Roadmap-Based Motion Planning in Dynamic Environments

To implement path planning of any robot or
robots in configuration spaces with dynamic and static obstacles.

Author Name
Harsh Bharat Kakashaniya
Koyal Bhartia
Aalap Rana

We use different path planning algorithms to find path of robots in
dynamic environment to simultaneously start from different points and reach different goals.

Date : 7/3/2019

Contents

List of Figures

1 Introduction

Motion planning is a term used in robotics for the process of breaking down a desired movement task into discrete motions that satisfy movement constraints and possibly optimize some aspect of the movement.[1] As mentioned in above definition that the desired movement must satisfy the constraints , there are two main kind of obstacles in planning which are namely static obstacles and dynamic obstacles. As the name suggests that static obstacles have a fixed position and orientation in configuration space which does not vary with time,in contrast to dynamic obstacles. Numerous planning algorithm have been suggested and successfully implemented for motion planning with static obstacle space but due to the complexity associated with dynamic constraints this part of planning is relatively untouched and has huge scope of exploration.

This project aims to find an optimal solution to the planning problem which has both static and dynamic obstacle in the environment with no constraints in movements, provided we have prior knowledge about the initial position of the obstacles. This paper uses probabilistic road-map planner (PRM) algorithm for finding the feasible path. [3] The main principle behind PRM algorithm is that it takes random samples from the configuration space of the robot, tests them for whether they are in the free space, and uses a local planner to attempt to connect these configurations to other nearby configurations. The starting and goal configurations are added in, and a graph search algorithm is applied to the resulting graph to determine a path between the starting and goal configurations.[2] The implementation of PRM algorithm is done in two stages. The first stage is known as construction phase where a road-map is formed by approximating the possible movement,and then it is connected to neighbors depending on the condition mentioned as parameters such as the nearest neighbour. The configurations and connections are added to the graph when the network becomes dense. The second stage is the query phase, where the path is created for movement from initial state to goal state. The algorithm is highly proficient for fixed obstacles, however when applied to moving obstacles it produces undermined results as the configuration space is extremely transient. Therefore, the implementation of traditional PRM algorithm is slightly modified by using it in levels. The first level is dedicated to road-mapping for static obstacles which is pre-processed without additional dimension of time and the second level does planning in query state for dynamic obstacles. Hence in this level we take into consideration the space and time parameter simultaneously.[4]

The proposed method is effective for both free-flying robots and articulated robots which have six degree of freedom.The algorithm can also be applied to multiple-robot motion planning problem in a confined dynamic environment.The proposed algorithm can be applied to robots which can be used in real situations at industries, airports, restaurants and hospitals.

2 Method

Following is a brief overview of the steps that will be implemented in the project. The broad two approaches are:

1. **Local Approach:** Also known as the pre-processing phase, This approach basically considers the objects to be stationary i.e the map is static in real time. The roadmap is built for this static part of the scene without the dynamic obstacles and without any additional dimension for time. A two-level approach is used to find the trajectory of the objects. Here the trajectories on single edges of the roadmap are found in an implicit grid in state-time space. The global approach is an extension of this

local approach. On the global level, the local trajectories are coordinated using an A*-like search to find a near-time-optimal global trajectory in the entire roadmap.

2. **Global Approach:** To find a trajectory from a point to another, algorithms like Dijkstra search does not solve the purpose. This is because it is not always best to arrive as early as possible on each of the vertices of the road-map as a scenario may occur that there may exists an obstacle at the next stage. The implementation is as described below:

   - **The State-Time Grid** The distance travelled by the robot along the edge of a roadmap, represents the robot's configuration as a variable x. This ranges from 0 to 1 corresponding to the source and destination respectively. Thus the resulting state-time space followed by the robot is 2D consisting of the pair (x,t) $\in$ [0,1] x [t_0,∞] Following is an illustration of the same:

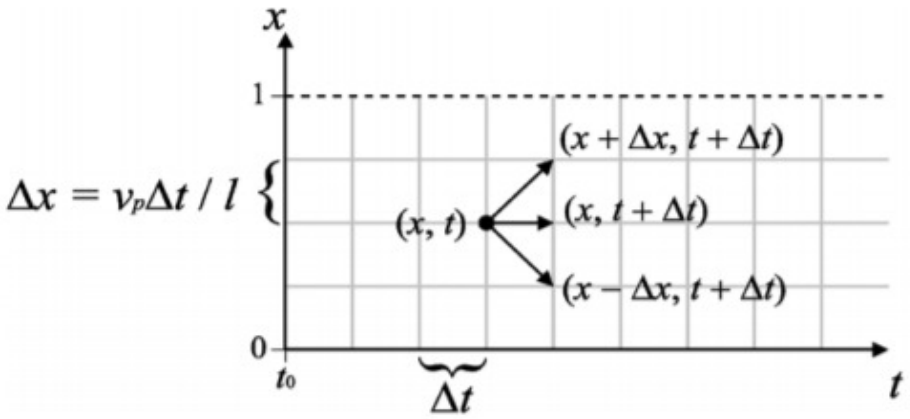

Figure 1: State Time grid of a roadmap edge

   - **Finding a Local Trajectory** Following is the algorithm followed: A path is first chosen towards goal. If there is an obstacle in the path, the second option is for the robot to wait till the object passes and then continue travelling towards the goal. Pre -calculation of the trajectory is done if the object is huge. Once the robot reaches the goal, and then it tracks an obstacle, it comes back if needed and again traverses towards the goal. This has been illustrated below in the diagram. The computation of these steps uses the A* algorithm with consideration of cost of motion.

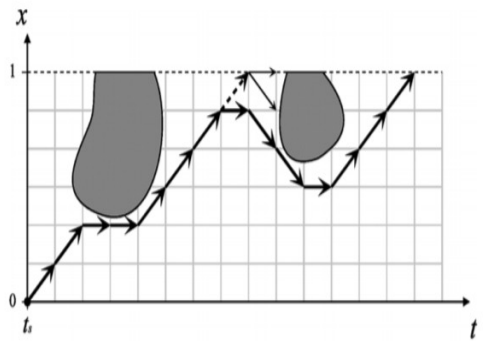

Figure 2: Finding the shortest trajectory from (0,t_s) to x = 1

- **Global Trajectory** In the previous section, computation of a local trajectory on a single edge was discussed. Here, we will show how the global trajectory from a start vertex to a goal vertex through the roadmap is found. As with the local trajectories, the algorithm will find an approximately time-optimal trajectory.

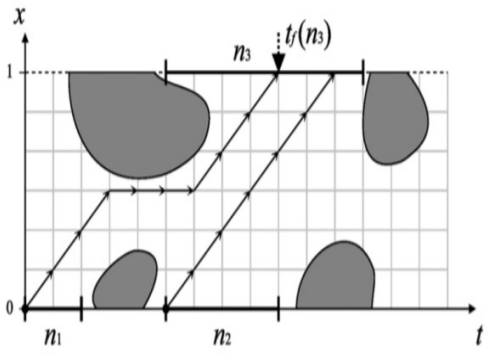

Figure 3: State-time grid showing free intervals on the vertices and trajectories between them

As shown in the figure above, we have a interval tree to track all robots motion. Interval tree has start vertex as roots. For example as shown in the figure robot A which starts from the origin reaches the goal in the time $t_f(n_3)$. Now Robot B which reaches there at some time after $t_f(n_3)$ will not be allowed as robot A is already reached at the same position. This is possible only if A displaces itself from that position. Else robot B will have to wait or take an alternative path to reach its goal. In other words, the first trajectory subsumes the latter.

- **A Probe** The probe is the main conceptual object of our algorithm. It saves priority in path of Robots. It explores the roadmap in search of a global trajectory. During the usage of A8 algorithm to track the probes, the following function is used.

$$f(p) = g(p) + h(p) \tag{1}$$

Here, f(p) gives an estimate of the cost of the time-optimal global trajectory. g(p) is the cost of the trajectory between the start vertex and the current state-time of p , and h(p) is a lower-bound estimate of the cost of the time-optimal trajectory between the current state-time of and the goal vertex. As explained above, the probe concept is used to find the global optimum global trajectory considering collision free path.

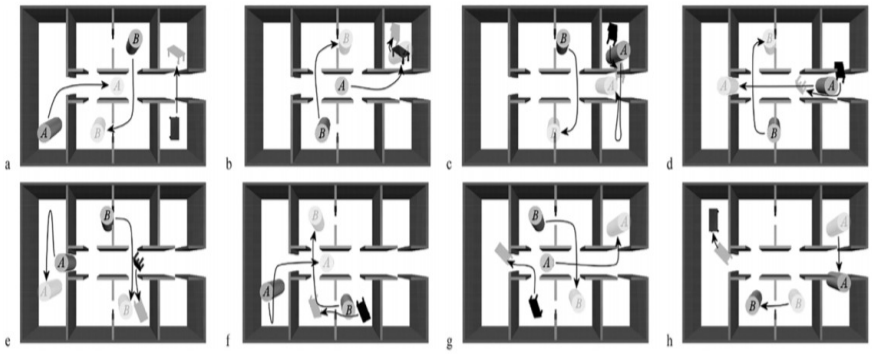

Figure 4: Illustration of Algorithm implementation

3 Assumptions

- There is a upper bound on the velocity of the robot.
- There is no constraints on shape and motion of dynamic obstacles as long as we have prior knowledge of its motions.
- There is no restriction on robot's motion.
- The road map for static obstacles is computed in prepossessing phase without time as a parameter.

4 Goal

The goal of the project is to implement motion planning for multiple robots with any dimension in a confined environment that has both static and dynamic obstacles,using Probabilistic Road-map Planner (PRM) algorithm.The path is formulated in two levels. On the local level the path is found by using depth-first search considering all static obstacles. On the other hand,In global level, A^* algorithm is applied on the local trajectories of a robot which results in conversion of optimum path to state-time space. Above conversion is done for all robots[5]. Hence, finally we choose the optimal path to reach goal state.[6]

5 Software

The algorithm mentioned in the paper is implemented using Python 3.

6 Conclusion

The algorithm works in two level in the local level we use depth-first search to find trajectories and on the global level we have implemented A* algorithm.The method implemented through this paper can over come the drawbacks of Fujimura method[7] for path planning which is applicable only to point robots. The algorithm could be applied to multirobot application where computational time is very critical in path planning.The computation time for finding optimal path is very less when the obstacles are not present or away from the computed path.

7 References

References

[1] "http://en.wikipedia.org/wiki/Probabilisticroadmap"

[2] http://en.wikipedia.org/wiki/Motionplanning

[3] *P. Svestka,"Robot motion planning using probabilistic road maps,"* Ph.D. dissertation, Utrecht Univ., Utrecht, The Netherlands, 1997.

[4] P. Fiorini and Z. Shiller, "Time optimal trajectory planning in dynamic environments *" J. Appl. Math. Comput. Sci.,*. vol. 7, no. 2, pp. 101–126, 1997

[5] M. Erdmann and T. Lozano-Pérez *"On multiple moving objects,"* Algorithmica, vol. 2, pp. 477–521, 1987.

[6] *"Time-minimum routes in time-dependent networks,IEEE"* Trans. Robot. Autom., vol. 11, no. 3, pp. 343–351, Jun. 1995.

[7] *K. Fujimura, Motion Planning in Dynamic Environments. Tokyo, Japan: Springer-Verlag, 1991.*