# Motion Planning

This report focusses on solving a unique motion planning problem related to moving a group of joined robots from start configurations to goal configuration going through rectangular obstacles in a normalized Cartesian plane. The program uses Probabilistic Roadmap planner to find an optimal path between a start and goal configuration. The program is developed in two phases. First phase is to generate random configurations in workspace close to obstacles in the world. The second is the query phase to connect these randomly generated configurations and find an optimal path.

## Defining C-space

The workspace here is a normalized Cartesian plane with rectangular obstacles. The configuration space is simply the free space where a given robot configuration can be moved. With the restrictions of shape, minimum area and max length between robots, the c-space is all the possible movements with n degrees of freedom where n is the number of robots in a particular configuration. Moreover, it is the set of all possible configurations reached by the robots. In this particular case, the movement is restricted by above mentioned constraints.

The problem defined here is represented as (q0, q\*, O) where q0 is the initial configuration and is represented by where d is the degrees of freedom for a robot configuration and goal configuration . Each obstacle o belongs to O and contains a set of collision configurations . For the set of obstacles O, the collision free configurations are defined as

## Search method in continuos space

As the workspace is a normalized Cartesian plane, I have used java’s 2D framework to find a collision free path from a start configuration to the goal configuration. Every point in this Cartesian plane is represented by Point2D class. The points can be joined together by a Line2D object. For a collision free path, I have taken the head (first point robot) of the start configuration and the head of goal configuration and join those with a straight line. If the line connecting these two points does not collide with any of the obstacles in the world, the path is a straight line.

If the line collides with one or more obstacles, Probabilistic Planner is used to generate N random configurations along the obstacle bounds. These configurations are then connected and assign their respective children. The children have just one property, they are reachable from parent. After that, A\* search on the graph is performed to find the optimal path from start to goal. The search graph uses heuristics as the Euclidean distance between current configuration and goal configuration.

## Details of discretization method

The problem space is continuous. It is discretized by using the maximum constraints available in the solution requirements. The workspace is divided in points where each point has a maximum distance of 0.001. This allows to have a finite workspace and the movement can be restricted to this constraint.

The search method gives the collection points which can be joined with straight lines to reach towards the goal. In the movement phase, these straight lines are then divided into finite number of points distanced as maximum step constraint (0.001). Now that I have all points on a line and the points are 0.001-point distance away from next and previous points, I use forward kinematics approach to move the configuration along the path.

The kinematic movement on each step is then restricted to achieve a valid configuration. These restrictions include checking collision, bounds checking, minimum area checking and inter-robot distance checking. The degrees to which a particular robot can rotate along its neighbor is restricted by a maximum expansion angle. Rotation is performed on both directions and a valid rotated configuration is selected. The configuration is then redrawn with new polar coordinates and configuration validity is performed. The random polar coordinate shifting is performed till the time target configuration is reached. The manipulation along the polar coordinates in performed as the configuration is moved each step.

The main part of selecting the children of an ASV configuration is to check whether the 2 configurations are reachable or not. For this purpose, 2 things are checked, first is that every point of respective second configuration can be joined together with straight lines and the second is the difference of angles between two points from a normalized center should not exceed 50 degrees in Cartesian coordinates.

The kinematics approach is used because it is more efficient than any of the roadmap planners. This is true because the movement is restricted to cover a very small magnitude on each step. This allows the step movement to be restricted within bounds and there is a very high probability that a valid configuration is achieved with minimal number of iterative movement checking.

## possible solution for class of scenarios

The method used to solve the problem is limited in terms of the size of ASV configurations. This means that the problem is more likely to get solved for fewer number of ASVs even in cluttered environments. But as the number of ASVs increase, the movement possibility for forward kinematics become limited. The configuration can still be moved and taken to goal but it will require to use some extra movement strategies that can sense if the configuration is stuck somewhere between obstacles and will recover. But this restricts the number of connections in the graph and hence the probability of a full connected graph towards goal is low.

This accounts for the problem of solving the problem with narrow passageways. When a configuration is formed with the convexity direction towards one side, it is difficult to pass through the narrow passage if the target point lies in the opposite direction of the convexity direction. For example, if the configuration is stuck between two obstacles with narrow passage, with convexity direction facing downwards, it is difficult for this method to reach to a point that lies above the configuration without violating any of the constraints.

The path where on each point, the obstacles are at a distance equal to minimum diameter of the ASV configuration, the program will be able to move the configurations freely in the workspace.

As the Probabilistic roadmap used in this program will find enough points closer to obstacle bounds in order to have a free movement of ASVs from one configuration to another. There is a high probability in this scenario to find a path in open environments

## failure of the program

The method used in the program has limitations when it comes to space available for ASV configurations to transform from one configuration to another. This means that if there is not enough space available for ASV configuration to transform, the program will get stuck and the goal configuration will not be reached. Unfortunately, the program does not implement any failsafe strategy to know if the goal configuration can be reached or not. The path is based on a minimum estimate of the possibility of solving the problem.

The search method will not always find a path from a start position to goal position if it exists. This is true for the fact that path finding is dependent upon number of ASVs and the obstacle bounds. The path finding only accounts for number of ASVs when calculating rectangular obstacle bounds. So, how much further a possible point can be from an obstacle depends on the minimum area of ASV configurations.