INTELLIGENT CONTROL OF ROBOTIC SYSTEMS

Prof. M. Felix Orlando

Department of Electrical Engineering

Indian Institute of Technology Roorkee

Lecture 19: Sampling based Path Planning Methods

Good morning. Today, we are going to see a lecture on sampling-based path planning methods. First, let us see the introduction: why we need sampling in path planning. Sampling algorithms are well-suited for high-dimensional configuration spaces where traditional grid-based approaches become computationally infeasible. These sampling-based path planning approaches can handle large and complex environments, adapting well to different problem sizes without a significant increase in computational resources.

These algorithms are versatile and can be easily adapted to various constraints and environments, including dynamic or partially known spaces. Today, we are going to see the key algorithms in sampling-based path planning, which include rapidly exploring random trees, shortly called RRT. These algorithms efficiently explore the space by incrementally building a tree structure from the start node. RRT Connect is an extension of RRT that aggressively tries to connect the start and goal nodes by improving the convergence speed. The third approach we are going to see today is RRT*, where the star indicates the optimal path solution obtained from this algorithm, and hence RRT* is an optimization-based RRT.

It refines the tree over time to ensure an optimal path. Finally, we are going to see probabilistic roadmap (PRM), which creates a global graph by connecting randomly sampled nodes in the free space. This is suitable for multi-query planning. First, we will see rapidly exploring random trees (RRT) today. The introduction goes as follows.

It is an incremental sampling and search-based approach. And here, no tuning parameters are required. This approach incrementally constructs the search tree from the start node. And the exploration here in this approach is that in the region of interest, the tree densely covers this space. This method is very suitable for high-dimensional space.

So, using this method, motion planning for both algebraic constraints, which arise from obstacles, and differential constraints, which arise from non-holonomic dynamics, can be addressed. Coming to the definition of RRT, a dense sequence of samples is used as a guide

in the incremental construction of the tree. The main goal here in RRT is that it incrementally constructs a tree that finally connects the start point to the goal point by backtracking. If the sequence is random, the resulting tree is called a rapidly exploring random tree. The main idea in RRT is to bias the exploration toward unexplored portions of the space.

This family of trees, whether the sequence is random or deterministic, will be referred to as rapidly exploring dense trees (RDT). Now, let us see the working steps involved in RRT. First, we have the construction of the tree. Let us build a rapidly exploring random tree in the search space from the start node.

This is the start node and from which we have the nodes which are connected to the start node thereby we have the initial tree and then we pick a random node from the search space then search the nearest node of the tree to this random node then we are going to extend the RRT in such a way that after finding the nearest node, say this is the nearest node to this random node, extend from the nearest node towards the random node by small steps as long as possible by avoiding the obstacle. Then connect this new node as a vertex and edge of the tree. So the tree shape is this is the tree shape now.

Now repeat the steps 2 to steps 5. Step 2, 3, 4, 5 to explore the search space by extending more and more nodes of the tree. So, this is the animated graph showing how the tree is getting built or incrementally built from the initial node. Let us say that this is the initial node. and we found a random node and nearest node is this one so move towards the nearest node as long as the path is free from obstacles till that you continue then you continued here till that it becomes the vertex and edge of the tree now with this new node

the tree is extended incrementally. So, the iterative process from step 2 to step 5 will enable to explore the search space. Once the RRT reaches the goal, say this is the goal and this is the start point. So, it starts like this.

So, once it reaches the goal after starting and exploring the space, we backtrack the path from the goal point to the starting node. That is how we get the path. But this path is not an optimal path. So the algorithm goes here. We need to build the RRT tree, or rapidly exploring random tree, with the initial node point.

The tree initially has that initial node point, as we have seen in the schematic. This is the initial tree with the starting node and the tree nodes. Now, we are going to run in order to explore the search space around this tree to reach the goal point, which is here. And we

need to explore this search space in order to build this tree incrementally. To reach the target or the goal point, we need to iterate a great number of times, say K, so that the iteration starts from 1 towards K. For that loop, you have to generate a random node using the random state function.

And extend towards the random node from the nearest node of the tree. The extension must also take care of obstacle-free space. Which means the space connecting the nearest node and the random node must be free of obstacles. If the tree, which is the updated incremental tree, reaches the goal, then the stop condition terminates the search process. Then we return the tree, which has branches leading to the goal point.

And the extend function is such that it receives the t with the initial point. And find the nearest neighbor, then it continues to add this vertex and a random vertex, and you increment it to form the edge. If the nearest node and the random node are reached, then you return a reached statement; if not reached, And if there is no obstacle between the current state and the target state or the random state, you continue towards the command advance. Once it is fully reached, return trapped, then go to the next incremental iterative process to incrementally build the tree towards the goal point. This is the process to build The programming part to have this algorithm getting run and built. Now, let us see the second path planning approach in our lecture today, which is RRT connect.

So, the introduction goes like this. This method constructs incrementally two RRTs from the starting node and from the goal node. The trees each explore space around them and also advance towards each other through the use of a simple greedy heuristic approach, thereby exploring the space very well. One tree will have the start node as its initial point. Another tree will have the goal node as its start node.

And each tree is extended to reach the target of the other tree. Let us see this algorithm in the next forthcoming slides. And this approach grows the trees towards each other rather than towards random configurations. So the tree growth, incremental tree growth, is confined towards the other tree. For this tree, which has the initial point as its starting node, and another tree, which has the goal point as its starting node, will have the tree built incrementally towards the other tree from its point of view.

And the path is found when two trees meet. In RRT connect, it grows with multiple epsilons so that the greediness becomes stronger and the two trees advance faster towards each other. Once it advances faster towards each other and reach and gets connected, we will stop this process of searching further. This technique is more probable to find a solution

faster with respect to the normal RRT approach. The RRT Connect planner is useful for path planning problems that involve no differential constraints.

Let us see the working steps associated with RRT Connect. Let us see the first step is the construction of tree. We build a rapidly exploring random tree in search space from both nodes which are starting node and goal node. From the starting node, a tree is built initially. From the goal node, a tree is built initially.

Now, we have to have this algorithm or RRT connect in such a way that this tree is extended and this tree is extended and if they meet then we need to stop that process so here pick a random node in the search space for both trees to explore in the search space random node for a tree which has starting node as its initial node and a random node for tree 2 which has the goal node as its initial node Then extend RRT as in the case of simple RRT approach. So, extend from the nearest node towards the random node by a small step as long as possible by avoiding the obstacle. So, nearest node.

So, keep going towards the random node. And similarly, random node here, nearest node is this one. So, keep moving towards it. As long as it is not avoiding the obstacle. This is the new node now.

Now connect and make that new node as the vertex and edge of this tree and correspondingly for the starting node based tree as well. Now the tree is extended to incorporate the new nodes. Now these new nodes become the target for the other trees other tree. What I mean to say is here for the starting node based tree this new node is the target for this tree which is having the gold node as its starting node. Similarly the new node of the gold node based tree will be the target for the tree starting

That starts from the starting node or the initial node. Now, find the node nearest to the target node. For this target, the tree with the starting node has this node as the nearest node. Now, increment towards the target node by delta. As long as we are not encountering any obstacle in between the path.

Until then, you continue and make that the new node toward the target node. Then connect it and extend the tree. In the same way, we can have the connecting subroutine, which has an extension and an advanced path. So the extension function will make the tree extend toward the random node. So here we have to extend and connect it.

Extend and connect. Once we have this extend and connect, then we have to swap the path, which means ta and tb. Then we have this swapping-based approach, in that way, we go

for extending ta. That means ta, let us say ta is the one with the start node as its initial node, and tb is the tree with the goal node as its initial. So you start extending one to reach the new node as its target. Then, once it is reached, you can swap it and continue this procedure.

If not, return failure and continue freshly with the new iteration. That is all. Now, let us talk about something called RRT star. As I indicated, the star is meant for the optimal path. The introduction to this RRT star is such that both RRT and RRT connect methods are not able to find the optimal path between the starting node and the goal node.

But the RRT start planning approach is ensures asymptotic optimality and hence this approach will find the optimal path. This RRT star method is useful to deal with differential constraints as well. There are two main techniques involved in this approach of RRT planar. One is path optimization, another one is intelligent sampling.

Now let us see the working steps associated with this RRT star. First step as the construction of tree which is build a rapidly exploring random tree in the search space from both nodes that is starting and goal node. And then pick a random node in the search space. This is a random node. So, here what happens we need to find the node of the tree which is closest or nearest to the random node and then we are supposed to approach towards this random node by small increments in the space without colliding the obstacles.

Now instead of directly reaching this new node now instead of connecting this new node to become a vertex and edge of the tree we are going to have the vicinity of this new node which is red in color and in the neighborhood we are going to pick some random So that from the starting node we get a measure to reach this nearest node to the nodes in the neighborhood of the new node. Then we see which length is minimum in the neighborhood of the new node. the path with the smallest length, minimum length is finally chosen and the other paths connecting the nearest node to the other nodes in the nearest vicinity or the neighborhood regions are ignored. So,

what remains here is basically this part of the tree now because this new node is now obtained by having a neighborhood region from the new node and those neighborhood region based nodes are connected to the initial node And based on the path length which is minimum, we finally connect it and ignore the other paths. In this way, we continue to reach the goal point. So that the path length to reach the goal point is minimum. So here...

Repeat the steps 2 to 6 to explore the search space by extending more and more nodes to the trees. The iterative process from step 2 to step 5 will able to explore the search space.

And again, once the path or the path reaches the goal or the tree is extended in such a way that it incrementally reaches the goal, Then backtracking will be done along the tree to identify the edges that lead from start node to the goal node. The algorithm has these type of subroutines which is we initialize the tree and name it as capital T which is having the initial tree nodes and

Then we insert node to the tree by having the iteration which involves a search in the search space, deeper search in the search space. By selecting the random node using sample function, we can find the nearest node in the tree from the sample function. random node then we need to steer that nearest node to the random node by a control input which is u nu u nu is the control input that connects the nearest node to the random node and that incremental path is given by x nu Based on delta. To reach towards.

The random node. And based on the. Obstacle checking. We can continue this process. And in the process.

We need to have the. Neighborhood. To be chosen. In such a way that. For the random node.

We are approaching. And once the node is ready to get connected, instead of connecting it directly as in the case of RRT and RRT connect approaches, we take a neighborhood of this new node. And in the new node neighborhood, we have chosen several nodes. And from that node to the initial node we connect and see the length which is minimum that will be the final node thereby ignoring the or avoiding the other nodes. And finally by inserting node and rewire subroutines we can finish and explore entirely till n number of iterations to explore the search space deeply we can return the

T or T, which has all the nodes associated with the tree, will be in that structure T. Then, these are the algorithms or subroutines associated with finding the RRT star-based approaches, which are 'choose parent' and 'rewire.' Choose the parent with the minimum distance from the neighborhood nodes and, after finding that, connect by rewiring functions. That is all. And now, let us move on to the probabilistic roadmap, which is the last topic of today's class.

It is shortly called PRM. Here, it is a random sampling-based approach, and this random sampling is used to generate nodes in the configuration space. You have a space—a search space—randomly select the nodes. By connecting them, you form a graph.

Finally, choose a path from the graph, which will be the end solution of this PRM. How do we choose that path? PRM constructs an indirect graph, which is a roadmap where nodes represent collision-free configurations and edges represent valid paths between them. Nodes are also termed vertices. Edges are the lines that connect the nodes.

And this PRM provides global convergence in the free space, and it is suitable when there are multiple goal points. This approach is very effective for high-dimensional spaces as it samples the space probabilistically. Let us see the working principle of this approach. A random node or point is generated in the configuration space. So, let's say we are in the middle of the algorithm.

These are the obstacles, and these nodes are already selected by this approach. Now we are in the middle of this algorithm, and here a new node is selected from the search space. And now we need to check whether the new node lies in the free space or not. If the new node is in the free space, consider it. If the new nodes are lying in the position of the obstacles, it is better to ignore them. So, as per the schematic,

Here is a new node, here is a new node, here is a new node. And these two red nodes are invalid nodes because they are located in the position of obstacles. This means they are already colliding with the obstacles. And hence we have chosen this green node, which is a free node without collision with the obstacles. And now this newly generated node is then connected to the closest nodes through straight lines.

Closest note: this, this, and this. Say this, this, this. And you connect them with straight lines. And we are going to say valid connection and invalid connection. As you can see here clearly,

there are three lines from the new node to the existing nodes that are valid because they do not have any interference from the obstacles. Whereas there are two nodes or two lines that are getting interfered with or interrupted by the obstacle. And these three lines connections—straight line connections—are without obstacles, and hence these three are only considered, whereas the obstacle-interfered line connections are ignored. The iteration for this new node ends here. Now we connect them, then this is the iteration. For this new node.

And we connected them successfully to the roadmap. So the roadmap is now extended. And here, you add more nodes.

As per your requirement. And repeat this process. Let us have this. As the goal node. And you have your

tree extended. From the beginning. This is the roadmap extended. For executing the path, we can use one of the various standard search algorithms, such as the breadth-first search approach, which uses a first-in-first-out strategy; the depth-first search approach, which uses a last-in-first-out approach; and Dijkstra's algorithm to find the path between the two nodes. And you may recall what the breadth-first search approach is, which means it starts here and goes layer by layer.

So this is the first layer, and then if there are other nodes, the second layer. So layer by layer, the breadth-first search method moves forward to extend the path. So this path which is shorter—that is the path in the breadth-first search approach. Then the second layer.

So, connect it. Then which is shorter, that will be the path chosen. Whereas in the depth first search approach, you take the path like this. And which is the shortest that will be chosen in the depth first search approach. Now it has to be noted that the optimal path found using graph search algorithms may not be optimal in a global sense.

Which means the path that we have obtained here in PRM using one of the standard search approaches will be local optimum will not be global optimum that is the note one has to be taken care of and the algorithm goes in this manner which we have seen we have chosen a node in the search space and the path is extended through straight lines and the path extended to the new node will be through shorter nodes connected without collision free space then you connect it and return the graph which is connected with the new node then generate a random node And return the random node. Then.

We find the nearest. Neighbors. By connecting through straight lines. And avoiding the obstacles in between them. Then.

Connect it. To the existing path. Then you continue. To reach the goal. So.

In this lecture. As a concluding remark. I want to say that. Here we have seen the sampling-based path planning approaches, mainly three topics: Rapidly Exploring Random Tree, Rapidly Exploring Random Tree Connect, and Rapidly Exploring Random Tree Star, which are RRT, RRT Connect, and RRT Star. Among these three approaches, RRT Star will give the optimal path.

To reach from the start point to the goal point. Finally, we have seen in today's lecture the Probabilistic Roadmap, which provides a locally optimal solution to reach the goal from the start node. In the next class, we will see the application of these approaches for a minimally invasive surgical needle. For percutaneous cancerous intervention. Thank you so much for your time.