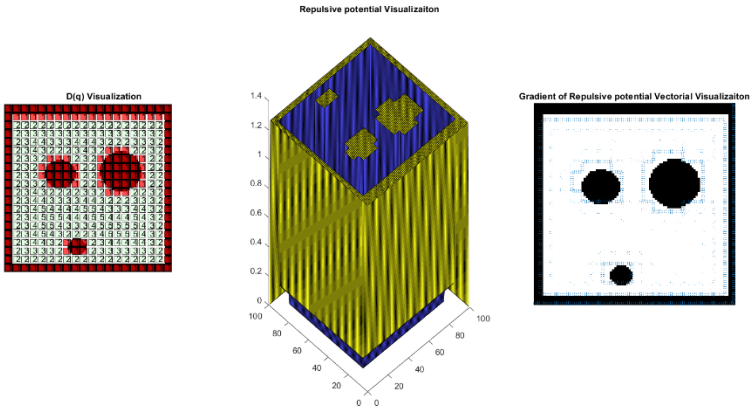
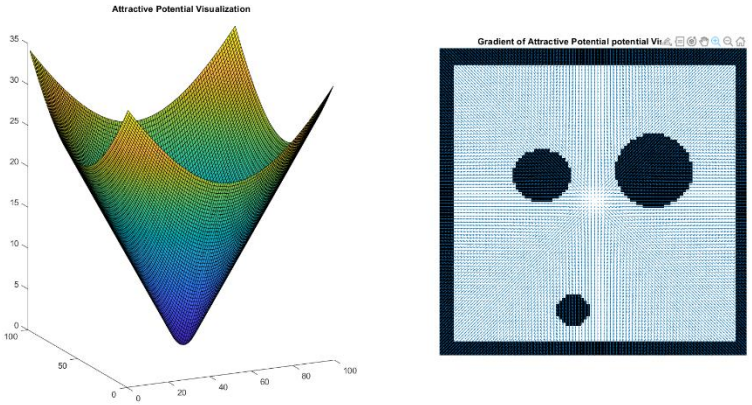
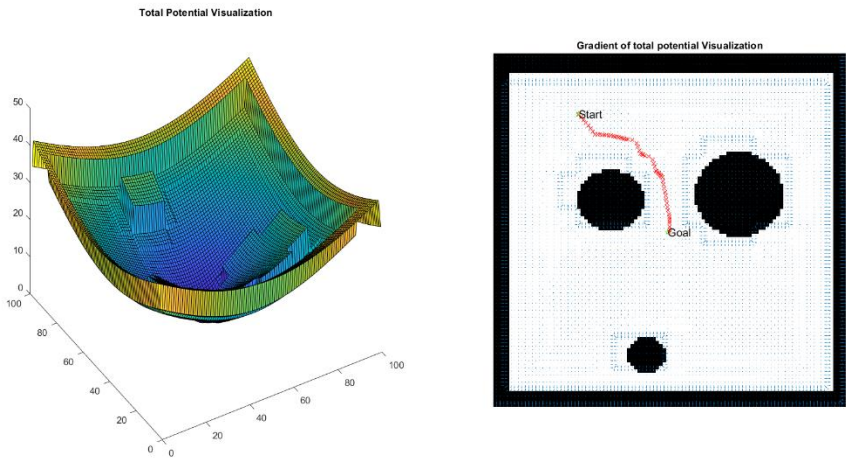
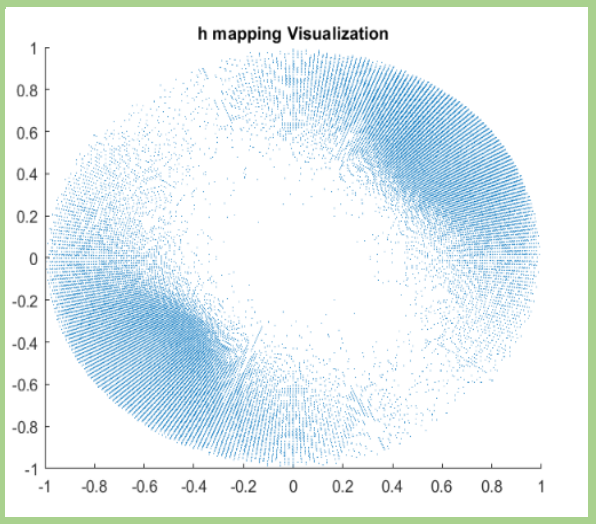
ROBOT MOTION PLANNING (ME510)

Assignment 2









ADITYA SHAH (Roll:2011mt02)

Department of Mechatronics

For the Academic Year 2021 - 2022

Code-1,2

```
1. %
2. %
3. %-------------------------------------------------------------------------
   ------------------------------------------------------------
4. %
5. % 1.For an R^2configuration space of size 100m × 100m,
6. % define an additive potential field with attractive potential as a
   combination of conic as well as quadratic potential
7. % and repulsive potential determined using Brushfire algorithm.
8. % Also determine the gradient of the potential function.
9. % Now implement your equations in a Matlab or any other programming
   language of your choice.
10.% Display the gradient vector field using arrows (in Matlab use quiver
   command).
11.% The arrows should be made at (1,1), (2,1)…(100,1), (1,2),
   (2,2)…(100,2),…(100,100)
12.% with the length of each arrow representing the magnitude of the gradient
   and
13.% direction of the arrow the direction of the gradient.
14.% You may use the enclosed Matlab implementation of Brushfire algorithm
   for determining minimum distance function.
15.% Note that the obstacles can be loaded via the enclosed image file which
   can be edited in MS Paint.
16.% In your report clearly describe the steps of your entire approach with
   all the equations.
17.% Make a list of the parameters  that need to be tuned.
18.% Also, show how/whether the vector field changes with variation of
   tunable
19.% parameters and describe. (25 points)
20.
21.%-------------------------------------------------------------------------
   ------------------------------------------------------------
22.% Goal Position input from user
23.xgx =input('Type in a value of x goal: ');
24.xgy=input('Type in a value of y goal: ');
25.Q_goal=[xgx,xgy];
26.
27.%starting position
28.xsx =input('Type in a value of x start: ');
29.xsy=input('Type in a value of y start: ');
30.QStartPosition=[xsx,xsy];
31.
32.
33.
34.figure;
35.    I=rgb2gray(imread('cs.png')); %load obstacle file
```

```
    subplot(1,3,1);
    title('D(q) Visualization');
    hold on;
    imshow(I);
    hold off;
    PixelResolution=20;
    nu=6;                       %Weightage to Repulsive Potential
    Imazenormalized=(I/255);
    WorldMap=zeros(PixelResolution,PixelResolution);
    hold on
    for i=1:(100/PixelResolution):100
    for j=1:(100/PixelResolution):100
        if min(min(Imazenormalized(j:j+(100/PixelResolution)-1,
   i:i+(100/PixelResolution)-1)))==0
            hold on;
            fill([i i+(100/PixelResolution)-1 i+(100/PixelResolution)-1
   i],[j j j+(100/PixelResolution)-1 j+(100/PixelResolution)-1],'r',
   'facealpha', 0.7);
            WorldMap(1+floor(i/(100/PixelResolution)),
   1+floor(j/(100/PixelResolution)))=1;
            text(i-1+(100/PixelResolution)/2,j-1+(100/PixelResolution)/2,
   '1');
            hold off;
        else
            hold on;
            fill([i i+(100/PixelResolution)-1 i+(100/PixelResolution)-1
   i],[j j j+(100/PixelResolution)-1 j+(100/PixelResolution)-1],'g',
   'facealpha', 0.1);
            WorldMap(1+floor(i/(100/PixelResolution)),
   1+floor(j/(100/PixelResolution)))=0;
            hold off;
        end
    end
end
hold off
%Defining Pixel Connectivity 4/8 connectivity concept
%Neighbors = [1 0;1 1;0 1;-1 1;-1 0;-1 -1; 0 -1;1 -1];
Neighbors = [1 0;0 1;-1 0; 0 -1];
%Implenmentation of Brushfire to find distToObst
counter = 1;
while min(min(WorldMap))==0
    for i=1:PixelResolution
        for j=1:PixelResolution
            if WorldMap(i,j)==counter
               %scan each neighbor
               for n=1:size(Neighbors,1)
                   if i+Neighbors(n,1)>0 && j+Neighbors(n,2)>0 &&...
                      i+Neighbors(n,1)<=PixelResolution &&
   j+Neighbors(n,2)<=PixelResolution
                       if WorldMap(i+Neighbors(n,1), j+Neighbors(n,2))==0
```

```
                            WorldMap(i+Neighbors(n,1), j+Neighbors(n,2)) =
   ...
                                WorldMap(i, j)+1;
                             % write the values of pixels
                            hold on
                            text(
   (100/PixelResolution)*(i+Neighbors(n,1))-(100/PixelResolution)/2,...
                                (100/PixelResolution)*(j+Neighbors(n,2))-
   ...
                                (100/PixelResolution)/2,
   num2str(WorldMap((i+Neighbors(n,1)), (j+Neighbors(n,2)))));
                            hold off
                        end
                    end
                end

            end
        end
    end

    counter = counter + 1;
end


% Finding Repulsive Potential

eta=4.5;
     WorldMap=WorldMap';
     [dworldx,dworldy]=gradient(WorldMap);

     [X,Y]=meshgrid(1:100,1:100);
     Urepulsive=zeros(size(X));
     Qstar=4;
     GradUrep=zeros([size(X),2]);
     for i=1:99
         for j=1:99

   Dq=WorldMap(1+floor(i/(100/PixelResolution)),1+floor(j/(100/PixelResolutio
   n)));

   GradDq=[dworldx(1+floor(i/(100/PixelResolution)),1+floor(j/(100/PixelResol
   ution))),dworldy(1+floor(i/(100/PixelResolution)),1+floor(j/(100/PixelReso
   lution)))];

              if Dq<=Qstar
                 Urepulsive(i,j)=0.5*eta*((1/Dq)-(1/Qstar))^2;
                 GradUrep(i,j,:)=eta*((1/Qstar)-(1/Dq))*(1/Dq^2)*GradDq;
             else
                 Urepulsive(i,j)=0;
                GradUrep(i,j,:)=[0,0];
```

```
            end
        end
    end
    [GradUrep(:,:,1),GradUrep(:,:,2)]=gradient(Urepulsive);
    subplot(1,3,2);
    title('Repulsive potential Visualizaiton');
    hold on;
    surf(X,Y,Urepulsive);
    hold off;
    %axis([1 100 1 100]);
    %daspect([1 1 1]);
    subplot(1,3,3);
    hold on;
    imshow(I);
    title('Gradient of Repulsive potential Vectorial Visualizaiton');
    quiver(X,Y,GradUrep(:,:,1),GradUrep(:,:,2));
    hold off;
    axis([1 100 1 100]);
    daspect([1 1 1]);


    figure;
    % Quadraturatic +Conic potential

    Zeta=0.1;
    Dstar=5;


    [X,Y]=meshgrid(1:100,1:100);
    Uattractive=zeros(size(X));
    gradUatt=zeros([size(X),2]);
    for i=1:100
        for j=1:100
        q=[i,j];

        if norm(Q_goal -q) <=Dstar              %Quadratic potential
            Uattractive(i,j)=0.5*Zeta*norm(Q_goal -q)^2;
            gradUatt(i,j,:)=Zeta*(q-Q_goal);
        else
            Uattractive(i,j)=Dstar*Zeta*norm(Q_goal -q) -
  0.5*Zeta*Dstar^2;          %Conic potential
            gradUatt(i,j,:)=Dstar*Zeta*(q-Q_goal)/norm(Q_goal -q);
        end


        end
    end
    [gradUatt(:,:,1),gradUatt(:,:,2)]=gradient(Uattractive);
    subplot(1,2,1);
    title('Attractive Potential Visualization');
```

```
167.      hold on;
168.      surf(X,Y,Uattractive);
169.      hold off;
170.      subplot(1,2,2);
171.      title('Gradient of Attractive Potential potential Visualization ');
172.      hold on;
173.      imshow(I);
174.      quiver(X,Y,gradUatt(:,:,1),gradUatt(:,:,2));
175.      hold off;
176.      axis([1 100 1 100]);
177.      daspect([1 1 1]);
178.      hold off;
179.      % Calcultaing Total potential Effect=Attractive + Repulsive in the
    whole
180.      % enivironment with some more weightage to Repulsive Potential.
181.      figure;
182.      U=Uattractive  + nu*Urepulsive;
183.      [gradU(:,:,1),gradU(:,:,2)]=gradient(U);
184.      %gradU=gradUatt+ gradUrep;
185.
186.      subplot(1,2,1);
187.      title('Total Potential Visualization');
188.      hold on;
189.      surf(X,Y,U);
190.      hold off;
191.      subplot(1,2,2);
192.      title('Gradient of total potential Visualization');
193.
194.      hold on;
195.      imshow(I);
196.      quiver(X,Y,gradU(:,:,1),gradU(:,:,2));
197.      hold off;
198.      %-----------------------------------------------------------------
    ----------------------------------------------------------------
199.      %
200.      % 2.Now, implement a gradient descent approach to calculate a path
201.      % from any given start point to goal location using the results in
    (1)% .
202.      % Assume that start and goal location will never be specified on any
    obstacles.
203.      % Test your algorithm for several start and goal locations.
204.      % Comment on completeness of the approach you developed. (25 points)
205.      %-----------------------------------------------------------------
    ----------------------------------------------------------------
206.
207.      % Gradient descent Calculation
208.
209.      hold on;
210.      plot(QStartPosition(1),QStartPosition(2),'gp');
211.      text(QStartPosition(1),QStartPosition(2),'Start',"FontSize",12);
```

```
212.      plot(Q_goal(1),Q_goal(2),'gp');
213.      text(Q_goal(1),Q_goal(2),'Goal',"FontSize",12);
214.      hold off;
215.      q=QStartPosition;
216.      alpha=2.1;
217.      grad=[gradU(q(2),q(1),1),gradU(q(2),q(1),2)];
218.      epsilon=0.1;
219.      while norm(grad)>epsilon
220.          hold on;
221.          plot(q(1),q(2),'rx');
222.          pause(.1);
223.          hold off;
224.
   grad=[gradU(round(q(2)),round(q(1)),1),gradU(round(q(2)),round(q(1)),2)];
225.          q=(q-(alpha*grad));
226.      end
227.
228.      axis([1 100 1 100]);
229.      daspect([1 1 1]);
```

Code-3

```
1. %-------------------------------------------------------------------------
   ------------------------------------------------------------
2. %
3. % 3.Bonus problem: Solve the problem in 1 and 2 using
4. % navigation function approach with star domain mapping. (25 points)
5. %-------------------------------------------------------------------------
   ------------------------------------------------------------
6.
7. %Defining Parameter
8. %si is the center of the star-shaped set
9. %qi and ri are, respectively, the center and radius of the spherical
   obstacle
10.  r0=1;
11.  ri=0.1;
12.  q0=[0,0];
13.  q1=[-0.5,-0.4];
14.  s1=[0.4 0];   %centre of star
15.  q2=[0.6,0.5];
16.  s2=[0 0.4];    %centre of star
17.
18.  qgoal=[0.0,0.0];
19.
20.
21.  kappa=3;
22.  % Creating a MeshGrid For Analysis purpose ,not require although.
23.  [r,theta]=meshgrid(0:0.01:1,0:1*pi/180:2*pi);
24.  [x,y]=pol2cart(theta,r);
```

```
% Calculaing Beta's.
Beta0=-((x-q0(1)).^2 + (y-q0(2)).^2) +r0^2;
Beta1=((x-q1(1)).^2 + (y-q1(2)).^2) -ri^2;
Beta2=((x-q2(1)).^2 + (y-q2(2)).^2) -ri^2;


Beta=Beta0.*Beta1.*Beta2;
% Calculating Vi.
v1=(((1+ Beta1).^0.5).*(ri./(((x-s1(1)).^2 + (y-s1(2)).^2).^0.5)));
v2=(((1+ Beta2).^0.5).*(ri./(((x-s2(1)).^2 + (y-s2(2)).^2).^0.5)));
% Calculating Ti
T1x=v1.*(x-s1(1)) + q1(1);
T1y=v1.*(y-s1(2)) + q1(2);
T2x=v2.*(x-s2(1)) + q2(1);
T2y=v2.*(y-s2(2)) + q2(2);
% Beta Visualization
subplot(2,3,1);
hold on;
title('Beta Visualization');
surf(x,y,Beta);
hold off;
% Gamma Visualization.

Gamma=((x-qgoal(1)).^2 + (y-qgoal(2)).^2).^kappa;
subplot(2,3,2);
hold on;
title('\Gamma');
surf(x,y,Gamma);
hold off;
% Gamma/Beta Visualization.

subplot(2,3,3);
hold on;
title('\Gamma/Beta');
surf(x,y,Gamma./Beta);
hold off;




% Phi Calculation
lambda=1;
phi1=(Gamma.*(Beta0.*Beta2))./(lambda.*Beta1 +Gamma.*(Beta0.*Beta2));
phi2=(Gamma.*(Beta0.*Beta1))./(lambda.*Beta2 +Gamma.*(Beta0.*Beta1));
phigoal=1-phi1 -phi2;
hx=phigoal.*qgoal(1) +phi1.*T1x+phi2.*T2x;
hy=phigoal.*qgoal(2) +phi1.*T1y+phi2.*T2y;




% Plotting h mapping.
figure;
hold on;

title('h mapping Visualization')
```

```
82.  quiver(x,y,hx,hy);
83.  hold off;
```

Path Planning Using Potential Functions

- Robot is assumed as a positive charge
- Goal as negative charge
- Obstacle as positive charge
- Positively charged robot will get attracted towards negatively charged goal
- But Positively charged robot gets repelled by positively charged obstacle
- Robot follows a path by following negated gradient of potential function
- We know, Conic Potential poses discontinuity problem at the origin causing chattering and Quadratic Potential grows without bound as q moves away from q_goal.
- To overcome the limitation of both Quadratic and Conic Potential, I have used both the potential to get more desired Attractive Potential Function.
- So, Conic potential attracts the robot when it is very distant from q_goal and the Quadratic potential attracts the robot when it is near $qgoal$,ths 8vercoming each other limitation.

$$U_{\text{att}}(q) = \begin{cases} \frac{1}{2}\zeta d^2(q, q_{\text{goal}}), & d(q, q_{\text{goal}}) \leq d^*_{\text{goal}}, \\ d^*_{\text{goal}}\zeta d(q, q_{\text{goal}}) - \frac{1}{2}\zeta (d^*_{\text{goal}})^2, & d(q, q_{\text{goal}}) > d^*_{\text{goal}}. \end{cases}$$

$$\nabla U_{\text{att}}(q) = \begin{cases} \zeta(q - q_{\text{goal}}), & d(q, q_{\text{goal}}) \leq d^*_{\text{goal}}, \\ \frac{d^*_{\text{goal}}\zeta(q - q_{\text{goal}})}{d(q, q_{\text{goal}})}, & d(q, q_{\text{goal}}) > d^*_{\text{goal}}, \end{cases}$$

Where, d^*_{goal} is the threshold from the goal where the planner switches between conic and quadratic potential.

- Repulsive Potential repel the robot when robot comes under domain of influence of Obstacles ,else no repulsion.

$$U_{\text{rep}}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{D(q)} - \frac{1}{Q^*}\right)^2, & D(q) \le Q^*, \\ 0, & D(q) > Q^*, \end{cases}$$

$$\nabla U_{\text{rep}}(q) = \begin{cases} \eta\left(\frac{1}{Q^*} - \frac{1}{D(q)}\right)\frac{1}{D^2(q)}\nabla D(q), & D(q) \le Q^*, \\ 0, & D(q) > Q^*, \end{cases}$$

Where, Q^* is the region of influence of obstacles for the robot.

- Finally using Gradient Descent technique, Robot finds a path to goal and at goal we are using epsilon , considering tolerance of 0.1, because $\nabla U(qi)$ reduces but does not vanish at goal during implementation.

Description of Brushfire:

Algorithm Descriptions

- The brushfire algorithm relies upon a grid-based world, either 4 or 8 connected.
- Obstacles are initialized to a value of 1, while the free space is initialized to zero.
- The grid is iteratively searched, when a pixel is found with a value of i, its adjacent zero-valued pixels are set to i+1.
- This process continues until no more zero-pixels are found in the graph.
- The value in the cell represents the distance to the closest obstacle and this is used in Repulsive Potential for D(q) .

Equation:

$$U_{att}(q) = \begin{cases} \frac{1}{2}\zeta d^2(q, q_{goal}), & d(q, q_{goal}) \le d^*_{goal}, \\ d^*_{goal}\zeta d(q, q_{goal}) - \frac{1}{2}\zeta (d^*_{goal})^2, & d(q, q_{goal}) > d^*_{goal}. \end{cases}$$

$$\nabla U_{att}(q) = \begin{cases} \zeta(q - q_{goal}), & d(q, q_{goal}) \le d^*_{goal}, \\ \frac{d^*_{goal}\zeta(q - q_{goal})}{d(q, q_{goal})}, & d(q, q_{goal}) > d^*_{goal}, \end{cases}$$

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{D(q)} - \frac{1}{Q^*}\right)^2, & D(q) \le Q^*, \\ 0, & D(q) > Q^*, \end{cases}$$

$$\nabla U_{rep}(q) = \begin{cases} \eta\left(\frac{1}{Q^*} - \frac{1}{D(q)}\right)\frac{1}{D^2(q)}\nabla D(q), & D(q) \le Q^*, \\ 0, & D(q) > Q^*, \end{cases}$$

Parameter Description:

Eta: gain on the Repulsive gradient, **set to 4.5**

Zeta: gain on the attractive force, **set to 0.1**
DStar: threshold distance from the goal where the planner switches between conic and quadratic potentials, **set to 5**

Nu: Weightage to Repulsive Potential, **set to 6**

Dq: Value **read from the brushfire graph**

QStar: Minimum distance to consider the obstacle, **set to 4**

Alpha: Gradient Descent Parameter, **set to 2.1**

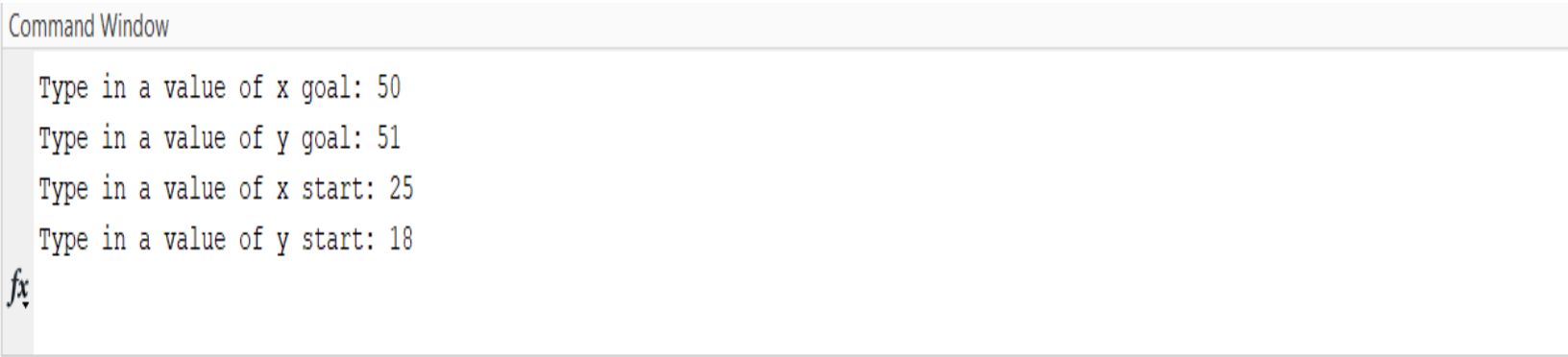
Epsilon: Tolerance for Grad at goal position, **set to 0.1**

<u>Attractive Repulsive potential technique is not complete</u>

- **Problem of Local Minima**

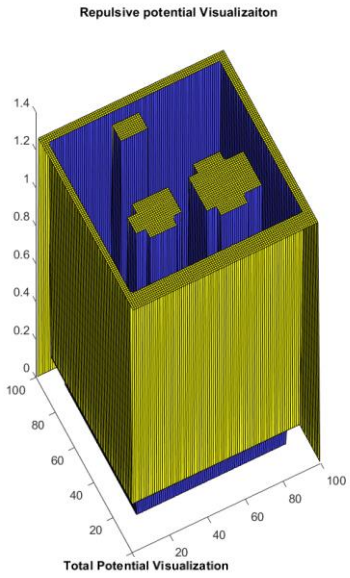
Illustrations:

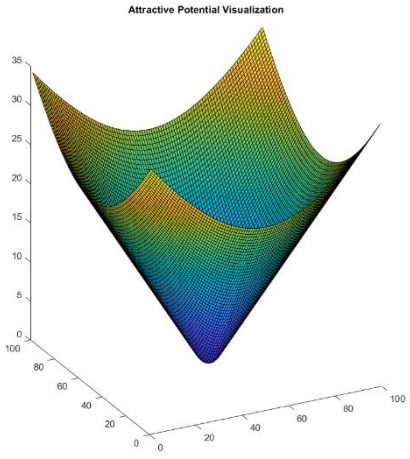
For following Configuration:

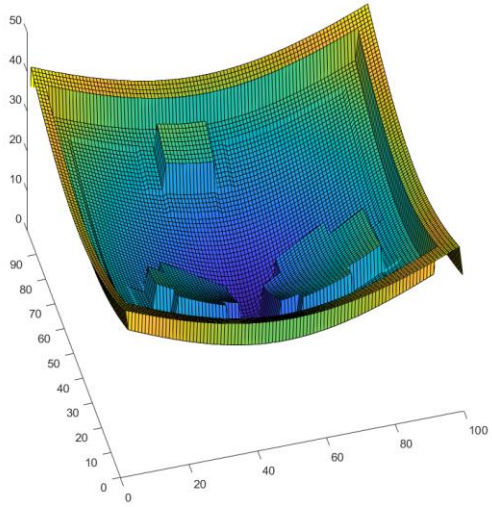


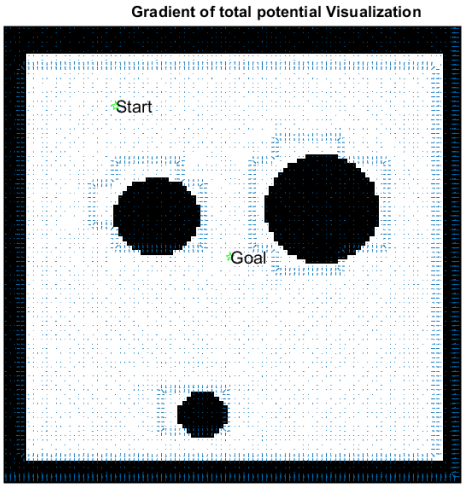
1st Case:

- Eta:4.5
- Zeta:0.1
- DStar:5
- Nu:6
- QStar:4
- Alpha:2.1
- Epsilon:0.1





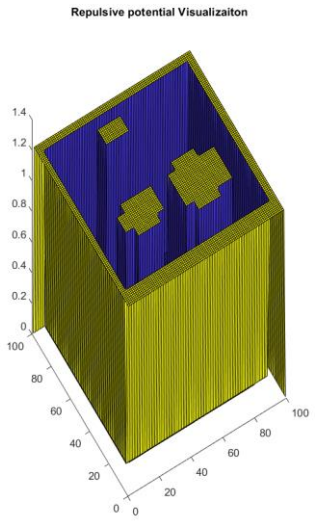


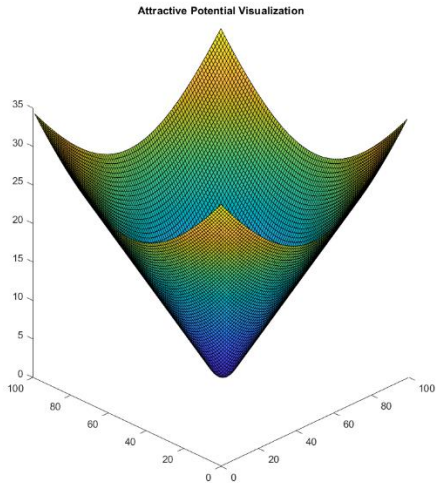


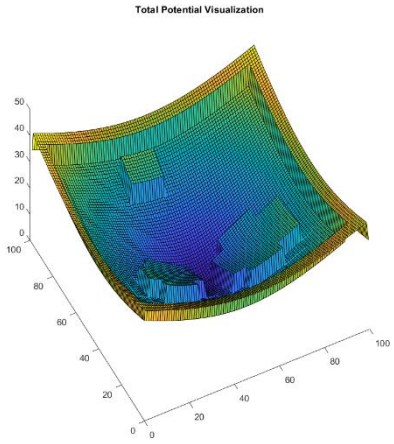
Analysis: We can see that near the obstacles, we are having high repulsive potential and its influence decreases gradually as distance to obstacles increases. And at Goal position the Potential is minimum and smooth.

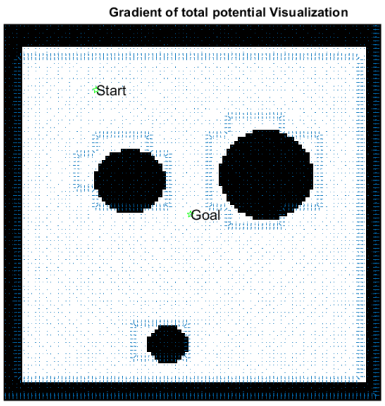
2nd Case:

- Eta:5.5
- Zeta:0.1
- DStar:5
- Nu:6
- QStar:3
- Alpha:2.1
- Epsilon:0.1





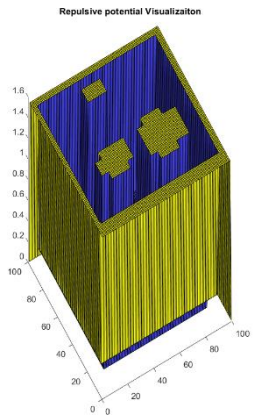


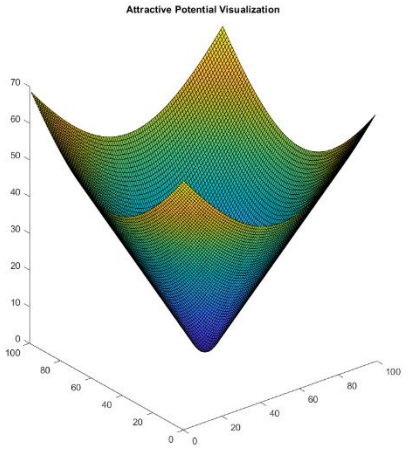


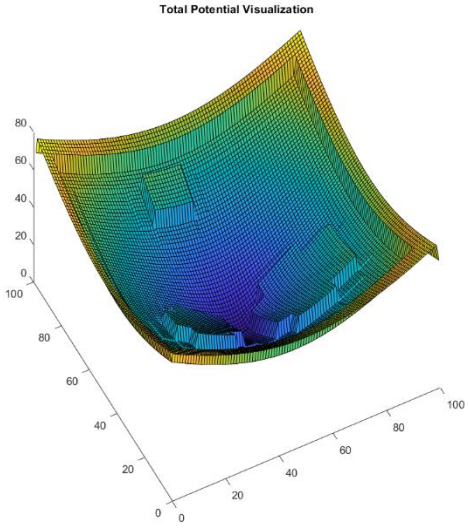
Analysis: Here we will observe because of lower QStar and higher Eta, Repulsive Potential has more stronger influence but the region of influence has reduced,w.r.t to previous one.

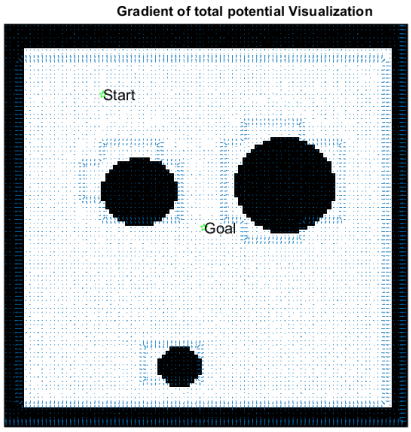
3rd Case:

- Eta:5.5
- Zeta:0.2
- DStar:5
- Nu:6
- QStar:4
- Alpha:2.1
- Epsilon:0.1





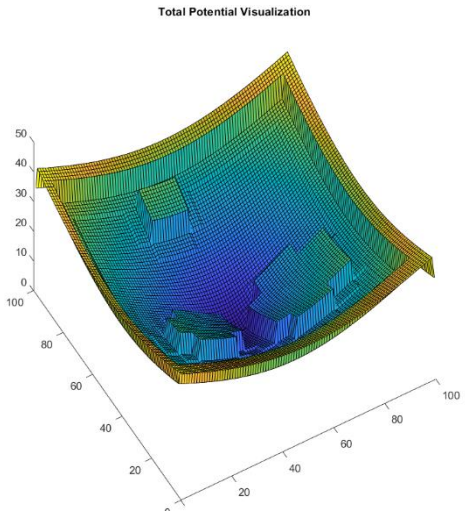


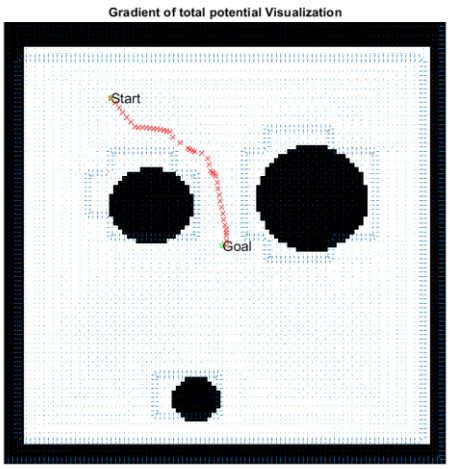


Analysis: Here we observe that the attractive potential effect has increased as Zeta has increased w.r.t previous case .

4th Case:

- Eta:4.5
- Zeta:0.1
- DStar:5
- Nu:6
- QStar:4
- Alpha:2.6
- Epsilon:0.1





Analysis: Here we observe w.r.t case 1, the iteration time has decreased significantly as the alpha value is increased little bit. Thus, convergence rate has increased but the it has to be tuned in such a manner such that it doesn't lead to oscillation due to several overshoot near goal location.

Decision for Proper Tuning taken During Final Implementation:

- Eta: gain on the Repulsive gradient, **set to 4.5**

Given an weightage to Repulsive potential and for conservative approach given a , higher value >1 is selected and by analysis ,I found 4.5 to be good as it gave good results .

- Zeta: gain on the attractive force, **set to 0.1**

Given an weightage to Attractive Potential, but < 1 as high attractive potential may lead to instability and also to get a smoother gradient(used both quad and conic potential function). So, given 0.1 after analysis.

- DStar: threshold distance from the goal where the planner switches between conic and quadratic potentials, **set to 5**

Choosing the key point for shift of Attractive potential from Conic to Quadratic Attractive Potential,considering the distance from q to q_goal.

- Nu: Weightage to Repulsive Potential, **set to 6**

Conservative Approach→ environment with some more weightage to Repulsive Potential given and after analysis found as 6.

- QStar: Minimum distance to consider the obstacle, **set to 4**

Chosen from brushfire algorithms, for considering the region of influence of obstacles and thus chosen 4 .

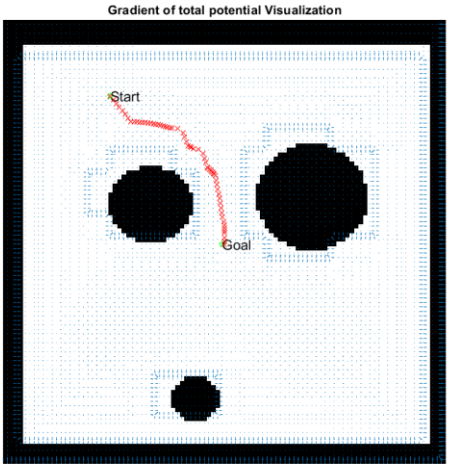
- Alpha: Gradient Descent Parameter, **set to 2.1**

Grad Descent Parameter ,chosen in such a manner so no overshoot /oscillation occur and iteration rate is also good. So chosen a value >1 ,giving good results with no overshoot. Found to be 2.1.

- Epsilon: Tolerance for Grad at goal position, **set to 0.1**

Considering the tolerance considerable ,for the iteration to end for gradient descent .

Testing Completeness With Given Parameter:

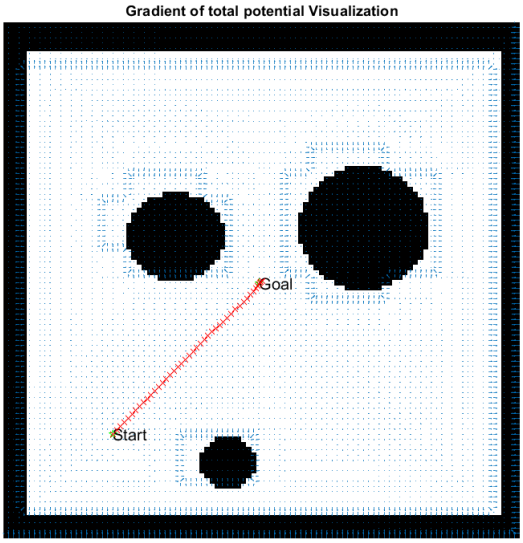


- Eta:4.5
- Zeta:0.1
- DStar:5
- Nu:8
- QStar:4
- Alpha:2.1
- Epsilon:0.1

- Case 1:

- Q_goal=[50,51];
- QStartPosition=[25, 18];

➢ **Reached Goal**



- Case 2:

❖ Q_goal=[50,51];
❖ QStartPosition=[22,80];

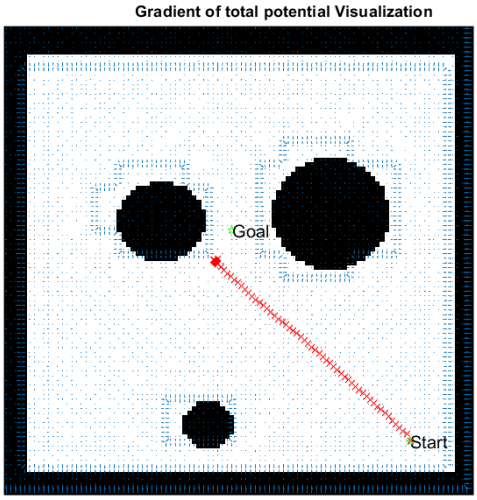
➢ **Reached Goal**

- Case 3:

❖ Q_goal=[49,44];

❖ QStartPosition=[87,89];

➢ **Goal not reached.**

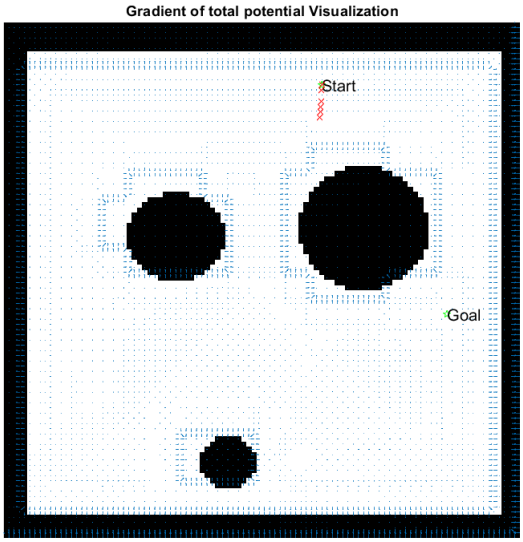


- Case4:

❖ Q_goal=[86,57];

❖ QStartPosition=[62,13];

➢ **Goal not reached.**



<u>Inferences:</u>

Thus, from above we can say that the given solution for Path Planning is **not complete.**

- **Attractive Repulsive potential technique is not complete.**

It led to local minima instead of reaching Goal. Also, after proper tuning for parameter, it can have local minima problem in the space.

3rd Question: **Navigation Function Approach with Star Domain Mapping.**

Equation used:

- h mapping between the star- and sphere-spaces constructed using a translated scaling map

$$T_i(q) = \nu_i(q)(q - q_i) + p_i, \text{ where } \nu_i(q) = \left(1 + \beta_i(q)\right)^{0.5} \frac{r_i}{d(q,q_i)}$$

Where, qi is the centre of the star-shaped set,

pi and ri are, respectively, the centre and radius of the spherical obstacle

and βi (q) defines a star-shaped set

- For star-shaped obstacle OQi, analytical switch is defined as

$$s_i(q,\lambda) = \left(\sigma_\lambda o \frac{\gamma_\kappa \bar{\beta}_i}{\beta_i}\right)(q) = \frac{\gamma_\kappa \bar{\beta}_i}{\gamma_\kappa \bar{\beta}_i + \lambda \beta_i} \bar{\beta}_i(q) = \prod_{j=0, j\neq i}^{n} \beta_j$$

- For goal the analytical switch is defined as:

$$s_{q_{goal}}(q,\lambda) = 1 - \sum_{i=0}^{M} s_i$$

- The mapping is thus defined as:

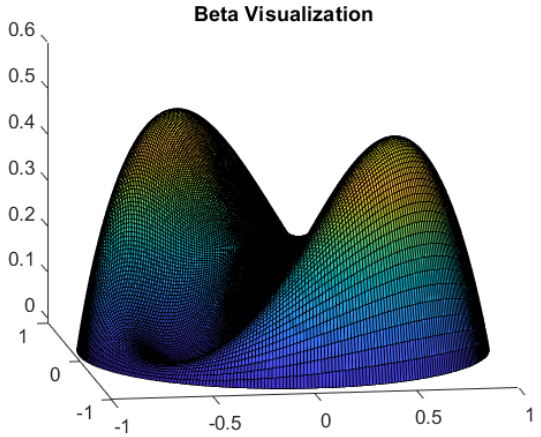
$$h_\lambda(q) = s_{q_{goal}}(q,\lambda) T_{q_{goal}}(q) + \sum_{i=0}^{M} s_i(q,\lambda) T_i(q)$$

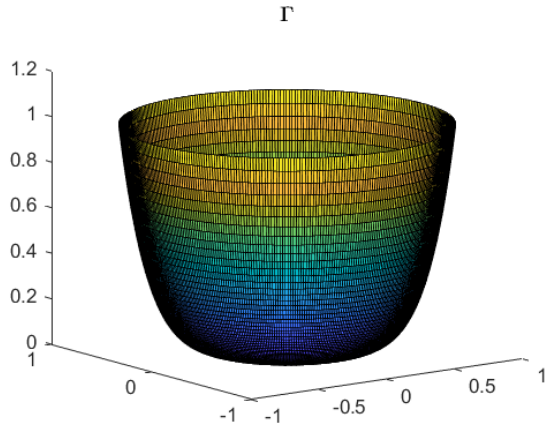
where $T_{\text{qgoal}}(q)$=q is just the identity map

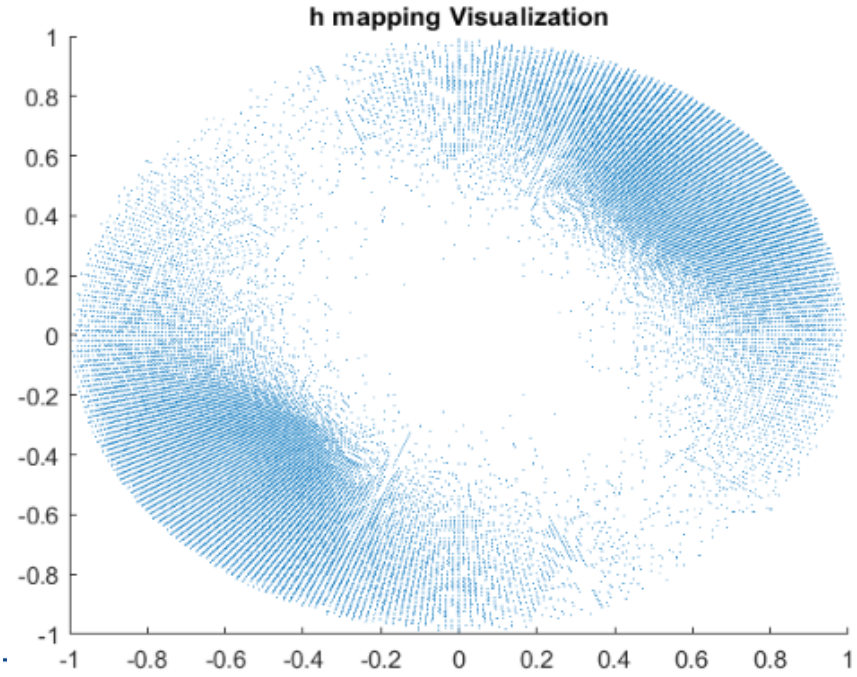
Parameter Description:

- Kappa: Associated with Attractive part of navigation function, **set as 3**.
- Lambda: Associated with analytical switch function ,**set as 1**.

Testing:







Inferences:

- We can observe here that there are two humps in beta visualization, signifying two obstacles.
- From the 2nd plot, we can observe the attractive part of navigation function, having a smooth minimum at the goal location.
- From the 3rd plot, we observed the h mapping between the star- and sphere-spaces and distribution of potential in the space and guaranteed to have a single minimum at q_goal for a sufficiently large κ.

To View the Simulation Video, Plot and Code, click on the following link:

Assignment -2