Two Didactic Tools for Robot Motion Planning Using Visibility Graphs

Hector Gonzalez-Garcia and Victor Ayala-Ramirez
Department of Electronics Engineering, Universidad de Guanajuato DICIS
Carr. Salamanca-Valle, Km. 3.5+1.8, 36700 Salamanca, Mexico
hectorg@laviria.org, ayalav@ugto.mx

Abstract

In this work, we present implementation details of two tools developed to understand the path planning problem using visibility graphs. We present here the motivations to use visibility graphs in the targeted mobile robotics problem, the theoretical basis of the visibility graphs and how to use them in mobile robotics path planning problems and some examples of the use of the developed tools in some specific problems.

1. Motivation

Our goal application is to develop a garbage collector robot. One of the main capabilities needed by an autonomous mobile robot is path planning and execution. As an educational tool to understand the path planning problem, we have developed a didactic system to show how the problem is solved for different scenarios. The system comprises two tools: a scenario generator that provides different scenarios in a random manner and a path planner tool that uses visibility graphs. We have chosen to use visibility graphs as the roadmap representation of the environment because they are efficient computationally and they are specially useful for polygonal environments.

2. Visibility Graphs in Robot Motion Planning

2.1. Visibility graphs

Visibility graphs are a general approach to solve the robot motion path planning problem (see for example [1]).

Main assumption of the visibility graphs is that the environment is composed of polygonal entities. There have been also extensions proposed in the literature to cover the case of non-polygonal entities.

The use of the visibility graph for robot motion path planning consists of two phases: The first stage is the construction of the graph by itself and a second stage having as goal to choose the optimal path between its end points.

The nodes of the visibility graph are the vertices of the polygonal entities in the environment. We connect two nodes if they are visible between them. Visibility between two nodes exists if and only if we can join the associated vertices of the polygonal entities with a straight line segment that does not collide with any other entity in the environment.

In the case of using the visibility graphs for robot motion problems, we need to dilate the obstacles with the robot shape in order to avoid collisions near the obstacles, providing the a safe path. Figures 1(a) and 1(b) show the visibility graph of a particular environment without augmenting the robot shape and after dilation of the robot shape for a particular environment, respectively.

When we are querying for a path between an initial position and a goal position for the robot, we add these points as nodes of the environment visibility graph and we add all the edges connecting them to the visible nodes from them of the roadmap previously computed. Here is when we need to search the graph for the optimal path connecting the initial node and the goal node. For this purpose, we have implemented the Dijkstra algorithm.

2.2. Dijkstra’s algorithm

Dijkstra’s algorithm uses a data structure for each of the nodes in the graph that contains: a label for the node, the distance to the node from the initial node, a pointer to the previous node and a state variable for the node, that can take the values of permanent and modifiable. Our implementation is as follows [2]:

1. To initialize for all nodes: its distance from the initial node as infinity, its previous node as null and its state variable as modifiable.

2. To set initial node data structure with zero as distance, pre-
vious node as the initial node. Set the initial node as the current node.

3. For all the neighbor nodes of the current node, consider all the unvisited neighbors (where the state variable is set to modifiable) and compute the distance from initial node, by adding the distance of the current node to the distance of the edge connecting the current node to the unvisited neighbor. If this distance is lower than its previous value, overwrite the distance and record its previous node as the current node. Mark the current node state variable as permanent.

4. If all nodes have been visited (state variable set to permanent) the search is finished. If not, set the modifiable node with the smallest distance as the current node and repeat step 3.

The shortest path is found by back-tracking the previous node of the goal node up to find the initial node. This path is the shortest path connecting the initial node to the goal node.

3. Implementation Details

In order to study the motion path planning problem, we have developed two software tools:

- i) A 2D scenario generator.
- ii) A path planner using visibility graph.

In the case of the scenario generator, we provide it with the number of polygons and the dimensions of the environment. For each of the obstacles, we generate them using random parameters. A text file interface is also provided.

In the case of the path planner, we use as input the definition of the scenario of the robot, the initial and goal poses of the robot and the shape of the robot.

The path planner is charged of computing the visibility graph of the environment taking into account the obstacle dilation step and of finding the optimal path using the Dijkstra algorithm described above.

The result of this process is a graphical representation of the best path for the path planning problem.

4. Examples

We can observe some examples of using our tools in path planning problems for several scenarios in Figs. 2, 3 and 4. The first scenario (Fig. 2) includes only four obstacles, the second environment has 19 obstacles (Fig. 3) and the third example shows a more complex scenario where there are 50 obstacles (see Figure 4). We can observe here how the graph becomes very dense with the increase in the number of obstacles. Nevertheless, we observe that the robot finds the best path for the path planning problem in each case.

References