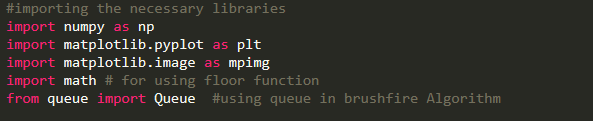
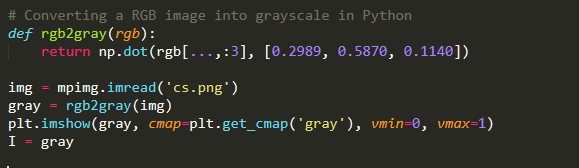
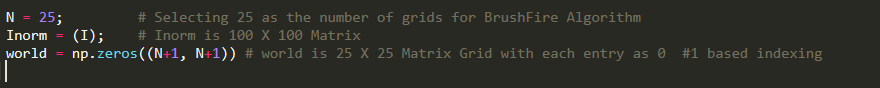
**Code Explanation:**

First importing the necessary libraries required.



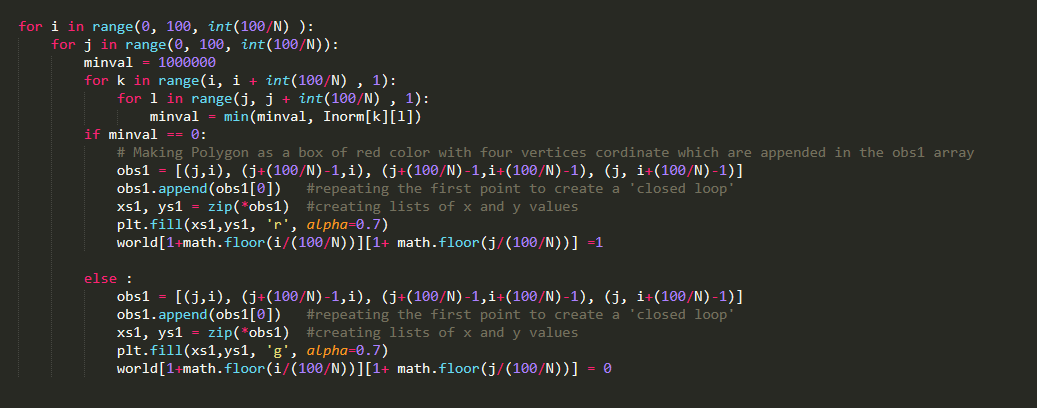
Next by taking the data from the image (PNG) file and converting the RGB image into gray scale and so we get a **Inorm** as a 100 x 100 Matrix of Boolean value



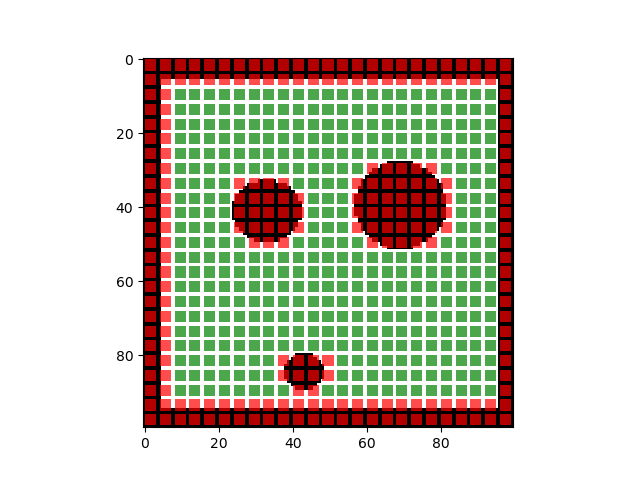


So we have created a 25 x 25 grid matrix of name **world**. If Inorm is 0 in that world[i][j]

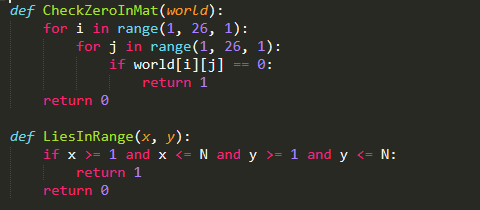
So we iterate over for every possible location (i, j) and find the value of world at that location (i, j) and also change the background to red when world[i][j] = 1, i.e. at obstacle, otherwise green background.



The output of the program looks



**Brushfire Algorithm:**

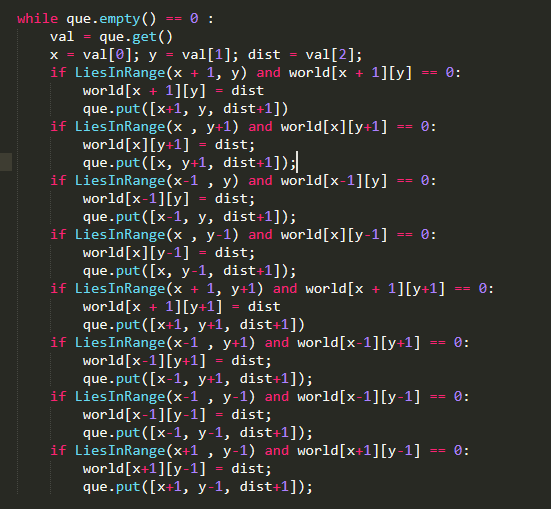


Defining the function **CheckZeroInMat(world)** which takes the world matric and check if there is still zero or unvisited node in the world matrix.

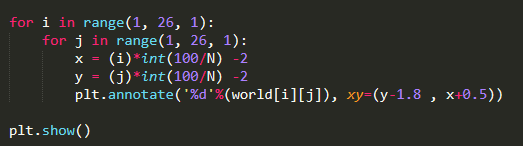
Defining the function **LiesInRange(x, y)** which takes the parameter location x and y and check whether x and y are valid matrix index or not.

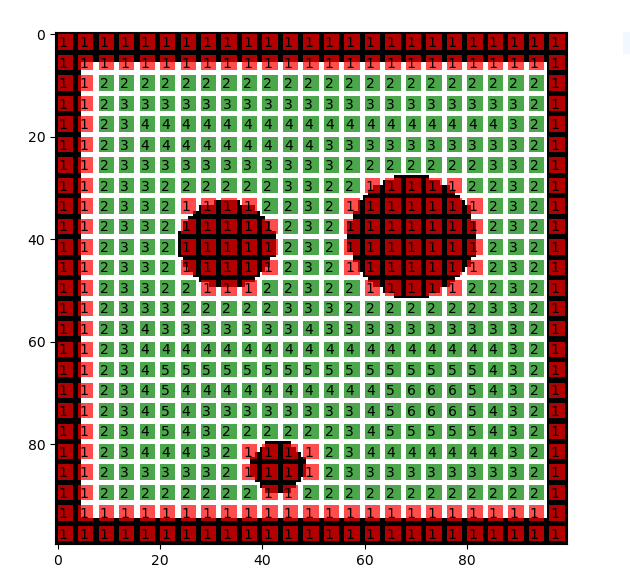
Using the BFS approach to find the distance at each node from the nearest obstacle.

1. First we create a queue and push the obstacle location (i, j, 1) in the queue.
2. Firstly, we pop an element from the queue with the node where world[i][j] = 1 i.e. there is an obstacle. We add the neighbor in the queue. Then we visit the eight neighbors of the that location (i,j) and if not visited and not the obstacle we assign them as 2.
3. Similarly, at world[i][j] = **k** i.e., we visit the eight neighbors of the that location (i,j) and if not visited and not the obstacle we assign them as **k+1**.
4. We follow the step 4 until all the nodes are visited. i.e. the queue is empty



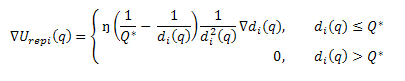
The above code we visit the eight neighbors of then node and assign the value as last + 1. And update the world matrix.

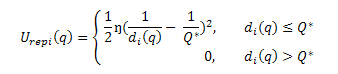


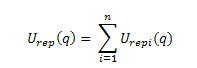


The above is the result of the BFS approach that finds the distance of nearest obstacle from any node.

**Repulsive Potential Function:**

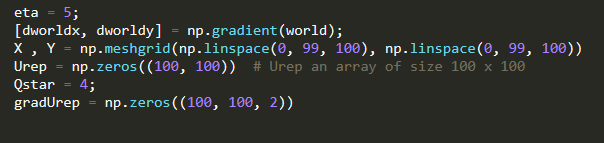


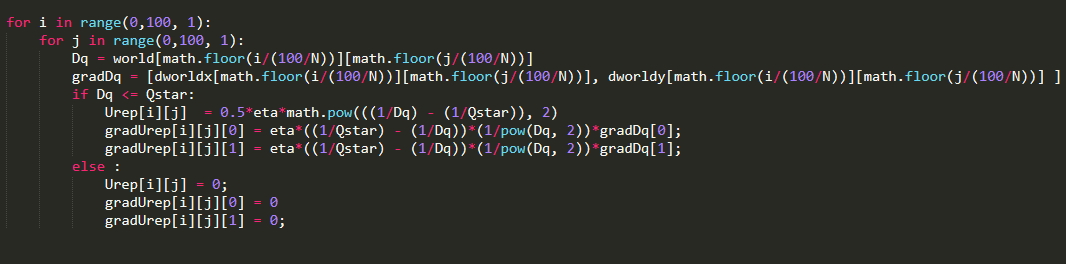




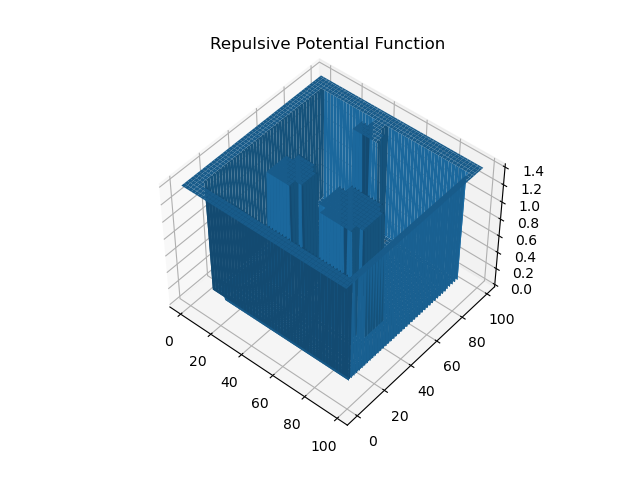
**References :** [**http://www.cs.bilkent.edu.tr/~culha/cs548/hw2/**](http://www.cs.bilkent.edu.tr/~culha/cs548/hw2/) all equations available here

I have assumed the value of eta and Qstar.

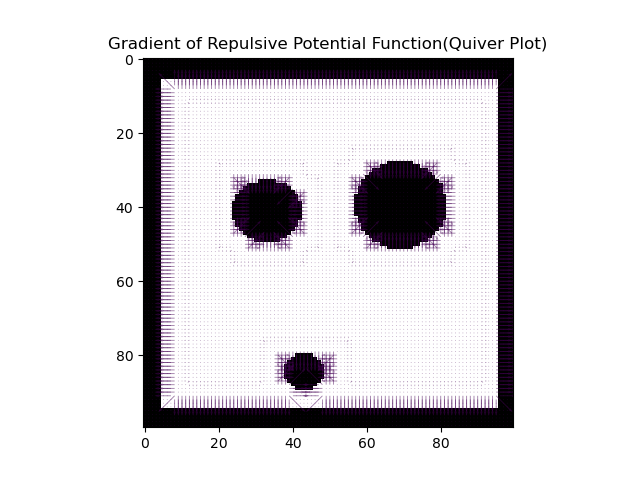




The following uses the above equation to find the value of Urep (Repulsive potential) and the gradient of repulsive potential(gradUrep)

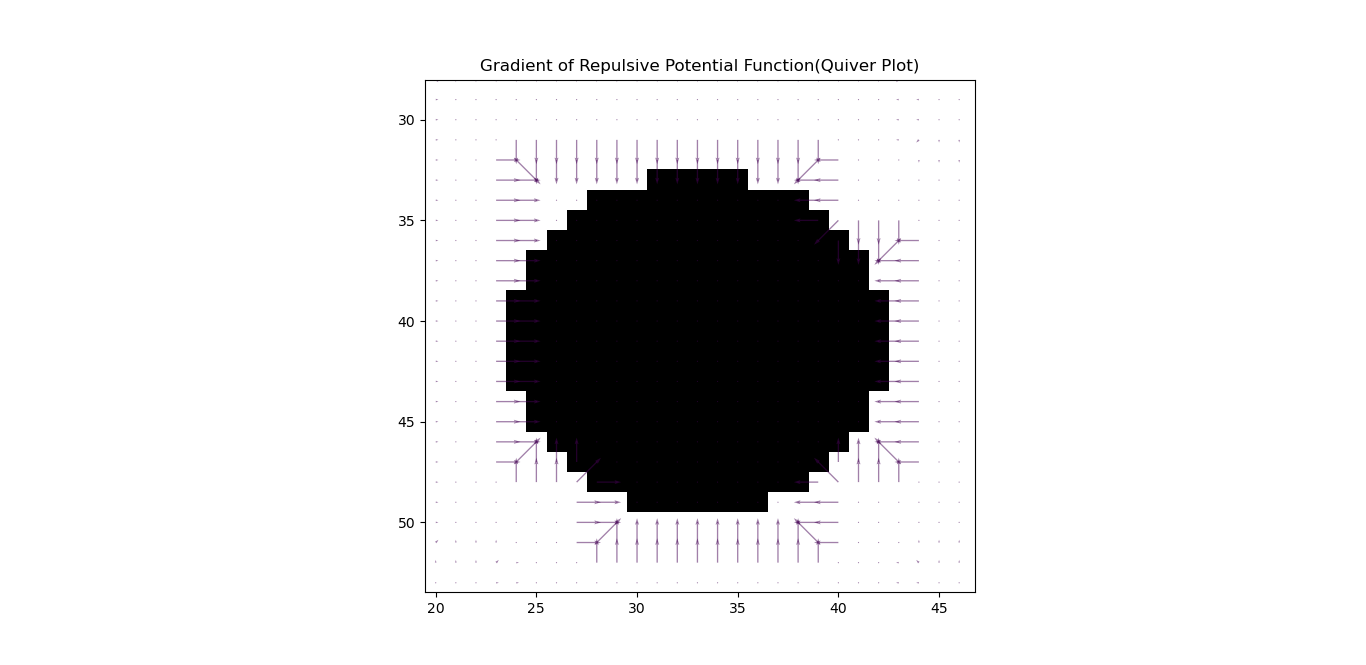
****

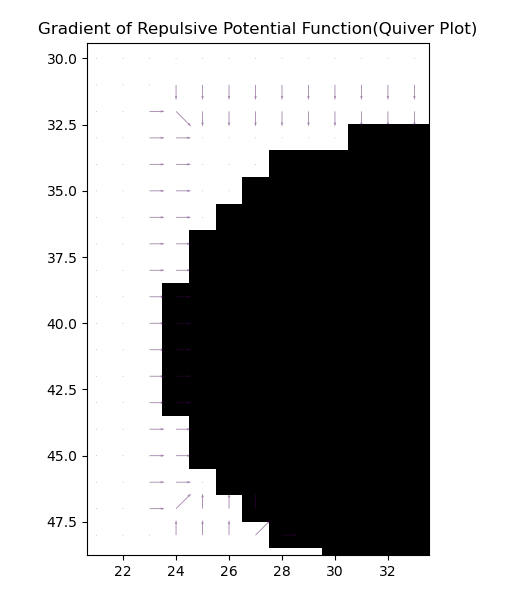
From the repulsive potential plot we can clearly infer that at the obstacle and the boundaries of the space, the potential is high and we get humps.

****

The above plot of the gradient of Repulsive potential represent the negative of the force to be applied at that location due to repulsion from the obstacle.

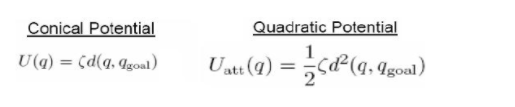
So as we move close to the obstacle the potential increase which mean that a greater force is experienced by the robot.

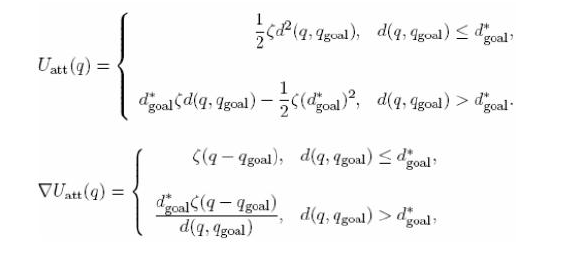
****

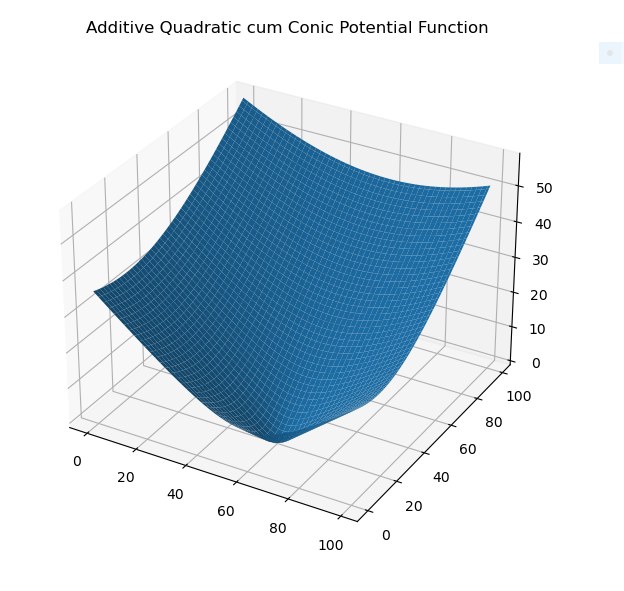


**Fig :** Gradient of Repulsive potential function close to the obstacle

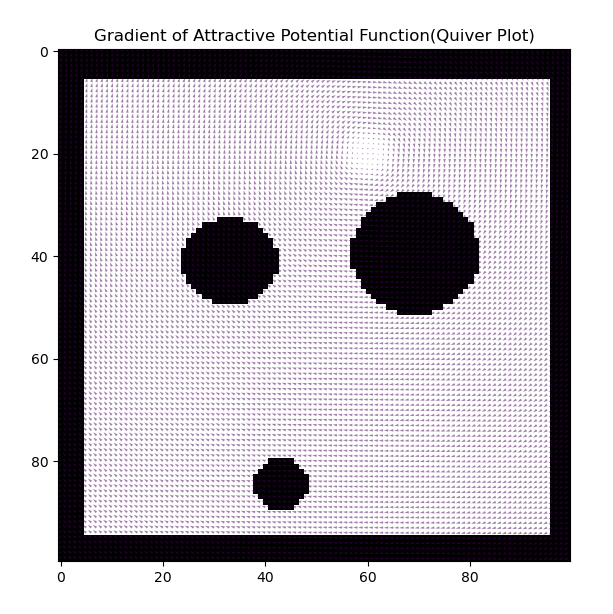
**Attractive Potential Function:**





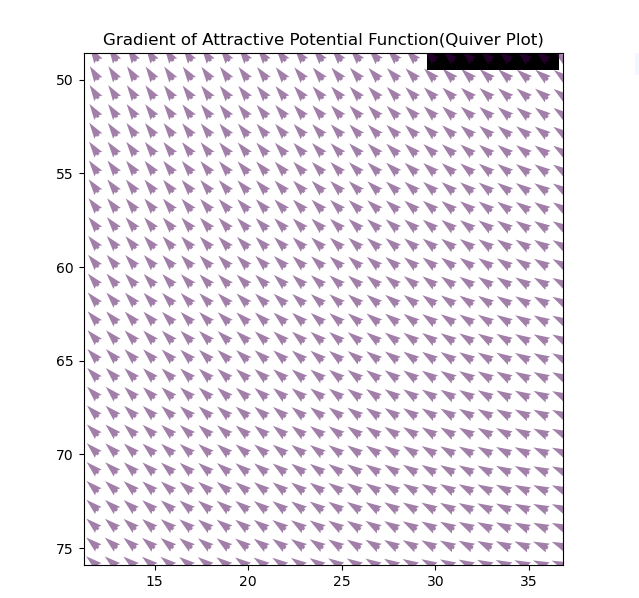


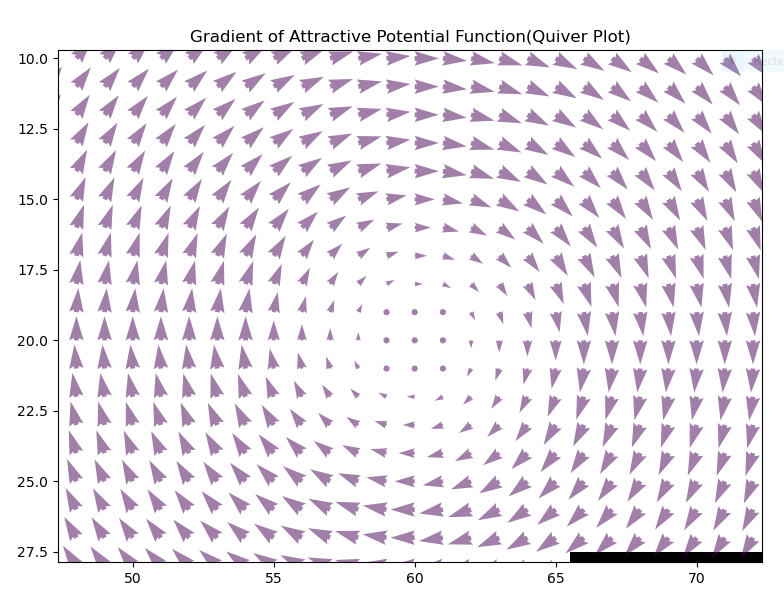
The following uses the above equation to find the value of Uatt (Attractive potential) and the gradient of attractive potential(gradUatt).



**Fig :** Gradient of Attractive Potential function

Clearly we can see that that the quadratic attractive function is working fine as we are getting zero potential at the (60, 20) and the variation of potential field gradient is quadratic.



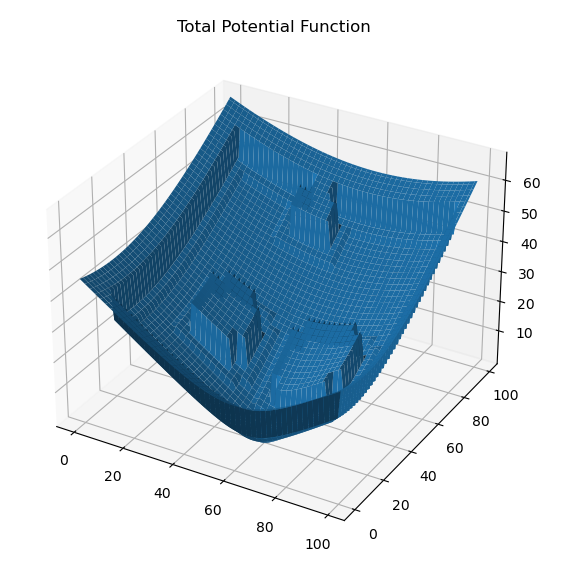


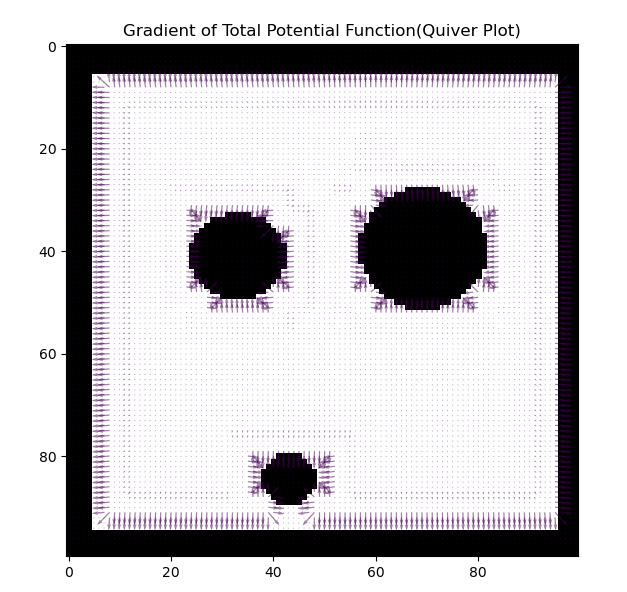
**Fig :**  Gradient of Quadratic Potential Function near the goal i.e. (60, 20)

**Total Potential Function:**



Getting the total potential at any location (i, j) as the algebraic sum of the attractive and repulsive potential.





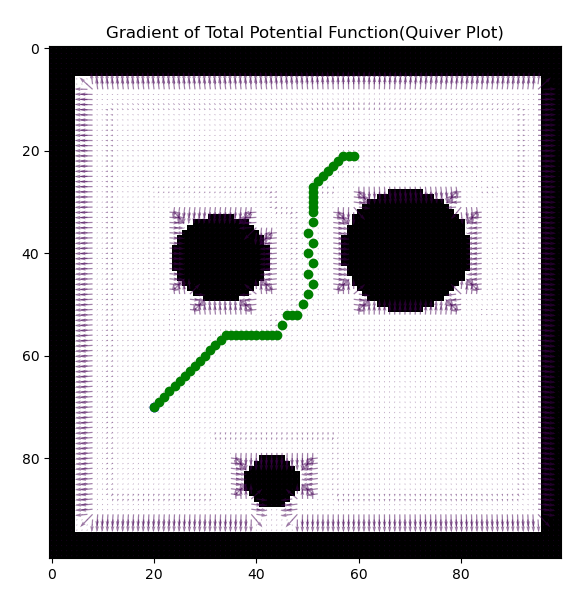
**Tunable Parameters:**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Effect on the vector Field** |
| **1.** | Size of grid (N) into which the space is divided into | As the number of grid increases space complexity and time complexity of the brushfire increases. Also for curved obstacle we will get now more close geometry to the curved of the world matrix. |
| **2.** |  | On increase of value , the vector field follow quadratic function at a distance lesser distance than from the goal, than the one with lesser |
| **3.** |  | On increase the gradient of attractive potential increases. i.e. the |
| **4.** |  | On increase of the value of Qstar, the repulsive potential is not zero at a greater distance from the obstacle. i.e. the influence of obstacle is felt now at far locations also. |
| **5.** | η | On increase, the value of the repulsive potential and its gradient increases. |

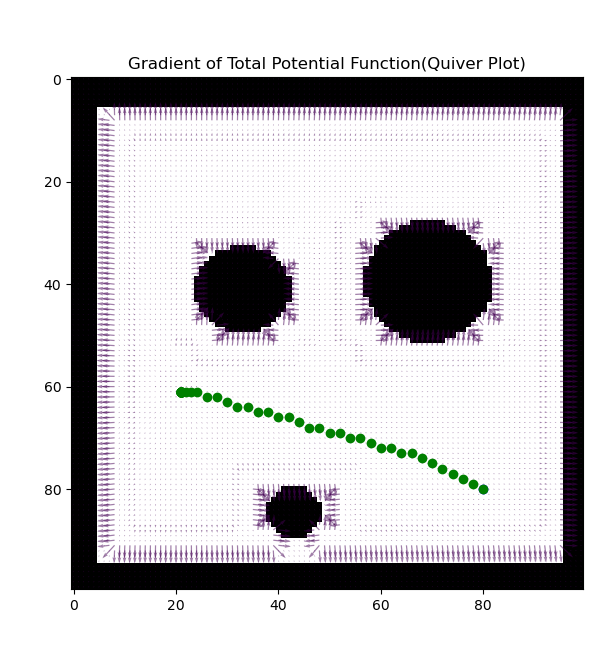
**Question 2.**

**Path Planning Using Brushfire, Potential Function and Gradient Descent :**

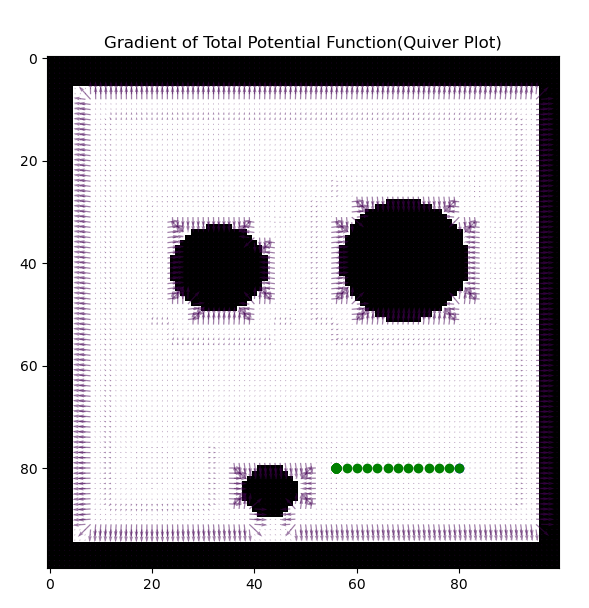
1. **Start = (20, 70) and Goal = (60, 20):**  Path in green



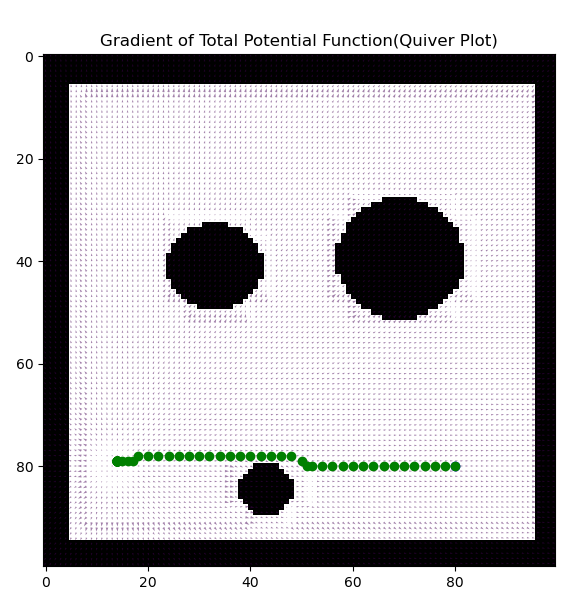
**Fig: Start = (20, 70) and Goal = (60, 20) and path in green**



**Fig: Start = (80, 80) and Goal = (20, 60) and path in green**



**Fig: Start = (80, 80) and Goal = (20, 80) and path in green (For this collision so occurring so I tweaked the parameter Qstar to avoid collision and reduce the effect of repulsive potential shown in next figure)**



**Fig: Start = (80, 80) and Goal = (13, 80) and path in green**