**Path Planning for Mobile Robot with trolleys**

**in Indoor Environment**

**Introduction**

Path planning is the method of finding the optimal path from one point to another in space. It optimizes the path between source and destination by determining shortest path between them. It is sometimes also termed as motion planning as it helps to decide the motion of any object in an environment. An object can be a robot which is autonomous in nature as it makes use of the path finding algorithm to determine its traversing points in space. Such a robot is referred to as mobile robot. Path planning can also be defined as the process of breaking down a desired path into number of iterative steps to make discrete motions to optimize some entities. Here we address Path Planning in a static environment for non-holonomic car like robot, with trolleys.

**Problem Formulation**

The Path Planning problem can be stated as follows: Given a map ***M****,* find the cost-optimal path from a vehicle’s current location ***agv\_initial*** including the pose*,* to a goal location ***goal*** subject to the constraints given by ***Vehicle and the Trailer Kinematics***. The discrete grid map ***M*** is an occupancy grid obtained from /map\_server node in ROSconsisting of values – [0, -1, 100] where, 0 – free space, -1 – unexplored region and 100 – obstacle.

The Path generated should be optimal (least cost), traversable (considering vehicle constraints), collision free (from static obstacles) and inbound collision free (collision among trolleys and robot).

**Proposed Approach**

***A\**** algorithm, which is limited to discrete state spaces is used for generating a *Cost Map* i.e. a grid, with each cell having a cost to reach the goal. Along with generating a cost map it’s also used to identify whether a goal is ‘Reachable’ or ‘Not Reachable’.

***Hybrid A\**** algorithm is used for generating a continuous path in the region searched by A\*. Each node *n* of the search graph i.e. the state of the robot is completely defined by,

robo\_state = (x, y, psi, v, w, vr, vl, )

**Related Diagrams**