Robot Motion Planning

Prof. Ashish Dutta
Department of Mechanical Engineering
Indian Institute of Technology – Kanpur

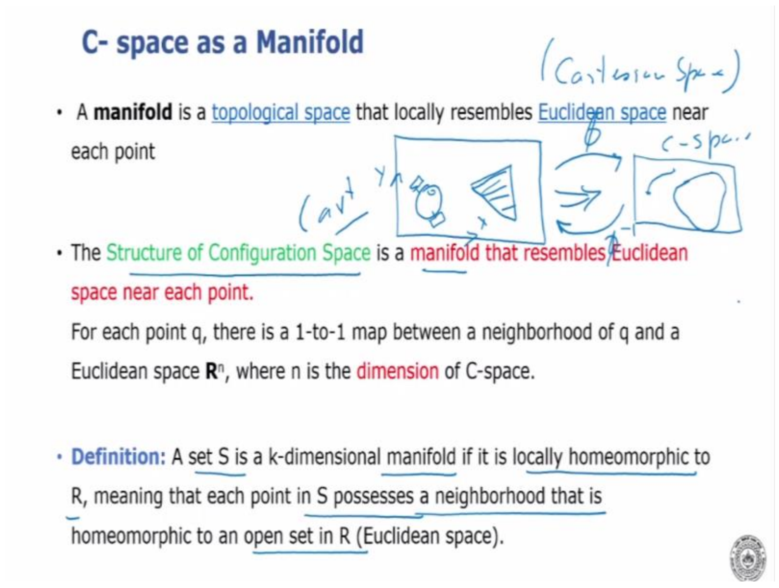
Lecture – 10
Road Map Methods

Hello and welcome to this lecture number 10 in the course of Robot Motion Planning. In the last class we looked at the topological configuration space that when you go from one space we have seen that we go from the Cartesian space which is the real world in which the robot is to the configuration space find the path in the configuration space and then come back to the Cartesian space again.

So, this mapping from one space to another space is very important in robot motion planning and that is why we need to understand the topology of different spaces. So, today we will very quickly complete what we were doing in the last class and then we will continue with the next part which is the various methods of path planning to take the robot from an initial point to a final point.

So, in the last class we were looking at topology of configuration space and very quickly we will revise what we were doing in the last class and then we will move on to various methods of path planning and today we will look at road map methods.

(Refer Slide Time: 01:13)



C- space as a Manifold

- A **manifold** is a topological space that locally resembles Euclidean space near each point
- The Structure of Configuration Space is a manifold that resembles Euclidean space near each point.
  For each point q, there is a 1-to-1 map between a neighborhood of q and a Euclidean space $\mathbf{R}^n$, where n is the dimension of C-space.
- **Definition:** A set S is a k-dimensional manifold if it is locally homeomorphic to R, meaning that each point in S possesses a neighborhood that is homeomorphic to an open set in R (Euclidean space).

But before that let us very quickly complete what we were doing in the last class. So, very quick revision we said that the manifold is a topological space that locally resembles Euclidean space near each point. Now we have seen that for example we have a robot which is in the Cartesian space let us take the example of mobile robot which is in the space and this is my mobile robot and there is an obstacle here.

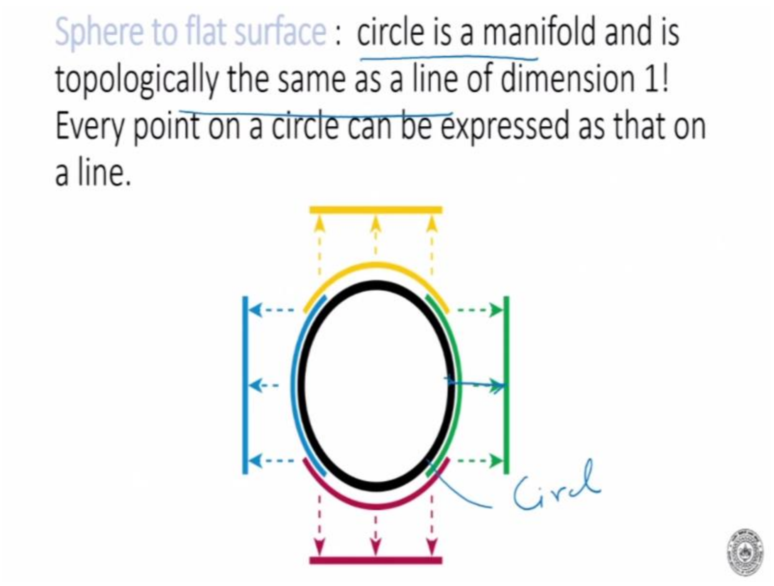
And what we do is this is my Cartesian space this is my X axis and that is my Y axis there is a mobile robot and this is my Y axis and this is an obstacle. So, what we do from the Cartesian space so this is Cartesian space we move to the C-space in which the workspace become smaller and the obstacle becomes larger let us say the obstacle has become larger like this.

And the robot becomes a point then what we do is this is my C-space. We find a path from the initial point to the goal point may be like this and then what we need to do is we need to come back to the real world again in which the robot moves. So, we have a forward transform which is given by Φ and we have inverse which is Φ^{-1} and this is the forward mapping and the inverse mapping between two spaces.

Now you can very well understand why this is very important because if you find the path in C-space and you cannot come back to the Cartesian space it is of no use. So, we need to very carefully understand this topology of different spaces. Now a manifold is a topological space that locally resembles Euclidean space near each point. Euclidean space or Cartesian space they mean the same thing here.

So, we say a manifold is a topological space that locally resembles Cartesian space. Now the structure of configuration space is a manifold that means it resembles Euclidean space near each point that means you can go from the configuration space to the Euclidean space. Now by definition a set k is a k-dimensional manifold it is locally homeomorphic to R meaning that each point in S possess a neighborhood that is from homeomorphic to an open set in R that means every point can be locally mapped on to the Euclidean space.

(Refer Slide Time: 03:30)

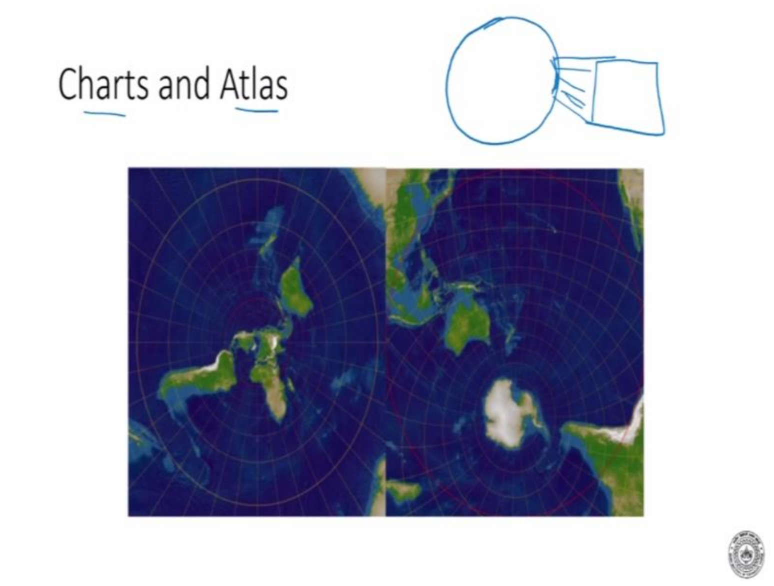


Sphere to flat surface : circle is a manifold and is topologically the same as a line of dimension 1! Every point on a circle can be expressed as that on a line.

Now this is an example a circle. Now a circle is a manifold and that is topologically the same as the line of dimension 1. So, in terms of topology a circle and a straight line is the same anyway in terms of topology and every point on the circle can be expressed as that on a line say for example when we look at this black curve this circle here. Every point here can be projected on to a straight line that means this is locally.

Hence, manifold is every point is locally can be mapped on to the Cartesian space. So, this is very important term here that a manifold is a topological space that locally resembles Euclidean space. So this term is very, very important here. So, if you locally look at this small region of a circle then each point here can be mapped on to a straight line.

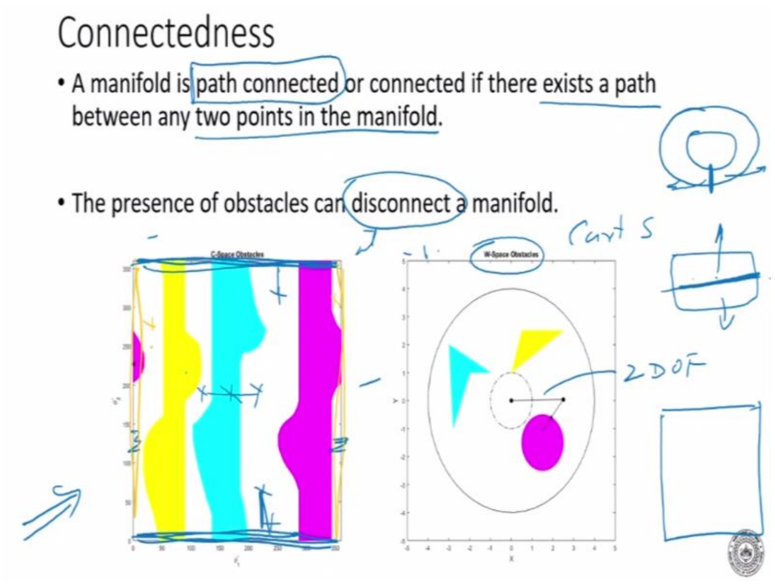
(Refer Slide Time: 04:21)



Charts and Atlas

Similarly, if you look at a sphere the Earth is a sphere it is spherical. Now how do you make maps? You locally map a local region to a flat map which is a flat surface. So, this is locally possible and that is why the term charts and atlas comes from. So, it is basically saying that a configuration space is a manifold and it is a manifold because it is locally like Cartesian space and that is why we can go from one space to another space.

(Refer Slide Time: 04:50)



Connectedness

- A manifold is path connected or connected if there exists a path between any two points in the manifold.
- The presence of obstacles can disconnect a manifold.

Now connectedness a manifold is path connected or connected if there exist a path between any two points in the manifold. So, like what we talked about in the last class we said that this is my C-space and this is my Cartesian space. So this is my Cartesian space and this is my C-space. Now in this case we have three obstacles the pink one, yellow one and the blue one. So, we have a two-link manipulator here, this is two degree of freedom.

And we can draw the C-space like this and in the C-space we have obstacles here so this is my obstacle, obstacle and obstacle. Now this is telling us a manifold is path connected or connected if there exist a path between any two points in the manifold. So, this point and this point there exist a path between these two points in the manifold and hence it is path connected.

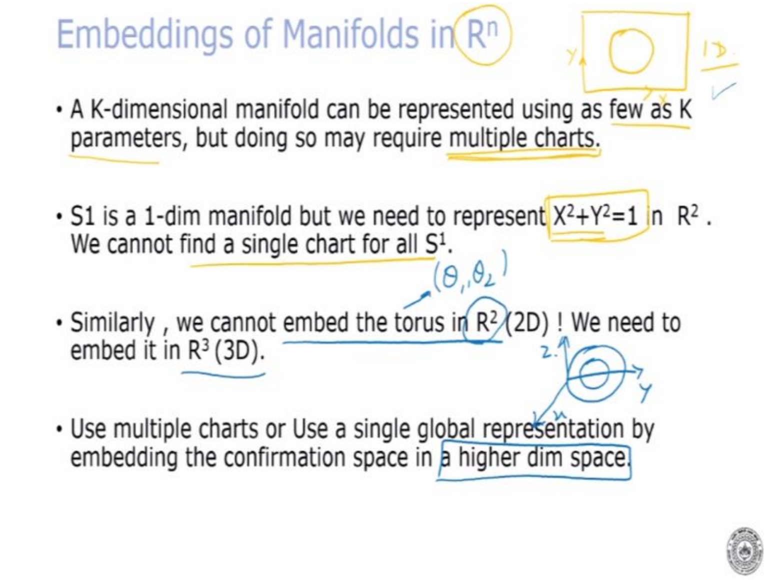
Now if I say this point and this point now there is no path from here and here it cannot go through an obstacle. So, these two are not path connected that is the meaning of path connected. Now why is this important? It is important because if there is no path between the two points then the robot cannot go in the real world between those two points. So, a manifold is path connected or connected if there exist a path between any two points in the manifold.

The presence of obstacles can disconnect a manifold as shown here. Now very clearly please remember that how did we get the manifold we got a torus we cut the torus like this opened it made it into a cylinder and then what we did we cut it further and opened it like this so I first cut it like this open like this then I cut along the center of the cylinder opened it and we made it like this and that is what is shown here.

So, it basically means that this side is connected through this side so you can have a path here and here it can go like this and you can come from here also it is also path connected and from this side and this side are path connected, but this and this are not path connected. So, please do remember that this side and this side are connected actually as far as the manifold is considered.

And in terms of the manifold, in terms of the torus this side and this side is also connected. So, this part and this part is path connected.

(Refer Slide Time: 07:16)



Embeddings of Manifolds in R^n

- A K-dimensional manifold can be represented using as few as K parameters, but doing so may require multiple charts.
- S1 is a 1-dim manifold but we need to represent $X^2+Y^2=1$ in R^2. We cannot find a single chart for all S^1.
- Similarly , we cannot embed the torus in R^2 (2D) ! We need to embed it in R^3 (3D).
- Use multiple charts or Use a single global representation by embedding the confirmation space in a higher dim space.

Now, embeddings of manifolds in R^n or in Euclidean space. Now a K-dimensional manifold can be represented using as few as K parameters, but doing so may require a multiple chart. So, the dimension of the manifold could be K, but you may not be able to represent. So, you may be able to represent using as few as K-parameters, but doing s may require multiple charts what does this mean let us take an example.

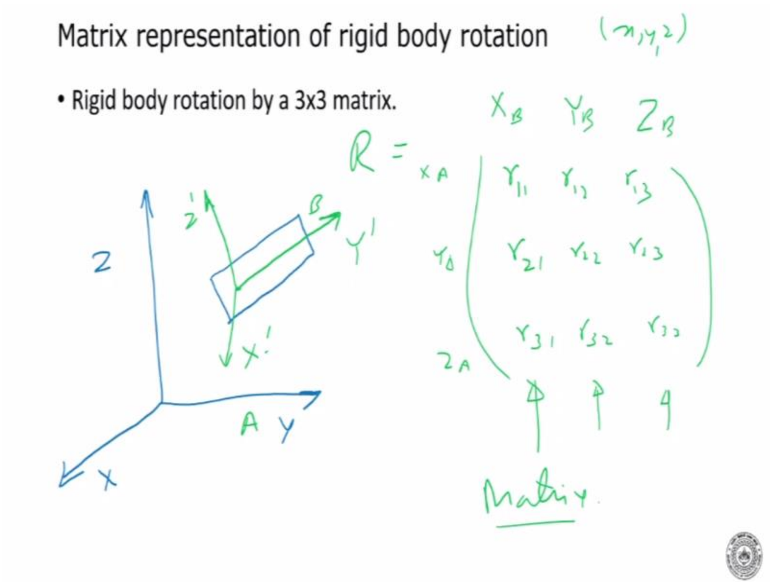
For example S 1 is a one dimensional manifold now as far as topology is considered a circle is a one dimensional manifold it is only 1-D, but to represent a circle you actually need two parameters X and Y. So, I am drawing it there is X axis there is Y axis. So, to represent this I used two parameters actually you can see that this is the equation of a circle. Although a circle is a one dimensional as far as topology is considered one dimensional manifold, but to represent it you need two parameters that is X and Y.

We cannot find the single chart for all S^1. Similarly, we cannot embedded the torus one of the example was the torus very good example is this torus. So, we cannot embed the torus in R^2 we know the torus is basically made up of θ_1 and θ_2 that is how we got the torus by varying θ_1 and θ_2 for a two degree of freedom system and when we draw the torus it is something like this, this is my torus.

So, this side is θ_1 and this one is θ_2, but when I am drawing it if you note that you cannot draw it in 2D it has to be drawn in 3D or torus can be drawn only in 3D. This is my X, this is my Y and this is my Z. So, similarly we cannot embed a torus in 2D that is R^2 we need to embed in 3D now. So, we can use multiple charts or use a single global representation by embedding the configuration space in the higher dimensional space.

First example was that of a circle which is 1D, but you require two parameters to represent it then a torus which is 2D in fact you require three parameters to represent it.

(Refer Slide Time: 09:30)



Matrix representation of rigid body rotation

- Rigid body rotation by a 3x3 matrix.

(x, y, z)

$$R = \begin{array}{c|ccc} & X_B & Y_B & Z_B \\ \hline x_A & r_{11} & r_{12} & r_{13} \\ y_A & r_{21} & r_{22} & r_{23} \\ z_A & r_{31} & r_{32} & r_{33} \end{array}$$

Matrix

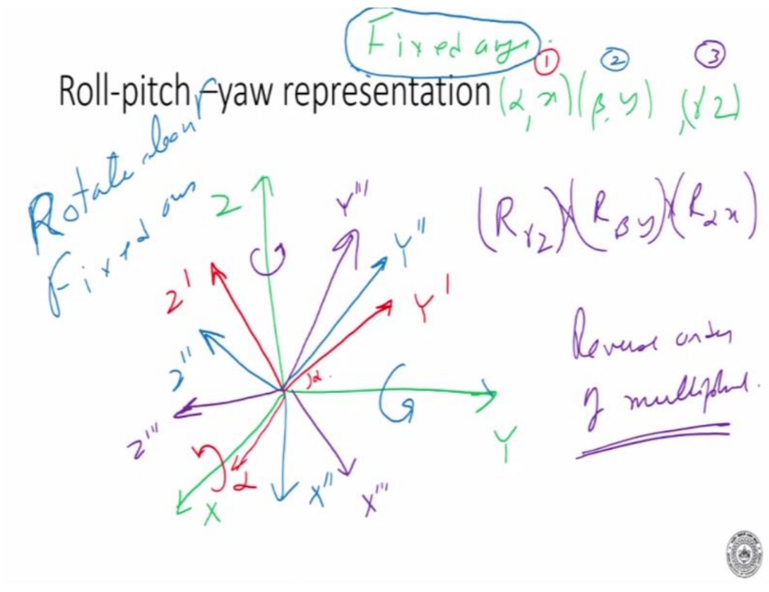
And what about the rotation of a rigid body. Now to represent the rotation of a rigid body you need a 3 X 3 matrix so what you have seen is that if you have a rigid body like this and we have an axis X, Y and Z and we have a rigid body, how do I represent the rigid body the angles of a rigid body you fix another X is there X, Y and this is Z sorry let me follow the same convention.

This is X', Y' and Z' and then what we do is we find the angles the X axis is making with the X, Y, Z of A this is A and that is B and then we write it as r $_{11}$, r $_{12}$, r $_{13}$, r $_{21}$, r $_{22}$ and r $_{23}$ and here it is r $_{31}$, r $_{32}$, r $_{33}$. Now what is this? This is the X of B on the X of A, Y of A and Z of A that is the meaning of this column and this is the Y of B on projected on to the X,Y, Z of A frame and this is the Z of B we project it on to the X, Y, Z of A frame.

So, what we are seeing here is that you can represent the position of a point by three coordinates X, Y, Z, but to represent the rotation of a rigid body and space you need how many parameters? You need a matrix so this is a matrix that we have seen. So, this we have seen earlier in the first class I guess first or second class when we talked about transformations.

Now this is showing that to represent a manifold you may require a higher dimensional space this is the meaning of the last statement here that you might need a higher dimensional space to represent the manifold.

(Refer Slide Time: 11:18)


Roll-pitch-yaw representation
Fixed axis
① (α, x)
② (β, y)
③ (γ, z)
Rotate about Fixed axis
$(R_{\gamma z})(R_{\beta y})(R_{\alpha x})$
Reverse order of multiplication.
Z
Z'
Z''
Z'''
Y
Y'
Y''
Y'''
X
X''
X'''
α

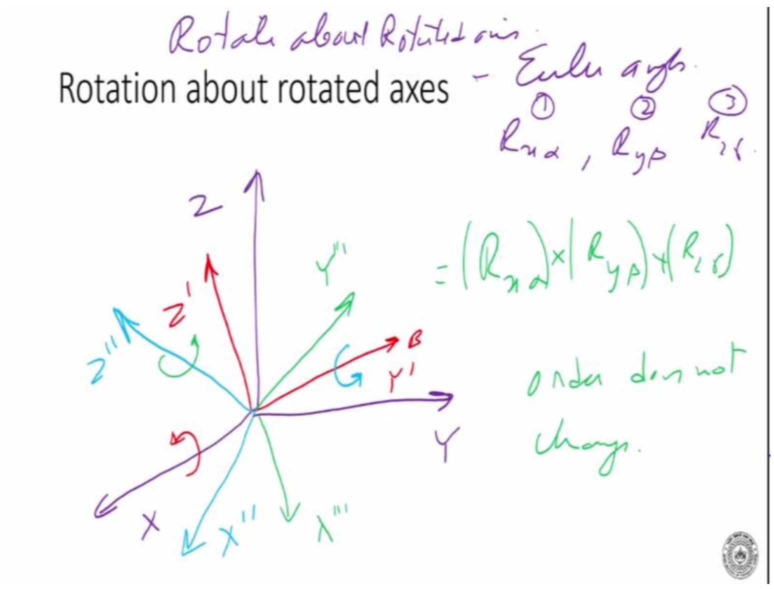
Now when we are representing the rotation or multiple rotations we can represent as roll-pitch-yaw. We can call this as rotation about the α about the X axis, Y axis and Z axis by angle α let us say, by an angle β and by an angle γ. So, I have two axes now and they are being rotated about a fixed angle. So, let us say this is my first fixed axis X, Y and Z and we have another axes which is the red colour which is going to be rotated.

So, this is my axis B. So, first what I do is I rotate about the X axis by an angle α. So, what will happen is B will come here now so Y' and Z' just come here this is an angle α then what I do is I rotate about let me change the colour I rotate by an angle β about the Y axis. Now which Y axis this is called fixed angle rotation we are rotating so rotate about fixed axis.

Rotating about fixed axis so I am rotating about the Y axis which is here so what will happen my X'' will come here Y'' will go there and Z'' has gone there then number three is what number three is the rotation of γ about Z axis which Z this Z. So, what happens it goes here, here and here that is my X''', Y''' and Z'''.

Now when I make the combined rotation matrix I am going to $(R_{\gamma z}) \times (R_{\beta y}) \times (R_{\alpha x})$. So, I reverse the order so we reverse order of multiplication please note this. So, this is called the fixed angle rotation also called the roll-pitch-yaw rotation.

(Refer Slide Time: 13:38)



Rotation about rotated axes

Rotate about Rotated axis
- Euler angles
① ② ③
$R_{x\alpha}$, $R_{y\beta}$ $R_{z\gamma}$
$= (R_{x\alpha}) \times (R_{y\beta}) \times (R_{z\gamma})$
order does not change.
Z
Z'
Z''
Y''
B
Y'
Y
X
X''
X'''

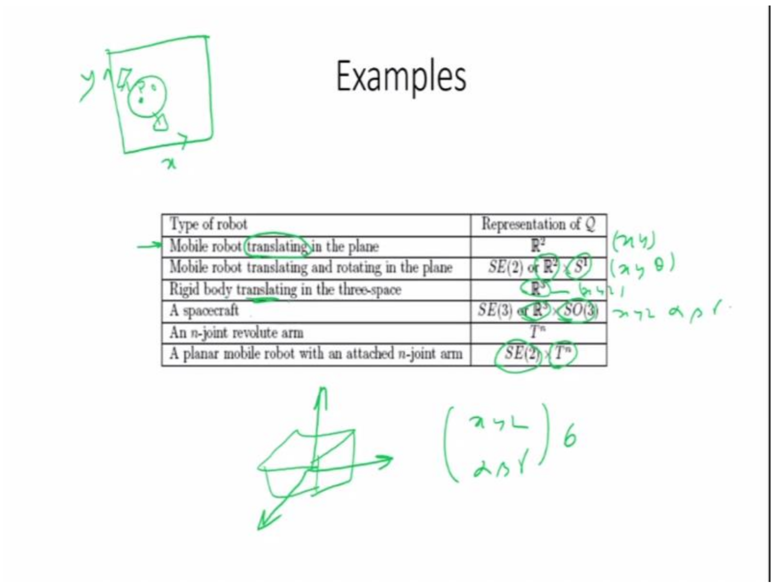
Now you can also rotate about the rotated system rotation of the rotated axes. These are some terms for Euler angles. Now in this particular case we are rotating about the rotated system that means the first axis is always going to remain the same. So, $R_{x\,\alpha}$ then we are going to rotate the $R_{y\,\beta}$ and $R_{z\,\gamma}$. So it is my first, second, third that rotate about rotated axes. So, this is my X, Y and Z.

Now, if I take my first rotation I rotate about the X axis like this so this fellow goes there towards my B, Y'so the first one is the same why because we are rotating about the fixed axes only because there is no other option initially they were the same. The second one now what happens is I am rotating about Y axis now. So, what will happen is Z will go there and X will come here and the third one I am rotating about the rotated Z axis.

So, the rotated Z axis is this one so I am rotating like this now. So, what will happen is this will come here and that will go there X''' and Y''', but in this case the order of multiplication remains the same. So, please remember that here order does not change. So, this is something you should note and why this happens because you have to be very careful about the order, but these are matrixes.

So, rotations do not commute in space and there is something you have to be very, very careful about.

(Refer Slide Time: 15:27)



Examples

Type of robot	Representation of Q
Mobile robot translating in the plane	$\mathbb{R}^2$
Mobile robot translating and rotating in the plane	$SE(2)$ or $\mathbb{R}^2 \times S^1$
Rigid body translating in the three-space	$\mathbb{R}^3$
A spacecraft	$SE(3)$ or $\mathbb{R}^3 \times SO(3)$
An n-joint revolute arm	T^n
A planar mobile robot with an attached n-joint arm	$SE(2) \times T^n$

y
x
(x y)
(x y θ)
x y z
x y z α β γ
$\begin{pmatrix} x\ y\ z \\ \alpha\ \beta\ \gamma \end{pmatrix}$ 6

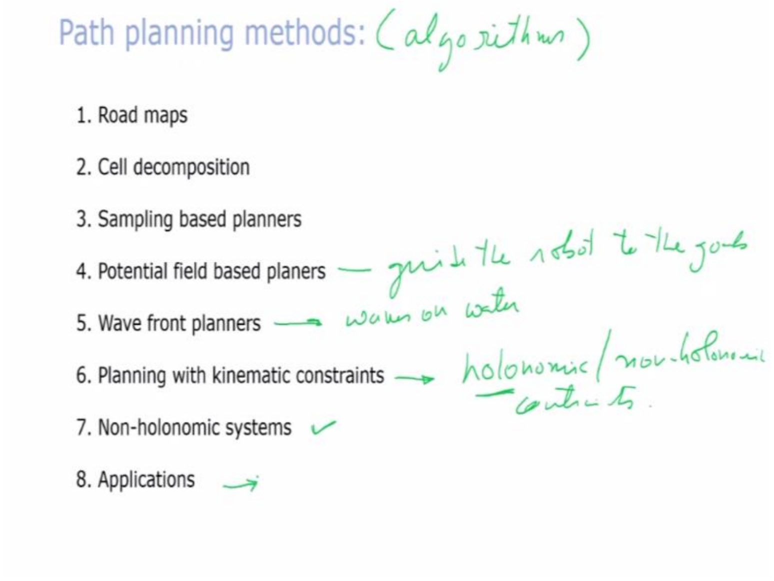
Now, these are some examples of robots and the degrees of freedom. So, let us look at the mobile robot translating in a plane. So, you have a plane which is x and y this is my x and this is my y and we have a mobile robot. So, there are two wheels here the Castor wheel is a point P. Now, it can only translate it is only translating in a plane. So, this point is translating it cannot rotate in that case degree of freedom is 2 so only x and y.

Now mobile robot translating and rotating in a plane so it is 3. So, x, y and θ so that is an R^2 which is x and y in Cartesian space and S is one rotation which is S. Now rigid body translating in 3D space so you have a rigid body which is translating in free space and this is also called a freely floating body. So, in this case how many degrees of freedom? It has three rotations 3 x, y, z and α, β, γ so this 6.

So, rigid body only translating is 3 a spacecraft has 6 so 3 rotations and 3 translations, 3 rotations. So, a rigid body is only translating then it has only three degrees of freedom my x, y, and z here it has x, y, z + α, β, γ. Now n-joint revolute arm it has T^n. A planar mobile robot with an attached n-joint arm has a mobile robot the planar mobile robot this is planar.

So, it has a SE (2) and plus the arm has T^n. So, this is basically giving you an example of the degrees of freedom of different kinds of robotic systems.

(Refer Slide Time: 17:25)



Path planning methods: (algorithms)

1. Road maps
2. Cell decomposition
3. Sampling based planners
4. Potential field based planers — guide the robot to the goal
5. Wave front planners → waves on water
6. Planning with kinematic constraints → holonomic / non-holonomic constraints.
7. Non-holonomic systems ✓
8. Applications →

Now, let us move on further from here and look at the path planning method and algorithms. As I said the whole objective of this course on motion planning or path planning is to write a program by an algorithm which is going to find the path that will take the robot from initial point to the goal point and to do that what we basically need is that we have to go to C-space the robot becomes a point and then we find the path and then it comes back.

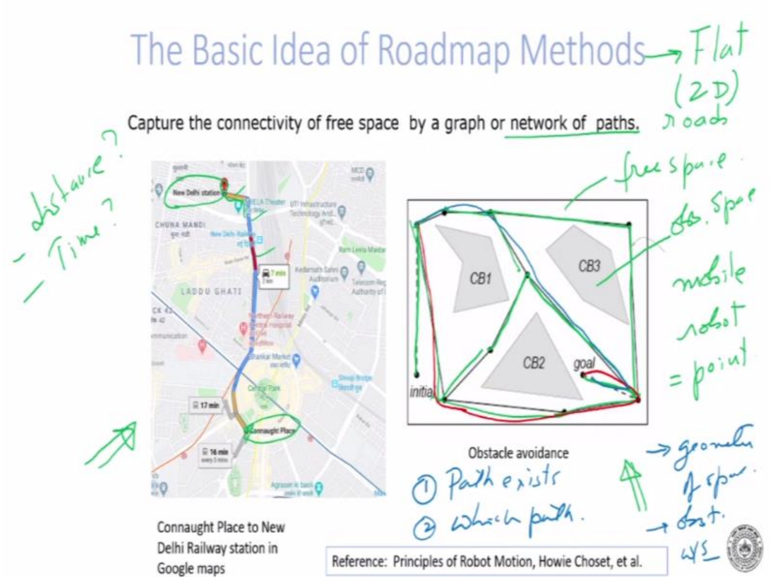
So, there are various methods of finding paths they are depending on situations and we will look at each one of them one by one. So, the first one is what we call road map methods. Road map method is the name suggested as the name suggests it is something like finding a road in a map like today we have Google maps we cannot find a road to go from one point to another point.

Then cell decomposition where you are decomposing the cell or the work shell to different shells and then finding the paths. Sampling based planners where you do not have a map, but you start sampling and see where you can go and where you cannot go and try and make it connected graph. Potential field based planners we use potential field to guide the robot so this is basically to guide the robot to the goal using potentials that means, for example, the object has repulsive potential.

So, when the robot goes near the object it gets repelled and the goal has an attractive potential. The robot is always pulled towards the goal wave front planners these are something like waves, the waves on the water they tend to move towards the lower potential, planning with kinematic constraint this is like cars we talked about kinematic constraints for example holonomic and non-homonomic constraints, like real cars they do not go sideways.

So, planning with kinematic constraints involve holonomic constraints and non holonomic constraints and then we look last on interest on applications. So, we will start up today with the roadmaps in the next couple of lectures are on basic methods as I just talked about.

(Refer Slide Time: 19:48)



The Basic Idea of Roadmap Methods → Flat (2D) roads

Capture the connectivity of free space by a graph or network of paths.

- distance ?
- Time ?

Connaught Place to New Delhi Railway station in Google maps

Obstacle avoidance

① Path exists
② which path.

free space
obs. space
mobile robot = point

→ geometry of space
→ obst.
WS

Reference: Principles of Robot Motion, Howie Choset, et al.

So, today we will start off with roadmaps. Now the basic idea of roadmap method is to capture the connectivity of free space by a graph or network of paths or roads. This is very simple to understand because this is we are very familiar with Google maps these days or if you look at any map how do you go from one place to another place you look at a map. Now, for example, on the left hand side here you see the Google map of New Delhi.

Now you have to go from Connaught place to New Delhi railway station. So, Connaught place is there and New Delhi railway station is there. Now how does Google map find out which way you should go? So, what it basically does or what is a map or what is the roadmap? Roadmap is a set of connected graphs or connected roads. So, basically you can see that these are all the various roads in that area.

And there is a network of roads and by following this network of road you can go from this initial point to the final point. Now there could be more than one road for example you could go from here like this, like this, like this then cross over from that side go like this and then go there. So, this could be one way. Now there could be other ways for example you go like this, like this, like this, like this these are small, small paths so you can find some part which will take you there. So, then the next question that come this which part will we take.

Now, in the Google map one of the parameters could be the distance, but as you can see that it has different colours, it has orange colour, it has blue colour and it has red colour. Red colour means it is very congested road so we will take more time. So, you have to worry about time. So, it is not only a question of finding a path it is a question of how you are going to go on the path which one is going to take which has a longer distance to a travel which is going to take more time to travel.

The particular distance could be less, but the time could be more. So, this is basically a very brief idea of how roadmaps work. Now on the right hand side we have a problem of a mobile robot. So, we have a mobile robot which is an initial point let us call it a mobile robot. So, mobile robot and these are obstacles which are called CB 1, CB 2, CB 3. Now the mobile robot is a point and I want to go from one point the initial point to the goal point here.

The initial point is here goal point is there. Now you can see that this free space the first thing that you should note here is that this is free space where it can go, and a couple of other things. This is flat ground so it is 2D. So, 90% of the algorithm we will talk about for 2D applications not for 3D and most of 3D also can be reduced to 2D. So, there is free space and obstacle space this is what you are seeing here.

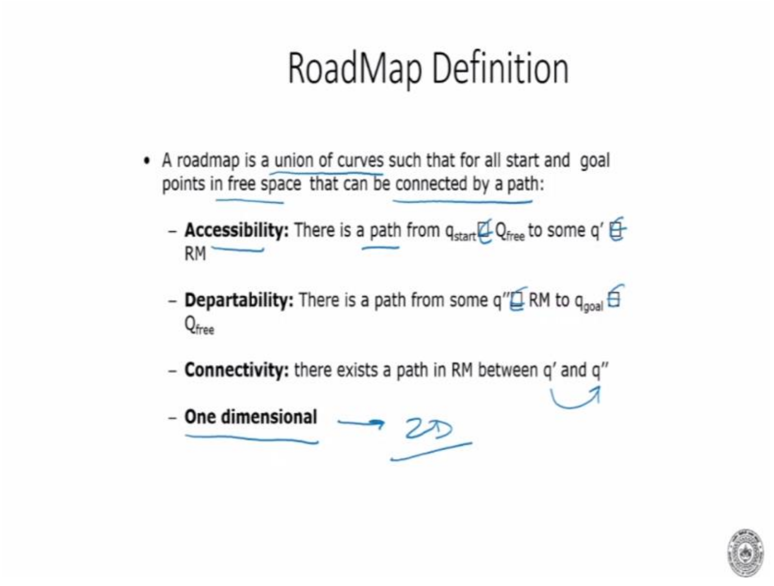
So, the free space is connected based network of path so this is a network of paths. This is already connected by network of paths. How do we get these paths that we will come to later. So, we have given this network of paths now what do I do is I know this is a network of paths. So, what I will do is from initial point I will connect to this network of paths here and from the goal point also I will connect to the network of paths.

Then how many ways are there to go from the initial point to the final point there could be more than one. For example I can go like this, like this, like this, like this or I could take some other path and go like this, like this, like this, like this, like this I could take even more paths other paths are there. So, then the next question comes that which path will you take? So, the first thing that we need to know is whether a path exists.

So, in this particular case we can see that the path exists. Number two then which path? Now for us having eyes it is very easy to see and say okay this is a path just connect it and it can go from like this, like this it will be shorter, but this poor algorithm does not have eyes. So, an algorithm is a program a program will be run and the program output has to be that this is the path and this is how you should go.

What is the input of the program? Now you can guess from here that the input to the program is the geometry of space that means where is the robot, where is the obstacle, where is the work space. So, essentially path planning is a geometrical problem we solve it geometrically. So, we have eyes so we can see very easily and say whereas for a robot which does not have eyes it has to geometrically figure out or the algorithm has to systematically figure out how many paths are there, which path I will take and should not hit obstacles that is the whole objective of writing an algorithm or a program for path planning. Let us proceed.

(Refer Slide Time: 24:40)



RoadMap Definition

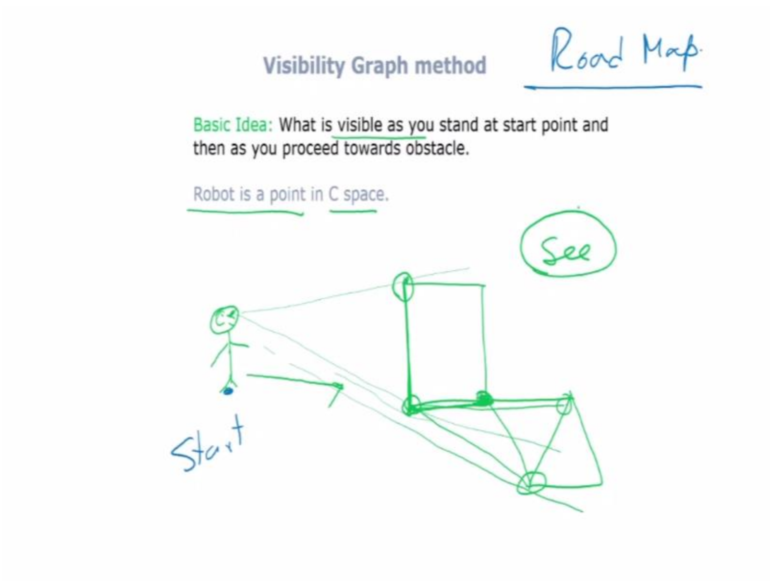
- A roadmap is a union of curves such that for all start and goal points in free space that can be connected by a path:
  - **Accessibility:** There is a path from $q_{start} \in Q_{free}$ to some $q' \in$ RM
  - **Departability:** There is a path from some $q'' \in$ RM to $q_{goal} \in Q_{free}$
  - **Connectivity:** there exists a path in RM between q' and q''
  - **One dimensional** $\longrightarrow$ 2D

Now the definition of a roadmap is that a road map is a union of curves such that for all start and goal points in free space that can be connected by a path it has accessibility that means there is a path from q_{start} which belongs to Q_{free} to some q' which also belongs to the roadmap. So, accessibility there is a path from the start point which will join the roadmap. Departability there is a path from the q'' to the road map to goal.

So, from q' it has joined at q'' now from q'' which belongs to the roadmap to the goal there is a free path and connectivity there exists a path in road map between q and q''. Now please note this is one dimensional. This path is one dimensional in terms of topology, but to represent it you need 2D, but the path itself is 1D only.

Now, this is very easy to understand accessibility, departability and connectivity. So, accessibility essentially means that from here you should be at this point you should be able to access the network. Departability you should be able to go from here and go to the goal point and connectivity is this connection should be there between the accessibility and the departability. So, that is the basic idea of a road map.

(Refer Slide Time: 26:19)


Visibility Graph method

Basic Idea: What is visible as you stand at start point and then as you proceed towards obstacle.

Robot is a point in C space.

Now visibility graph methods. So, how do you make this road map that is what we are looking at now. So, we tried to get the road map we know once we have the road map so we have the road map like this. Now once you have the road map then you can go from the initial point join the network then go towards the goal points and from that you can depart again and then go to the goal.

But then you need the network of paths how do you get that. So, there are some methods of getting the network of paths. So, the first method we look at is called visibility graph method. So, the basic idea is very simple you see what is visible as you stand at start point and then as you proceed towards the obstacle. So, this means that basically you stand suppose I am at this point this is my initial point or start point.

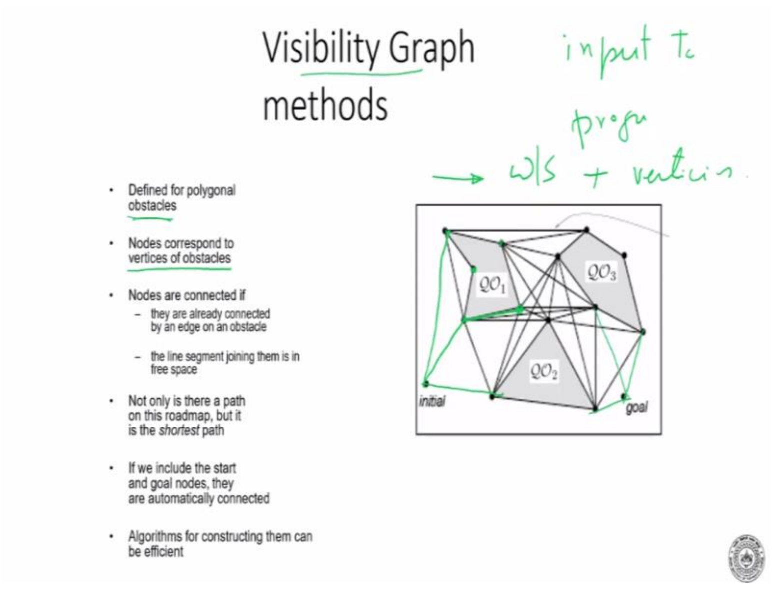
And there is an obstacle let me draw it with some other colour so there is an obstacle here and there is another obstacles here. It says that we are standing, imagine yourself standing here so this we are standing there and this is your eyes and from there you can see. Now what is visible to you is this is visible to you that is visible to you, this is visible to you. So, these are visible to you as you are standing there.

So, what is visible as you stand at the start point and then as you proceed towards the obstacles. So, we can see what can you see you can see this obstacle here you can see this side, you can see this side. Now when you proceed towards the obstacle you come here and from here what can you see we can see this you may be able to see that so you can see this, you can see that.

Now the edges here are also visible there because from standing here you can see that also then you move here and from here you can see that, you can see that. So, it is like saying you are standing at the start point looking at the obstacle then going to vertices to this obstacles and seeing what more can you see and whenever you see a vertices whenever you see a vertex basically you are connecting back with a straight line which is your road.

So, this basically means that the robot is a point in C space and you basically stand and see it is very easy to understand whenever you see a vertex you know there is an obstacle and the obstacle has a vertex. So, you go to that vertex and then you see again what more can you see and then you connect all of these vertices by straight lines and that is where you get your collection of paths.

(Refer Slide Time: 29:01)



Visibility Graph methods

input to progr

→ w/s + vertices

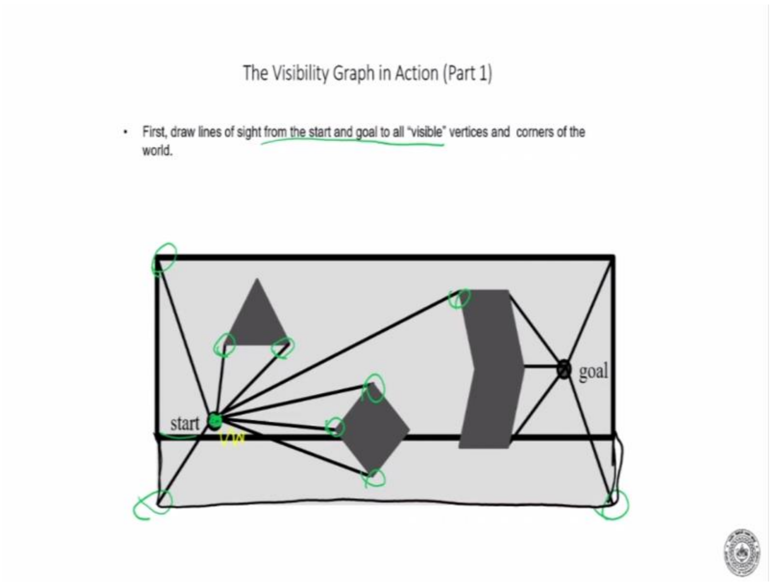
- Defined for polygonal obstacles
- Nodes correspond to vertices of obstacles
- Nodes are connected if
  - they are already connected by an edge on an obstacle
  - the line segment joining them is in free space
- Not only is there a path on this roadmap, but it is the *shortest* path
- If we include the start and goal nodes, they are automatically connected
- Algorithms for constructing them can be efficient

Now in the visibility graph methods this is defined for polygonal obstacles, there are various obstacles which you can see Q 1, Q 2, Q 3 then nodes correspond to vertices of obstacles. So, each of these nodes are the vertex of the obstacles, the vertices. Nodes are connected if they are already connected by an edge or an obstacle. So, you can get connected nodes, nodes are connected if they are already connected by edge of an obstacle so you have a node here, node here and this is already connected by the edge.

The line segment joining them is in free space not only is there a path on this roadmap, but it is the shortest path. Now, if we include the start and goal nodes they are automatically connected. Algorithms for constructing them can be efficiently written. So, you can have a computer program, what is the input to your program? So, the input to the program is a work space plus vertices.

Once you give this to the program the computer program knows that these are the vertices and these is the work space. So, what it can do is it can connect all the nodes by straight lines and then from the initial point we can connect to the goal point by following one of the road maps that is how we got our road map let us see how this is done.

(Refer Slide Time: 30:25)



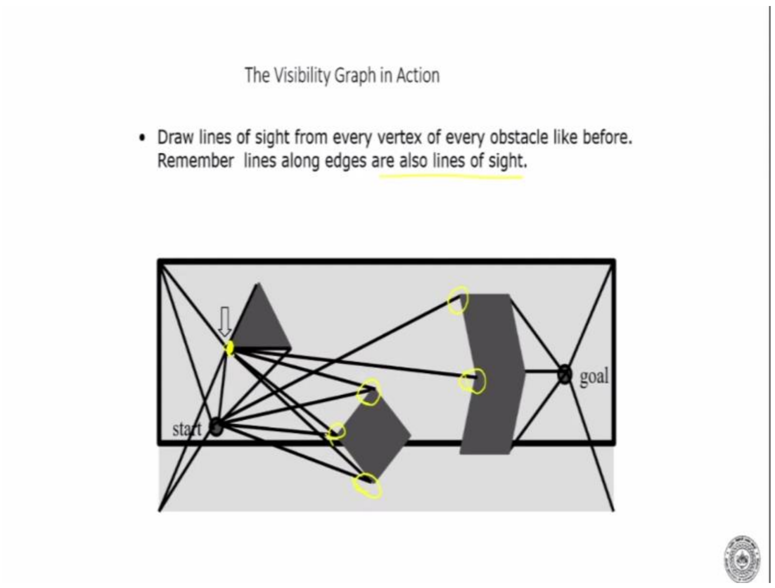
The Visibility Graph in Action (Part 1)

- First, draw lines of sight from the start and goal to all "visible" vertices and corners of the world.

First of all draw lines of sight from start and goal to all visible vertices and corners of the world. So your start point here. So, we are drawing straight lines to all the visible vertices. So, what can you see you can see this one, you can see this one, you can see this one, you can see this one, this one, this one that is visible. So, first of all draw lines of sight from those start and goal to all visible vertices and corners of the world that we have done that is the corner of the world this line should have been here.

Let me just correct it so the corners of the world is here like this and that part is now there. So, this is not there. So, we are drawing lines from the start point to all the vertices that are visible.

(Refer Slide Time: 31:15)

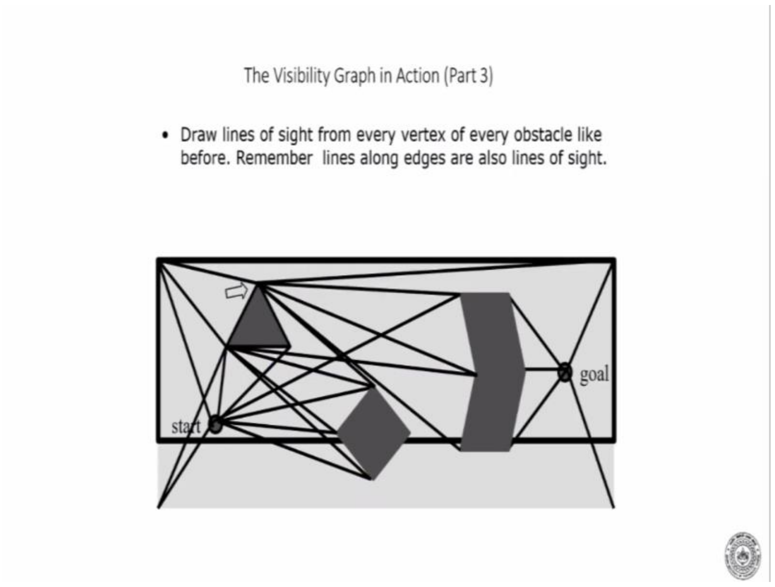


The Visibility Graph in Action

- Draw lines of sight from every vertex of every obstacle like before. Remember lines along edges are also lines of sight.

Then what we do is draw lines of sight from every vertex of every obstacle like before. Remember lines along edges are also lines of sight. So, first of all you see which vertices are you can see then you go to that vertex and what more can you see. See from here you can see this, you can see that, you can see that, you can see this, you can see this. So, this vertices we can see and you are also supposed to draw lines along the edges. So, you can see this also.

(Refer Slide Time: 31:45)

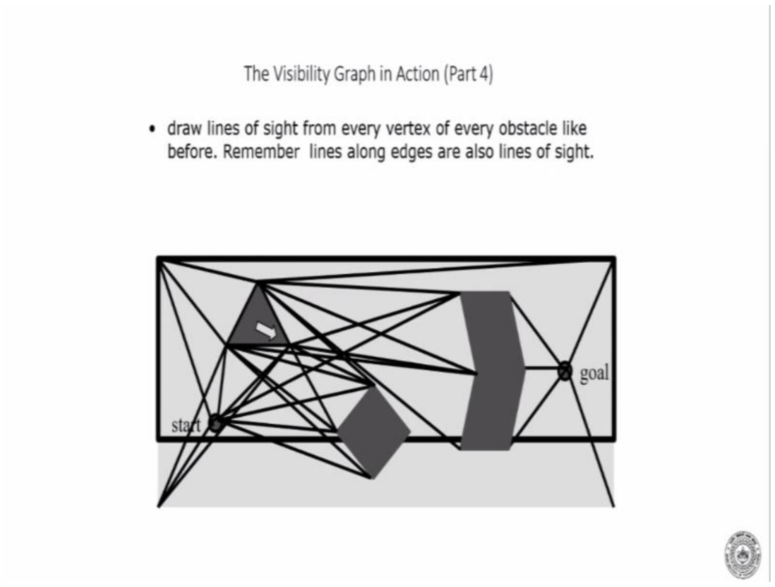


The Visibility Graph in Action (Part 3)

- Draw lines of sight from every vertex of every obstacle like before. Remember lines along edges are also lines of sight.

Now, continue doing that now draw lines of sight from every vertex of every obstacles. Remember lines of edges are also lines of sight.

(Refer Slide Time: 31:54)

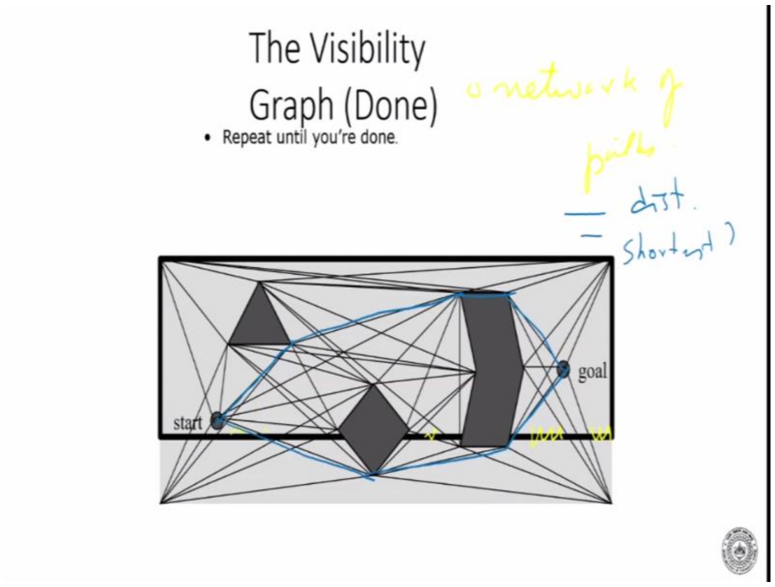


The Visibility Graph in Action (Part 4)

- draw lines of sight from every vertex of every obstacle like before. Remember lines along edges are also lines of sight.

So, if we do this what will happen is what we will see is a total connected ray of lines.

(Refer Slide Time: 32:03)



The Visibility Graph (Done)

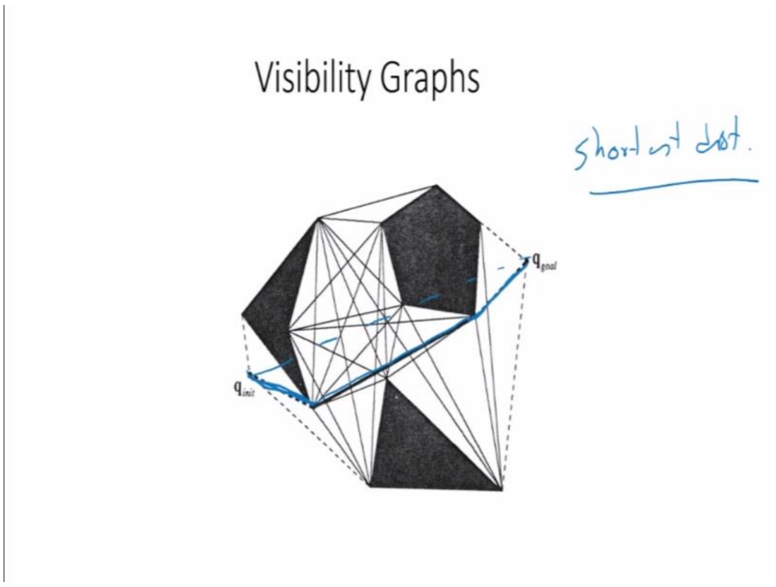
- Repeat until you're done.

And finally what we get is this. Please ignore this line, this line is not there. So, the visibility graph is complete now. So, this is showing your complete network of paths. So, this is giving you a complete, complete network of paths. Now once you have got the network of paths now what you can do is you know you have to choose you want to go to the goal point. So, how can you go to the goal point?

There would be so many you can go this, this, this, this, that or we could go this way, you could that way, you could that way, you could go in so many ways here. The next question would come in terms of distance, so which is the shortest path may be. So, this is basically called the visibility graph method where we have done the visibility graph simply by connecting all the vertices by straight lines.

It is like you go and stand in the vertices and which are the vertex is visible to you just connect it with a straight line.

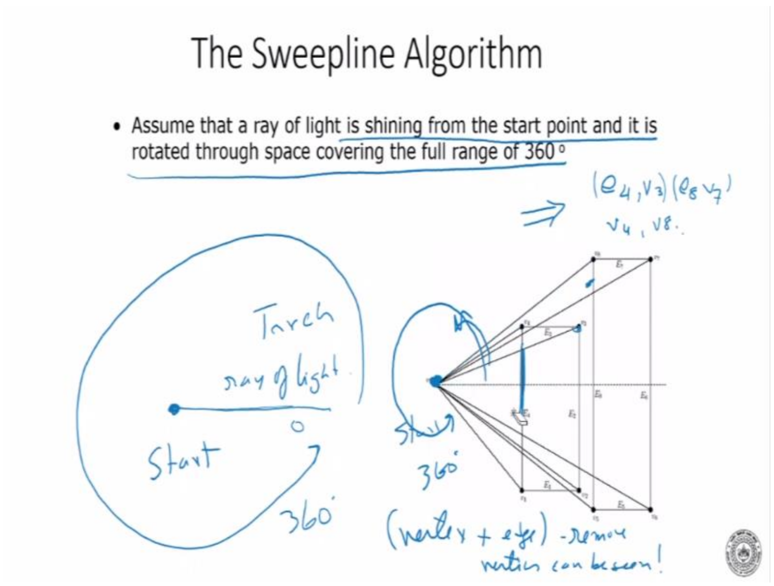
(Refer Slide Time: 33:15)



Now, this is a visibility graph where we have drawn the complete graph. Now we have to find a path now what you can do is you can look at the path length so this path length here to here to here probably be the shortest. How do you know? Probably this is the if you draw straight line like that the one closer to the straight line will probably give the shortest distance.

So, this is the probably the shortest distance path. This is the straight line that was connecting. Now, so this is geometrically how it is done.

(Refer Slide Time: 33:51)



The Sweepline Algorithm

- Assume that a ray of light is shining from the start point and it is rotated through space covering the full range of 360°

Now let us look at in terms of an algorithm when we are drawing this connection of graph interconnected road map the algorithm that works is basically which is called the sweep line algorithm. Now assume that a ray of light is shining from start point and it is rotated through space covering the full range of 360^0. So, imagine we are standing here start and there is a ray of light that is shining from start.

And it is rotated through space covering the full range of one 360^0. So, let us say this is my 0 and this is a ray of light it is like saying we can consider this to be a torch. Suppose you are shining a torch and this ray of light is rotated to full 360^0. Now what is it that you will see. As you start off from 0^0 you will first hit this obstacles then as it is moving upward what will happen you will see okay the logical the way it works.

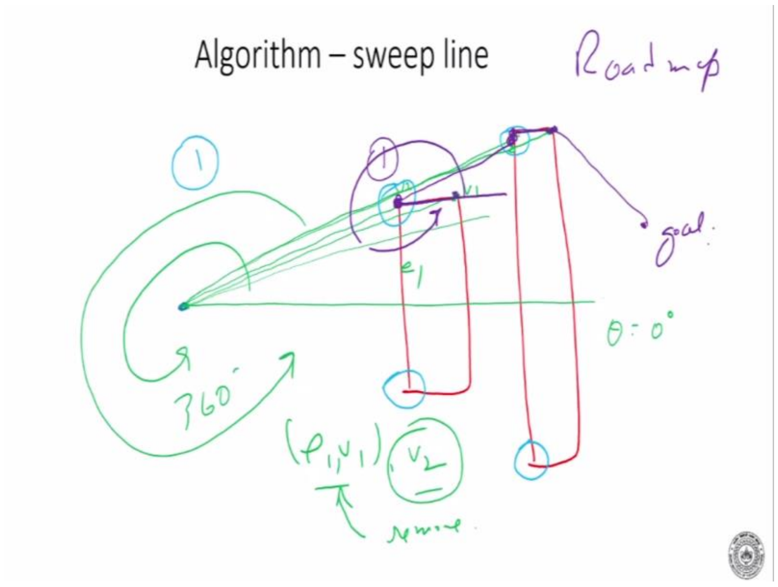
But in this particular case you are standing here the example that is shown here so we are standing here this is my start point and shining a ray of light. So, what we are seeing is that it is hitting the obstacles like this, like this, like this, like this. You can see that vertex how many vertices you are shining this ray of light and how many vertices can you see? So, you are seeing one is an edge that is e_4 and there is a vertex which is let us say v_3.

So, this one and this one and then as a sweeping mode that side it is hitting this edge which is e_8 and it is getting the vertex v_7. Now sweep a little bit more in this point what is happening it is hitting no edge, but is hitting the vertex v_4 that was hitting v_4. Further we sweep mode we are hitting v_8. So, in this way the ray of light is being rotated by 360^0 and we are making this if it is an edge or a vertex or only a vertex that we are noting as it is making a full sweep.

Now, in this what we can do is wherever there is a vertex plus an edge we can remove because it is going through the edge it cannot go through the edge, but geometrically basically the ray of light is a ray of light. So, this is equation of straight line. So, we are basically drawing a straight line and then seeing whether this straight line is intersecting this edge or it is hitting that point.

So, geometrically we are doing this and we make these pairs and whenever we have a vertex plus an edge we remove so we are left with only the vertices. So, the vertices which can be seen so the vertices which only can be seen those are the ones that we are keeping.

(Refer Slide Time: 37:17)



So, let me draw it again just explain this is a very simple method a very nice geometrical method. So, here we have one obstacle here I have another obstacles this is an example and we started off by using a ray of light like this $\theta = 0^0$ then we start sweeping it like this 360^0. So, wherever it hit an edge and a vertex we record and keep. So, this is a straight line and these are all straight line and edges.

So, geometrically we can find out the intersection to it. So, as it is going up what we are seeing is that it is going like this and then it hit that one it hit that, but it is also hitting this edge let us call this e_1 and let us call this v_1 so e_1 v_1 is there then it goes little bit higher it hits v_2 so there is v_2, but there is no edge then goes little bit higher it hits that one it hits the inside fellow first this one, but it also hits this edge.

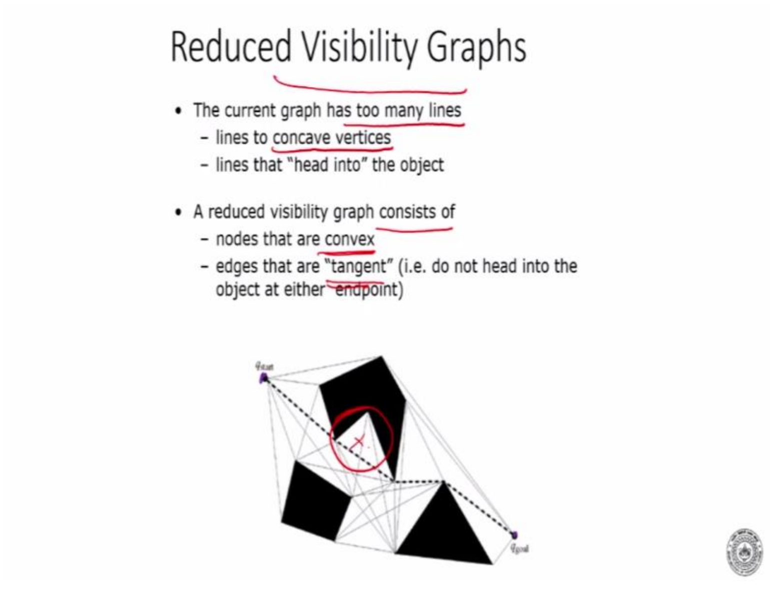
Then what we do is we make this full sweep and make this wherever it is hitting the edge and the vertex then we remove. So, if I remove this what will be left only this vertices will be left because the vertices through which there is no edge and vertex there is one here and one here. So, my sweep line is often is basically giving me the connection between the start points and the edges.

Then what do I do this is step number one, step number two is I come to the edge one at a time I come to this edge this is the first one then I do the same thing sweep line algorithm now I start sweeping from here to 360^0. What does it see? It sees that edge then where do I go I go here it sees this edge it also sees the edges. So, it goes here it sees that edge and then basically what it can do is suppose my goal is somewhere here it connects it with that.

So, this is basically giving us the sweep line algorithm which basically tells us what are the various interconnections between the nodes and that is giving my road map. So, in the previous example we said that how to get the road map this is how we can get the road map. Basically we can shine a ray of light and see wherever it is hitting and identify this vertices then this is actually like a ray of light so it is hitting that, it is hitting that, it is hitting this.

So, that is like a road in the road map and we are making the connections to all of them. Now something which you have seen here is that there this network includes a lot of roads for networks. Now all of them are useful is getting very clumsy and congested. For example you may not need in this particular case.

(Refer Slide Time: 40:18)



Reduced Visibility Graphs

- The current graph has too many lines
  - lines to concave vertices
  - lines that "head into" the object
- A reduced visibility graph consists of
  - nodes that are convex
  - edges that are "tangent" (i.e. do not head into the object at either endpoint)

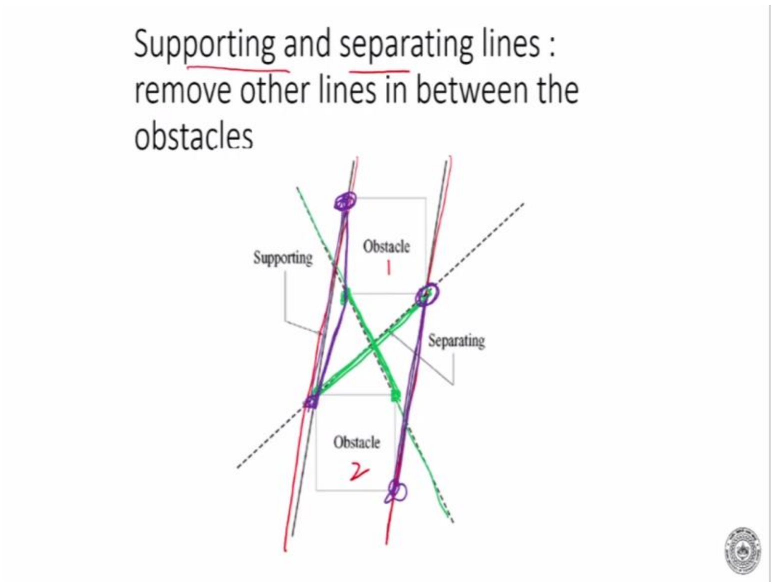
q_{start}
q_{goal}

Suppose I want to go from the start point to this goal point. I could have gone like this, like that, like this, like this and then gone like that or I could have gone from here to here and here to there and there to there then there to here and there to here and then there that is also a part it is possible, but something that we notice here is that I if you use the sweep like algorithm or this method the current graph has too many lines that means there is just too many lines.

Now something that you will note very easily is that whenever you have concave there is no point in going there. So, the lines that go into concave vertices actually not of much use because they only lengthen the path they do not make it shorter. So, the next step is to reduce the visibility graph and this is called the reduced visibility graph. So, a reduced visibility graph consists of nodes that are convex.

Edges that are tangent and do not hit into objects at either end points. So, first of all we are reducing it such that anything that is roads which are going into the concave path we will remove it because they are taking more distance the increasing the length without doing anything useful.

(Refer Slide Time: 41:38)

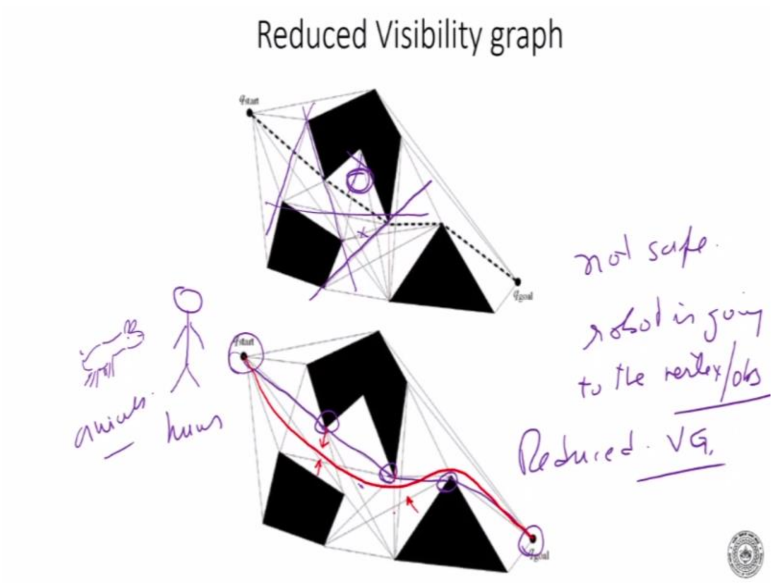


Supporting and separating lines : remove other lines in between the obstacles

So, how do we do that we use the supporting and separating lines. This is called supporting line between any two obstacles let us call obstacle 1, obstacle 2 that is a supporting line and these are separating lines which are separating these two obstacles. Now, if you can imagine if you want to go from here to here then this is the shortest path similarly if I want to go from here to there this is the shortest path.

Whereas if I want to go on the outside if I want to go from here to here then this is the shortest path or from here to here this is the shortest path that is the logic of the separating and the supporting edges. So, basically if you just have supporting edges and separating edges you do not need any other lines because any other line coming in between we will just add length nothing else. Suppose, from here I go here then I come there this is longer than going straight like this.

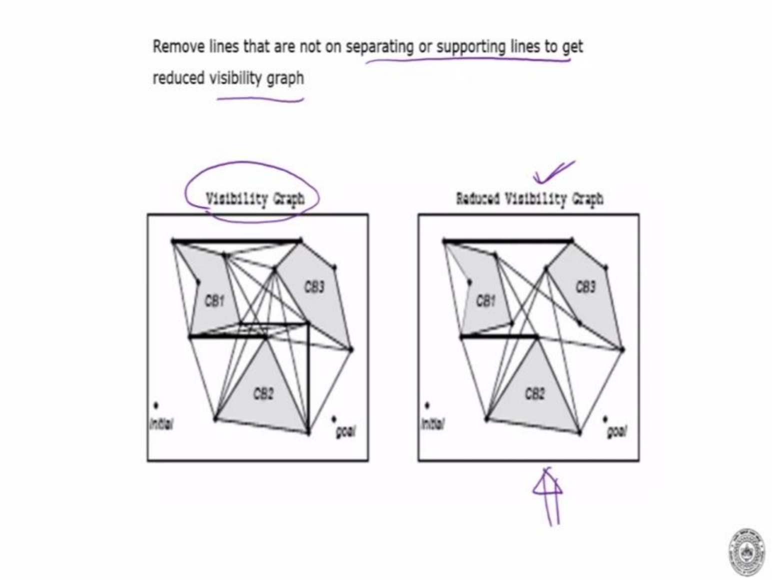
(Refer Slide Time: 42:40)



So, what I can do is I look at all the concave path use separating and supporting edges and remove the other lines. So, in this particular case which is my supporting line? This is a supporting line and which is a intersecting line? Intersecting line will be this and that one supporting and separating. So, whatever else is coming I remove it especially which is here that is removed you see this is removed here and this one is removed that one is not there.

So, this is a reduced visibility graph this is reduced visibility graph. Now in this reduced visibility graph now if you want to go from some point to some point you can simply take this route, this route, this route, that route you need not go from here to here go there, go here these are not required. So, we can have a visibility graph and then we can have a reduced visibility graph by simply using separating lines and supporting lines.

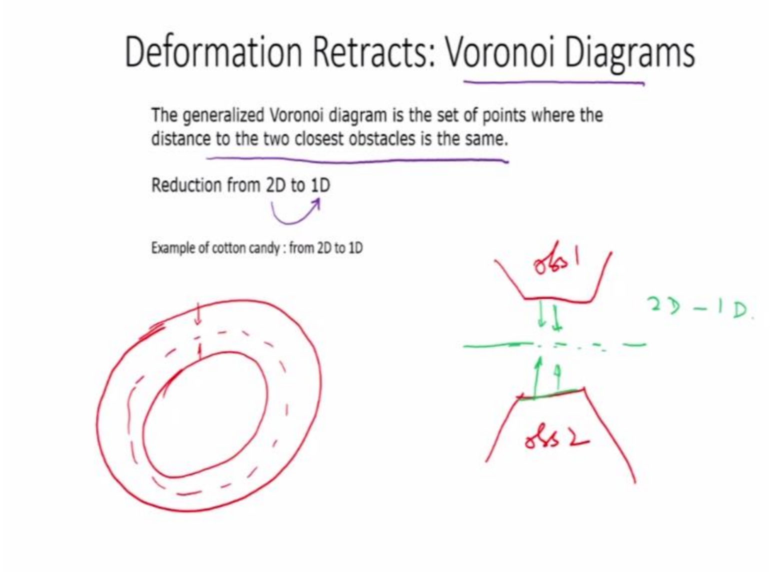
(Refer Slide Time: 43:42)



Remove lines that are not on separating or supporting lines to get reduced visibility graph

So, remove lines that are not on supporting or separating edges to get the reduced visibility graph. So, this is also another example which is showing a visibility graph this showing a reduced visibility graph. So, we can see all unnecessary lines have been removed here.

(Refer Slide Time: 44:00)



Deformation Retracts: Voronoi Diagrams

The generalized Voronoi diagram is the set of points where the distance to the two closest obstacles is the same.

Reduction from 2D to 1D

Example of cotton candy : from 2D to 1D

Now visibility graph method is a very simple method to find the roadmap and then take you from point to another point as we can very easily find which obstacles to avoid the obstacles. So, this is a very interesting and very easy method, but if you think a little bit suppose you are going from one point to another point suppose you are starting from this point and you are going from this point a human being or maybe even a dog.

Well it is a dog, a dog is going from the start point to the goal point would it follow a path like this if I have to ask you would a dog this path or would a human being take this path which path will you take. Now we would not take this path now why would not you take this path why is this actually that path. Okay we have found it using visibility graph, using sweep line algorithm and all that, but then why if you think a little bit why is this a bad path?

Now this is a bad path because it is going to the edge of the obstacle here, here, here. When you go from one point to another point you do not go and brush against an obstacle. You go far away from the obstacle why because it is safer. This is not safe why is it not safe? Because the robot is going to the vertex and the vertex is on the obstacle. So, if there is any sensor noise or there is any kind of problem in estimation then the robot is going and hit the obstacle. We know that in the real world there are always errors.

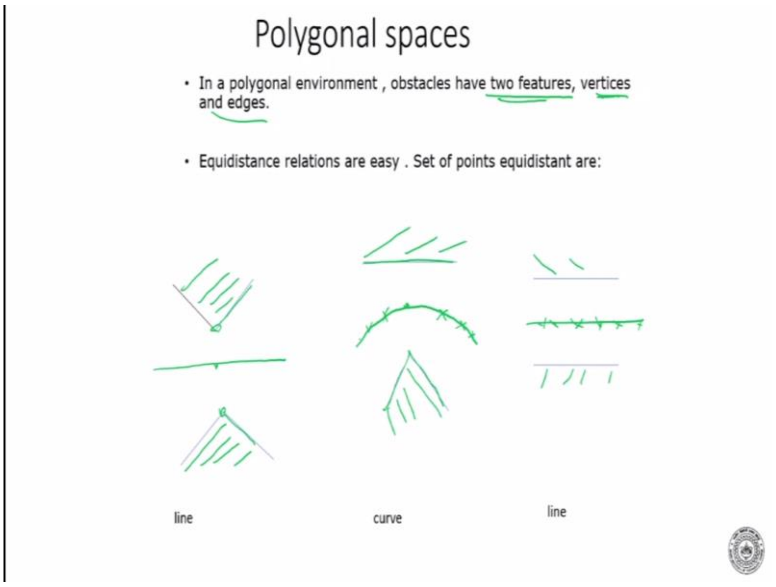
So, the sensors which are there on the robot would have little bit of error. Let us say sensor or positioning sensor so if there is small error also it will go hit the obstacle and that is also one reason why human beings, humans, animals do not do this. We look at the obstacle may be make road maps, find the path and then go, but that path will not be going and touching the obstacle because that is a bad idea.

So, the next thing that we do is we look at what is called the Voronoi diagram. So, the generalized Voronoi diagram is a set of points where the distance to the two closest obstacle is the same. So, here reduction from 2D to 1D. Now, if I go back to this earlier example again of a human being which path would you take. So, we will probably take this path we go like this, like this, like this and go there. So, what we did is we were equidistance from here if you see carefully.

We now go to close to that obstacle we were equidistance from the obstacles that is the safest path. Now, if you look at this deformation retract example of cotton candy or sugar candy you might have seen that if you have the sugar candies which are like this. So, this start shrinking from both sides what will happen is they will go somewhere in the center. Now, if I have an obstacle like this so obstacle 1 and I have another one like this.

Which is in between both of this so the path right in between both of this would be here. So, it is like we are going it is like this is shrinking this side this is shrinking that side and then what is coming from 2D it has become 1D. Now this is basically how we produce a Voronoi diagram.

(Refer Slide Time: 47:48)



Polygonal spaces

- In a polygonal environment , obstacles have two features, vertices and edges.
- Equidistance relations are easy . Set of points equidistant are:

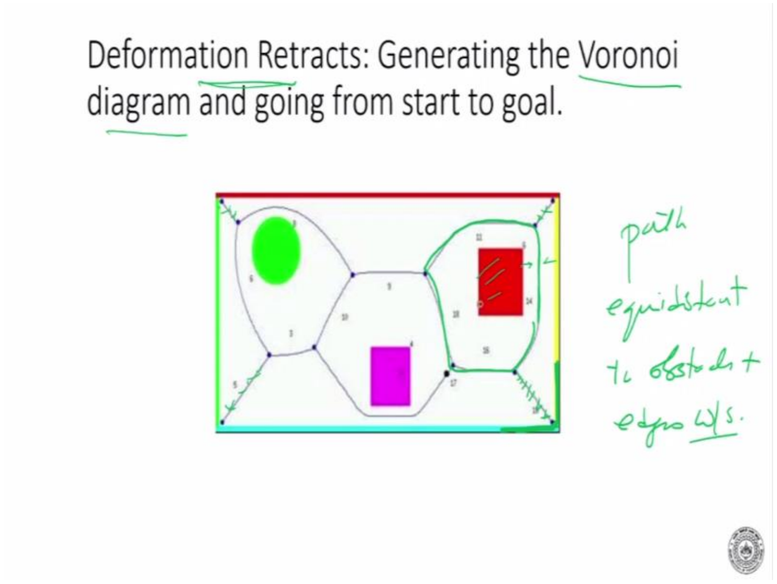
For different kind of obstacles we have two features. We have edges and we have vertices. So, any kind of obstacle would be made up of vertices and edges and if you want to find a distance relationship which is equidistant now that is very easy. So, basically what we are trying to do is we are trying to get a set of points equidistance. Now suppose this is an obstacle and this is an obstacle and this is another obstacle which is the path which is equidistance from here.

So, if you have a vertex here, vertex here very easy this point is equidistance and this line and this line is the same the angle is the same. So, it is like this is the path which is equidistant from both these obstacles. Now, if you have a straight line this is an obstacle and you have an obstacle like this now which is first of all start with the vertex which is the equidistant point this is the equidistant point.

Now this is inclined this is flat now what would be a set of equidistance point this is going to be curve like this. So, these points would be equidistant. Now, what about edge like this an edge like this. Now this is very simple it will be equidistant point will be equidistant from both these edges which is a straight line again. So, in polygonal spaces obstacles have two features they can be vertices and they can be edges.

So, this obstacles are made to the polygonal they are made up of vertices and edges and we can have a equidistance relation between them very easily and another equidistance relation is going to be the path.

(Refer Slide Time: 49:20)



Deformation Retracts: Generating the Voronoi diagram and going from start to goal.

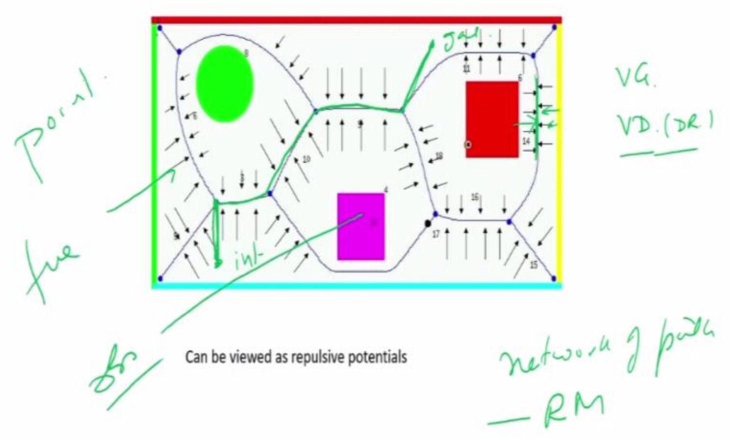
Now look at this Voronoi diagram these are called Voronoi diagrams and this is also called deformation retract. Why is there deformation retract because the example I just gave sometime back is that if you are thinking of this shrinking or deforming in that direction this is deforming in this direction what will be left with is this circle or this curve. So, from 2D it has become 1D actually and that is why it is called a deformation retract these are called retracts. So, we have one side here, another side here I will start from here.

So, points which are equidistance from both these points is one line. Similarly it is one line there one line here, one line here this point are equidistance from two sides. Now there is an obstacle here so if you can imagine this obstacle is expanding and this is also expanding where will it come? It will come here so this is my network of path now which are equidistance from the obstacles and the sides.

So, path equidistance to obstacles plus edges it is work space. Now this is basically called a Voronoi diagram. Now in this Voronoi diagram you are seeing that it is same because why it is going in between the obstacles is going in between the edge and obstacle also. So, it is not going very near to the obstacle or the edge.

(Refer Slide Time: 50:54)

Deformation Retraction: GVG in Plane



Can be viewed as repulsive potentials

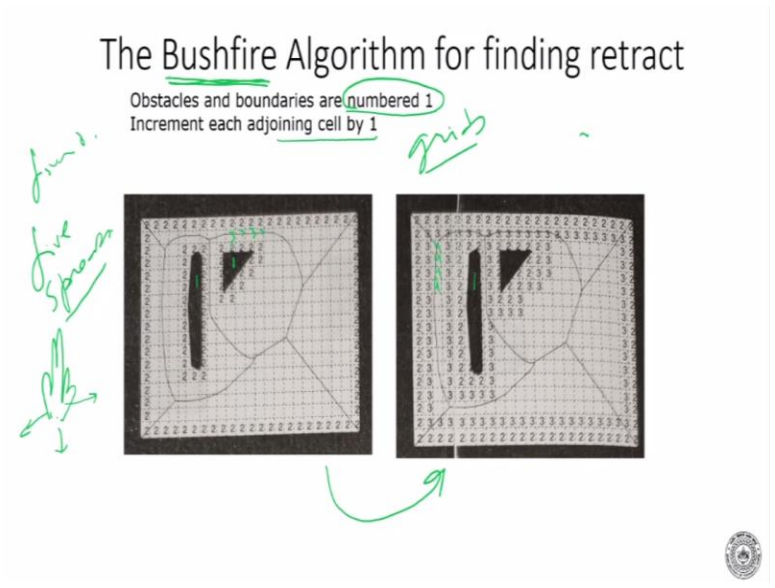
And this is the deformation retract because you can think about this as the side is good it is deforming in this direction the object is deforming in that direction where they are meeting they are meeting in between there. So, this is basically called the deformation retract generating Voronoi diagram or this is called a generalized Voronoi diagram so this is very often called as generalized Voronoi diagram.

Now once we have got the network of paths we got the network of paths which is my road map is called a RM then I can go from the start point and join. So, here I can suppose I am here initial point I want to go here this is my goal point what I need to do is try this node and go and join the node from here I can find the node and join the node and then I have different paths I can go like this, like this, like this that is my path.

There can be more than one path so this is how we can find either by using the visibility graph we can use Voronoi diagram, we can use deformation retract Voronoi diagram or deformation retract they are more or less the same and to find network of paths once I have the network then it is a question of just joining the node the nearest to your start point and then from the goal which is nearest to the goal and then you can follow the path and go from the start point to the end point.

Please note here again that the robot is a point and we are basically dividing into free space and obstacle space that is how we are operating.

(Refer Slide Time: 52:44)



The Bushfire Algorithm for finding retract

Obstacles and boundaries are numbered 1

Increment each adjoining cell by 1

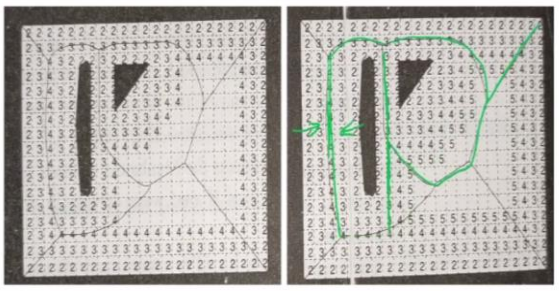
Now these are some examples another example of finding equidistance point between obstacles is by what is called the Bushfire algorithm for finding retracts. Now this is very easy way of doing it because in the earlier case you saw that it was something like it was continuous space. So, what you have this is a geometrical problem why you have the equation of this line, you have the equation of this line.

Now, if you take points one point here, one point here then find equidistance point so we have to go point by point whereas suppose we can decide we can divide the whole space into grids then it is much easier for us the job is simplified and this is basically called Bushfire algorithm why Bushfire because you say for example you light a fire. So, we are lighting a fire here.

And how the fire will spread is it will start spreading the fire spreads like in the forest so you start one point and then it start spreading sideways in all directions actually. So, what we do is we divide it into grids and you start putting the obstacle as one so obstacle is numbered by obstacles boundaries and numbered by one and increment each adjoining cell by one. So, this is 1 the next will be 2 in there will be 3 like that and this can be done very easily by the computer.

So, this is 1, this is 2 so any program can do this very quickly. So, obstacle was 1 the next one was 2, then 3 then 4 like that.

(Refer Slide Time: 54:27)

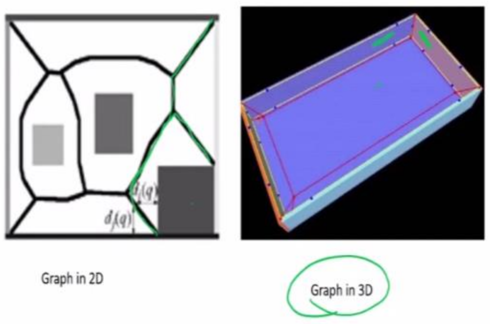


Now, if you see this kind of a numbering it would automatically when you have covered the full grid the highest numbers will cross point to in between points. This is 4, this is 5, 5, 5, 6 now this is automatic. So, between this subject and this subject you see this is a deformation retract actually now that is automatic because wherever the number becomes highest, it can go anymore and it is going in both direction from here and here.

So, the highest number is the one that is in the middle. So, we basically have made a road map as shown here. So, this is a road map which you have made by in this grid format. Now you can start from the start point join to the node from the goal point you can join to the node and then follow the path.

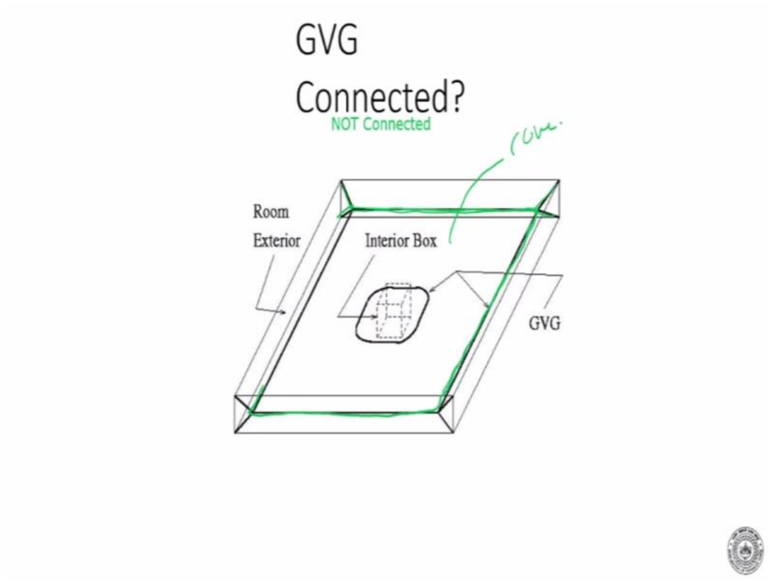
(Refer Slide Time: 55:12)

General Voronoi Graph-
Extrude the 2D graph to 3D



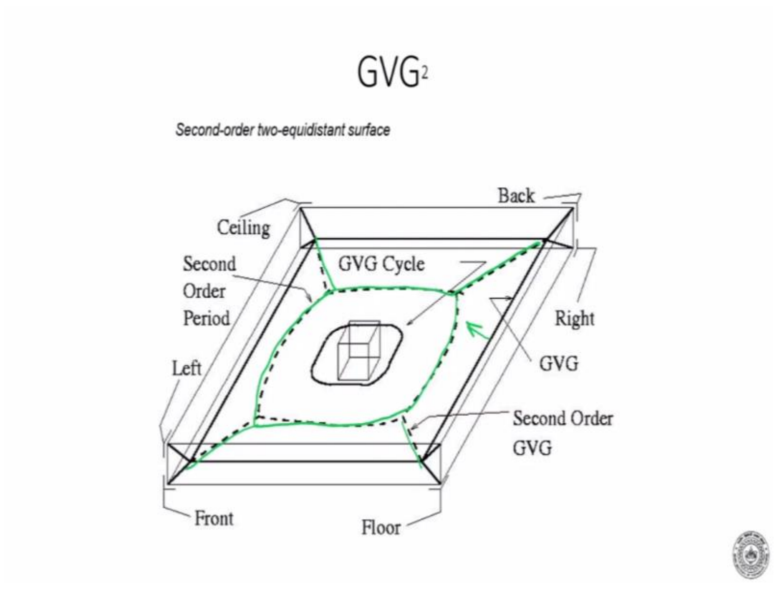
Now here we can also have a 3D so what I just explained was in 2D we can exactly do this in 3D also. So, in 3D case what we do is we deform it in this direction and this is the wall the side wall and this is the obstacle. So, when we are deforming it this is my retract now. In case of 3D how will it come you can see that this is a side wall is there, this is the floor. So, we have three lines now one line here, one line here, one line here the three. So, deformation retract what will it look like?

(Refer Slide Time: 55:47)



So the deformation retract will look something like this, this line is equidistant and the central line is here. Now there is of course at top also there is a cover is also there of this. So, this line is giving you the equidistant line from there. Now, if there is an obstacle there then this is my for the obstacle this will be like that.

(Refer Slide Time: 56:10)



So, at the end of the day what we get so it is going to deform like this, like this and what we get is my deformation retract which is like this. These are set of points which are equidistant from the site and the obstacles. So, we will stop here today and in the next class today we looked at manifold then we looked at the rotation matrix and from the rotation matrix we then moved on and looked at the roadmap method.

And with the road map method we basically saw that once you have a roadmap then the roadmap from the start point we can connect to the road map and from the goal also we can connect with road map and then find different path that we get here from the initial point to the final point and we looked at a few methods of finding the road map for example using visibility graph then by using deformation retracts. So, we will stop here today and we will continue in the next class. Thank you.