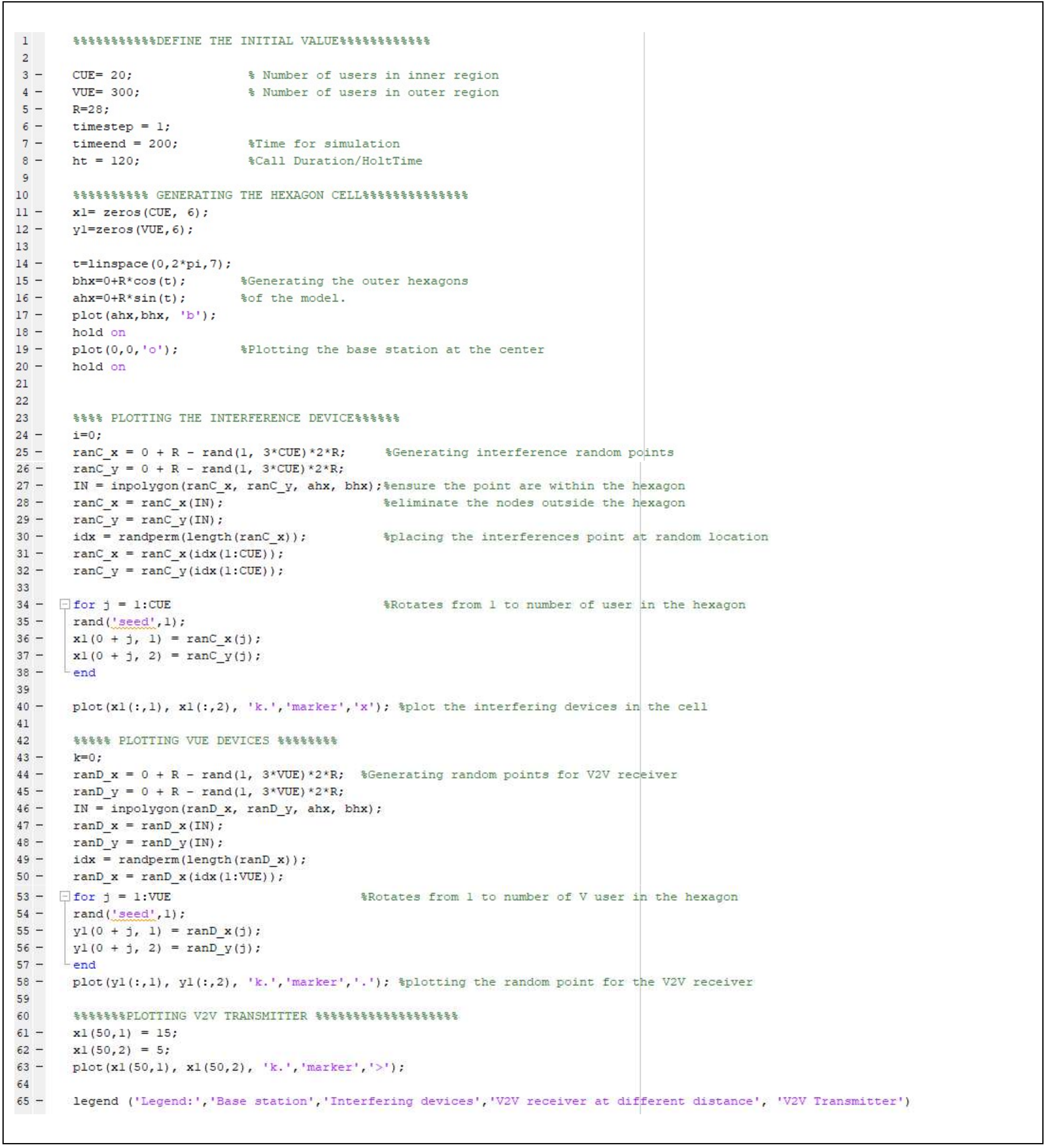
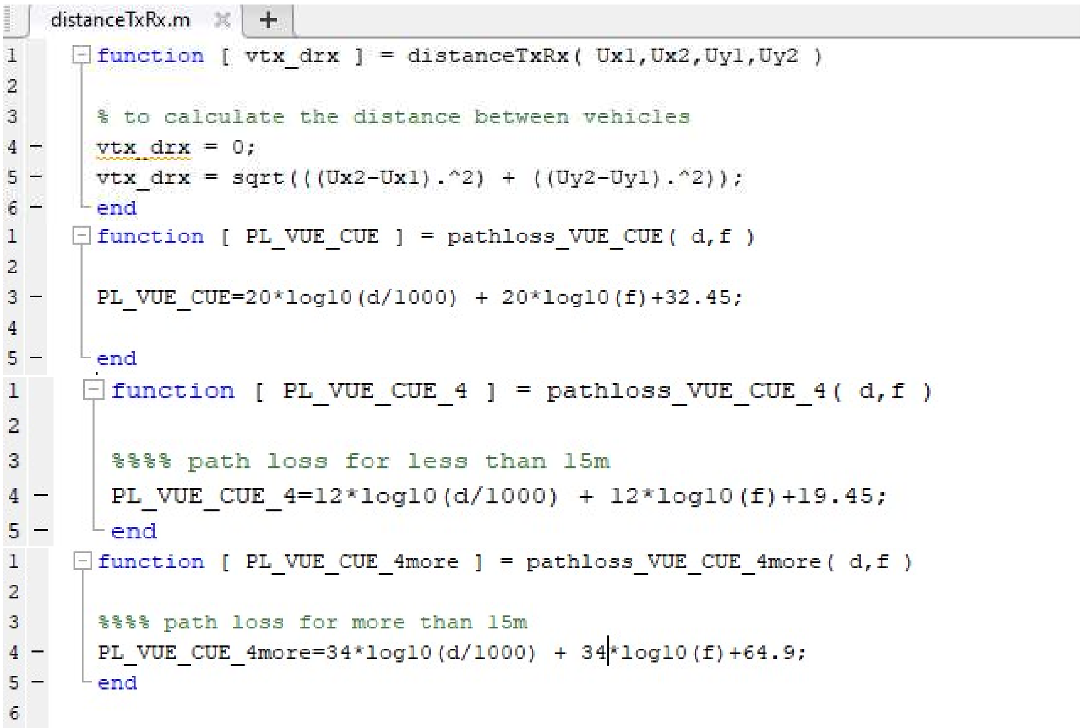
MATLAB CODING

Coding for V2V Network Setup in the Dynamic Spectrum Allocation



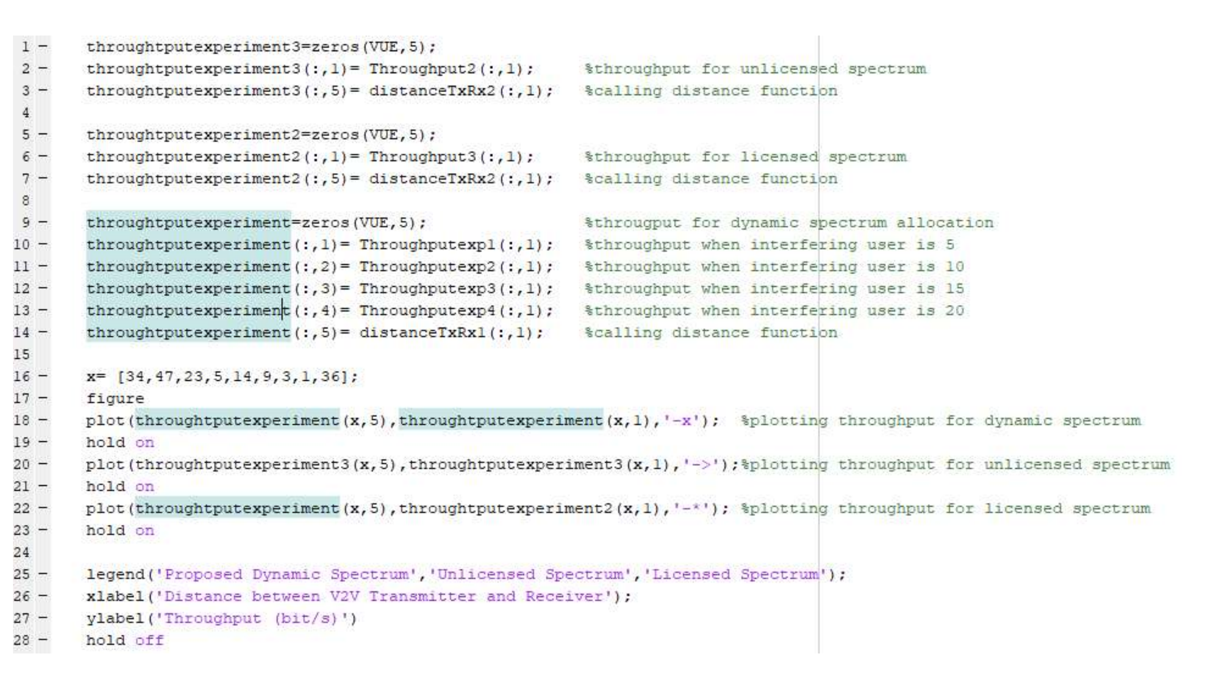
```
%%%%%%%%%%%%DEFINE THE INITIAL VALUE%%%%%%%%%%%%%

CUE= 20;                % Number of users in inner region
VUE= 300;               % Number of users in outer region
R=28;
timestep = 1;
timeend = 200;          %Time for simulation
ht = 120;               %Call Duration/HoltTime

%%%%%%%%%%% GENERATING THE HEXAGON CELL%%%%%%%%%%%%%%%
x1= zeros(CUE, 6);
y1=zeros(VUE,6);

t=linspace(0,2*pi,7);
bhx=0+R*cos(t);        %Generating the outer hexagons
ahx=0+R*sin(t);        %of the model.
plot(ahx,bhx, 'b');
hold on
plot(0,0,'o');         %Plotting the base station at the center
hold on


%%%% PLOTTING THE INTERFERENCE DEVICE%%%%%%
i=0;
ranC_x = 0 + R - rand(1, 3*CUE)*2*R;     %Generating interference random points
ranC_y = 0 + R - rand(1, 3*CUE)*2*R;
IN = inpolygon(ranC_x, ranC_y, ahx, bhx);%ensure the point are within the hexagon
ranC_x = ranC_x(IN);                     %eliminate the nodes outside the hexagon
ranC_y = ranC_y(IN);
idx = randperm(length(ranC_x));          %placing the interferences point at random location
ranC_x = ranC_x(idx(1:CUE));
ranC_y = ranC_y(idx(1:CUE));

for j = 1:CUE                            %Rotates from 1 to number of user in the hexagon
rand('seed',1);
x1(0 + j, 1) = ranC_x(j);
x1(0 + j, 2) = ranC_y(j);
end

plot(x1(:,1), x1(:,2), 'k.','marker','x'); %plot the interfering devices in the cell

%%%%% PLOTTING VUE DEVICES %%%%%%%%
k=0;
ranD_x = 0 + R - rand(1, 3*VUE)*2*R;  %Generating random points for V2V receiver
ranD_y = 0 + R - rand(1, 3*VUE)*2*R;
IN = inpolygon(ranD_x, ranD_y, ahx, bhx);
ranD_x = ranD_x(IN);
ranD_y = ranD_y(IN);
idx = randperm(length(ranD_x));
ranD_x = ranD_x(idx(1:VUE));
for j = 1:VUE                         %Rotates from 1 to number of V user in the hexagon
rand('seed',1);
y1(0 + j, 1) = ranD_x(j);
y1(0 + j, 2) = ranD_y(j);
end
plot(y1(:,1), y1(:,2), 'k.','marker','.'); %plotting the random point for the V2V receiver

%%%%%%%PLOTTING V2V TRANSMITTER %%%%%%%%%%%%%%%%%%%
x1(50,1) = 15;
x1(50,2) = 5;
plot(x1(50,1), x1(50,2), 'k.','marker','>');

legend ('Legend:','Base station','Interfering devices','V2V receiver at different distance', 'V2V Transmitter')
```

Simulation setup files



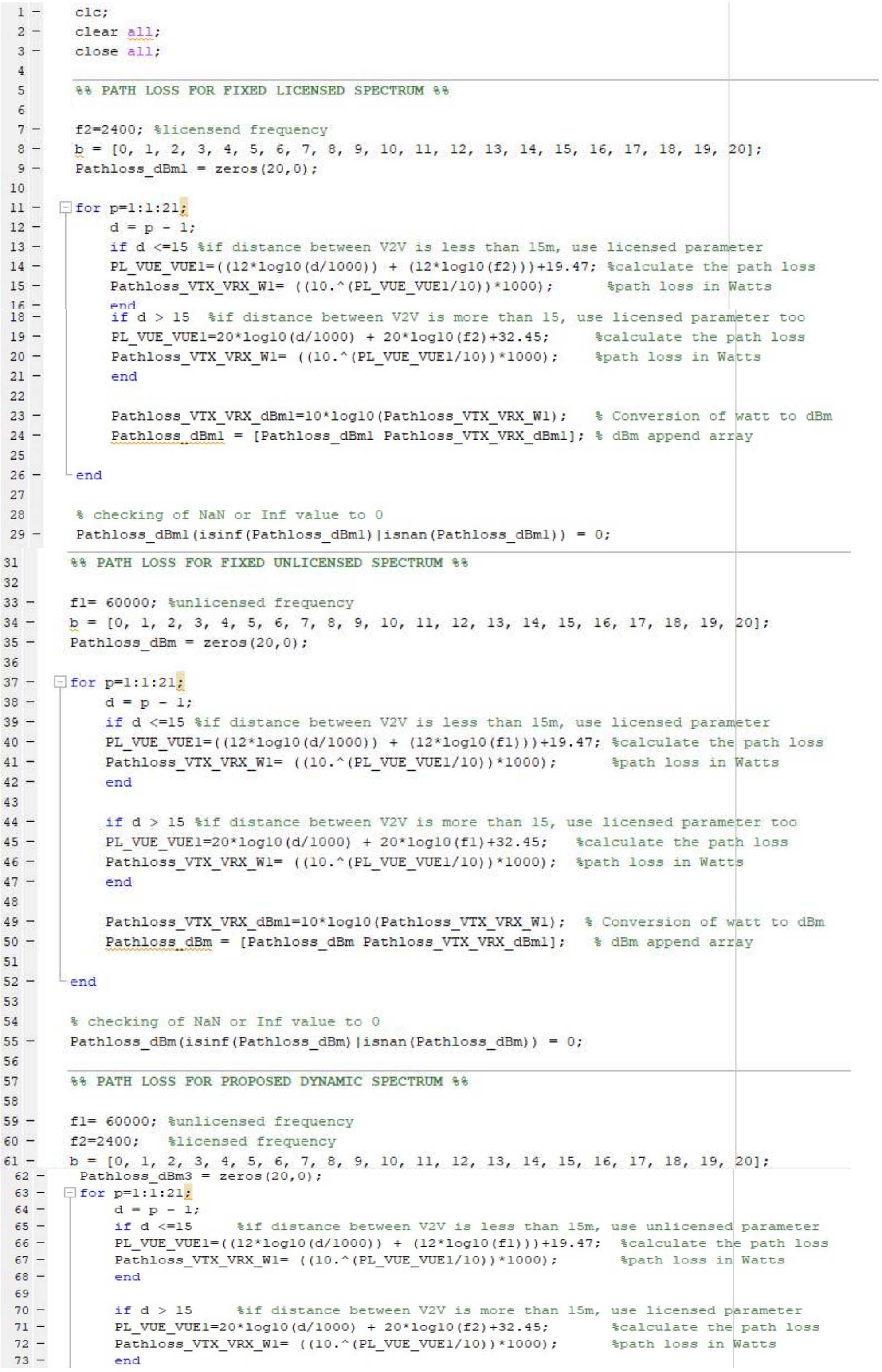
```
function [ vtx_drx ] = distanceTxRx( Ux1,Ux2,Uy1,Uy2 )

% to calculate the distance between vehicles
vtx_drx = 0;
vtx_drx = sqrt(((Ux2-Ux1).^2) + ((Uy2-Uy1).^2));
end
function [ PL_VUE_CUE ] = pathloss_VUE_CUE( d,f )

PL_VUE_CUE=20*log10(d/1000) + 20*log10(f)+32.45;

end
function [ PL_VUE_CUE_4 ] = pathloss_VUE_CUE_4( d,f )

%%%% path loss for less than 15m
PL_VUE_CUE_4=12*log10(d/1000) + 12*log10(f)+19.45;
end
function [ PL_VUE_CUE_4more ] = pathloss_VUE_CUE_4more( d,f )

%%%% path loss for more than 15m
PL_VUE_CUE_4more=34*log10(d/1000) + 34*log10(f)+64.9;
end

```

Coding for Throughput of the Proposed Dynamic Spectrum

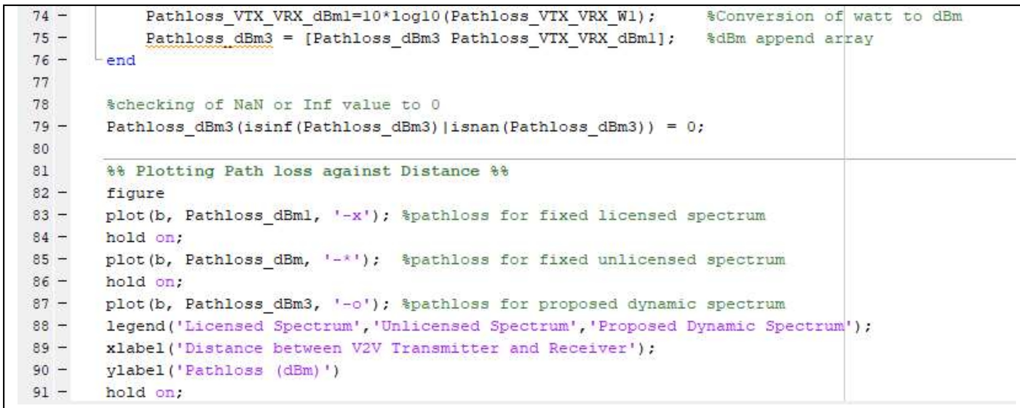


```
throughtputexperiment3=zeros(VUE,5);
throughtputexperiment3(:,1)= Throughput2(:,1);      %throughput for unlicensed spectrum
throughtputexperiment3(:,5)= distanceTxRx2(:,1);    %calling distance function

throughtputexperiment2=zeros(VUE,5);
throughtputexperiment2(:,1)= Throughput3(:,1);      %throughput for licensed spectrum
throughtputexperiment2(:,5)= distanceTxRx2(:,1);    %calling distance function

throughtputexperiment=zeros(VUE,5);                 %througput for dynamic spectrum allocation
throughtputexperiment(:,1)= Throughputexp1(:,1);    %throughput when interfering user is 5
throughtputexperiment(:,2)= Throughputexp2(:,1);    %throughput when interfering user is 10
throughtputexperiment(:,3)= Throughputexp3(:,1);    %throughput when interfering user is 15
throughtputexperiment(:,4)= Throughputexp4(:,1);    %throughput when interfering user is 20
throughtputexperiment(:,5)= distanceTxRx1(:,1);     %calling distance function

x= [34,47,23,5,14,9,3,1,36];
figure
plot(throughtputexperiment(x,5),throughtputexperiment(x,1),'-x');  %plotting throughput for dynamic spectrum
hold on
plot(throughtputexperiment3(x,5),throughtputexperiment3(x,1),'->');%plotting throughput for unlicensed spectrum
hold on
plot(throughtputexperiment(x,5),throughtputexperiment2(x,1),'-*'); %plotting throughput for licensed spectrum
hold on

legend('Proposed Dynamic Spectrum','Unlicensed Spectrum','Licensed Spectrum');
xlabel('Distance between V2V Transmitter and Receiver');
ylabel('Throughput (bit/s)')
hold off
```

Coding for Path Loss vs Distance



```
clc;
clear all;
close all;

%% PATH LOSS FOR FIXED LICENSED SPECTRUM %%

f2=2400; %licensend frequency
b = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20];
Pathloss_dBm1 = zeros(20,0);

for p=1:1:21;
    d = p - 1;
    if d <=15 %if distance between V2V is less than 15m, use licensed parameter
    PL_VUE_VUE1=((12*log10(d/1000)) + (12*log10(f2)))+19.47; %calculate the path loss
    Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000);      %path loss in Watts
    end
    if d > 15  %if distance between V2V is more than 15, use licensed parameter too
    PL_VUE_VUE1=20*log10(d/1000) + 20*log10(f2)+32.45;      %calculate the path loss
    Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000);     %path loss in Watts
    end

    Pathloss_VTX_VRX_dBm1=10*log10(Pathloss_VTX_VRX_W1);    % Conversion of watt to dBm
    Pathloss_dBm1 = [Pathloss_dBm1 Pathloss_VTX_VRX_dBm1]; % dBm append array

end

% checking of NaN or Inf value to 0
Pathloss_dBm1(isinf(Pathloss_dBm1)|isnan(Pathloss_dBm1)) = 0;
%% PATH LOSS FOR FIXED UNLICENSED SPECTRUM %%

f1= 60000; %unlicensed frequency
b = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20];
Pathloss_dBm = zeros(20,0);

for p=1:1:21;
    d = p - 1;
    if d <=15 %if distance between V2V is less than 15m, use licensed parameter
    PL_VUE_VUE1=((12*log10(d/1000)) + (12*log10(f1)))+19.47; %calculate the path loss
    Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000);      %path loss in Watts
    end

    if d > 15 %if distance between V2V is more than 15, use licensed parameter too
    PL_VUE_VUE1=20*log10(d/1000) + 20*log10(f1)+32.45;   %calculate the path loss
    Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000);  %path loss in Watts
    end

    Pathloss_VTX_VRX_dBm1=10*log10(Pathloss_VTX_VRX_W1);  % Conversion of watt to dBm
    Pathloss_dBm = [Pathloss_dBm Pathloss_VTX_VRX_dBm1];    % dBm append array

end

% checking of NaN or Inf value to 0
Pathloss_dBm(isinf(Pathloss_dBm)|isnan(Pathloss_dBm)) = 0;

%% PATH LOSS FOR PROPOSED DYNAMIC SPECTRUM %%

f1= 60000; %unlicensed frequency
f2=2400;   %licensed frequency
b = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20];
 Pathloss_dBm3 = zeros(20,0);
 for p=1:1:21;
     d = p - 1;
     if d <=15     %if distance between V2V is less than 15m, use unlicensed parameter
     PL_VUE_VUE1=((12*log10(d/1000)) + (12*log10(f1)))+19.47;  %calculate the path loss
     Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000);       %path loss in Watts
     end

     if d > 15     %if distance between V2V is more than 15m, use licensed parameter
     PL_VUE_VUE1=20*log10(d/1000) + 20*log10(f2)+32.45;        %calculate the path loss
     Pathloss_VTX_VRX_W1= ((10.^(PL_VUE_VUE1/10))*1000);       %path loss in Watts
     end
```



```
    Pathloss_VTX_VRX_dBm1=10*log10(Pathloss_VTX_VRX_W1);     %Conversion of watt to dBm
    Pathloss_dBm3 = [Pathloss_dBm3 Pathloss_VTX_VRX_dBm1];   %dBm append array
end

%checking of NaN or Inf value to 0
Pathloss_dBm3(isinf(Pathloss_dBm3)|isnan(Pathloss_dBm3)) = 0;

%% Plotting Path loss against Distance %%
figure
plot(b, Pathloss_dBm1, '-x'); %pathloss for fixed licensed spectrum
hold on;
plot(b, Pathloss_dBm, '-*');  %pathloss for fixed unlicensed spectrum
hold on;
plot(b, Pathloss_dBm3, '-o'); %pathloss for proposed dynamic spectrum
legend('Licensed Spectrum','Unlicensed Spectrum','Proposed Dynamic Spectrum');
xlabel('Distance between V2V Transmitter and Receiver');
ylabel('Pathloss (dBm)')
hold on;
```