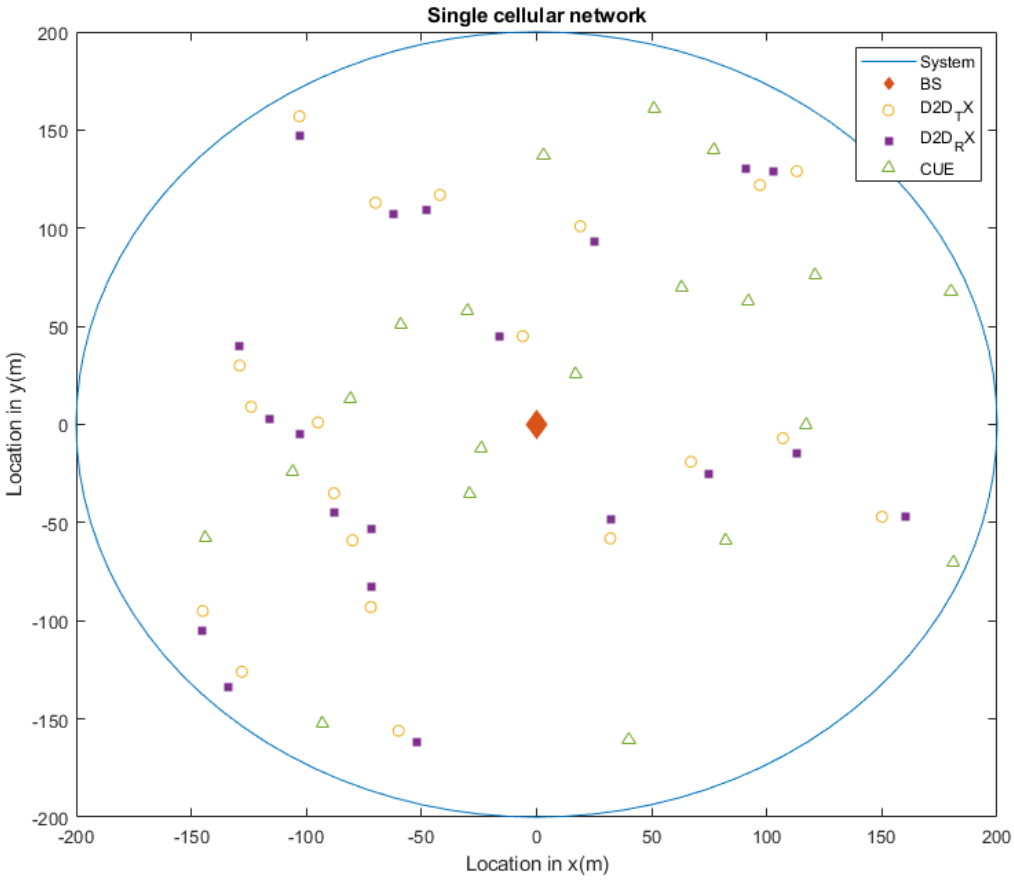
System generation

- Random D2D TX points
- Random CUE points
- Random D2D RX points

This package will be used to generate the system model. So, in this system, the user devices will be distributed uniformly. So we can think of this system as a giant circle where there are a lot of randomly distributed points.



Firstly, we need to think about how to generate randomly distributed and make them try not to collide with each other. The key point of that is the location. For example, if you already have a D2D point at [2 3], it means that you should have some other points at some other locations.

In this function, we take the required number of D2D links CUEs and communication distance as input argument.

Random D2D points

To generate randomly distributed D2D points that won't collide with each other, the key is randperm(), which will generate the random integers within the settled range. And to ensure that there will not be the same D2D points colliding with each other, we will use a boolean logic to check if the generated points are the same as that of the pervious generated points.

Firstly, it will check if the D2D_TX set is empty, if so it will select a random point within the selected range and put it into the empty set. If not, then it will find a randomly point that will not conflict with the existing D2D TX points.

```
for i=1:number_D2D
    x=randperm(400,1)-200;
    y=randperm(400,1)-200;
    check1=x==0;
    check2=y==0;
    check=check1*check2;
    isempty(D2D_TX_coord)
    if isempty(D2D_TX_coord)
        if x^(2)+y^(2)<=200^2 && check==0
           D2D_TX_coord(i,1)=x;
           D2D_TX_coord(i,2)=y;
        else
            while x^(2)+y^(2)>200^2 ||check==1
              x=randperm(400,1)-200;
              y=randperm(400,1)-200;
              check1=x==0;
              check2=y==0;
              check=check1*check2;
            end
            D2D_TX_coord(i,1)=x;
            D2D_TX_coord(i,2)=y;
        end
    elseif isempty(D2D_TX_coord)==0
        find=false;
        while find==false
            if x^(2)+y^(2)<=200^2 && ~any(D2D_TX_coord(:,1)==x)&& check==0
            D2D_TX_coord(i,1)=x;
            D2D_TX_coord(i,2)=y;
            find=true;
            else
             x=randperm(400,1)-200;
             y=randperm(400,1)-200;
             check1=x==0;
             check2=y==0;
             check=check1*check2;
            end
        end
    end
end
```

Random CUE points

This is similar to the method of generating random D2D TX.

```
for j=1:num_CUE
    x_C=randperm(400,1)-200;
    y_C=randperm(400,1)-200;
    check_x_C=x_C==0;
    check_y_C=y_C==0;
```

```
    check_C=check_x_C*check_y_C;
    if isempty(CUE_coord)
        if x_C^(2)+y_C^(2)<=200^2 && check_C==0
           CUE_coord(j,1)=x_C;
           CUE_coord(j,2)=y_C;
        else
            while x_C^(2)+y_C^(2)>200^2 ||check_C==1
              x_C=randperm(400,1)-200;
              y_C=randperm(400,1)-200;
              check_x_C=x_C==0;
              check_y_C=y_C==0;
              check_C=check_x_C*check_y_C;
            end
            CUE_coord(j,1)=x_C;
            CUE_coord(j,2)=y_C;
        end
    else
        find_C=false;
        while find_C==false
            if x_C^(2)+y_C^(2)<=200^2 &&
~any(CUE_coord(:,1)==x_C)&&~any(D2D_TX_coord(:,1)==x_C)&& check_C==0
                CUE_coord(j,1)=x_C;
                CUE_coord(j,2)=y_C;
                find_C=true;
                break;
            else
                x_C=randperm(400,1)-200;
                y_C=randperm(400,1)-200;
                check_x_C=x_C==0;
                check_y_C=y_C==0;
                check_C=check_x_C*check_y_C;
            end
        end
    end
end
```

Random D2D RX points

The location of the D2D RX points is affected by the required communication distance and location of the generated D2D TX points

So the RX() function will be used to do that. It will first generate two random points, then calculate the distance between the generated points, then it will compare the calculated distance to the required distance, if they are the same,then it will break the while the loop, otherwise it will still continue with the loop.

```
function [RX_x,RX_y]=RX(x,y,distance)
a=randperm(400,1)-200;
b=randperm(400,1)-200;
dist=hypot(x-a,y-b);
check1=(a==0);
check2=(b==0);
check=check1+check2;
while dist~=distance || a>200||a<-200 || b>200 ||b<-200||(check==1)
```

```
    a=randperm(400,1)-200;
    b=randperm(400,1)-200;
    dist=hypot(x-a,y-b);
    check1=(a==0);
    check2=(b==0);
    check=check1+check2;
end
RX_x=a;
RX_y=b;
end
```