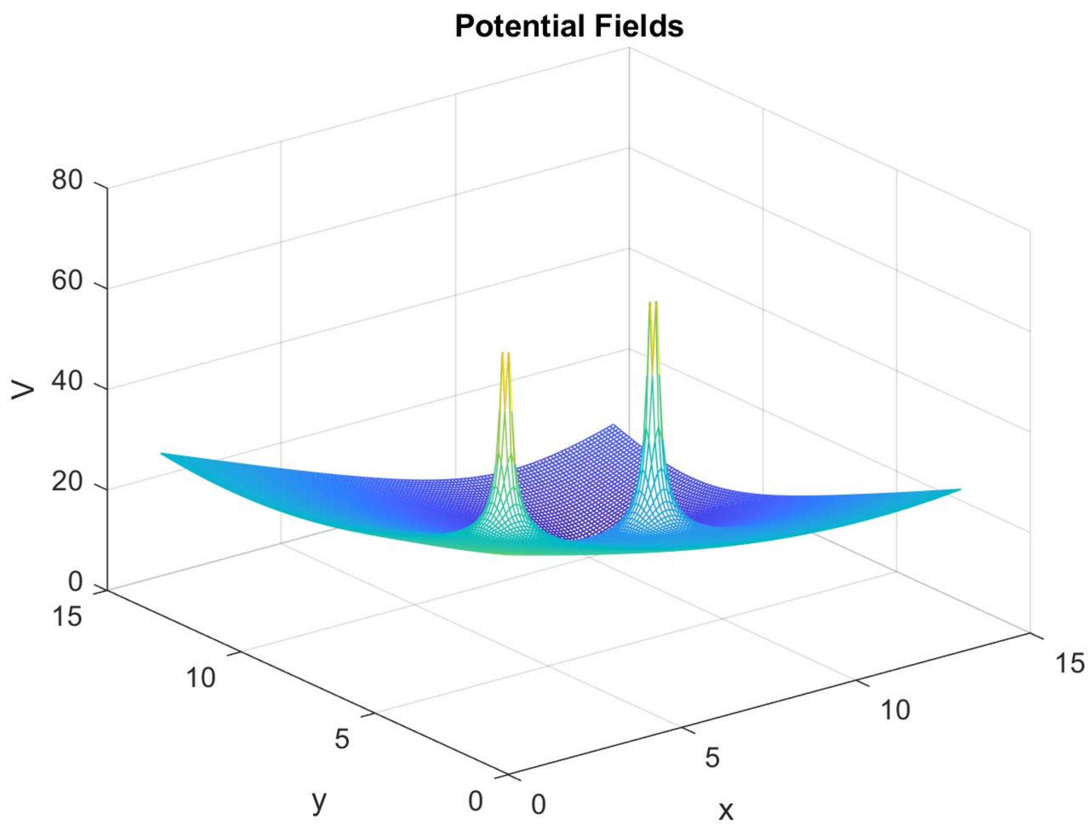
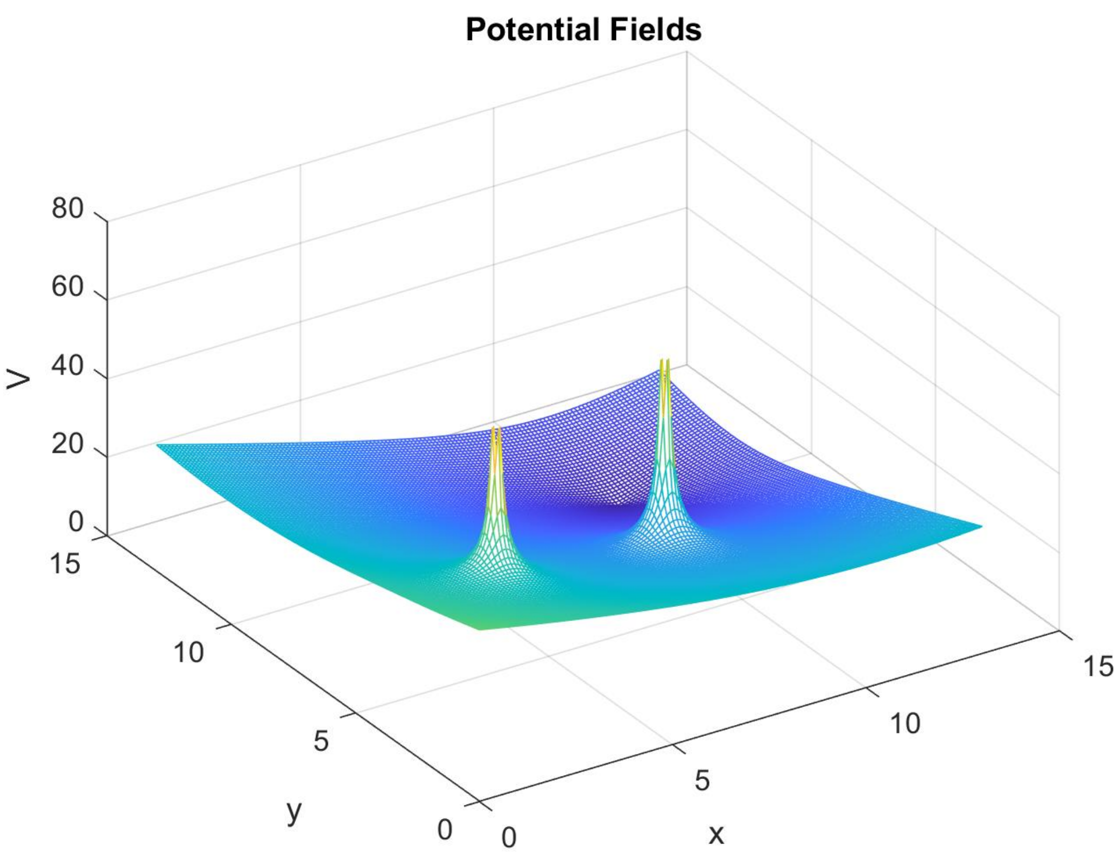
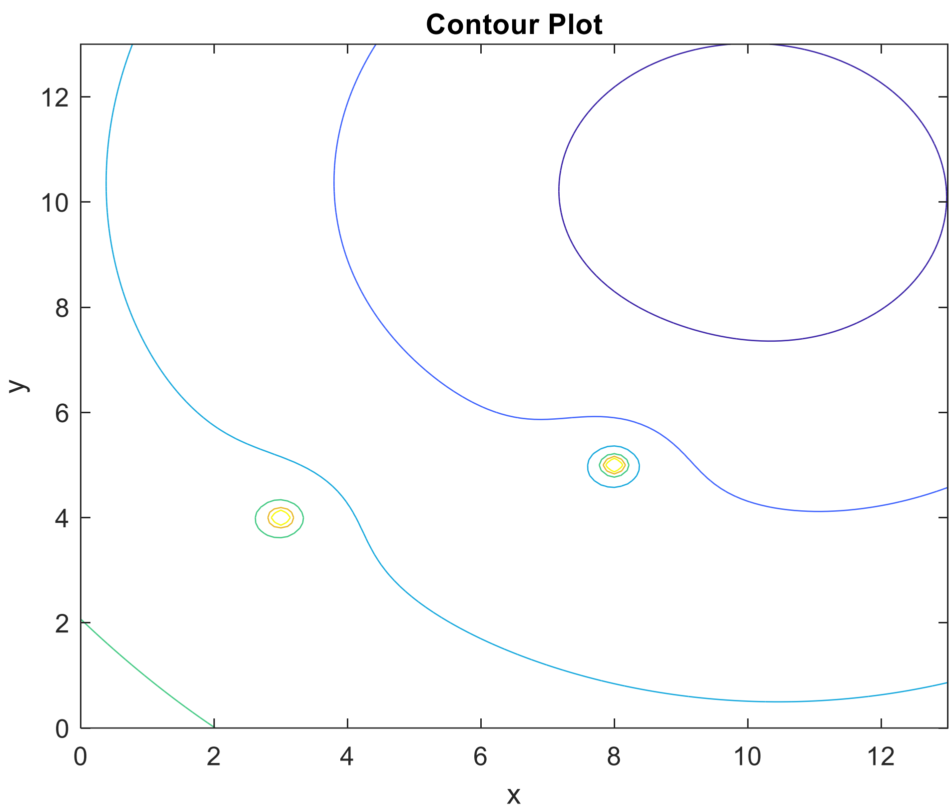
Generating the Potential Field with two obstacles







MATLAB Code

```
function PotentialFieldGeneration

%work area

wa=[0:0.1:13;0:0.1:13];

%locations of obstacles and target

T=[10,10];

obs=[3,4;8,5];

n=length(wa(1,:));

%gains

kT=3;

ko=[4,5];


for p=1:n

    for q=1:n

        VT(q,p)=kT*sqrt(((wa(1,p)-T(1,1))^2)+(wa(2,q)-T(1,2))^2);

        Vo1(q,p)=ko(1,1)/sqrt(((wa(1,p)-obs(1,1))^2)+(wa(2,q)-obs(1,2))^2);

        Vo2(q,p)=ko(1,2)/sqrt(((wa(1,p)-obs(2,1))^2)+(wa(2,q)-obs(2,2))^2);
```

```
        V=VT+Vo1+Vo2;
    end
end
figure
mesh(wa(1,:),wa(2,:),V)

figure
contour(wa(1,:),wa(2,:),V)
end
```

Calculating Forces and Control System to guide the UGV to goal with front steered dynamics

a. Note: All the forces can be broken into x and y components using the matrices so I didn't write the formulae again and again to avoid redundancy.

Force due to obstacle 1

$$F_{o1} = -\frac{k_{o1}}{{r_1}^2}\begin{bmatrix} \frac{3-x}{r_1} \\ \frac{4-y}{r_1} \end{bmatrix}$$

where k_{o1} = gain for obstacle 1

$$r_1 = \sqrt{[(3-x)^2 + (4-y)^2]}$$

Force due to obstacle 2

$$F_{o2} = -\frac{k_{o2}}{{r_2}^2}\begin{bmatrix} \frac{8-x}{r_2} \\ \frac{5-y}{r_2} \end{bmatrix}$$

where k_{o2} = gain for obstacle 2

$$r_2 = \sqrt{[(8-x)^2 + (5-y)^2]}$$

Force due to goal

$$F_G = k_g\begin{bmatrix} \frac{10-x}{r_g} \\ \frac{10-y}{r_g} \end{bmatrix}$$

where k_g = gain for goal

$$r_g = \sqrt{[(10-x)^2 + (10-y)^2]}$$

Total Force

$$F_{total} = F_{o1} + F_{o2} + F_G$$

$$F_{total} = -\frac{k_{o1}}{{r_1}^2}\begin{bmatrix} \frac{3-x}{r_1} \\ \frac{4-y}{r_1} \end{bmatrix} - \frac{k_{o2}}{{r_2}^2}\begin{bmatrix} \frac{8-x}{r_2} \\ \frac{5-y}{r_2} \end{bmatrix} + k_g\begin{bmatrix} \frac{10-x}{r_g} \\ \frac{10-y}{r_g} \end{bmatrix}$$

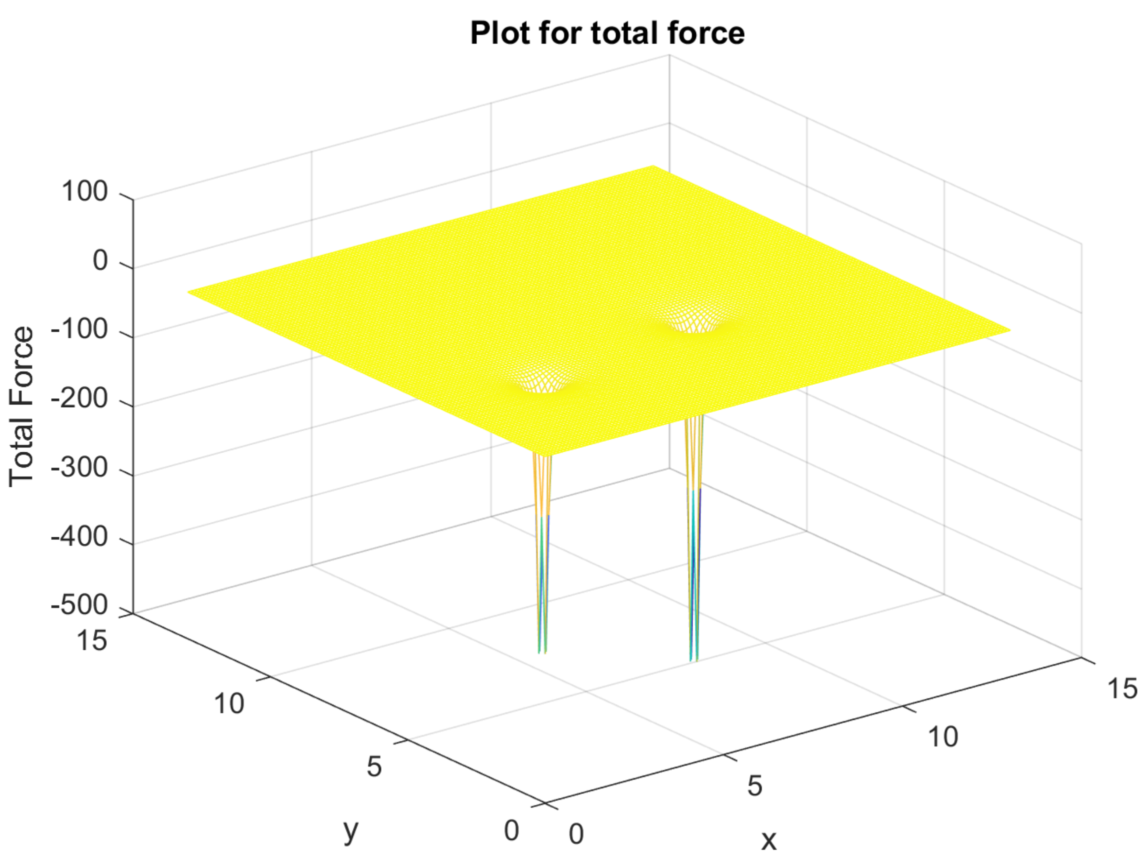
This can be simplified and written without the individual x and y components as

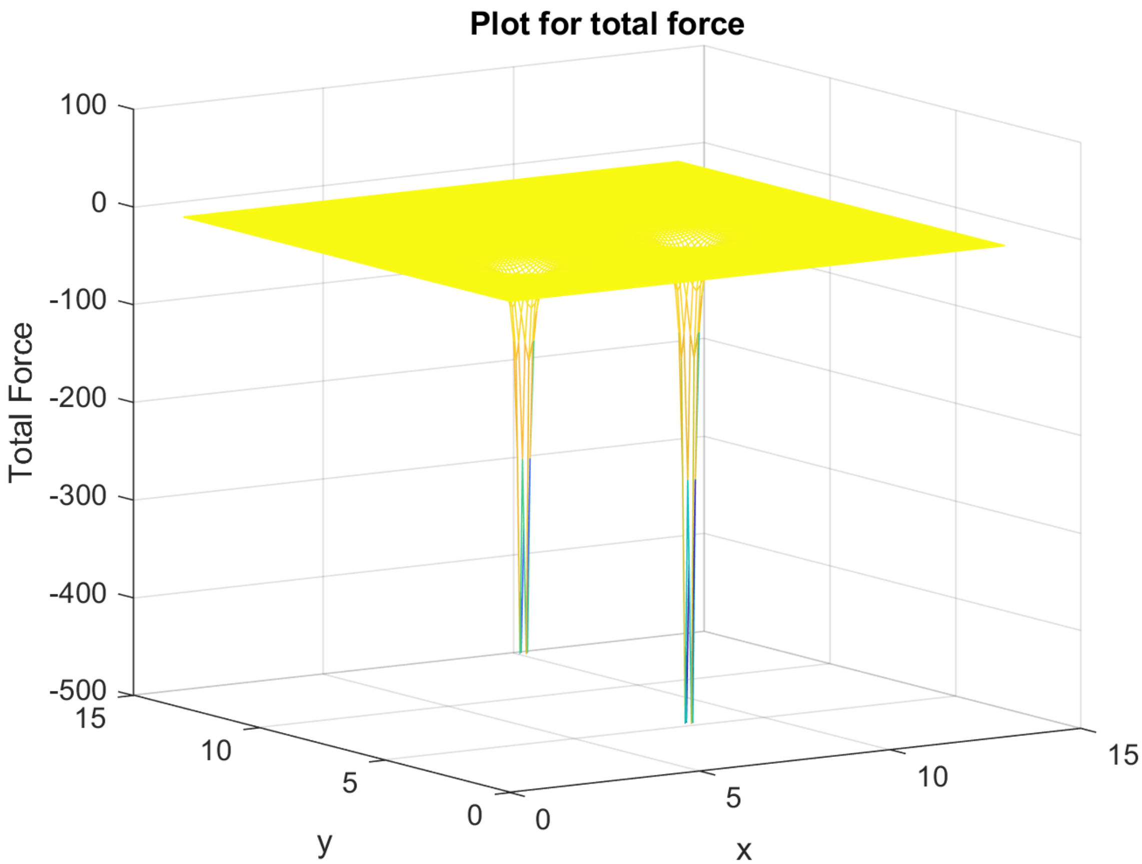
$$F_{total} = -\frac{k_{o1}}{{r_1}^2} - \frac{k_{o2}}{{r_2}^2} + k_g$$

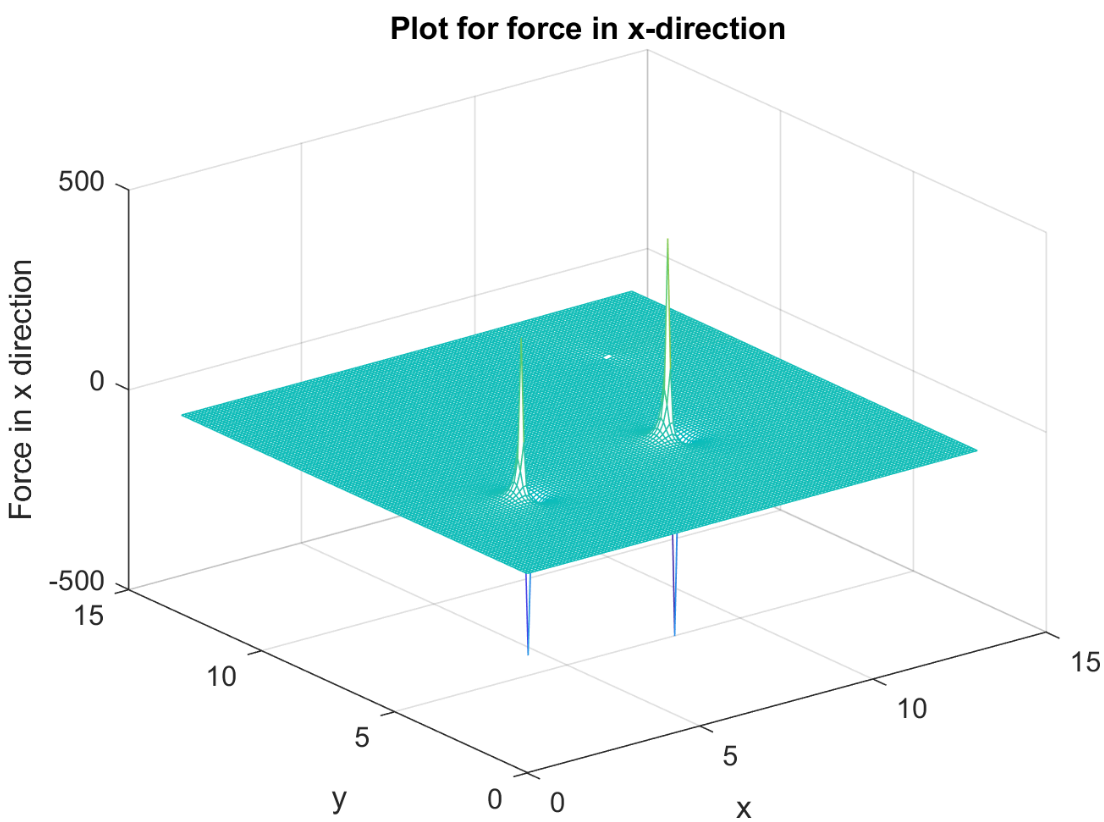
It shows that attractive force is linear whereas the repulsive forces follow the inverse square law.

```
function DefiningForces

%work area

wa=[0:0.1:13;0:0.1:13];

%locations of obstacles and target

T=[10,10];

obs=[3,4;8,5];

n=length(wa(1,:));

%gains

kT=3;

ko=[4,5];


for r=1:n

    for s=1:n

        FT(s,r)=kT;

        Fo1(s,r)=-ko(1,1)/(((wa(1,r)-obs(1,1))^2)+(wa(2,s)-obs(1,2))^2);

        Fo2(s,r)=-ko(1,2)/(((wa(1,r)-obs(2,1))^2)+(wa(2,s)-obs(2,2))^2);

        F=FT+Fo1+Fo2;

    end

end


for t=1:n

    for u=1:n

        FTx(u,t)=kT*(wa(1,t)-T(1,1))/(sqrt(((wa(1,t)-T(1,1))^2)+(wa(2,u)-T(1,2))^2));

        Fo1x(u,t)=-ko(1,1)*(wa(1,t)-obs(1,1))/((((wa(1,t)-obs(1,1))^2)+(wa(2,u)-
obs(1,2))^2)^1.5);

        Fo2x(u,t)=-ko(1,2)*(wa(1,t)-obs(2,1))/((((wa(1,t)-obs(2,1))^2)+(wa(2,u)-
obs(2,2))^2)^1.5);

        Fx=FTx+Fo1x+Fo2x;

    end

end


for a=1:n

    for b=1:n

        FTy(b,a)=kT*(wa(2,b)-T(1,2))/(sqrt(((wa(1,a)-T(1,1))^2)+(wa(2,b)-T(1,2))^2));
```

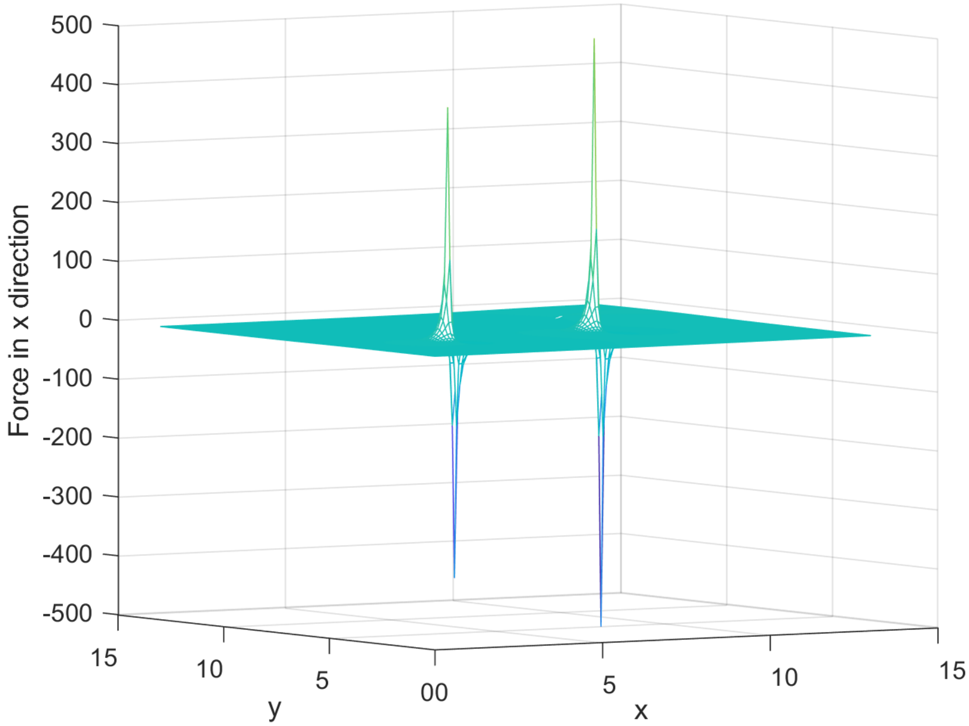
```
        Fo1y(b,a)=-ko(1,1)*(wa(2,b)-obs(1,2))/((((wa(1,a)-obs(1,1))^2)+(wa(2,b)-
obs(1,2))^2)^1.5);

        Fo2y(b,a)=-ko(1,2)*(wa(2,b)-obs(2,2))/((((wa(1,a)-obs(2,1))^2)+(wa(2,b)-
obs(2,2))^2)^1.5);

        Fy=FTy+Fo1y+Fo2y;

    end

end


figure(1)

mesh(wa(1,:),wa(2,:),F)

figure(2)

mesh(wa(1,:),wa(2,:),Fx)

figure(3)

mesh(wa(1,:),wa(2,:),Fy)


end
```



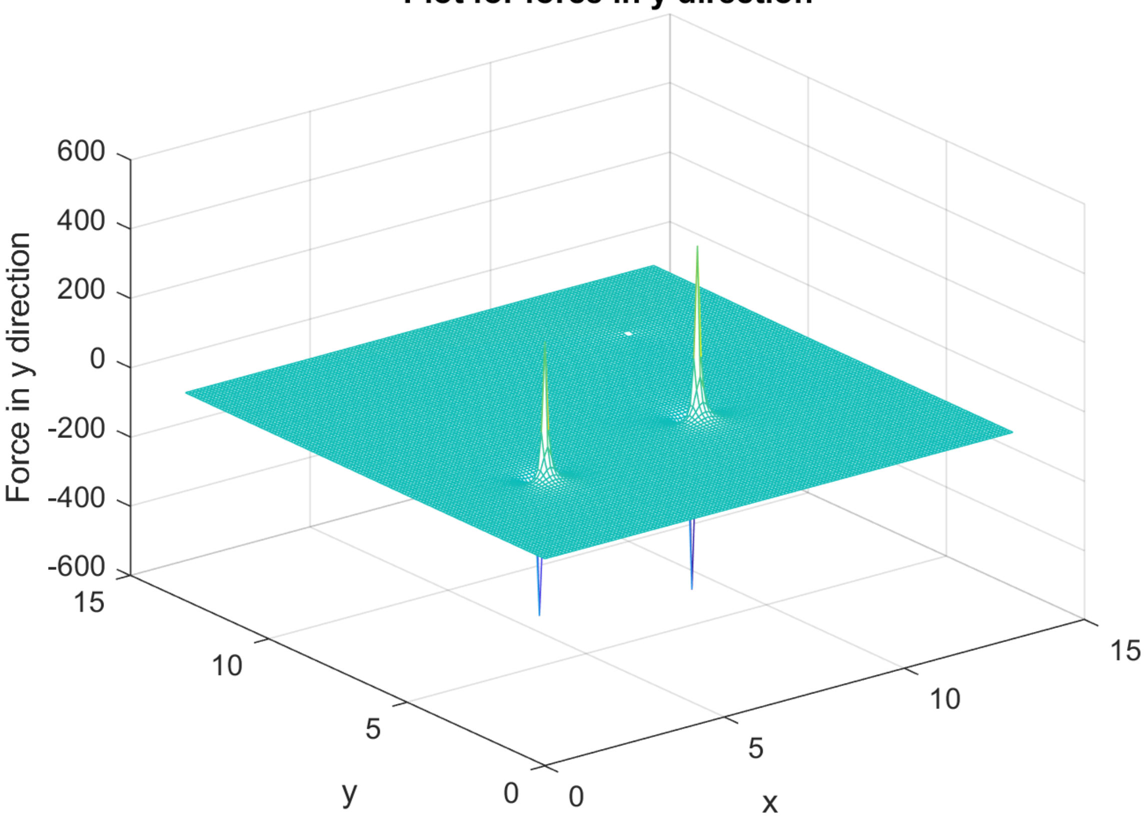

Plot for total force
Total Force
100
0
-100
-200
-300
-400
-500
15
10
5
0
y
0
5
10
15
x

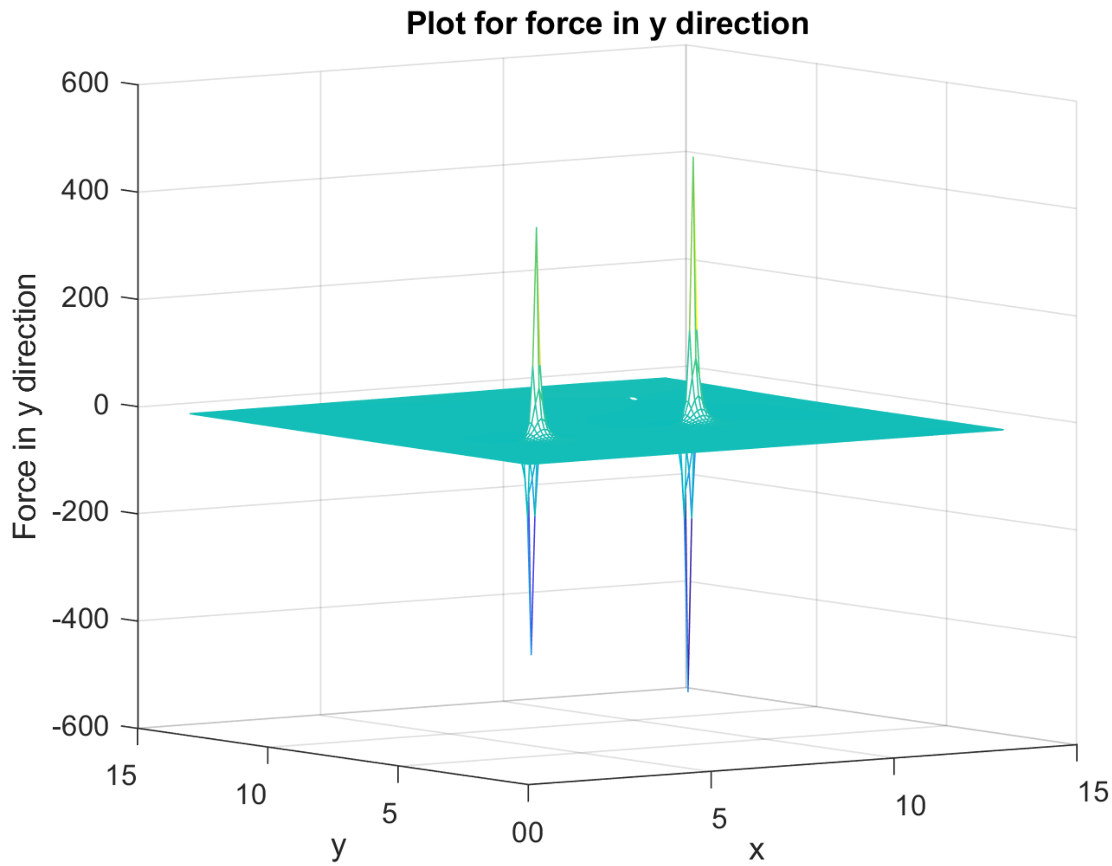

Plot for force in x-direction
Force in x direction
500
0
-500
15
10
5
0
y
0
5
10
15
x

Plot for force in x-direction



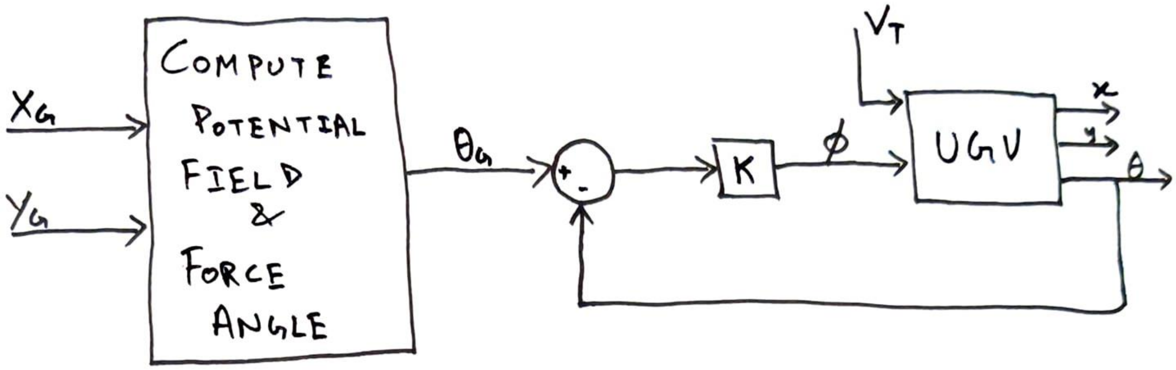
Plot for force in y direction





Plot for force in y direction
Force in y direction
600
400
200
0
-200
-400
-600
15
10
5
00
y
5
10
15
x

b. Control System



CONTROL SYSTEM

$(X_G, Y_G) \longrightarrow$ GOAL POSITION

$\theta_G \longrightarrow$ DESIRED ANGLE

$K \longrightarrow$ STEERING GAIN

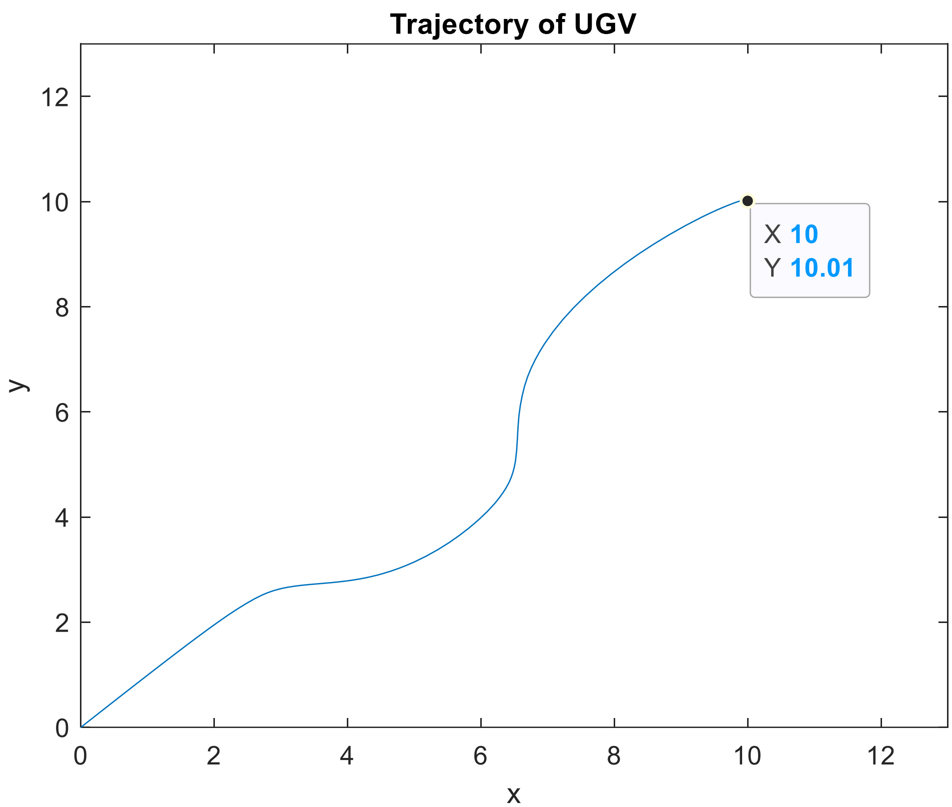
$\phi \longrightarrow$ STEERING ANGLE

$V_T \longrightarrow$ CONSTANT UGV VELOCITY

$(x, y) \longrightarrow$ ACTUAL POSITION OF UGV

$\theta \longrightarrow$ CURRENT ANGLE

c. Simulation

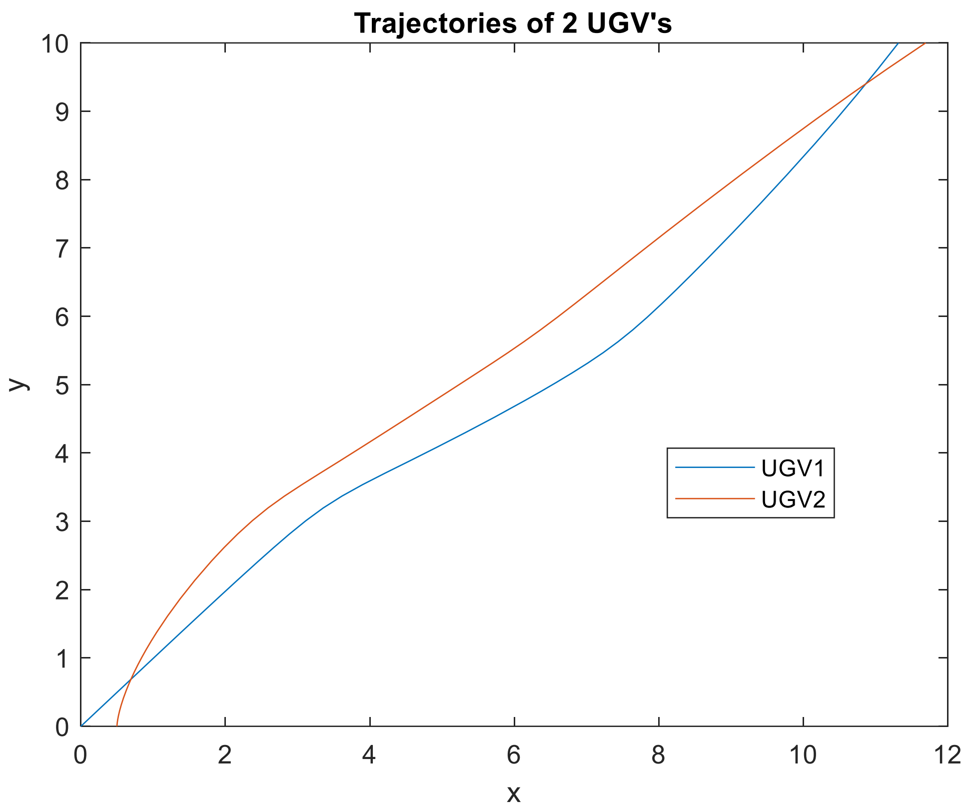


MATLAB Code

```
function FrontSteeringDynamicsImplementation
%work area
wa=[0:0.1:13;0:0.1:13];
%locations of obstacles and target
T=[10,10];
obs=[3,4;8,5];
n=length(wa(1,:));
%gains
kT=3;
ko=[4,5];
options=odeset('events',@StopSim);
Vp=4; kn=3; L=2;
init=[0;0;pi/4]; %initial robot condition
[t,pos]=ode45(@robot,[0 100],init,options);
figure
plot(pos(:,1),pos(:,2))
```

```
xlim([0 13]);
ylim([0 13]);
    function dp=robot(t,pos)
        dp=zeros(3,1);
        x=pos(1);y=pos(2);theta=pos(3);
        FTx=kT*(T(1,1)-x)/(sqrt(((T(1,1)-x)^2)+((T(1,2)-y)^2)));
        Fo1x=-ko(1,1)*(obs(1,1)-x)/((((obs(1,1)-x)^2)+(obs(1,2)-y)^2)^1.5);
        Fo2x=-ko(1,2)*(obs(2,1)-x)/((((obs(2,1)-x)^2)+(obs(2,2)-y)^2)^1.5);
        Fx=FTx+Fo1x+Fo2x;
        FTy=kT*(T(1,2)-y)/(sqrt(((T(1,1)-x)^2)+(T(1,2)-y)^2));
        Fo1y=-ko(1,1)*(obs(1,2)-y)/((((obs(1,1)-x)^2)+(obs(1,2)-y)^2)^1.5);
        Fo2y=-ko(1,2)*(obs(2,2)-y)/((((obs(2,1)-x)^2)+(obs(2,2)-y)^2)^1.5);
        Fy=FTy+Fo1y+Fo2y;
        thetades=atan2(Fy,Fx);
        fi=kn*(thetades-theta);
        dp(1)=Vp*cos(fi)*cos(theta);
        dp(2)=Vp*cos(fi)*sin(theta);
        dp(3)=(Vp/L)*sin(fi);
    end
    function [Val,Ister,Dir]=StopSim(t,pos)
        Val(1)=(pos(1)-T(1,1));
        Ister(1)=1;
        Dir(1)=0;
    end
end
```

Controlling multiple point mass UGV’s to reach goal




```
function hw4q3
%work area
wa=[0:0.1:13;0:0.1:13];
%locations of obstacles and target
T=[10,10];
obs=[3,4;8,5];
n=length(wa(1,:));
%gains
kT=3;
ko=[4,5];
options=odeset('events',@StopSim);
ktr=0.0001; m=1;
init=[0;0;0;0;0.5;0;0;0];
[t,pa]=ode45(@robota,[0 10],init,options);
plot(pa(:,1),pa(:,3),pa(:,5),pa(:,7))
    function dpa=robota(t,pa)
        dpa=zeros(8,1);
```

```
xa=pa(1);ya=pa(3); xb=pa(5); yb=pa(7);

FTax=kT*(T(1,1)-xa)/(sqrt(((T(1,1)-xa)^2)+((T(1,2)-ya)^2)));
Fo1ax=-ko(1,1)*(obs(1,1)-xa)/((((obs(1,1)-xa)^2)+(obs(1,2)-ya)^2)^1.5);
Fo2ax=-ko(1,2)*(obs(2,1)-xa)/((((obs(2,1)-xa)^2)+(obs(2,2)-ya)^2)^1.5);
Frbax=-ktr*(xb-xa)/(((xb-xa)^2+(yb-ya)^2)^1.5);
Fxa=FTax+Fo1ax+Fo2ax+Frbax;

FTay=kT*(T(1,2)-ya)/(sqrt(((T(1,1)-xa)^2)+(T(1,2)-ya)^2));
Fo1ay=-ko(1,1)*(obs(1,2)-ya)/((((obs(1,1)-xa)^2)+(obs(1,2)-ya)^2)^1.5);
Fo2ay=-ko(1,2)*(obs(2,2)-ya)/((((obs(2,1)-xa)^2)+(obs(2,2)-ya)^2)^1.5);
Frbay=-ktr*(yb-ya)/(((xb-xa)^2+(yb-ya)^2)^1.5);
Fya=FTay+Fo1ay+Fo2ay+Frbay;

dpa(1)=pa(2);
dpa(2)=Fxa/m;
dpa(3)=pa(4);
dpa(4)=Fya/m;

FTbx=kT*(T(1,1)-xb)/(sqrt(((T(1,1)-xb)^2)+(T(1,2)-yb)^2));
Fo1bx=-ko(1,1)*(obs(1,1)-xb)/((((obs(1,1)-xb)^2)+(obs(1,2)-yb)^2)^1.5);
Fo2bx=-ko(1,2)*(obs(2,1)-xb)/((((obs(2,1)-xb)^2)+(obs(2,2)-yb)^2)^1.5);
Frabx=-ktr*(xa-xb)/(((xa-xb)^2+(ya-yb)^2)^1.5);
Fxb=FTax+Fo1ax+Fo2ax+Frabx;

FTby=kT*(T(1,2)-ya)/(sqrt(((T(1,1)-xa)^2)+(T(1,2)-ya)^2));
Fo1by=-ko(1,1)*(obs(1,2)-ya)/((((obs(1,1)-xa)^2)+(obs(1,2)-ya)^2)^1.5);
Fo2by=-ko(1,2)*(obs(2,2)-ya)/((((obs(2,1)-xa)^2)+(obs(2,2)-ya)^2)^1.5);
Fraby=-ktr*(ya-yb)/(((xa-xb)^2+(ya-yb)^2)^1.5);
Fyb=FTay+Fo1ay+Fo2ay+Fraby;

dpa(5)=pa(7);
dpa(6)=Fxb/m;
```

```
        dpa(7)=pa(8);
        dpa(8)=Fyb/m;
    end
    function [Val,Ister,Dir]=StopSim(t,pa)
        Val(1)=pa(3)-T(1,1);
        Ister(1)=1;
        Dir(1)=0;
    end
end
```

References

1. Intelligent Control Systems – Dr. Frank L. Lewis (Professor, Electrical Engineering, The University of Texas at Arlington)
2. Chenyuan He (PhD Student, Electrical Engineering, The University of Texas at Arlington)