

Trajectory Planning

EE395 - Fall 2005

Cesar Barrios

Overview

- 1 - Path Planning
- 2 - Obstacle avoidance
- 3 - Navigation Architectures

1- Path Planning

- Path planning is important for robots to move or perform a certain action in the most effective way in the fastest time, making them more productive and reliable.
- Three main methods are:
 - 1.1 - Road Map path planning
 - 1.2 - Cell decomposition path planning
 - 1.3 - Potential field path planning

1.1 – Road Map Path Planning

- Captures robot's free space in a network of 1D curves or lines. Once the network is constructed it is used in segments for motion planning by finding best segment combination to take robot from start to goal.
- Two popular methods are:
 - 1.1.1 – Visibility Graph
 - 1.1.2 – Voronoi Diagram

1.1.1 – Visibility Graph

- This method first defines line segments between start, goal, and all vertices of polygonal obstacles.
- Then it uses these segments to find the best path between start and goal.
- This method does not take into account size of robot so it may bump into obstacles while trying to follow the determined path.

1.1.1 – Visibility Graph



Figure 6.2
Visibility graph (71). The nodes of the graph are the initial and goal points and the vertices of the non-figurative space obstacles (polygons). All nodes which are visible from each other are connected by straight-line segments, defining the road map. This ensures there are no edges along each polygon's sides.

1.1.2 – Voronoi Diagram

- This method works similar to the Visibility Graph method, but it tends to maximize the distance between the robot and the obstacles in the map.
- The robot may travel a slightly longer path but it reduces the number of bumps it might have during the trajectory.

1.1.2 – Voronoi Diagram

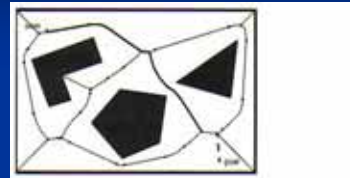


Figure 4.3
 Voronoi Diagram [21]. The Voronoi diagram consists of the lines connecting the set of points that are equidistant from two or more obstacles. The initial q_{start} and goal q_{goal} configurations are marked on the Voronoi diagram as q_{start} and q_{goal} , respectively, by drawing the line along which the distance to the boundaries of the obstacles increases the fastest. The direction of movement on the Voronoi diagram is also selected as the direction to the boundaries increases fastest. The points on the Voronoi diagram represent maximum clearance from the obstacles (maximum clearance between two lines is precisely maximum clearance between a line and a point).

1.2 – Cell Decomposition Path Planning

- The idea behind cell decomposition is to discriminate between geometric areas, or cells, that are free and areas that are occupied by objects.
- It determines path by checking if adjacent cells are connected to each other or not.
- There are two ways for cell decomposition:
 - 1.2.1 – Exact cell decomposition
 - 1.2.2 – Approximate cell decomposition

1.2.1 – Exact Cell Decomposition

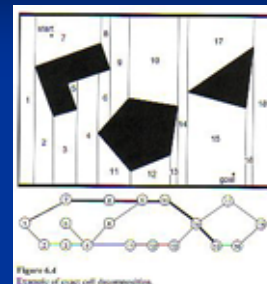


Figure 4.4
 Example of exact cell decomposition.

1.2.2 – Approx. Cell Decomposition

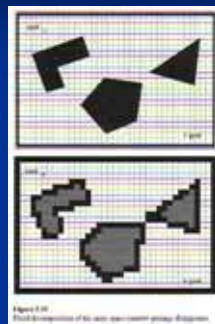


Figure 4.5
 Approximate decomposition of the same environment as in Figure 4.3.

1.3 – Potential Field Path Planning

- This method creates a field across the robot's map that directs it to the goal position from multiple prior positions.
- It works as if the goal pulls the robot towards it, and as the robot moves it is also influenced by the fields around the obstacles, making it avoid them and continue moving towards the goal.

1.3 – Potential Field Path Planning



2 – Obstacle Avoidance

- Local obstacle avoidance focuses on changing the robot's trajectory as informed by its sensors during robot motion.
- Some common algorithms are:
 - 2.1 – Bug algorithm
 - 2.2 – Bubble band technique
 - 2.3 – Lane curvature method

2.1 – Bug Algorithm

- It's perhaps the simplest obstacle avoidance algorithm.
- The basic idea is to follow the contour of each obstacle in the robot's way and thus circumnavigate it.
- Bug1 – the robot fully circles the object first, then departs from the point with the shortest distance towards the goal.
- Bug2 – the robot begins to circle the object, but departs immediately when it is able to move directly towards the goal.

2.1 – Bug1 Algorithm

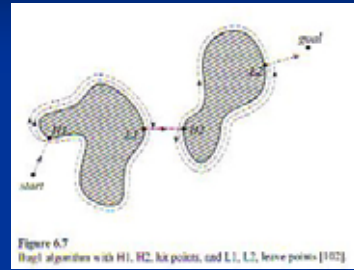


Figure 6.7 Bug1 algorithm with H1, H2, hit points, and L1, L2, leave points [102]

2.1 – Bug2 Algorithm

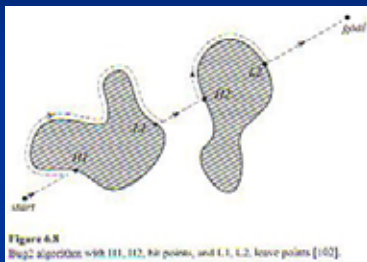
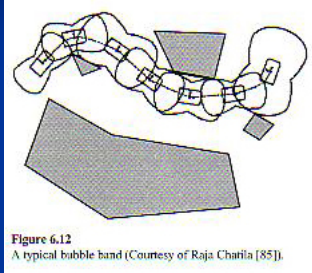


Figure 6.8 Bug2 algorithm with H1, H2, hit points, and L1, L2, leave points [102]

2.2 – Bubble Band Technique

- A bubble is defined as the maximum free space around a given configuration of the robot so there is no possibility of collision when traveling in any direction.
- The path is then calculated making sure the bubble fits perfectly without touching any obstacle.

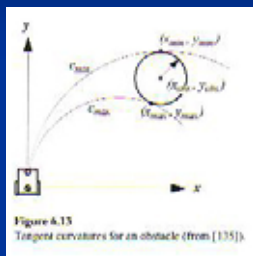
2.2 – Bubble Band Technique



2.3 – Lane Curvature Method

- LCM calculates a set of desired lanes, trading off lane length and lane width to the closest obstacle.
- The best lane is selected and the robot is transitioned to the selected lane if not already in it.

2.3 – Lane Curvature Method



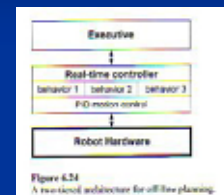
3 – Navigation Architectures

- Given techniques for path planning and obstacle avoidance, how do we combine them into one complete robot system for the real-world application?
- Here are some tiered robot architectures:
 - 3.1 – Off-line planning
 - 3.2 – Episodic planning
 - 3.3 – Integrated planning and execution

3.1 – Off-Line Planning

- Certainly the simplest possible integration of planning and execution is no integration at all.
- It requires a stored scheme to travel to desired location.
- It can not move to a new environment without a new navigation system instantiated.
- Used in manufacturing and space shuttle flights.

3.1 – Off-Line Planning



3.2 – Episodic Planning

- One of the most popular methods in mobile robot navigation today.
- It has previously stored navigation schemes, but it can recalculate its path when enough local information is gathered .
- Recalculating its path frequently can be a serious problem, so methods such as “deferred planning”, or turning robot 90 degrees are used to prevent recalculating path too often.

3.2 – Episodic Planning

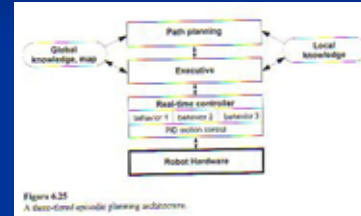


Figure 4.25
A two-level episodic planning architecture.

3.3 – Integrated Planning and Execution

- This method is a combination of the previous two, it combines predetermined paths with local information, so when it detects an unexpected object, instead of recalculating its path, it selects a previously optimized calculated path, making processing time faster and reliable.
- Again, this method is computationally challenging and not practical in very complex environments.

3.3 – Integrated Planning and Execution

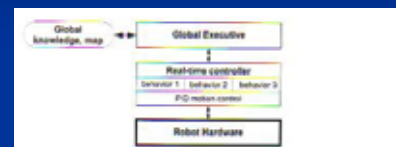


Figure 4.26
An integrated planning and execution architecture in which planning is working more than a real-time execution map (behaviors).

References

- Choset, Lynch, Hutchinson, Kantor, Burgard, Thrun, “Principles of Robot Motion”, Chapter 11, 2005.
- Siegwart and Nourbakhsh, “Autonomous Mobile Robots”, Chapter 6, The MIT Press, 2004.

Topic Question

- What type of architecture would be more appropriate for a vehicle collision prevention: an Episodic architecture or an Integrated Planning and Execution architecture? Think about the advantages and disadvantages of both and also factor in the available processor speeds we have today. Try to consider also a compromise between the two, how much could be predetermined so it does not have to be calculated in real time?