

MATLAB CODING

Coding for V2V Network Setup in the Dynamic Spectrum Allocation

```

1  %*****DEFINE THE INITIAL VALUE*****
2
3  CUE= 20;           % Number of users in inner region
4  VUE= 300;          % Number of users in outer region
5  R=28;
6  timestep = 1;
7  timeend = 200;     %Time for simulation
8  ht = 120;          %Call Duration/HoltTime
9
10 %***** GENERATING THE HEXAGON CELL*****
11 x1= zeros(CUE, 6);
12 y1=zeros(VUE,6);
13
14 t=linspace(0,2*pi,7);
15 bhx=0+R*cos(t);    %Generating the outer hexagons
16 ahx=0+R*sin(t);    %of the model.
17 plot(ahx,bhx, 'b');
18 hold on
19 plot(0,0,'o');      %Plotting the base station at the center
20 hold on
21
22
23 %*** PLOTTING THE INTERFERENCE DEVICE***
24 i=0;
25 ranC_x = 0 + R - rand(1, 3*CUE)*2*R; %Generating interference random points
26 ranC_y = 0 + R - rand(1, 3*CUE)*2*R;
27 IN = inpolygon(ranC_x, ranC_y, ahx, bhx); %ensure the point are within the hexagon
28 ranC_x = ranC_x(IN); %eliminate the nodes outside the hexagon
29 ranC_y = ranC_y(IN);
30 idx = randperm(length(ranC_x)); %placing the interferences point at random location
31 ranC_x = ranC_x(idx(1:CUE));
32 ranC_y = ranC_y(idx(1:CUE));
33
34 for j = 1:CUE %Rotates from 1 to number of user in the hexagon
35     rand('seed',1);
36     x1(0 + j, 1) = ranC_x(j);
37     x1(0 + j, 2) = ranC_y(j);
38 end
39
40 plot(x1(:,1), x1(:,2), 'k.','marker','x'); %plot the interfering devices in the cell
41
42 %*** PLOTTING VUE DEVICES ***
43 k=0;
44 ranD_x = 0 + R - rand(1, 3*VUE)*2*R; %Generating random points for V2V receiver
45 ranD_y = 0 + R - rand(1, 3*VUE)*2*R;
46 IN = inpolygon(ranD_x, ranD_y, ahx, bhx);
47 ranD_x = ranD_x(IN);
48 ranD_y = ranD_y(IN);
49 idx = randperm(length(ranD_x));
50 ranD_x = ranD_x(idx(1:VUE));
51
52 for j = 1:VUE %Rotates from 1 to number of V user in the hexagon
53     rand('seed',1);
54     y1(0 + j, 1) = ranD_x(j);
55     y1(0 + j, 2) = ranD_y(j);
56 end
57
58 plot(y1(:,1), y1(:,2), 'k.','marker','.'); %plotting the random point for the V2V receiver
59
60 %*****PLOTTING V2V TRANSMITTER *****
61 x1(50,1) = 15;
62 x1(50,2) = 5;
63 plot(x1(50,1), x1(50,2), 'k.','marker','>');
64
65 legend ('Legend','Base station','Interfering devices','V2V receiver at different distance', 'V2V Transmitter')

```

Simulation setup files

```
distanceTxRx.m  ✕  +
1  function [ vtx_drx ] = distanceTxRx( Ux1,Ux2,Uy1,Uy2 )
2
3      % to calculate the distance between vehicles
4  -   vtx_drx = 0;
5  -   vtx_drx = sqrt(((Ux2-Ux1).^2) + ((Uy2-Uy1).^2));
6  -   end
1  function [ PL_VUE_CUE ] = pathloss_VUE_CUE( d,f )
2
3  -   PL_VUE_CUE=20*log10(d/1000) + 20*log10(f)+32.45;
4
5  -   end
1  function [ PL_VUE_CUE_4 ] = pathloss_VUE_CUE_4( d,f )
2
3      %%% path loss for less than 15m
4  -   PL_VUE_CUE_4=12*log10(d/1000) + 12*log10(f)+19.45;
5  -   end
1  function [ PL_VUE_CUE_4more ] = pathloss_VUE_CUE_4more( d,f )
2
3      %%% path loss for more than 15m
4  -   PL_VUE_CUE_4more=34*log10(d/1000) + 34*log10(f)+64.9;
5  -   end
6
```

Coding for Throughput of the Proposed Dynamic Spectrum

```
1 - throughputexperiment3=zeros(VUE,5);
2 - throughputexperiment3(:,1)= Throughput2(:,1); %throughput for unlicensed spectrum
3 - throughputexperiment3(:,5)= distanceTxRx2(:,1); %calling distance function
4
5 - throughputexperiment2=zeros(VUE,5);
6 - throughputexperiment2(:,1)= Throughput3(:,1); %throughput for licensed spectrum
7 - throughputexperiment2(:,5)= distanceTxRx2(:,1); %calling distance function
8
9 - throughputexperiment=zeros(VUE,5); %throughput for dynamic spectrum allocation
10 - throughputexperiment(:,1)= Throughputexp1(:,1); %throughput when interfering user is 5
11 - throughputexperiment(:,2)= Throughputexp2(:,1); %throughput when interfering user is 10
12 - throughputexperiment(:,3)= Throughputexp3(:,1); %throughput when interfering user is 15
13 - throughputexperiment(:,4)= Throughputexp4(:,1); %throughput when interfering user is 20
14 - throughputexperiment(:,5)= distanceTxRx1(:,1); %calling distance function
15
16 - x= [34,47,23,5,14,9,3,1,36];
17 - figure
18 - plot(throughputexperiment(x,5),throughputexperiment(x,1),'-x'); %plotting throughput for dynamic spectrum
19 - hold on
20 - plot(throughputexperiment3(x,5),throughputexperiment3(x,1),'->');%plotting throughput for unlicensed spectrum
21 - hold on
22 - plot(throughputexperiment(x,5),throughputexperiment2(x,1),'-*'); %plotting throughput for licensed spectrum
23 - hold on
24
25 - legend('Proposed Dynamic Spectrum','Unlicensed Spectrum','Licensed Spectrum');
26 - xlabel('Distance between V2V Transmitter and Receiver');
27 - ylabel('Throughput (bit/s)')
28 - hold off
```

Coding for Path Loss vs Distance

```

1  clc;
2  clear all;
3  close all;
4
5  %% PATH LOSS FOR FIXED LICENSED SPECTRUM %%
6
7  f2=2400; %licensed frequency
8  b = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20];
9  Pathloss_dBm1 = zeros(20,0);
10
11  for p=1:1:21;
12      d = p - 1;
13      if d <=15 %if distance between V2V is less than 15m, use licensed parameter
14          PL_VUE_VUE1=((12*log10(d/1000)) + (12*log10(f2)))+19.47; %calculate the path loss
15          Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000); %path loss in Watts
16      end
17      if d > 15 %if distance between V2V is more than 15, use licensed parameter too
18          PL_VUE_VUE1=20*log10(d/1000) + 20*log10(f2)+32.45; %calculate the path loss
19          Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000); %path loss in Watts
20      end
21  end
22
23  Pathloss_VTX_VRX_dBm1=10*log10(Pathloss_VTX_VRX_W1); % Conversion of watt to dBm
24  Pathloss_dBm1 = [Pathloss_dBm1 Pathloss_VTX_VRX_dBm1]; % dBm append array
25
26  end
27
28  % checking of NaN or Inf value to 0
29  Pathloss_dBm1(isinf(Pathloss_dBm1)|isnan(Pathloss_dBm1)) = 0;
30
31  %% PATH LOSS FOR FIXED UNLICENSED SPECTRUM %%
32
33  f1= 60000; %unlicensed frequency
34  b = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20];
35  Pathloss_dBm = zeros(20,0);
36
37  for p=1:1:21;
38      d = p - 1;
39      if d <=15 %if distance between V2V is less than 15m, use licensed parameter
40          PL_VUE_VUE1=((12*log10(d/1000)) + (12*log10(f1)))+19.47; %calculate the path loss
41          Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000); %path loss in Watts
42      end
43
44      if d > 15 %if distance between V2V is more than 15, use licensed parameter too
45          PL_VUE_VUE1=20*log10(d/1000) + 20*log10(f1)+32.45; %calculate the path loss
46          Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000); %path loss in Watts
47      end
48  end
49
50  Pathloss_VTX_VRX_dBm1=10*log10(Pathloss_VTX_VRX_W1); % Conversion of watt to dBm
51  Pathloss_dBm = [Pathloss_dBm Pathloss_VTX_VRX_dBm1]; % dBm append array
52
53  end
54
55  % checking of NaN or Inf value to 0
56  Pathloss_dBm(isinf(Pathloss_dBm)|isnan(Pathloss_dBm)) = 0;
57
58  %% PATH LOSS FOR PROPOSED DYNAMIC SPECTRUM %%
59
60  f1= 60000; %unlicensed frequency
61  f2=2400; %licensed frequency
62  b = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20];
63  Pathloss_dBm3 = zeros(20,0);
64
65  for p=1:1:21;
66      d = p - 1;
67      if d <=15 %if distance between V2V is less than 15m, use unlicensed parameter
68          PL_VUE_VUE1=((12*log10(d/1000)) + (12*log10(f1)))+19.47; %calculate the path loss
69          Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000); %path loss in Watts
70      end
71
72      if d > 15 %if distance between V2V is more than 15m, use licensed parameter
73          PL_VUE_VUE1=20*log10(d/1000) + 20*log10(f2)+32.45; %calculate the path loss
74          Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000); %path loss in Watts
75      end
76  end

```

```

74 -     Pathloss_VTX_VRX_dBm1=10*log10(Pathloss_VTX_VRX_W1);    %Conversion of watt to dBm
75 -     Pathloss_dBm3 = [Pathloss_dBm3 Pathloss_VTX_VRX_dBm1];    %dBm append array
76 - end
77
78 %checking of NaN or Inf value to 0
79 - Pathloss_dBm3(isinf(Pathloss_dBm3)|isnan(Pathloss_dBm3)) = 0;
80
81 %% Plotting Path loss against Distance %%
82 - figure
83 - plot(b, Pathloss_dBm1, '-x'); %pathloss for fixed licensed spectrum
84 - hold on;
85 - plot(b, Pathloss_dBm, '-*'); %pathloss for fixed unlicensed spectrum
86 - hold on;
87 - plot(b, Pathloss_dBm3, '-o'); %pathloss for proposed dynamic spectrum
88 - legend('Licensed Spectrum','Unlicensed Spectrum','Proposed Dynamic Spectrum');
89 - xlabel('Distance between V2V Transmitter and Receiver');
90 - ylabel('Pathloss (dBm)')
91 - hold on;

```