may be your goal point and you will see that there are obstacles on its paths.

So, assume that these are all the obstacles and the path, so this on the configuration space. So, we assume that the robot is a point and it can actually move. Now, what we need to know is what are the various paths existing between the start point and goal point.

So, I can actually identify a path like this I can go like this I can go like this. So, there are multiple ways in which you can actually reach. So, all the potential connections from start to goal will be the graph. So, once you have all these then, we call this as the graph. So, we create a graph we call this is graph construction.

And then, this graph within this graph you do the search and then, check which one is the shortest or the less time taken or less cost of travel. So, that is basically how you do this. So, you create a graph first. So, construct the graph first and then, within this graph you search for the shortest path or the most feasible paths; that is the graph search methods.

And as you can see here, we need to have some kind of a formal procedure to do this, otherwise, you do not know whether to go like this or like this or what way we actually decide which path to be taken. So, we create some kind of nodes and then adjust to connect the start to goal position.

So, that is the most basic way of doing it. We need to connect create some kind of nodes and edges. So, that it can actually pass through these edges to reach the goal point. So, again there are different methods proposed in the literature to create the graph. So, first we need to have the graph created and then do the search.

(Refer Slide Time: 17:21)

Graph Search



In graph search, a connectivity graph in free space is first constructed and then searched.

The graph construction process is often performed offline.

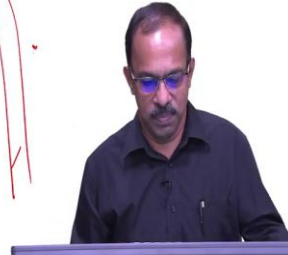
Involves two main steps:

Graph construction, where nodes are placed and connected via edges

Graph search, where the computation of an (optimal) solution is performed.

Graph Construction

- **Visibility graph**: Path come as close as possible to obstacles and resulting optimal paths are minimum-length solutions.
- **Voronoi diagram**: Path stay as far away as possible from obstacles.
- **Cell decomposition methods**: idea is to discriminate between free and occupied geometric areas.



So, in the graph search method, we will have multiple methods to create the graph and to identify all the potential paths existing ok. So, some of the commonly used methods are known as visibility graph, Voronoi diagram and cell decomposition methods. As I mentioned, all these are offline methods. So, we do not do it online. We do not do it as the robot moves. We will plan everything offline and then give the path to the robots.

And each one has got its own advantages and disadvantages; as saying you know visibility graph comes as close as possible to the obstacles and resulting optimal paths are minimal length solutions. So, we can actually get a minimum length solution as the robot will actually go very close to the obstacles and reach the target without hitting the obstacle.

But in the other case, Voronoi diagram; it will try to keep as much away as possible from the obstacle. So, this path need not be the shortest path, but it can be a path where actually the safety will be much more convey to the visibility graph.

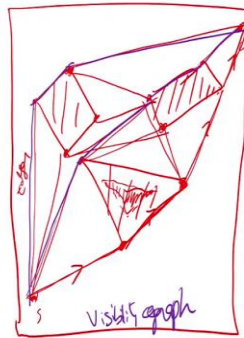
And then there is another method called cell decomposition; where actually it will try to distinguish between free and occupied sense and then try to identify all the free cells through which the robot can pass and then find a then create a graph and then find out the best feasible path methods best feasible path. So, let us look at the visibility graph first and then we go to once you understand visibility graph then, other methods will be much more easier to understand.

(Refer Slide Time: 19:11)

Visibility Graph

Concept:

The visibility graph for a polygonal configuration space consists of edges joining all pairs of vertices that can see each other (including both the initial and goal positions as vertices as well).



So, visibility graph for a polygonal configuration space; that means we assume that the configuration space has got all polygonal obstacles. So, the obstacles can actually be represented as polygons. And then edges joining all pairs of vertices that can see each other including both the initial and goal positions as vertices as well. So, what it will do? It will consider this as a polygonal configuration space. So, configuration space is where the obstacle size is large.

Now, this is the start point and this is the goal point. So, we consume this as a node or a we consider this as a vertex and then we know that there are obstacles. So, we assume that these are all the obstacles. There are configuration space obstacles. So, the size has been increased. Now, what it will do? It will connect using edges all pairs of vertices that can see each other ok.

So, now, this can actually see this vertex. So, I connect using the edges. So, I can see all these from here, so, I will connect it. Then, this can see here it can see here it can see here it can see this. Similarly, this can also see this can see this can see. There are always there are multiple ways in which it will see. So, it can actually see this one similarly this also can see here.

So, now I will connect these are all the edges these are all the edges connecting the vertices it is which can each see each other. These are the vertices, ok. So, it can all see. So, these are all the edges that can connect. Now, we can see that this robot can actually

start from here, go up to here and then pass along this and then, go along this and then, go from here to reach this position.

So, this may be a potential path. Since the actual obstacle is smaller than this, so, even if the robot is moving along this edge, it will not hit the obstacle, because there is already the actual obstacle may be only this much size. So, that is the assumption here. And therefore, the robot can actually take a path from here to here from here then here and then reach here or it can go like this then pass here and then go ahead with this and then reach here. So, like this there are multiple option for the robot to move ok.

So, these are all the potential option for the robot to move from here and here or here, here, here. So, this is basically known as the visibility graph. So, all the visible all the visible edges as well vertex are connected using edges and then we get a visibility graph. So, we have multiple paths possible. So, that is for the first phase stage of graph construction.

So, we have constructed a graph which shows multiple paths for the robot to go without hitting the obstacle and reaching the target. So, that is the first stage in visibility graph. It is a polygonal configuration space consisting of edges joining all pairs of vertices that can see each other including the starting and goal points ok. So, that is the thing.

(Refer Slide Time: 23:13)

Visibility Graph

Concept:

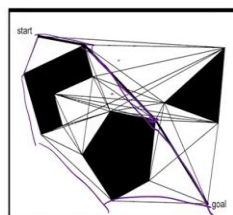
The visibility graph for a polygonal configuration space consists of edges joining all pairs of vertices that can see each other (including both the initial and goal positions as vertices as well).

The unobstructed straight lines (roads) joining those vertices are obviously the shortest distances between them. The task of the path planner is then to find a (shortest) path from the initial position to the goal position along the roads defined by the visibility graph.

Notes:

Size of the representation and the number of edges and nodes increase with the number of obstacle polygons.

Solution paths found by graph search tend to take the robot as close as possible to obstacles on the way to the goal.



Now, the unobstructed straight lines joining those vertices are obviously, the shortest distance between them. The task of the path planning planner is then, to find out the shortest paths from the initial position to the goal position. So, that is the second stage. First stage is to construct the visibility graph.

So, once a graph is constructed then, you can search within that graph to find out the shortest path or whatever the criteria you set you will be able to find out the paths ok. This is the one which I already explained. So, you can see there are multiple paths possible like this, like this.

So, one of these may be the shortest, but we need to find out which one is the shortest and then, that can be done using a again a search algorithm. So, first one is creation of the graph. And it is a simple process and which can be done offline. And then once you get the offline data, you can actually feed that coordinate points to the robot and the robot will be able to go along that path.

But the difficulty is that with the number of obstacles, as the number of obstacle increases, you will be having a lot much more complex graph. It may find bit difficult to handle it. Solution paths found by the graph search tend to take the robot as close as possible to obstacles on the way to the goal.

So, that is another disadvantage here, because as you can see here the robot may be going very close to the. For example, in this path you can see it is actually very close to the obstacle though it is the size is enlarged, it may be still close. If there is a small error in the robots position, there is a high possibility that the robot may hit the obstacle.

So, we need to take enough precautions to avoid that situation or either you increase the obstacle size more than the size of the robot or give some clearance in all the paths to be away from the obstacles. So, these are the practical issues we need to understand, when we do the visibility graph method of path plan.

(Refer Slide Time: 25:30)

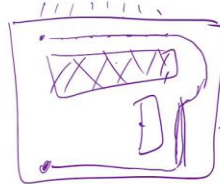
Voronoi Diagram



Concept:

For each point in free space, its distance to the nearest obstacle is computed. At points that are equidistant from two or more obstacles, such a distance plot has sharp ridges. The Voronoi diagram consists of the edges formed by these sharp ridge points.

Since its edges maximize the distance to obstacles, any short-range sensor on the robot will be in danger of failing to sense its surroundings.



So, to avoid this problem of the robot going so close to the obstacle, there is another method called Voronoi diagram methods. So, in the Voronoi diagram, the robot will be moving as much away as possible from the obstacles. So, that is basically the method of Voronoi diagram. So, what it will do?.

For each point in free space, its distance to the nearest obstacle is computed and that points that are equidistant from two or more obstacles, that such a distance plot has sharp ridges. The Voronoi diagram consists of the edges formed by these sharp ridge points.

For example, suppose this is the area and the robot is initially here and there is a an obstacle like this and what it will do is, it will try to find out what is the space in this the point which is equidistant from this one. This is again a boundary so, that is a consider an obstacle and between these two.

So, it will try to find out a path which is equidistant from these two. So, it will try to find a the points which are equidistant from these two and take that as the possible path from here to here and that is basically, the Voronoi diagram methods. So, once you can identify all those depending on the obstacle and the boundaries, you will be able to get all the path, which are equidistant from the obstacles.

So, in this case, it will be like this equidistant from this one and this one and then, it may reach here the target. These are the possible ways in which you can think off are the both possible paths which are equidistant from obstacles is basically the Voronoi diagram. Of

course, it has got its own limitation because the sensors are when you use the sensors to identify the obstacles.

So, the sometimes it maybe, it may not be able to identify the obstacle far away. It might not be knowing that there is an obstacle at that location. That is more in the online method, but offline method since everything is known, you will be able to create a good Voronoi diagram and then use it for path planning.

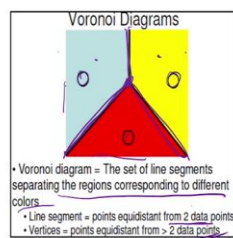
(Refer Slide Time: 27:57)

Voronoi Diagram

Concept:

For each point in free space, its distance to the nearest obstacle is computed. At points that are equidistant from two or more obstacles, such a distance plot has sharp ridges. The Voronoi diagram consists of the edges formed by these sharp ridge points.

Since its edges maximize the distance to obstacles, any short-range sensor on the robot will be in danger of failing to sense its surroundings.

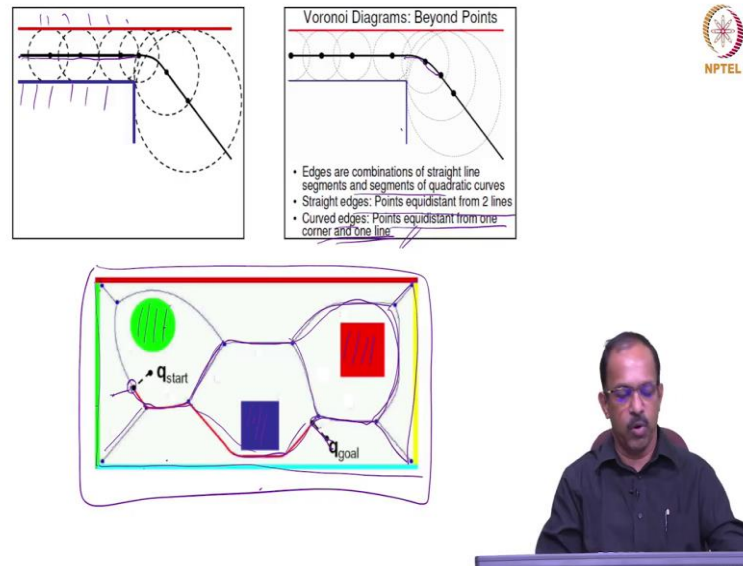


So, this is again a example of this Voronoi method. So, you can see that if there are there is an obstacle here and there is an obstacle here, it will identify this as the shortest paths. And similarly, this can actually be, because now, there is an obstacle here sorry there is a boundary here and here. So, it will identify this as the paths. So, this way you will be able to identify the short paths ok.

The set of line segments separating the regions corresponding to different colors. So, lines this is in this case it is different colors. So, each color will be separate. So, you will be able to get this as the possible path for the robot equidistant from all the place. So, you will be getting a line segment, when you have from two data points. So, if you want to take it from this one and this one then, it will be a straight line, but you have three conditions.

So, now, this one, this one and this one then, this becomes a vertex as the point which is equidistant from all the 3. So, but this is points equidistant from more than two data points and only two data points you will get a line segment. So, this principle can be used in the construction of Voronoi diagram and then you will be able to find out all the possible paths. So, we take an example and then show you how this is done.

(Refer Slide Time: 29:14)



So, for example, now if the robot is starting from here and you will see that there is an obstacle here and there is an obstacle here. So, between these two data points, you will be having a straight line ok. And then you will be having from this edge and then this one. So, that will be a curve. Then again between this one and this one, you will be having a straight line.

So, this is the way how you can actually create. Edges are combination of straight line segments and segments of quadratic curves and straight edges points equidistant from 2 lines and curve edges points equidistant from one corner and one line. So, one corner and one line will be a curve two data sets points will be a straight edge and then, this edges will be a combination of the like straight line and the curves curved edges.

Because, you may be having situations, where, it is equidistant from one corner and one line. So, in that situation you will be getting a curve also. So, primarily you will be having these edges as a combination of straight lines and curved edges curves. So, now

look at this situation. So, you have a area I mean a space configuration space with these are all the obstacles. And of course, it will consider these boundaries also as obstacles.

Now, it will try to find out the all the possible paths and if it is this is a q, then it will try to find out what is the path possible. So, from between these two, you have a straight line and between these three, you will be having a edge sorry a point. And then, you will be between these two, you will be getting this as the line and then, you have the line between these two and then, you have this line between these two.

So, like this you will be trying to find out a path which is equidistant from all the obstacles that is basically the Voronoi diagram. Now, you can see you can find out all these possible paths. Now, depending on where the robot is, if you even the robot is somewhere here, you can find out a; you can find out a path between these two and then, try to find out you can go straight from here and then, reach this point and then get into this path and then reach the goal.

So, this is the way how we can actually get the possible paths. So, you will be able to make multiple paths. Now, as you can see here there will be multiple paths, the robot can actually go like this and then reach here also. Again, the distance will be more, but then, the first task is to find out all the possible paths for the robot to travel. Now, we will be getting this as the potential path; that is basically the Voronoi diagram method of planning.

So, the advantage is that, it will be far away from the obstacle as much as possible from the obstacle. So, it will be far away from the obstacle. So, you can see here, it is nowhere close to this. So, it will be going far away. So, that is the advantage of using a Voronoi diagram method of path plan ok.

(Refer Slide Time: 32:31)

Exact Cell Decomposition



Exact cell decomposition is a lossless decomposition, whereas *approximate cell decomposition* represents an approximation of the original map. A graph is then formed through a specified connectivity relation between cells.



The other method is exact cell decomposition. We will discuss this in the next lecture.

Thank you.