Name: Tahsincan
Surname: Köse
ID: 2188423

Homework 4 - Probabilistic Roadmaps

Assigned - Jan 09, 2018, **Due - Dec 24, 2018**

1

In order to compute whether any intersection occurs between the line segment and circle, it is needed to devise a composite formula that possible values are plugged in an tested with predefined conditions.

To start with, recall the basic circle formula. Having $p_c = \begin{bmatrix} x_c & y_c \end{bmatrix}$, it can be written as $(x - x_c)^2 + (y - y_c)^2$.

Let's compute two additional vectors that will hugely ease our job when solving the problem.

$d = p_2 - p_1 \rightarrow$ direction vector starting from the first point.

$v = p_1 - p_c \rightarrow$ vector from circle center to the beginning of line segment. It will be used in substitutions latter in the solution.

Having all necessary identities in advance, we now need to assume an intersection point p on the line segment and solve the circle formula for it.

$p = p_1 + t * d.$
In parametric form:
$p_x = p_{1x} + t * d_x$
$p_y = p_{1y} + t * d_y$

When plugged into $(p_x - x_c)^2 + (p_y + y_c)^2 = r_c^2$, the formula becomes as:

$p_{1x}^2 + 2tp_{1x}d_x + t^2d_x^2 - 2p_{1x}x_c - 2td_xx_c$
$+x_c^2 + p_{1y}^2 + 2tp_{1y}d_y + t^2d_y^2 - 2p_{1y}y_c - 2td_yy_c + y_c^2 - r_c^2 = 0.$

After grouping with respect to t and t^2 terms:
$t^2(\underbrace{d_x^2 + d_y^2}_{d \cdot d}) + 2t(\underbrace{p_{1x}d_x + p_{1y}d_y - d_xx_c - d_yy_c}_{p_1 \cdot d - d \cdot p_c})+$

$$\underbrace{p_{1x}^2 + p_{1y}^2 - 2p_{1x}x_c - 2p_{1y}y_c + x_c^2 + y_c^2 - r_c^2}_{(p_1 - p_c \cdot p_1 - p_c) - r_c^2} = 0$$

From now on, we will convert back to vector notation for simplicity. Note that, all grouped terms can be written in the form of dot product. Before writing the most compact form, one further simplification should be made. The second underbraced part can be written as $d \cdot (p_1 - p_c)$. Now we have $(p_1 - p_c)$ in both second and third part. At the beginning of the question it is already computed as v. Applying that substitution gives the final formula:

$$t^2(\underbrace{d \cdot d}_{\text{a}}) + t\underbrace{2(d \cdot v)}_{\text{b}} + \underbrace{(v \cdot v) - r_c^2}_{\text{c}} = 0.$$

Computation of discriminant Δ is as follows:

$$\Delta = b^2 - 4ac$$

Δ is used in the first conditional statement in the script. If it is less than 0, we can directly conclude that no intersection occurs. If it is equal to or bigger than 0, we need to apply further tests to infer whether an intersection takes place. If we were dealing with lines, the further tests would not be needed. But since our materials are line segments, those tests are necessary. We should calculate t_1 and t_2 using Δ that is just computed above. Below are the identities:

$$t_1 = (-b - \sqrt{\Delta})/(2 * a)$$
$$t_2 = (-b + \sqrt{\Delta})/(2 * a)$$

Recall that, any point on a line segment can be found directly using a parameter t, having starting point and direction vector in hand. Therefore any valid point on the line segment should conform to the condition $0 \leq t \leq 1$. Grounding on this basis, there can be 3 more non-intersecting cases even when $\Delta > 0$. If $t_1 > 1$ and $t_2 > 1$, the line segment just falls too short to reach to the circle. If $t_1 < 0$ and $t_2 < 0$, then the line segment started too late and the direction it goes to its end point diverges from the circle. Finally, if $t_1 < 0$ and $t_2 > 0$, then the line segment is completely in the circle boundaries without touching any point on the circle. The line segments with particular t_1 and t_2 values that passes all the tests described above intersects with the circle given as input.

2

This problem is relatively easy compared to the first one. First we compute the distance between p_1 to p_2 and check it with the sum of radiuses $r1 + r2$.

If $\|p_1 - p_2\| \leq r_1 + r_2$
  then has intersection
else

then does not have an intersection

One edge case is that when circles have the same center. If they do have the same radiuses, then they are aligned and intersect with each other on infinite points. This is a open form solution after all. The solution only grounds from geometrical intuitions. However, it is sufficient for this problem to have such a solution. It resembles to the greedy-choice feature in *Shortest Path* problems. The particular script that implements above description is **hw4_script2_geo.m**. Aside with it, I also implemented a closed form solution that proceeds through rigorous math and considers all cases in **hw4_script2_mat.m**. Briefly, it uses two circle equations to reduce the number of variables to 1, writing one in terms of other. Solving the resulting quadratic formula with 1 unknown is simple root computation.

Consequently, both scripts produces the same result for given any input.

5

In order to implement Probabilistic Road Map Algorithm, I used **PGraph** class of Robotics Toolbox. I'm particularly using the distance, shortest path and display utility functions of the class. On the other hand; configuration sampler, collision checker, distance function and closeness criteria among the configurations belongs to me. I have implemented my own version of Dijkstra (Adjacency Matrix form). However, it fails to produce an outcome in reasonable time when sample size is bigger than 500. Therefore, I'm using A* implementation of PGraph. It supports the sample size up to 1500 nodes. Possibly, it could support more. However 1500 nodes are adequate to create paths between any two configurations in averagely-densed grids. The grid used in the tests is $40x40$ and θ values are restricted into $\begin{bmatrix}0 & \pi/2\end{bmatrix}$ range.

Actually, most of the functionality is produced through previous questions. The required work in this question was to bundle all particular functions into scripts to have a decent path from q_0 to q_f at the end.

The main script is **PRM.m** and utility function is **collision_checker.m**. Other functionalities described above are implemented inside of these scripts. I used a distance threshold of 1.5 meters including the joint configurations. For that purpose, angles are sampled in radians instead of degrees. In that way, smooth trajectory is constructed. For more smoother movement, one can apply a lower threshold, but that requires higher sample size in order to have enough valid nodes in the road map.

Below figures illustrate the roadmap and the shortest path in each case. Input configurations is as follows for ***Test 1***.

$n : 1000 \rightarrow$ sample size
$q_0 : \begin{bmatrix} -10.0000 & 0 & 0.5236 & 0.7854 \end{bmatrix}$
$q_f : \begin{bmatrix} 15.0000 & 10.0000 & 0.7854 & 1.0472 \end{bmatrix}$
$obstacles : \begin{bmatrix} 0 & 0 & 1 \\ 2 & 2 & 1 \\ 2 & -2 & 1 \\ -2 & -2 & 1 \\ -2 & 2 & 1 \\ -5 & 4 & 2 \end{bmatrix}$, first 2 columns represent p_i and 3^{rd} column represents r_i.

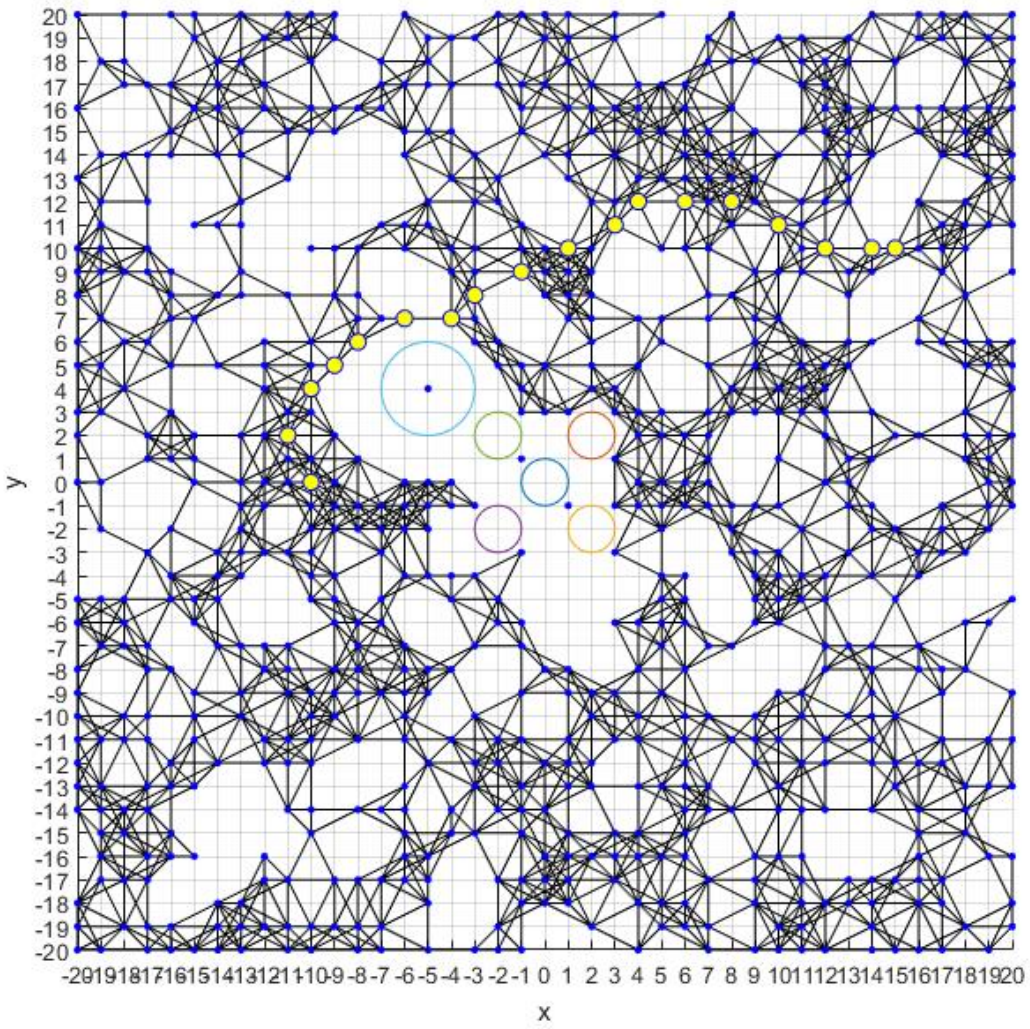

Figure 1: Sparse Environment with n=1000

Figure 1 illustrates the first successful test regarding to the PRM and shortest path

application.There are only 5 obstacles and accumulated on the middle of the grid. There exists no path from the narrow passages between obstacles.

Test 2 input configurations:

$n : 750 \rightarrow$ sample size

$q_0 : \begin{bmatrix} -10.0000 & 0 & 0.5236 & 0.7854 \end{bmatrix}$

$q_f : \begin{bmatrix} 15.0000 & 10.0000 & 0.7854 & 1.0472 \end{bmatrix}$

$$obstacles : \begin{bmatrix} 0 & 0 & 1 \\ 2 & 2 & 1 \\ 2 & -2 & 1 \\ -2 & -2 & 1 \\ -2 & 2 & 1 \\ -5 & 4 & 2 \\ 10 & 11 & 1.5 \\ 10 & 6 & 2 \end{bmatrix}$$

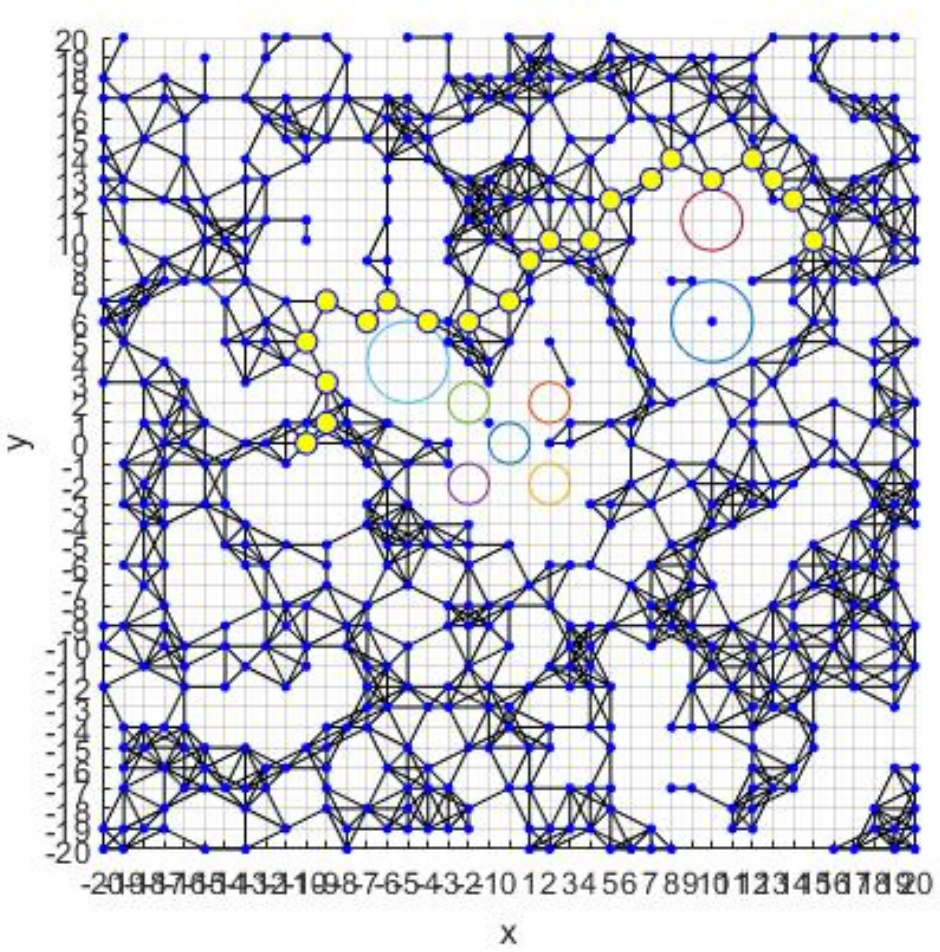

Figure 2: Dense Environment with n=750

In this test, algorithm also suffers from narrow passages. The main reason of it in this case is the sample size. In the following test, algorithm successfully finds a passage-route with higher sample size.

Test 3 input configurations:

$n : 1000 \rightarrow$ sample size

$q_0 : \begin{bmatrix} -10.0000 & 0 & 0.5236 & 0.7854 \end{bmatrix}$

$q_f : \begin{bmatrix} 15.0000 & 10.0000 & 0.7854 & 1.0472 \end{bmatrix}$

$$obstacles : \begin{bmatrix} 0 & 0 & 1 \\ 2 & 2 & 1 \\ 2 & -2 & 1 \\ -2 & -2 & 1 \\ -2 & 2 & 1 \\ -5 & 4 & 2 \\ 10 & 11 & 1.5 \\ 10 & 6 & 2 \end{bmatrix}$$

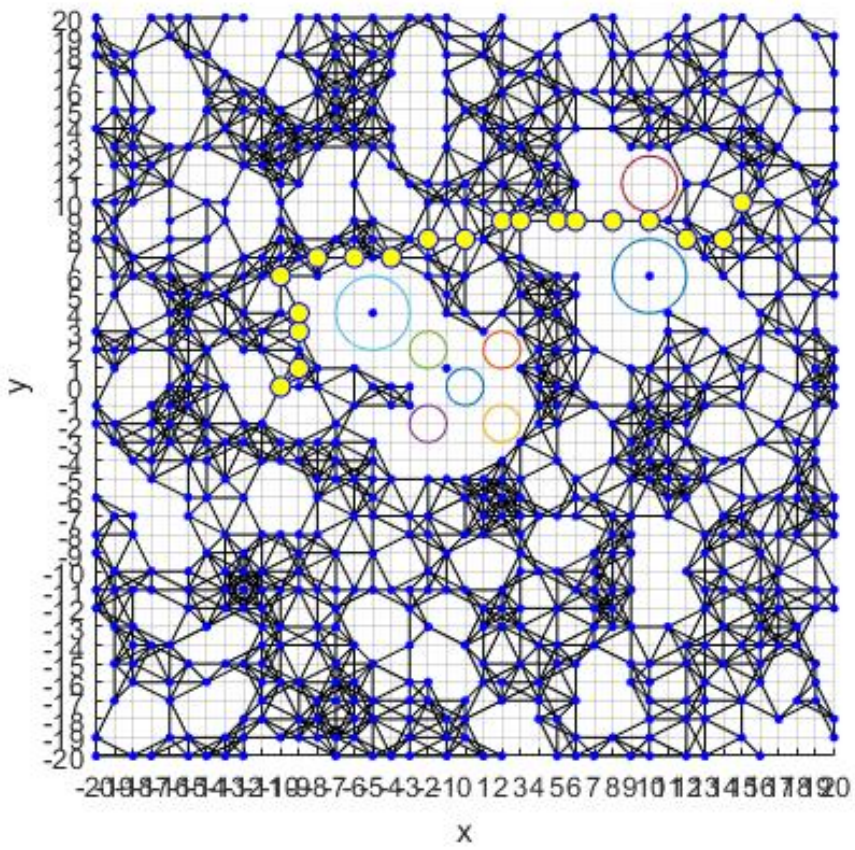

Figure 3: 9-Obstacles with n=1000

As it can be seen, all the configuration is same except the sample size. While it fails to

find a passage-route in the former case, it successfully finds it in this case because of the increased sample size. Another implication of increased sample size is the enhancement of optimality of the algorithm. Since passage-routes are shorter than circumferencing routes most of the time, the ability to finding them directly points to the optimality.

The subsequent test illustrates a shorter path even when a new obstacle is introduced, because of the increased sample size. One of the main drawbacks of PRM algorithm is its non-optimality. One can easily detect this by comparing *Figure 3* and *Figure 4*.

Test 4 input configurations:

$n : 1250 \rightarrow$ sample size

$q_0 : \begin{bmatrix} -10.0000 & 0 & 0.5236 & 0.7854 \end{bmatrix}$

$q_f : \begin{bmatrix} 15.0000 & 10.0000 & 0.7854 & 1.0472 \end{bmatrix}$

$$obstacles : \begin{bmatrix} 0 & 0 & 1 \\ 2 & 2 & 1 \\ 2 & -2 & 1 \\ -2 & -2 & 1 \\ -2 & 2 & 1 \\ -5 & 4 & 2 \\ 10 & 11 & 1.5 \\ 10 & 6 & 2 \\ 2 & 10 & 2 \end{bmatrix}$$

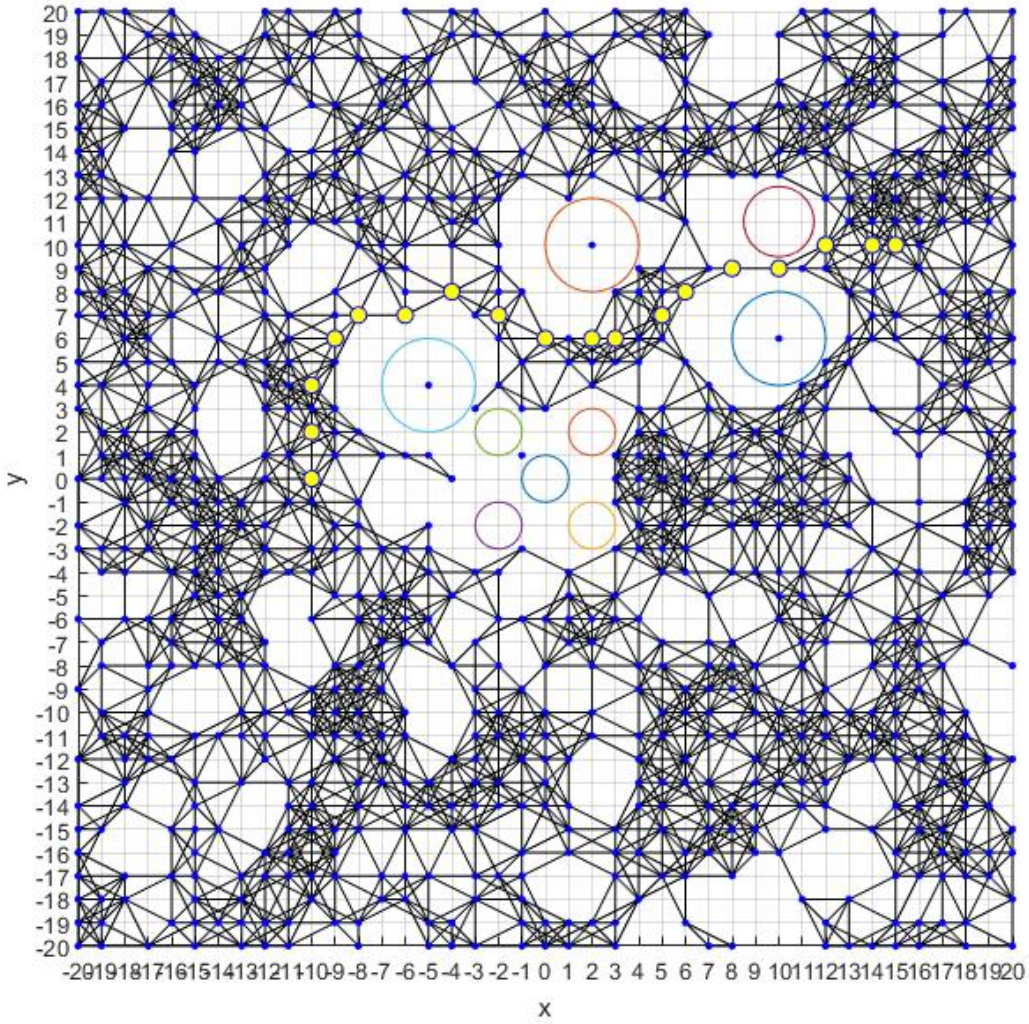

Figure 4: 9-Obstacles with n=1250

Since sample size is increased, the shortest path is shorter than of the ***Test 3***. Conclusion of this comparison is the non-optimality of PRM.

Test 5 input configurations:

$n : 400 \rightarrow$ sample size
$q_0 : \begin{bmatrix} -10.0000 & 0 & 0.5236 & 0.7854 \end{bmatrix}$
$q_f : \begin{bmatrix} 15.0000 & 10.0000 & 0.7854 & 1.0472 \end{bmatrix}$

$$obstacles: \begin{bmatrix} 0 & 0 & 1 \\ 2 & 2 & 1 \\ 2 & -2 & 1 \\ -2 & -2 & 1 \\ -2 & 2 & 1 \\ -5 & 4 & 2 \\ 10 & 11 & 1.5 \\ 10 & 6 & 2 \\ 2 & 10 & 2 \end{bmatrix}$$

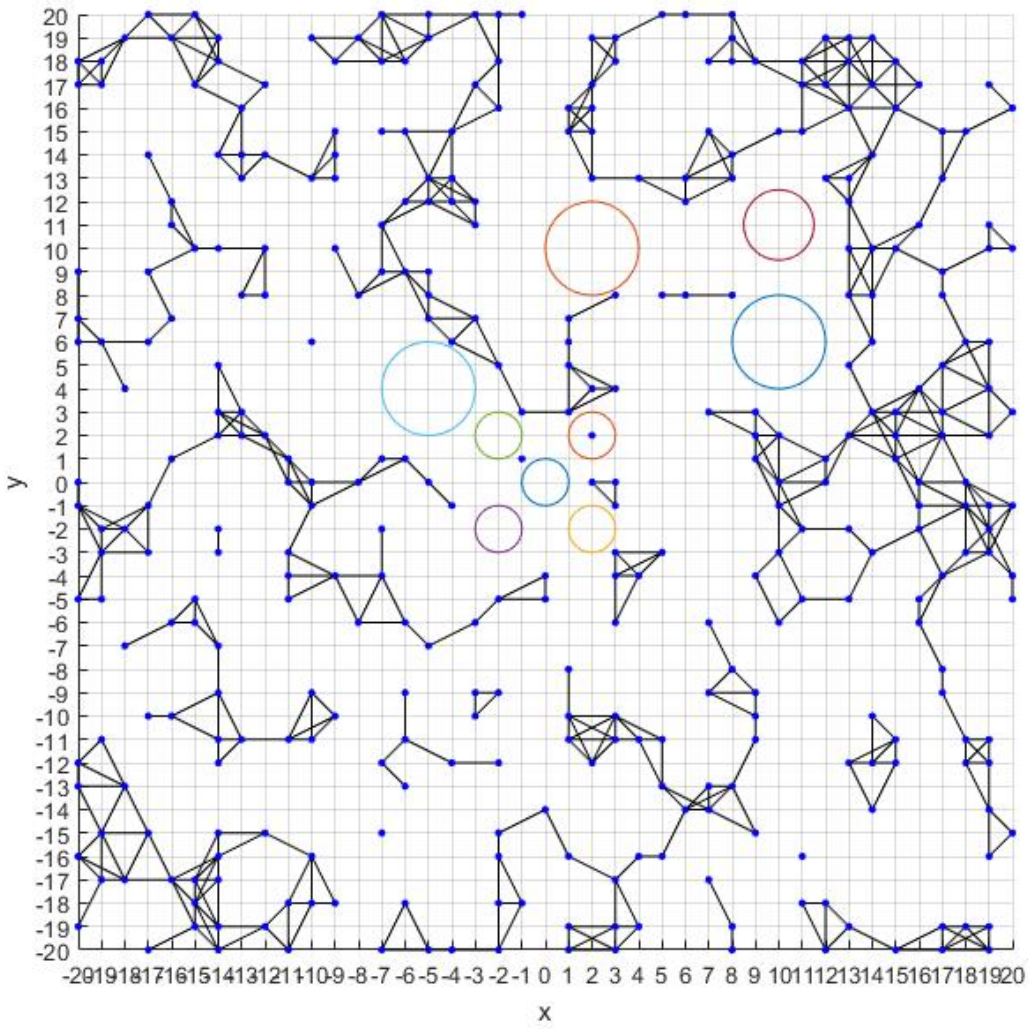

Figure 5: 9-Obstacles with n=400

This is the final test that illustrates the *incompleteness* of the PRM algorithm. The environment is the same of *Test 4*. However, algorithm fails to find a path because of the insufficient sample size. In order to have a valid path through lower sample size, the

distance threshold should be increased to consider more distant as valid edges. In effect, the outcomes of this adjustment are identical with increasing the sample size.

All input configurations with respect to tests above and other tests that are not illustrated in here can be inspected in ***inputConf.txt****.*