

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

**Path Planning: Graph Construction**  
**Lecture - 6.2**  
**Path Planning- Cell Decomposition Method**

(Refer Slide Time: 00:13)

Lecture 6.2



Path Planning- Cell Decomposition Method



Hello everyone, welcome back. So, in the last class, we talked about the Path Planning of mobile robots. And mentioned that there are two steps involved in navigation of the robot; one is the path planning and other one is the obstacle avoidance. And for path planning, we used different methods of path planning strategies, for identifying a feasible path for the robot to move from its current position to the goal position.

And graph search is one of the commonly used method, which is an offline planning method. So, in graph search, first we try to make a graph of all possible paths from the start to goal position and then, we search within that graph to find out the best feasible paths. That is basically the graph search method of path planning. And in this method, we saw two methods for finding out the paths.

So, one is the visibility graph; the other one is Voronoi diagram. Visibility graph and Voronoi diagram, uses the configuration space to identify feasible paths. In the visibility graph methods, we try to find out a path which is very close to the obstacle; but in the

Voronoi diagram, we try to get a path which is as far away from the obstacle. So, that is the primary difference between these two methods. So, we will see the third method which is known as the Cell Decomposition methods.

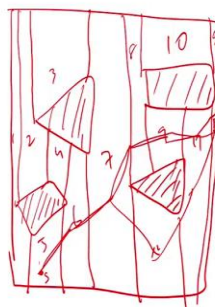
So, in cell decomposition, what we do is to we will decompose the whole workspace or the environment into multiple cells. And then, look at these cells to see which one is occupied, which one is not occupied and then, we try to find a path through all the unoccupied cells to reach the target. So, that is basically the cell decomposition method, ok.

(Refer Slide Time: 02:18)

### 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.



And there are two methods; one is known as the Exact Cell Decomposition and the other one is Approximate Cell Decomposition. So, in the exact cell decomposition, it is a lossless decomposition, where approximate decomposition represents an approximation of the original map. And then, a graph is then formed through a specified connectivity relation between cells.

So, that is in exact cell decomposition, we make sure that there are cells which are occupied or not occupied; but in approximate, there may be cells which are partially occupied and which are partially an unoccupied also. So, that is the difference between these two methods. There are advantages in using both these methods.

So, we will first look into the exact cell decomposition. So, what does exact cell decomposition say that you have a map of the environment with obstacle and this is the obstacle. So, like this. So, you divide them into exact cells of occupied or unoccupied. So, what I will do? I will divide this into this way ok I put another one here, another obstacle.

And then, I will try to connect all these and then, say which one is occupied. So, all the this will be occupied and I will connect divide this into cells like this. So, these are all and similarly, I will connect this and I will connect this. So, I had divide this into multiple cells. Now, you can see I can call this as cell 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 like this.

So, it is an exact cell decomposition, where I have divided this into different cells and then, I will say that ok all these cells are unoccupied; 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12. So, any path through these cells will be always feasible to reach the target point. Suppose, now this is the start and this is the goal, I can say that I can connect between this all unoccupied cells.

So, 5 and 6 can be connected; 6 and 7 can be connected; 7, 8 can be connected; 8, 9 can be connected; 9, 11 can be connected and 9, 11, 13, 13 can be connected. So, I can find a path through the free cells or I can go from here to 12 and then, go to here and then, do this.

So, all the decomposition primarily decomposes the whole area into cells which are occupied or unoccupied and then, all unoccupied cells can be connected to get a connectivity relation. And that connectivity relation between cells can be used for planning the path. That is basically the exact cell decomposition method, ok.

(Refer Slide Time: 05:30)



## 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 boundary of cells is based on geometric criticality. The resulting cells are each either completely free or completely occupied, and therefore path planning in the network is complete, like the road-map-based methods seen earlier. The basic abstraction behind such a decomposition is that the particular position of the robot within each cell of free space does not matter; what matters is rather the robot's ability to traverse from each free cell to adjacent free cells.

Number of cells and, therefore, the overall computational planning efficiency depends on the density and complexity of objects in the environment.



So, the cells are either completely or completely occupied or unoccupied. So, that is the thing, it is completely occupied free or completely occupied and therefore, path planning the network is complete. Like the road map-based method seen earlier. So, once that is done, then you can go for the graph construction using the other methods and then, get the graph. So, that is basically the exact cell decomposition.

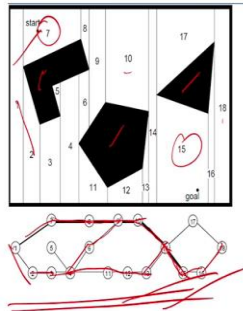
So, the basic abstraction behind such a decomposition that particular position of the robot within each cell of free space does not matter. So, we do not really worry whether the robot is in a particular position within the cell as long as the robot can actually reach that cell, then it can actually go to the next cell.

That is the understanding here. So, the robots ability to transfers from one from each free cell to the adjacent free cell; so, as long as the robot can move from one free cell to the other free cell, then it is a feasible path.

So, this is basically the exact cell decomposition method of path planning. So, the efficiency depends on the number of cells and the number of obstacles and basically number of obstacle decide number of the cells also. So, as this is very dense with obstacles, then you will be having large number of cells and that may become little bit complex for the path planning; otherwise, it is a good method of planning.

(Refer Slide Time: 07:03)

## Exact Cell Decomposition



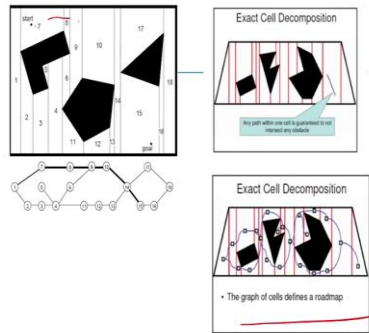
So, as you can see here, now I have this obstacles, these are the obstacles and then, you have this cells identified. So, now, you can see if this is the start. So, you can see 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 11, 12, 13, 14, 15, 16, 17 and 18. There are 18 cells which are not occupied and therefore, you can identify suitable paths along these. So, you can see that can be 1 to 7, can have 1 to 7 or 1 to 2. You can have 1 to 2, then 2 to 3, 3 to 4, then 7 to 8, 9, 10.

So, you will be able to create a graph like this in order to plan the path. So, this actually gives you the all feasible paths from the cells unoccupied cells. Now, if you want to travel from 7 to 15, you can find that 7, 8, 9, 10, 14, 15 is one path or you can take another path like this also and then, reach here 15 or there are multiple ways you can reach. And which one is the shortest depends on again other criteria which we need to decide.

But this is the first thing, where we create the graph and then, say that ok. These are the potential paths existing or the possible paths existing from the start to goal position. So, that is basically the exact cell decomposition ok. So, these are the road maps that you can create using the cell decomposition.

(Refer Slide Time: 08:31)

### Exact Cell Decomposition

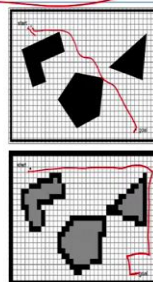


So, the advantage is that you will be having you can actually get or whether it is occupied or unoccupied so that you can always plan the path without any cells having any obstacles. But, that actually creates a bit difficulty because of the cells the way it is being arranged and sometimes you may find it very difficult to decide from which point this actually move to the next one, because the cell size may be bit big compared to the other methods.

(Refer Slide Time: 09:18)

### Approximate Cell Decomposition

Fixed-size approximate cell decomposition:



And to avoid that one, we go for something called an approximate cell decomposition, so that you will be having much more capability for to plan the path in a much more efficient way. So, here what we do? We will fix the size of the cell and then, divide the

whole area into grids and then, look at which are the grids occupied fully and which are the occupied grids occupied partially.

So, we will have partial occupation and full occupation of the grid and then, we will decide to go from start to goal based on the nearest grid. You try to find out the nearest grid and then, start moving and whenever there is a next grid is occupied, you try to move to the unoccupied grid and then, keep moving like that way you will be able to find out a path.

So, you will be able to get a path from here through any of these grids and then, reach the goal and again you will be having multiple possibilities to go through the grids and reach grid and reach here. So, that is the approximate cell decomposition. So, approximate cell decomposition has got its own benefits.

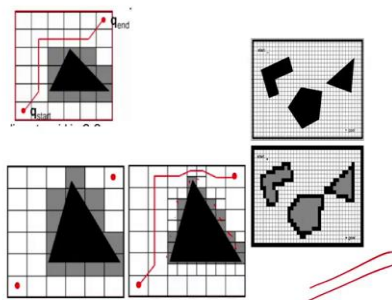
Because you will be able to get a much better search using the approximate cell decomposition. But the number of grids and number of cells increases and that may make the computation really little bit costlier; otherwise, it is a, good method of decomposition and then, getting the graph.

(Refer Slide Time: 10:55)



### Approximate Cell Decomposition


Fixed-size approximate cell decomposition:



So, as you can see here, you will be always using the unoccupied cells only, you will be always marking the partially occupied cell also, which is not which will not be using. It

will be going along the with look for the completely free grids and then, start a path or plan a path along that those grids. That is the approximate cell decomposition methods.

(Refer Slide Time: 11:37)




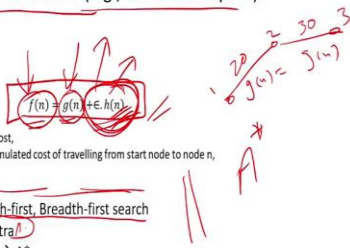
### Deterministic Graph Search

The environment map has been converted into a connectivity graph using one of the graph generation methods presented earlier. Whatever map representation is chosen, the goal of path planning is to find the best path in the map's connectivity graph between the start and the goal, where best refers to the selected optimization criteria (e.g., the shortest path).

$f(n) = g(n) + \epsilon \cdot h(n)$

$f(n)$  = expected total cost,  
 $g(n)$  = path cost (accumulated cost of travelling from start node to node  $n$ ),  
 $h(n)$  = heuristic cost

If  $g(n)$  is constant  $\rightarrow$  depth-first, Breadth-first search  
If  $g(n)$  is varying  $\rightarrow$  Dijkstra  
If epsilon not equal to zero  $\rightarrow$  A\*



So, now you can see that there are multiple methods to find out the to have a graph. So, basically you are trying to create the road maps from start to go using multiple methods. So, once you have this graph, the next question is to ok. How do I identify the best path or the most feasible path from start to goal and for that, we go for a graph search, ok.

We go for deterministic graph search, where we assume that we know the values of travel cost and other things and based on that, we search within the graph and then find out the best path. So, that is basically the graph search. So, we already have the connectivity graph using one of the graph searching methods. And then, the goal of path planning is to find the best path in terms of the connect in the maps connectivity graph and the best refers to the selected optimization criteria.

So, you can have it as a shortest path or the shortest time or the shortest cost of travel, whatever may be the criteria you use that becomes the best method, best path. So, we based on this criteria, we can actually have a an algorithm to search. So, normally what we do? We will have a expected cost and this cost will be minimized. So, the cost it can be the cost of travel, time of travel, distance to be traveled whatever it is.



So, we define a function, an expected cost function as  $f(n)$  and then, try to minimize this and try to find a path which gives you the minimum cost expected total cost, that is a search. So, you search in all the possible paths and then find out which one gives you the minimal cost that path will be the best path for you or that is the most feasible path for your application.

So, what we do? We have an expected cost  $f(n)$  and we will define it as a sum of  $g(n)$  which is the travel cost or the path cost, accumulated cost for traveling from start node to  $n$  nodes plus we define some more thing as heuristic cost  $\epsilon \times h(n)$  ok. So, the total cost will be  $f(n)$  and then,  $g(n)$  will be the path cost from moving from one point to the other point. And then, we will have an additional cost called the heuristic cost which can be added or it can be neglected also.

But this heuristic cost will be based on what additional criteria, you want to give in order to search for the best possible path. So, if  $h(n)$  is not there, then we put  $\epsilon = 0$ ; that means, its only  $f(n)$  is equal to  $g(n)$ , which is the total cost of travelling from point to point or if there is a  $h(n)$ , we want to be included, then we put a  $\epsilon = 1$  and then, we consider the heuristic cost also.

So, I will explain this when we take some examples of graph search methods. But primarily, we want to define a function of cost function which can be used for by finding out the best path, ok. So,  $g(n)$  is constant, then we have multiple algorithms available. So, when we keep  $g(n)$  is a constant that is traveling from one point to the next point, the next point if the cost of this  $g(n)$  is constant between this each one of these.

So, this one is equal to this cost is same, then we have many methods called depth-first, Breadth-first search methods. When  $g(n)$  is varying, then we have something called a Dijkstra algorithm and then, when  $\epsilon \neq 0$  that is when  $\epsilon = 1$  or you additional consider a additional heuristic cost, we have algorithms A star. Of course, there are multiple algorithms available for graph search in the literature.

I am just talking about only very few, where we have this something called depth-first and breadth-first approach in which we assume that the travel from one grid to the other grid is the cost of traveling from one grid to the other grid is always constant. There is no variation between grid to grid travel that is the depth-first and breadth-first approach.

And then, we have when this  $g(n)$  is varying; that means, the travel cost is this money may be 20 and traveling from I mean 1 to 2, 2 to 3. I said this is 20; but 2 to 3 is 30.

That means, the costs are not same between the grids, then that is  $g(n)$  is not constant and then, that kind of then in that case, we have some algorithm called Dijkstra algorithm. And then, even if it is varying or constant and still you need to have a  $h$  also that is when  $\epsilon \neq 0$ , then we called we have an algorithm called A star algorithm. Of course, there are many more such algorithms available in the literature.

So, we will look into these algorithms and then, see how these algorithms can be used for searching the most feasible paths. So, we will look into this in the next class and then, see how we can actually identify methods or how can we use these algorithms for finding out the most feasible path or the optimal path from start to go. That discussion, we will have in the next class.

Thank you.