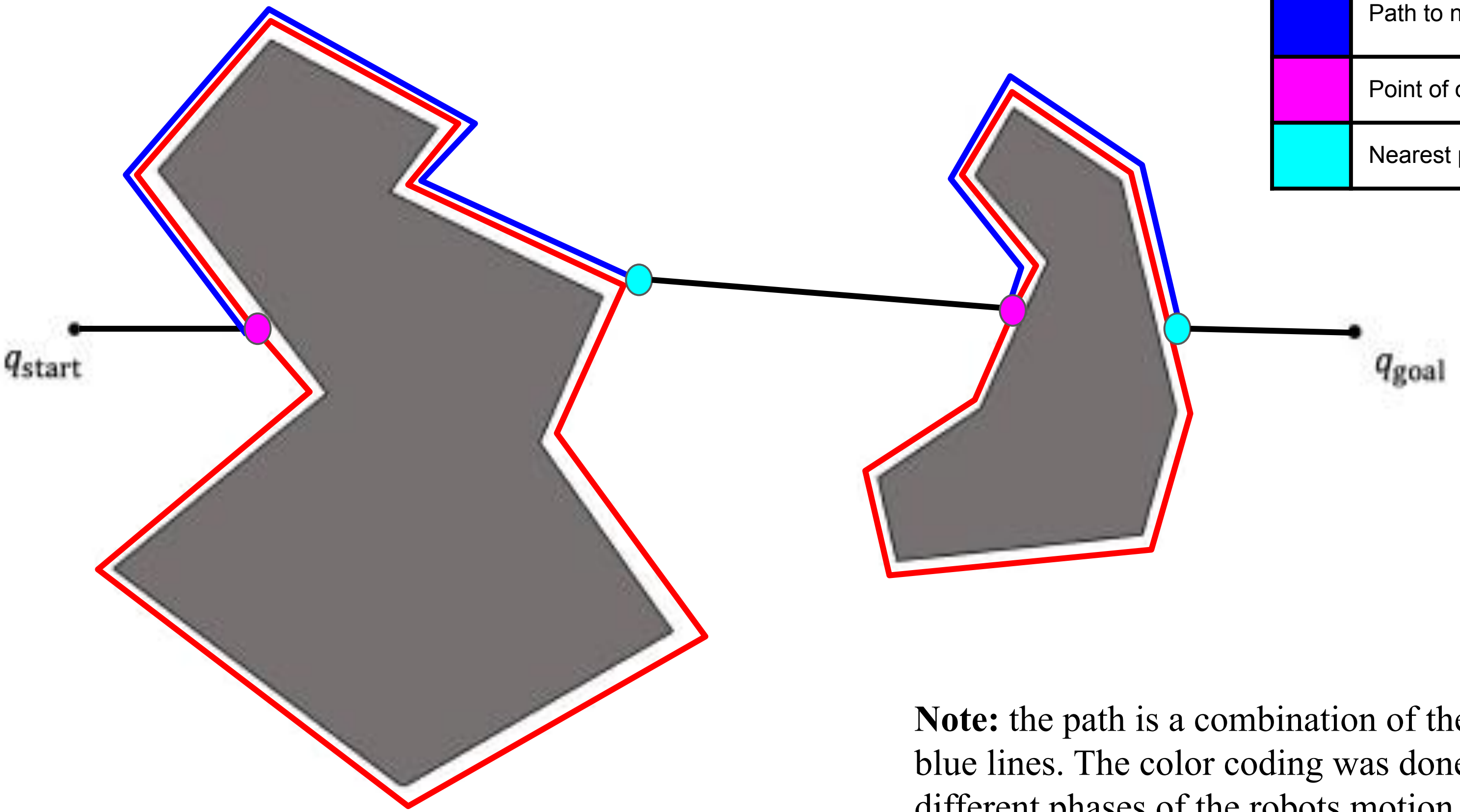
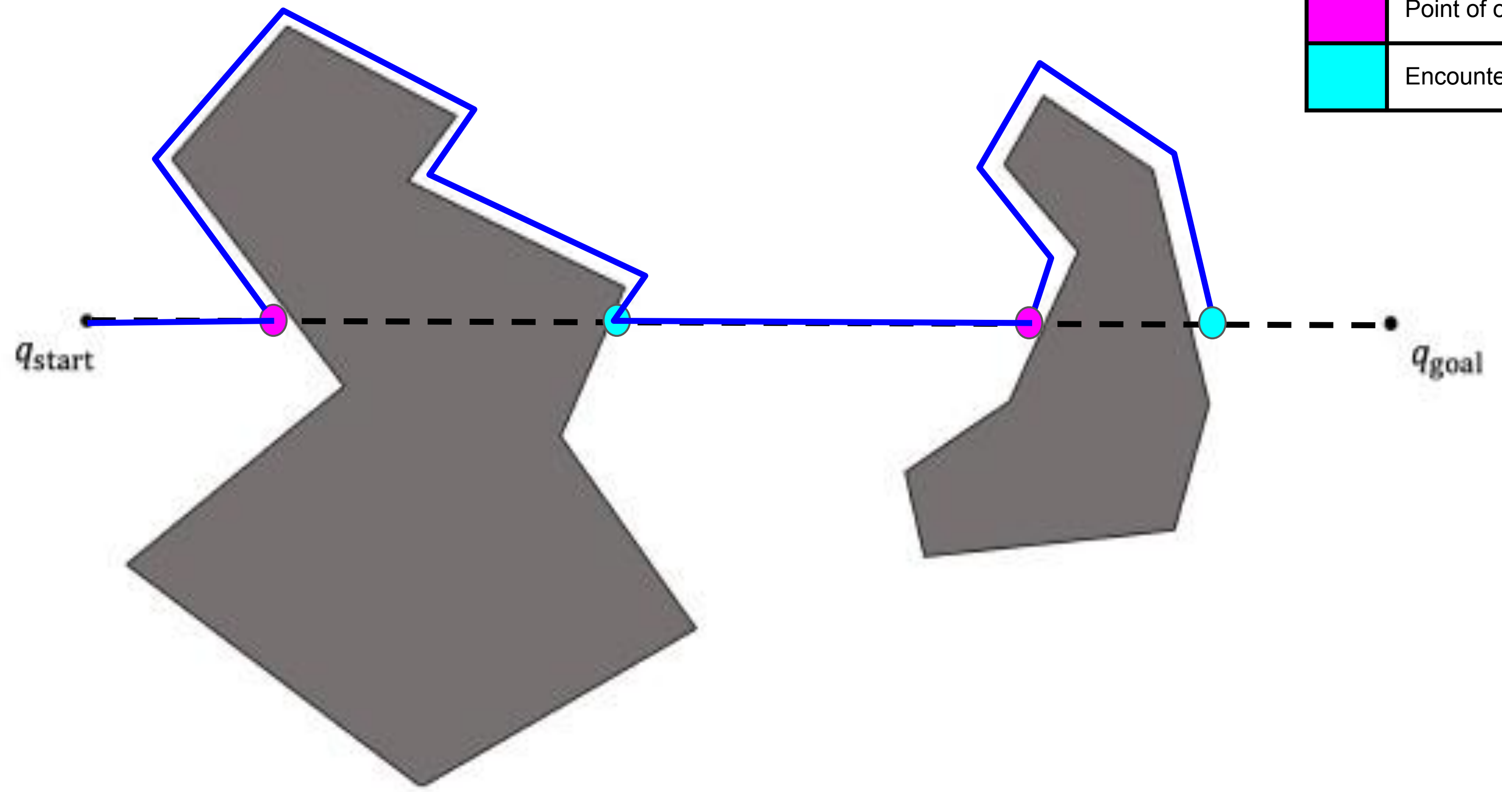
Exercise 1: Bug 1 Algorithm

	Straight path towards goal
	1st path circumnavigating
	Path to nearest point to goal on obstacle
	Point of obstacle intersection
	Nearest point on obstacle to goal



Note: the path is a combination of the black, red, and blue lines. The color coding was done to illustrate the different phases of the robots motion

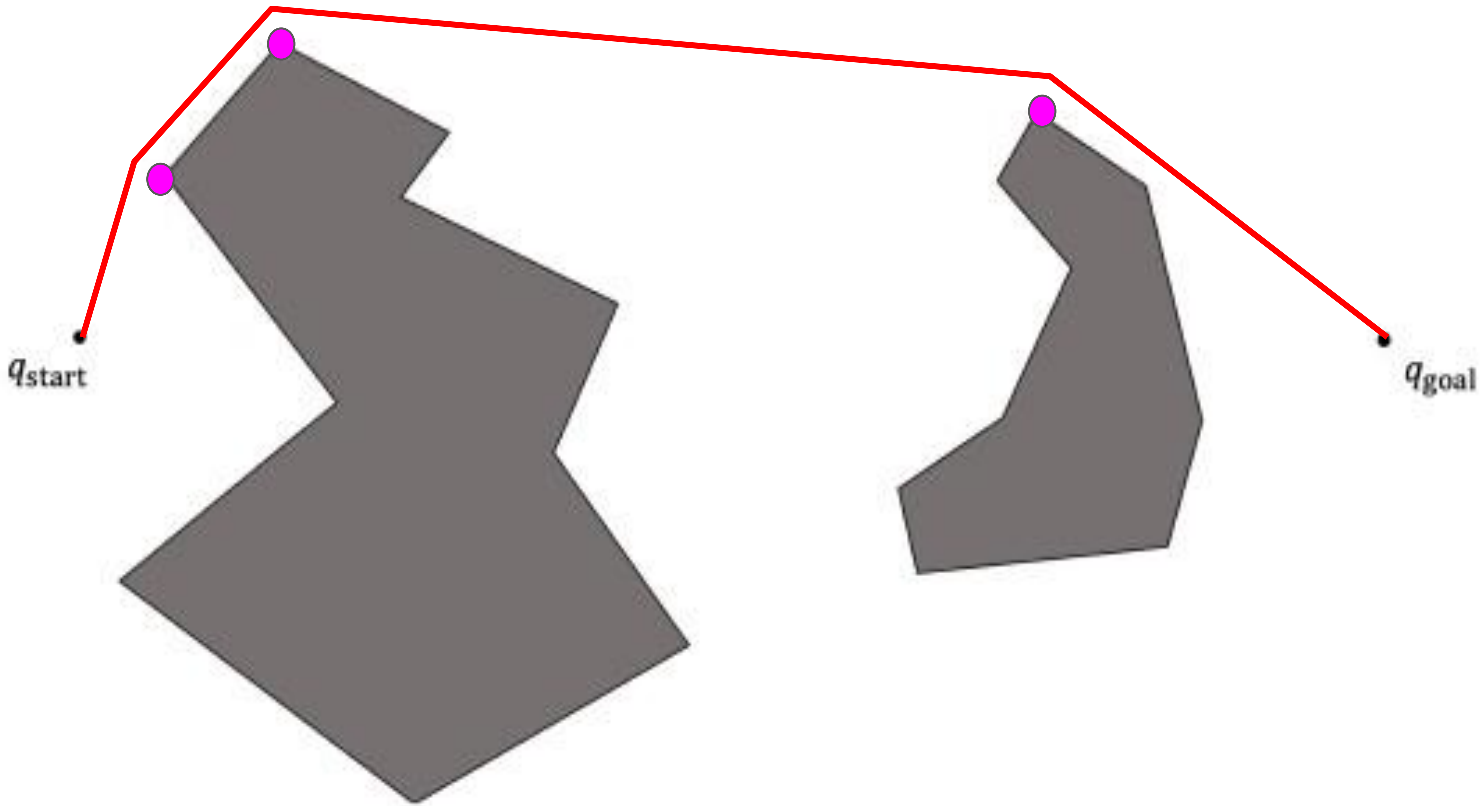
Exercise 1: Bug 2 Algorithm



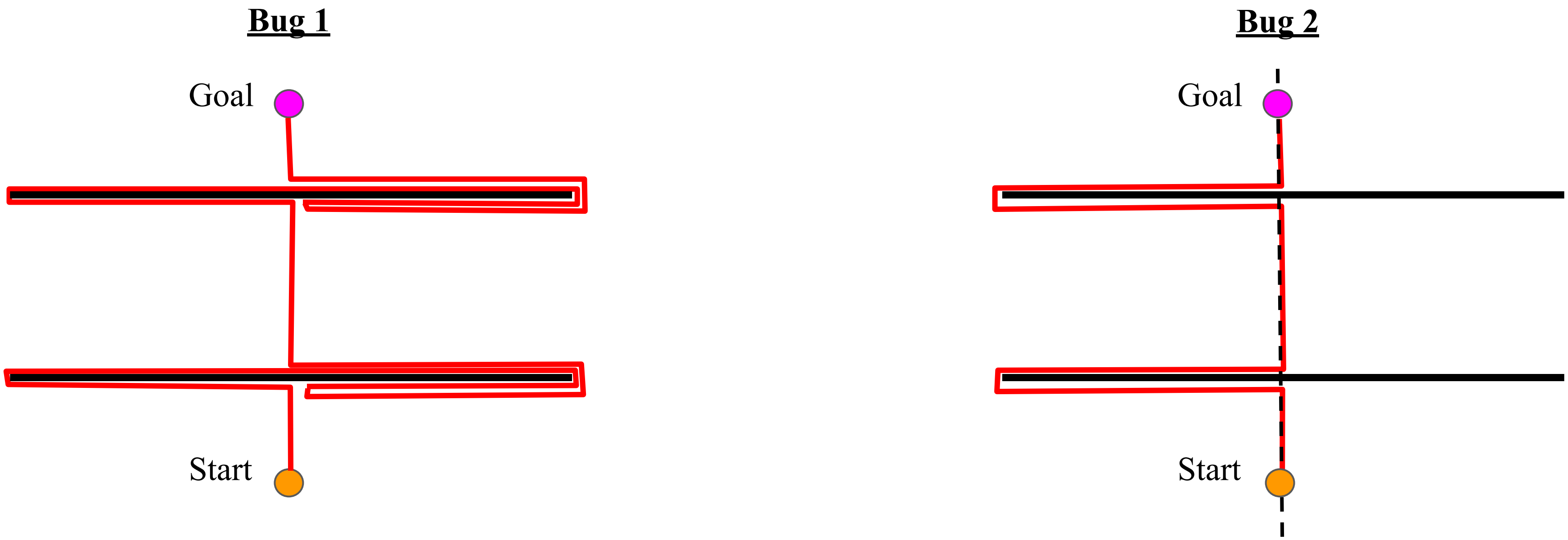
	m-line
	Path bug 2 takes
	Point of obstacle intersection from m-line
	Encounter m-line closer to goal

Exercise 1: Tangent Bug Algorithm

(red)	Tangent bug path
(magenta)	Discontinuities that minimize heuristic



Exercise 2

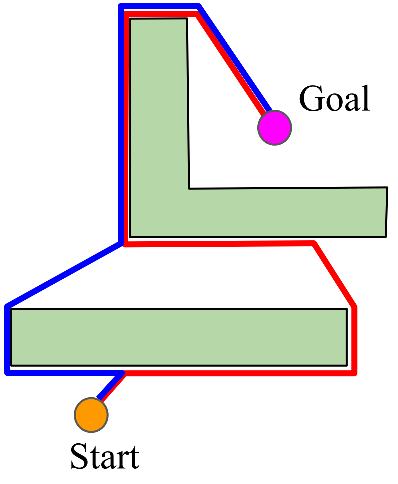


We can image obstacles of 1 dimension (a line of distance w) with perimeters equal to 2w(*). Here the Bug 1 algorithm takes a path of D + 1.5∑P_i - easily verified by inspection. Bug 2 outperforms Bug 1 by twice the addition of the obstacles' perimeters (2∑P_i).

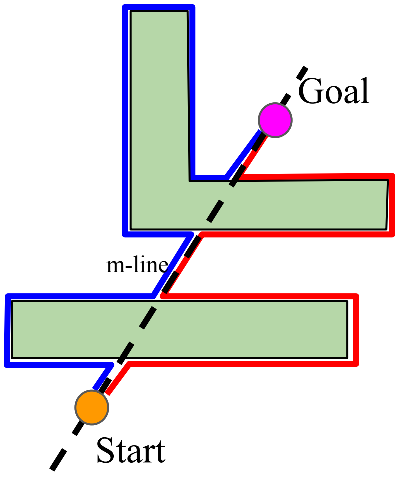
(*) Perimeter = 2w +2h, and we take the limit as h→0, we get that the perimeter of a line is 2w

Exercise 3

Tangent Bug



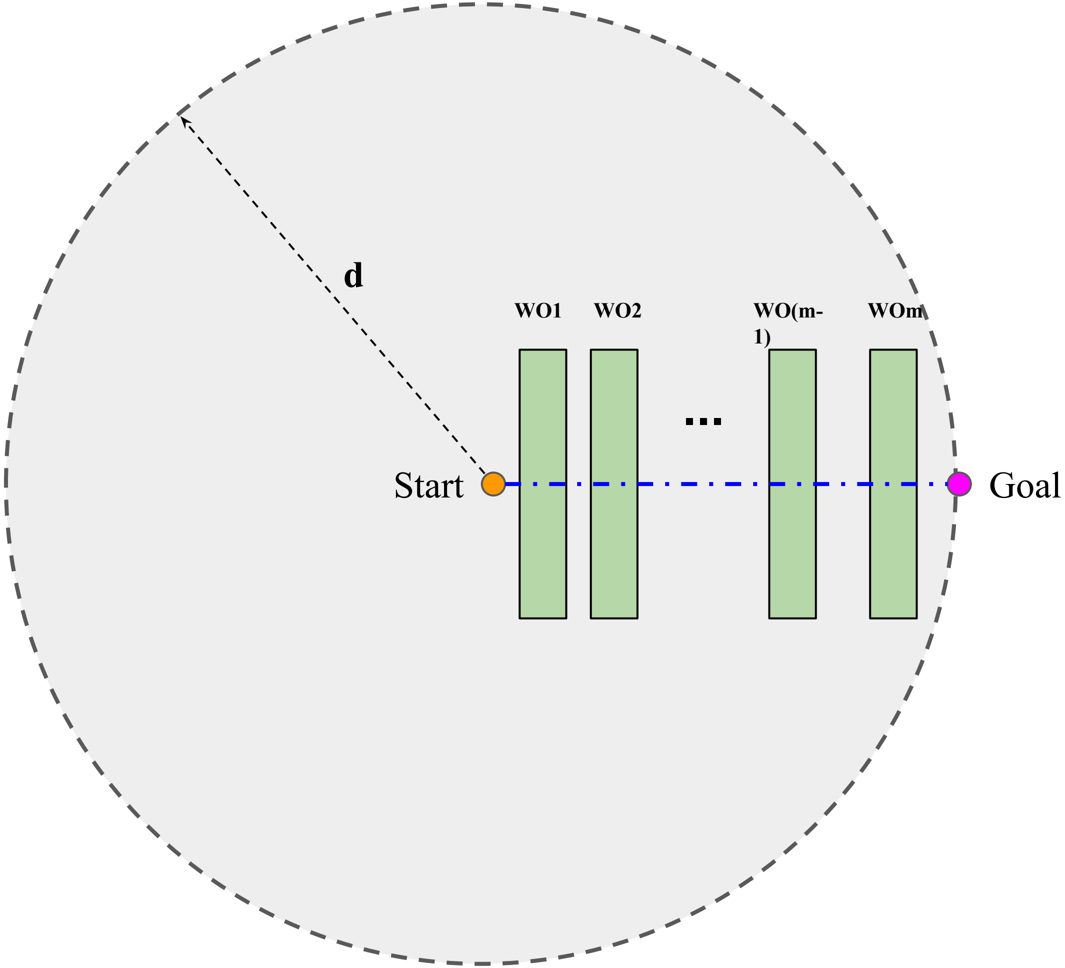
Bug 2



The path the Bug 2 algorithm takes is largely dependent on whether it goes CW or CCW when it encounters an obstacle, this similar for the Tangent Bug. We can see that because the Tangent Bug is allowed to move toward the goal once d_reach < d_follow, it goes back into motion-to-goal once the bug has found a way around the obstacle. Bug 2 always travels on the m-line (if it finds a point on the m-line closer to the goal).

Exercise 4

Suppose all of the obstacles are rectangles and the rectangles are all the same size and are parallel to each other. Arrange all of the interior obstacles WOi, i ∈ [1,m], between start and goal such that the line between start and goal intersects each object twice - seen on the right. This means that every hit point and leave point q_hi and q_li, respectively, will lie on the m-line*. This means that q_li will necessarily lead to q_h(i+1), until we reach q_lm, and then there should be a clear path to goal. For this reason, the maximum number of obstacles Bug 1 will encounter is m obstacles.



*** I understand that this definition of the m-line is used for Bug 2, but is a very convenient definition for this example**

Exercise 5

Is the Tangent Bug algorithm complete?

Proof:

1) Suppose that the Tangent Bug Algorithm (TBA) is NOT complete (proof by **contradiction**)
2) Therefore, there exists a path from start to goal
   a) By assumption it is finite length, and encounters obstacles a finite number of times
3) Let's say that TBA does not find a path to goal
   a) Either it **never terminates** or it **terminates**
   b) Suppose it **never terminates**
      i) If it does not terminate, then the robot keeps finding leave(L)/depart(D) points that are not the corresponding hit points(H)
      ii) Each L/D is closer to goal than the corresponding H. This means we are approaching the goal steadily but surely with each step
      iii) Because each step takes us closer, there are a finite number of H/D/L pairings we will exhaust before reaching goal
      iv) A goal will be found (**contradiction**)
   c) Suppose it **terminates** (incorrectly)
      i) The closest point after a H point must be a L point where it moves into the corresponding obstacle
         (1) Between this L point and the goal is another point on the obstacle close to the goal (Jordan Curve theorem)
         (2) Because we assumed that a path exists, we should have encountered this point, **contradicting** the definition of a leave point