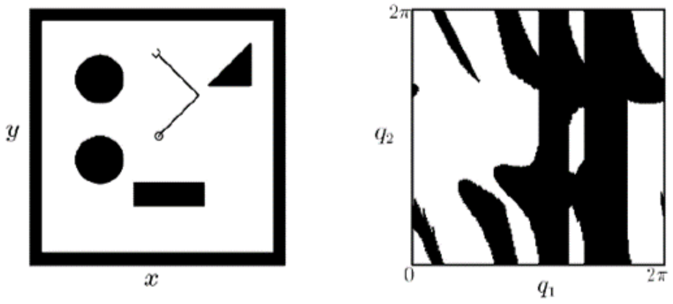
COMS W4733: Computational Aspects of Robotics

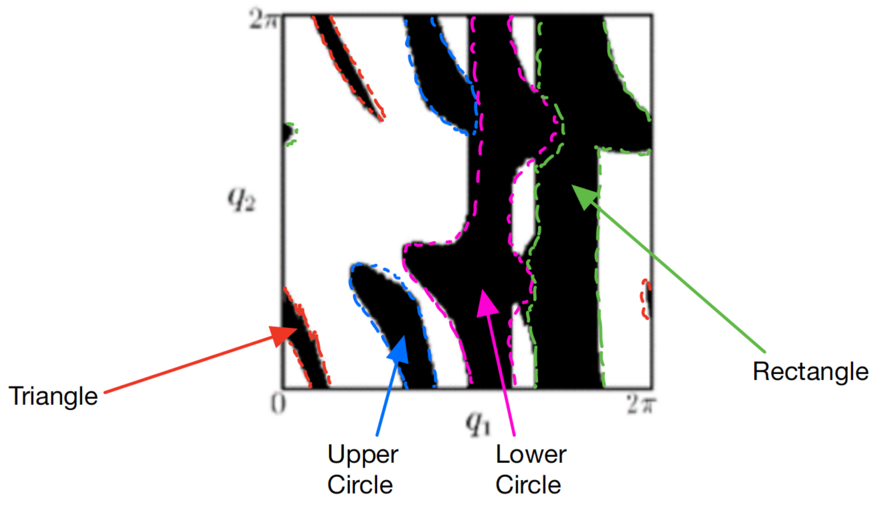
Homework 1 Solutions

Problem 1: C-Space (14 points)

Consider the planar, fixed-base RR manipulator shown on the left below with four obstacles of varying shapes in its workspace. The corresponding "flattened" configuration space is shown on the right, with q_1 being the first joint angle and q_2 being the second (defined relative to the first link).

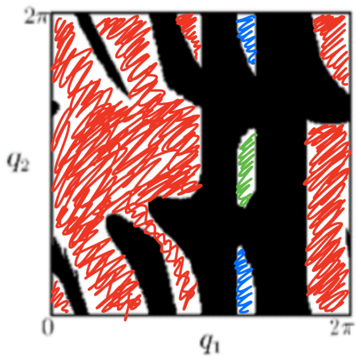


(a) You can determine which object is which by determining at which (q_1, q_2) your manipulator would encounter the obstacle. For example, the triangle in the top right should only be an issue approximately when $q_1 \in [0, \pi]$. Then we can look at the specific configurations of $q2$ in that range for q_1 to determine when a collision actually occurs. Thus, we can use this procedure for all the obstacle shapes in the C-space. See figure below (outlines of shapes are approximate in C-space)

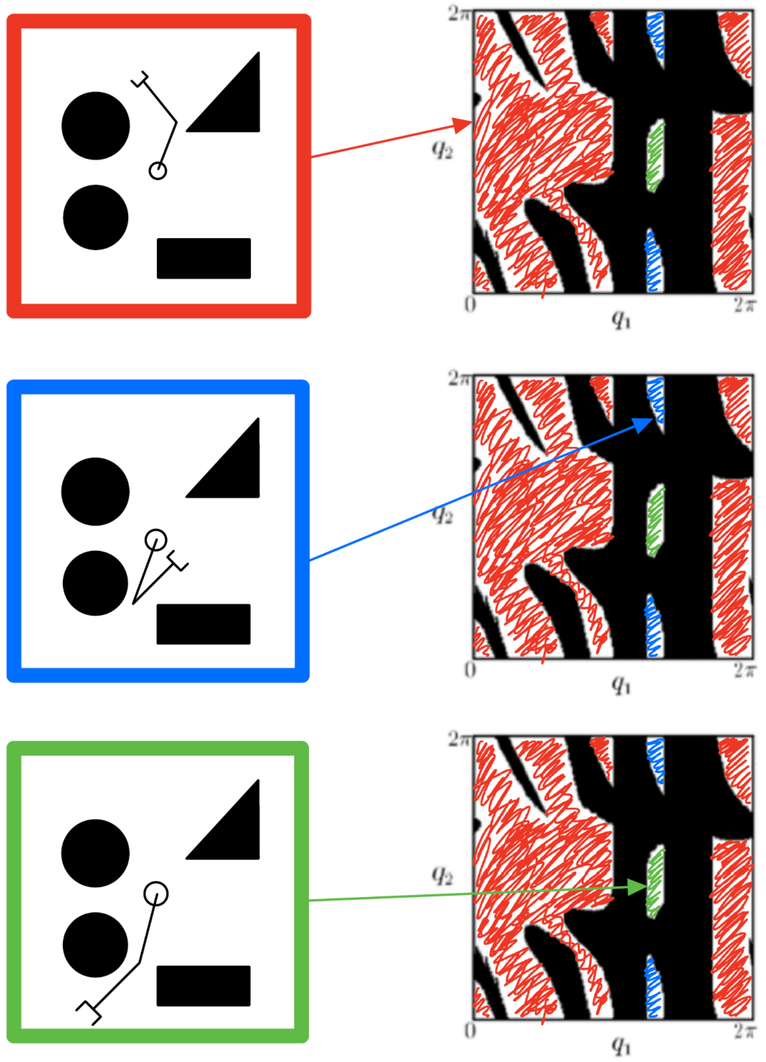


(b) The key to this question lies in recalling the true torus shape of the C-space from the lecture slides. This means that the Cartesian representation of the C-space actually wraps around in both the q_1 and q_2 directions. Thus components that end at the edges are actually connected to other components.

As the drawing below indicates, there are three distinct “connected” components in the manipulator’s free space.

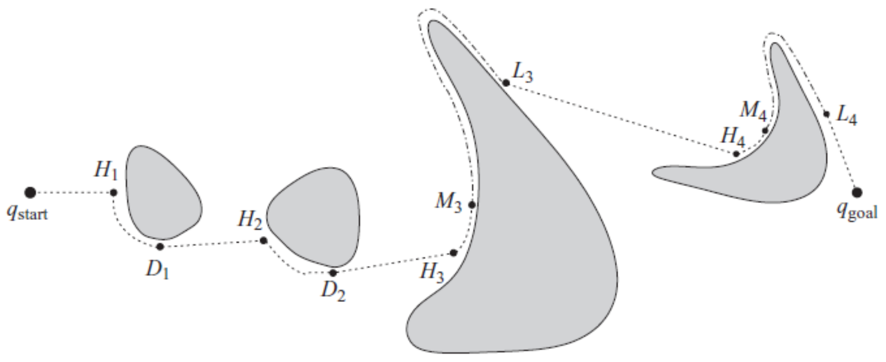


(c) Similar to part (a), you just need to determine a potential combination of (q_1, q_2) for each distinct component. Figures are not drawn to scale.



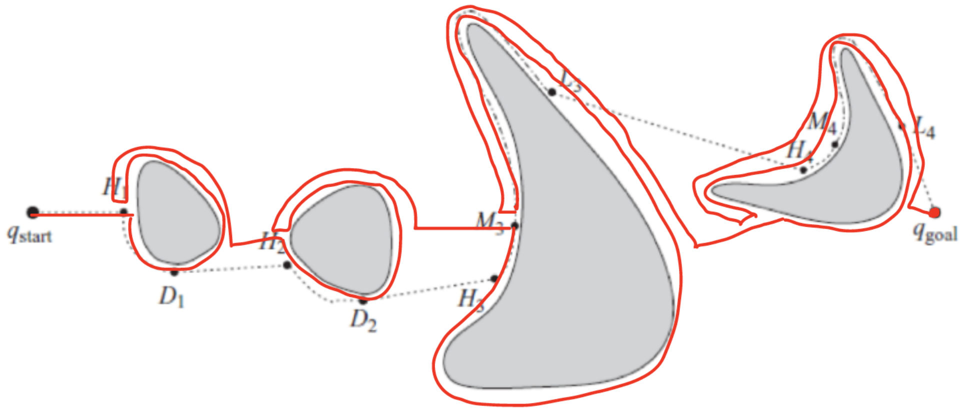
Problem 2: Bug Algorithms (16 points)

The path taken around the obstacles shown below is generated by the Tangent Bug algorithm with zero sensor range.

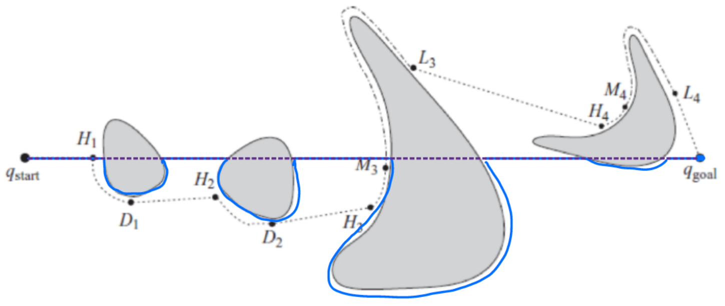


(a) *Note: Here, we assert that the robot turns around to reach q_{leave} once it has re-encountered q_{hit}.*

Bug 1:



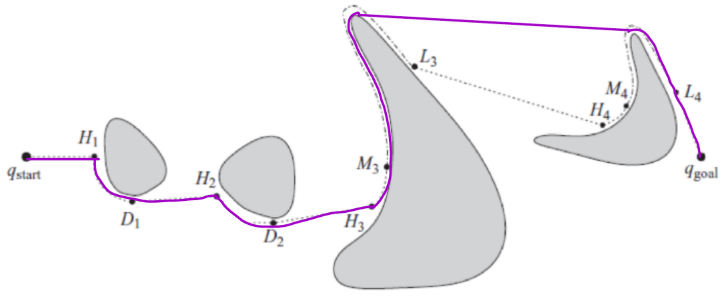
Bug 2:



where the purple dotted line is the m-line.

(b) Bug 2 and Tangent Bug with zero sensor range differ in how they leave the obstacle. Bug 2 always circumnavigates the obstacle until it hits the pre-calculated global m-line where its distance to the goal smaller than its previous hit. On the other hand, Tangent Bug continuously computes the local distance to goal and leaves the boundary when $d_{reached} < d_{followed}$.

(c) The infinite sensor range allows the robot to follow the shorter path using $d(x,n) + d(n, q_{goal})$ without having to follow the obstacle boundary, resulting in a behavior that travels from an obstacle corner to another.



A path from the peak of obstacle 3 to the bottom corner of obstacle 4 is also acceptable; here, we assume that path around the top of obstacle 4 is shorter.

Problem 3: Potential Functions (20 points)

Consider an additive attractive/repulsive potential function on $\mathbb{R}^2$. The goal configuration is located at $q_{\text{goal}} = (0, 0)$, which induces a combined conic and quadratic attractive potential with parameters $d^*_{\text{goal}} = 1$ and $\zeta = 1$. There is a single circular obstacle centered at $q_1 = (2, 0)$ with radius $r_1 = 1$. This induces a repulsive potential with parameters $Q^* = 1$ and $\eta = 1$. All distances are computed using the Euclidean metric.

You can find the solution code HERE.

(a) Please see this figure.

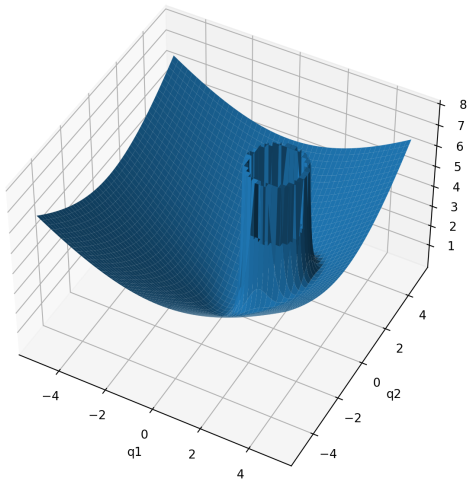


Figure 1: The potential function of the given workspace

(b) The question asks us to compute the potential and gradient at the configuration $q = (0.5, 0)$.

$$\begin{aligned} U_t(q) &= U_a(q) + \Sigma U_{r,i}(q) \\ U_t(q) &= \frac{1}{2}\zeta d(q, q_g)^2 + \frac{1}{2}\eta\left(\frac{1}{d(q)} - \frac{1}{Q^*}\right) \\ U_t(q) &= \frac{1}{2}\zeta(0.5)^2 + \frac{1}{2}\eta\left(\frac{1}{0.5} - \frac{1}{Q^*}\right) \end{aligned} \tag{1}$$

Please note that $d(q, q_g) = 0.5$, and $d(q) = 0.5$ is the distance from q to the closest point on the obstacle. Final answer, the potential is 0.625 or $\frac{5}{8}$.

$$\begin{aligned} \nabla U_t(q) &= \nabla U_a(q) + \Sigma\nabla U_{r,i}(q) \\ \nabla U_t(q) &= \zeta(q - q_g) + \frac{\eta}{d^2(q)}\left(\frac{1}{Q^*} - \frac{1}{d(q)}\right)\nabla d(q) \\ \nabla U_t(q) &= \zeta(q - q_g) + \frac{\eta}{d^2(q)}\left(\frac{1}{Q^*} - \frac{1}{d(q)}\right)\frac{q - c}{d(q, c)} \\ \nabla U_t(q) &= \zeta((0.5, 0) - (0, 0)) + \frac{\eta}{0.5^2}\left(\frac{1}{Q^*} - \frac{1}{0.5}\right)\frac{(0.5, 0) - (1.0, 0)}{1.5} \end{aligned} \tag{2}$$

The gradient is (4.5, 0). $d(q) = 0.5$ is the distance from q to the closest point on the obstacle. $\nabla d(q)$ is the gradient vector between the obstacle and q. Furthermore, $\nabla d(q) = \frac{q-c}{d(q,c)}$ where c is the center of the obstacle given in the problem statement.

Please note that because the point $q = (0.5, 0), d(q, q_g) \leq d^*$, hence, we must use the appropriate part of the piece-wise function.

(c)

$$\begin{aligned} \nabla U_t(q) &= \nabla U_a(q) + \Sigma\nabla U_{r,i}(q) \\ \nabla U_t(q) &= \frac{d^*\zeta(q - q_g)}{d(q, q_g)} + \frac{\eta}{d^2(q)}\left(\frac{1}{Q^*} - \frac{1}{d(q)}\right)\frac{q - c}{d(q, c)} \\ (0, 0) &= \frac{d^*\zeta(q - q_g)}{d(q, q_g)} + \frac{\eta}{d^2(q)}\left(\frac{1}{Q^*} - \frac{1}{d(q)}\right)\frac{q - c}{d(q, c)} \end{aligned} \tag{3}$$

Part C uses the same gradient formula mentioned in Part B. There are two ways to solve this problem. First, by using intuition and observation, one can recognize that the critical point must be lie on the x axis and on the right side of the obstacle. Therefore, $d(q, q_g) > d^*$ and $d(q) > 0$. You must use the appropriate parts of the piece-wise functions. Let $q = (x, y)$, $q = (x, 0)$. The point where the gradient vector is zero is at approximately (3.6823, 0).

The second way is to program the gradient functions using Python and then numerically compute the gradient using libraries such as scipy. Both methods are valid and if the steps are shown will receive credit.

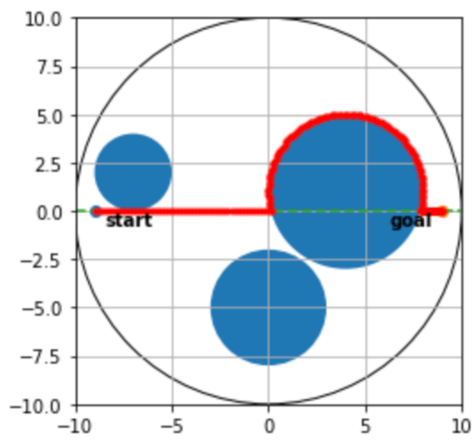
Problem 4: Path Planning

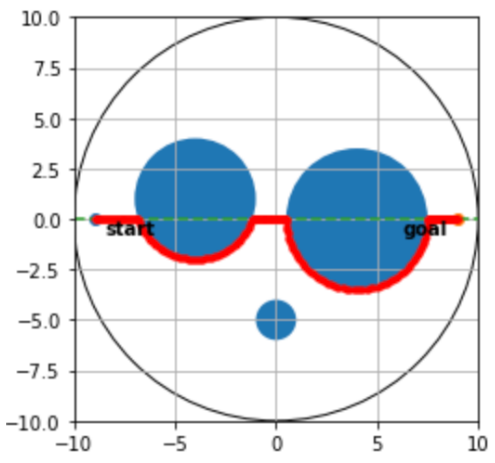
For this last part you will implement a bug algorithm and potential function motion planner for a configuration space on $\mathbb{R}^2$ with spherical obstacles. For the latter approach you will specifically be looking at sphere space navigation functions.

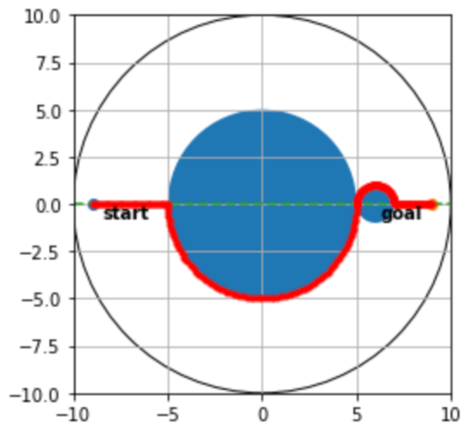
Specifications: The configuration space is bounded by a circle centered at $(0, 0)$ with radius 10. The start and goal configurations are $q_{\text{start}} = (-9, 0)$ and $q_{\text{goal}} = (9, 0)$. There is a number of circular obstacles in the configuration space with varying center locations and radii. The input to your programs is essentially just a list of center positions and corresponding radii. For example, `input = [((-7,2),2), ((0,-5),3), ((4,1),4)]` means that there are three obstacles at the specified locations with radii 2, 3, and 4, respectively. You may assume that all obstacles are disjoint, do not intersect the boundary of the configuration space, and do not contain q_{start} or q_{goal}, so that a solution path is guaranteed to exist.

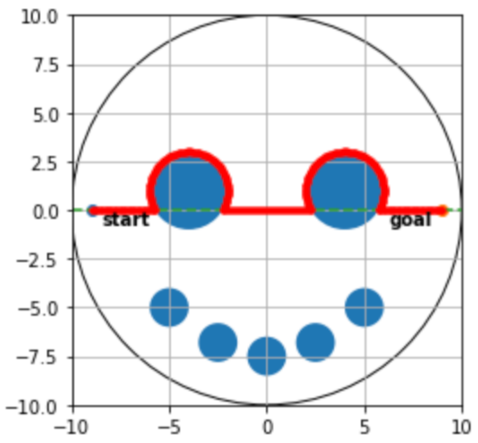
4.1: Offline Bug 2 (20 points)

Below are four examples of Offline Bug 2 at work. Find source code HERE.









4.2: Sphere-Space Navigation Function (20 points)

Refer to screenshots below. Source code can be found HERE.

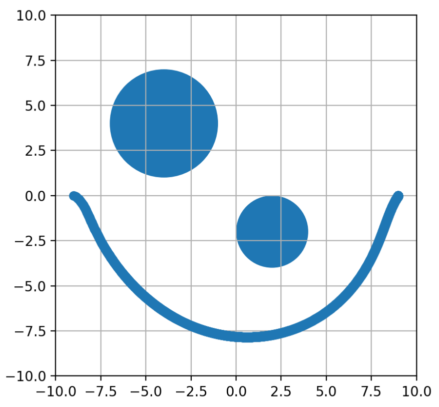
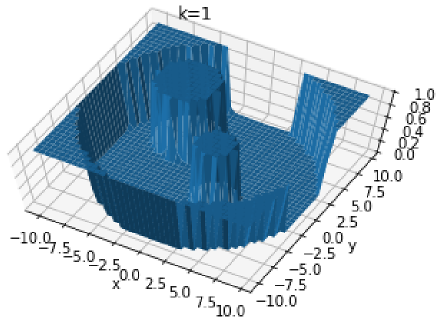
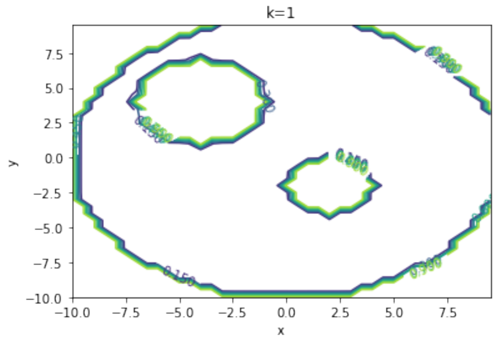


Figure 2: 4.2-1 Robot successfully navigates to the goal via a sensible trajectory; here, $\kappa = 1$.



(a) Navigation function



(b) Contour plot

Figure 3: 4.2-1 Sample navigation function plot $\kappa = 1$

We understand that navigation function is hard to visualize so we won't be penalize if visualization is not perfect.

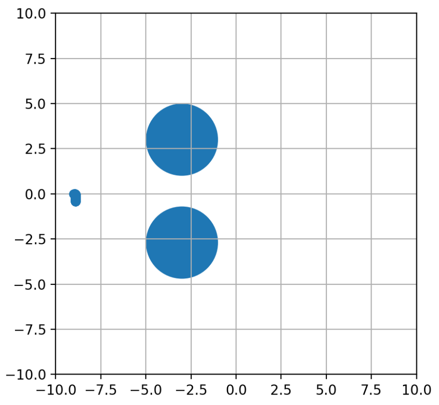


Figure 4: 4.2-2 Robot gets stuck in a local minimum; here, $\kappa = 1$.

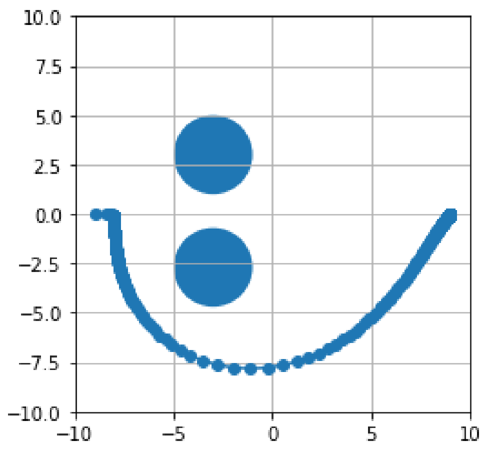


Figure 5: 4.2-3 Robot navigates out of local minimum; here, $\kappa = 2$.

4.3: Discussion (10 points)

Provide brief responses to the following questions.

(a) Use adapted step size. Large step size at the beginning, decrease step size in between, and use a relatively larger time step near the goal (to prevent overshooting).

(b) Along the m-line, arrows will point in the direction of q_{start} to q_{goal}. On obstacle, arrows will follow the obstacle boundary (either left and right) from hit to leave.