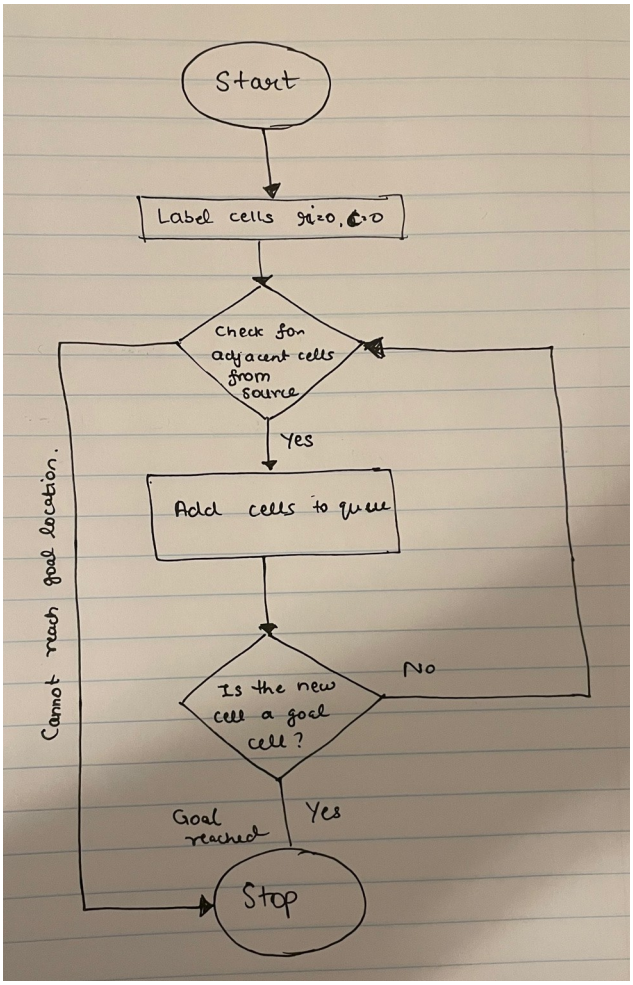
RBE 550: W03_Coding_Assignment_1

Breadth First Search Algorithm

Breadth-First search (BFS) is an algorithm for traversing or searching tree. BFS originates at the root node and explores all of its neighboring nodes prior to moving on to the neighboring nodes of the current nodes at the next level.

The flow diagram of the breadth first search algorithms is depicted below:



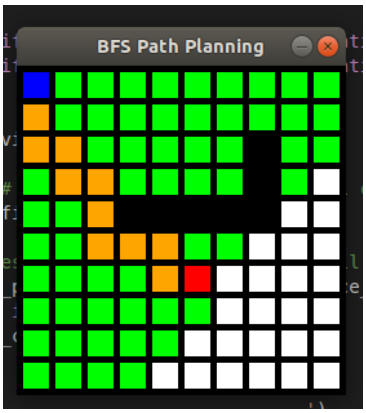
The pseudo-code of BFS algorithm is depicted below:

```
Set all nodes to "not visited";

q = new Queue();

q.enqueue(initial node);

while ( q ≠ empty ) do
{
   x = q.dequeue();

   if ( x has not been visited )
   {
      visited[x] = true;         // Visit node x !

      for ( every edge (x, y)   )
         if ( y has not been visited )
            q.enqueue(y);
   }
```

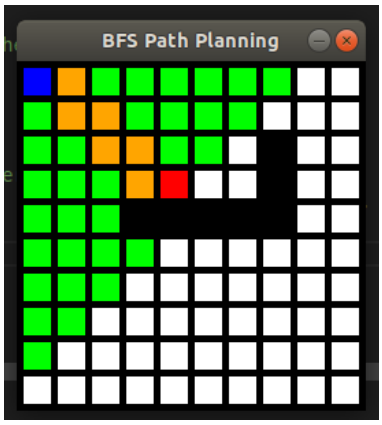
Results:

I have implemented, validated, and tested the BFS algorithm in a lattice based grid structure where a point robot can move in 4 directions (North/South/East/West). The algorithm implemented on Python was able to avoid static obstacles and optimally plan a path to its goal location. I have justified this by testing for various combinations of start point, goal point, and obstacle locations and a few test cases have been depicted below:

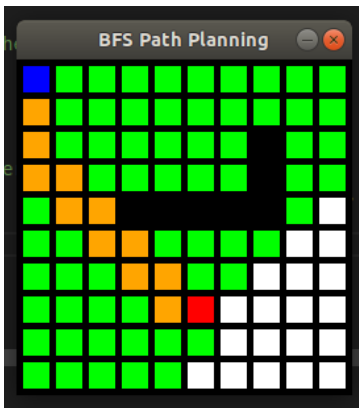
1. Start Point: (0,0)
   Goal point: (6,5)
   Iterations: 62



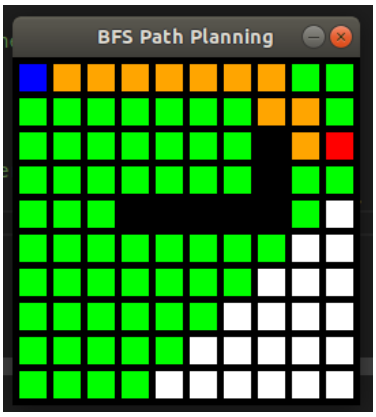
2. Start Point: (0,0)
   Goal point: (3,4)
   Iterations: 62



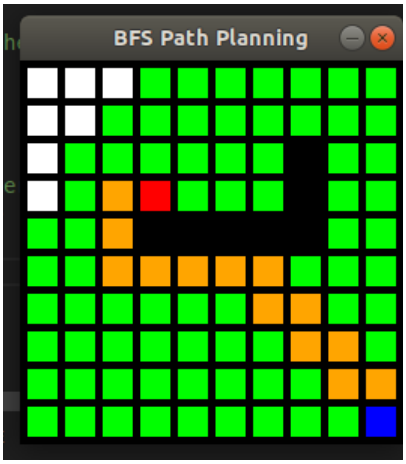
3. Start Point: (0,0)
   Goal point: (7,5)
   Iterations: 68



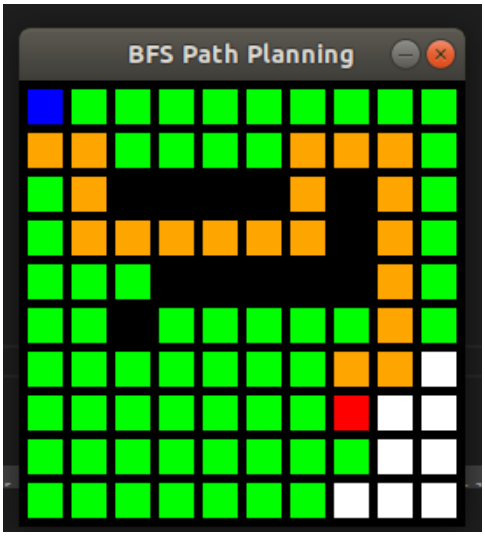
4. Start Point: (0,0)
   Goal point: (2,9)
   Iterations: 65



5. Start Point: (9,9)
   Goal point: (3,3)
   Iterations: 69



6. Start Point: (0,0)
   Goal point: (7,7)
   Iterations: 75



Legend:

Green – Explored grids/nodes
Orange – Optimal path connecting source to destination
White – Unexplored grids/nodes
Black - Obstacles
Blue – Robot's Starting point
Red – Robot's end/ goal point