**Motivation:**

The principle system requirements that impacts the path planning team are:

Req 1: The system shall perform a search for targets in an unknown environment

1.2: The system shall navigate unknown environment, avoiding collisions with obstacles

Req 4: The system shall be autonomous

These suggest important derived requirements:

1. We are operating in an unknown environment, therefore, re-planning will be necessary as obstacles are found
2. Therefore, computational load must be a minimum such that the time required for re-calculation of the path is on the order of milliseconds

**Proposed solution:**

While the overall search pattern has not yet been finalized, with no prior idea of where a target is the entire space will need to be searched. Therefore, it is likely we will employ a simple back and forth swath search. In order to navigate around local obstacles however, a path planning algorithm becomes necessary.

In order to achieve necessary planning time, a 2D path will be calculated in lieu of a 3D one.

Two algorithms are being evaluated, A\* and PRM.

**Algorithm Details:**

A\*/Voronoi

A\* is a widely used algorithm which uses weighted cost functions to determine which node to choose.

Cost:

Where g(n) is the known cost of getting from initial to node n and h(n) is an estimate of cost to get from n to goal node (shortest path).

Inputs:

A\* uses as list of nodes defining locations that are able to be visited. In the 2D case, these nodes are listed in terms of their x and y coordinates.

Outputs:

A\* will generate a path. This path is comprised of a list of nodes which will guide the vehicle from start to goal nodes.

The following Figure 1, demonstrates the A\* approach given a grid where each traversable node is defined MxN coordinates. The red circles are obstacles.

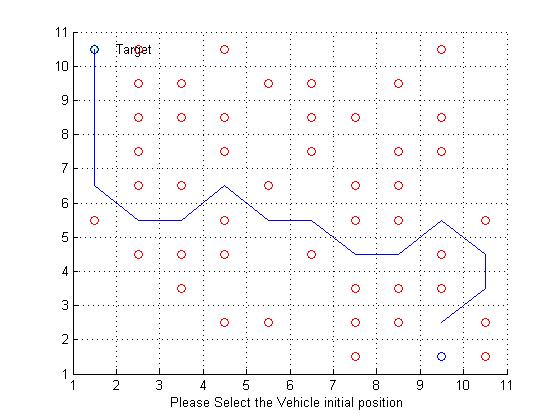


Figure . A\* Search Method Example provided by Mathworks File Share

Obstacle avoidance:

A raw A\* approach will yield a grid spacing that must be defined such that a diagonal path next to an obstacle must be greater than the radius of the vehicle in use. Therefore, each node must be the length of the diameter of the vehicle more some margin. This may or may not be a problem in our application.

What is likely a problem for us would be that the traversal of open space would be characterized by successive small steps across the grid which would yield a start stop motion that is undesirable. This could be smoothed out in higher level software.

An alternative to raw grid data that we are exploring employs the Voronoi algorithm to generate points which create paths equidistant between nodes. Figure 2 provides an example where the \*’s represent obstacles. The vertices of these polygons are outputs of the Voronoi algorithm which are then fed into the A\* search algorithm which generates the path found in Figure 3.



Figure . Voronoi generated paths

Next, A\* was implemented in Matlab. A\* plans the path of nodes which will be navigated to reach the goal. In the following, Figure 2, the origin is represented as a circle, the goal as X. The paths with dashed lines (that aren’t red) have been eliminated due to their proximity to obstacles.



Figure . Voronoi/A\*