```
% clear all
clearvars
clear
clc
clearvars -GLOBAL
close all
format shorte

global C

C.q_0 = 1.60217653e-19;             % electron charge
C.hb = 1.054571596e-34;             % Dirac constant
C.h = C.hb * 2 * pi;                % Planck constant
C.m_0 = 9.10938215e-31;             % electron mass
C.kb = 1.3806504e-23;               % Boltzmann constant
C.eps_0 = 8.854187817e-12;          % vacuum permittivity
C.mu_0 = 1.2566370614e-6;           % vacuum permeability
C.c = 299792458;                    % speed of light
C.g = 9.80665;                      %metres (32.1740 ft) per s²
C.meff = C.m_0 *0.26;
C.am = 1.66053892e-27;

TotalElectrons = 15000;


Tau = 0.2e-12;
length = 200e-9;
height = 100e-9;
time = 500;
dt = 15e-15;
OGTemp = 300;

%Create Vectors Sized Based on Electrons
Px= zeros(1,TotalElectrons);
NewPx= zeros(1,TotalElectrons);
Py= zeros(1,TotalElectrons);
NewPy= zeros(1,TotalElectrons);
Vx= zeros(1,TotalElectrons);
Vy= zeros(1,TotalElectrons);
Scat = zeros(1,TotalElectrons);
Differ = zeros(1,TotalElectrons);
PScat = (1-exp(-dt/Tau));
histArray=zeros(2, TotalElectrons);
TotalTemp = 0;
AverageTemp = 0;
counter = 1;
TempArray = zeros(20,10);


%Calculate the initial thermal voltage
Vth = sqrt(2*(C.kb*300)/(C.m_0*0.26));
lambda = Vth*Tau;
```

```
%Output the values for initial thermal velocity and MFP.
fprintf('The initial thermal velocity is equal to %d\n',Vth)
fprintf('The mean free path is equal to %d\n',lambda)

%Name the figure window
figure('Name','Electron Paths')

%Calculate the random starting position for every electron
Px = rand(1, TotalElectrons)*200e-9;       %Generate the random x
 location
Py = rand(1, TotalElectrons)*100e-9;       %generate the random y
 location

%Check if electron generated in a non-valid location
xlow = Px>.75E-7;
xlarge = Px<1.25E-7;
ylow = Py <.3E-7;
ytop = Py > .7E-7;

%Generate vector of x components that occupy suspect non-valid areas
xlocation = bitand(xlow,xlarge);

%Generate vector of x and y components that occupy non-valid areas
bottom = bitand(xlocation,ylow);
top = bitand(xlocation,ytop);
InBox = bitor(bottom, top);

%Generate a new random location for electrons that appear in the non-
valid
%areas
while any(InBox ==1)
    Px(InBox) = rand(1)*200e-9;       %Generate the random x location
    Py(InBox)= rand(1)*100e-9;       %generate the random y location

    %Check if electron generated in a non-valid location
    xlow = Px>.75E-7;
    xlarge = Px<1.25E-7;
    ylow = Py <.3E-7;
    ytop = Py > .7E-7;

    %Generate vector of x components that occupy suspect non-valid
 areas
    xlocation = bitand(xlow,xlarge);

  %Generate vector of x and y components that still occupy  non-valid
 areas
    bottom = bitand(xlocation,ylow);
    top = bitand(xlocation,ytop);
    InBox = bitor(bottom, top);

end
```

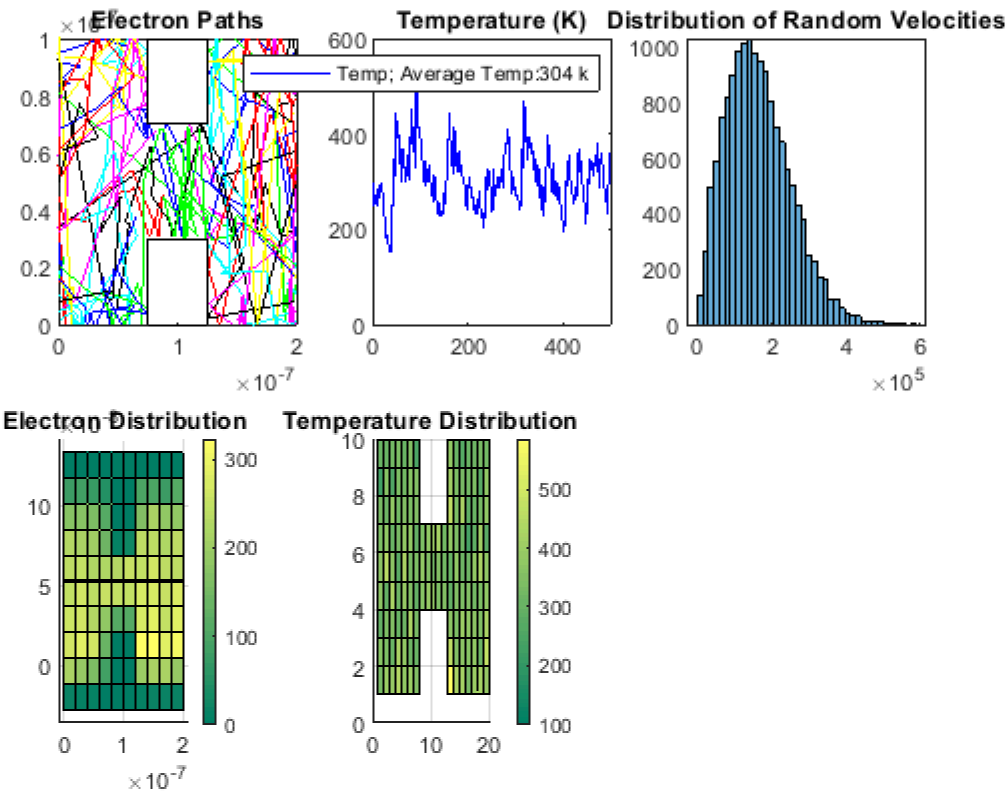
```
%Pick a random velocity in X and Y from the distribution
Vx = (sqrt(C.kb*300/(C.m_0*0.26))*randn(1,TotalElectrons)); %Generate
  x
Vy = (sqrt(C.kb*300/(C.m_0*0.26))*randn(1,TotalElectrons)); %Generate
%Calculate the Voltage
V2 = (Vx.^2)+(Vy.^2);
V = sqrt(V2);


subplot(2,3,3)
histogram(V,40);                        %Plot histogram of velocity
title ('Distribution of Random Velocities');
Vm = mean(V2);                          %Find the average velocity of
 electrons
TCalc = ((Vm)*(C.m_0*0.26))/2/C.kb;     %Calculate the temperature

subplot (2,3,1)
title('Electron Paths');
xlim([0 200e-9]);
ylim ([0 100e-9]);
for loops=1:time

        subplot(2,3,4)
     histArray(1,:) = Px(:);
     histArray(2,:) = Py(:);
    hist3(transpose(histArray), 'CdataMode','auto');
    colormap ('summer');
    colorbar
    view (2)
    title ('Electron Distribution');

    hold off
    %Calculate random probability of scattering
    Scat = rand(1,TotalElectrons);

    %If in scatter probability, scatter
    Vx(Scat<PScat) = (Vth/sqrt(2))*randn(1);
    Vy(Scat<PScat) = (Vth/sqrt(2))*randn(1);

    NewPx = Vx*dt+Px;                       %Calculate new x position
    NewPy = Vy*dt+Py;                       %Calculate new y position

    %Check right boundary
    ix = NewPx>length;
    NewPx(ix) = NewPx(ix)-length;
    Px(ix) = Px(ix)-length;

    %Check left boundary
    ix = NewPx<0;
    NewPx(ix) = NewPx(ix)+length;
    Px(ix) = Px(ix)+length;

    %Check bottom boundary
```

```
    ix = NewPy<0;
    Vy(ix) = -Vy(ix);

    %Check top boundary
    ix = NewPy>height;
    Vy(ix) = -Vy(ix);




    %Check  box boundaries
        xlow = Px < .75E-7;
        xNewLow = NewPx>.75E-7;
        xlarge = Px>1.25E-7;
        xNewLarge = NewPx<1.25E-7;
        ylow = Py >.3E-7;
        yNewLow = NewPy <.3E-7;
        ytop = Py < .7E-7;
        yNewTop = NewPy > .7E-7;

        %left side boundary condition
        ViolateLeft = bitand (bitand(xlow, xNewLow),
 bitor(yNewLow,yNewTop));
        Differ(ViolateLeft) = NewPx(ViolateLeft)-0.75E-7;
        Vx(ViolateLeft)= -Vx(ViolateLeft);
        NewPx(ViolateLeft) = 0.75E-7-Differ(ViolateLeft);

        %right side boundary condition
        ViolateRight = bitand(bitand(xlarge, xNewLarge),
 bitor(yNewLow, yNewTop));
        Differ(ViolateRight) = 1.25E-7 - NewPx(ViolateRight);
        Vx(ViolateRight)= -Vx(ViolateRight);
        NewPx(ViolateRight) = 1.25E-7+Differ(ViolateRight);

        %Middle Top Boundary Condition
        ViolateTop = bitand((bitand(ytop, yNewTop)),(bitand(xNewLow,
 xNewLarge)));
        Differ(ViolateTop) = NewPy(ViolateTop) - 0.7E-7;
        Vy(ViolateTop) = -Vy(ViolateTop);
        NewPy(ViolateTop) = 0.7E-7 - Differ(ViolateTop);

        %Middle Bottom Boundary Condition
        ViolateBottom = bitand((bitand(xNewLow, xNewLarge)),
(bitand(ylow, yNewLow)));
        Differ(ViolateBottom) = 0.3E-7 - NewPy(ViolateBottom);
        Vy(ViolateBottom) = -Vy(ViolateBottom);
        NewPy(ViolateBottom) = 0.3E-7 + Differ(ViolateBottom);



    %Plot electrons
    subplot (2,3,1)
    rectangle ('position', [.75E-7  0 .5E-7 .3E-7],'EdgeColor','k');
```

```
    rectangle ('position', [.75E-7
  .7E-7 .5E-7 .3E-7],'EdgeColor','k');

    plot([Px(1) NewPx(1)], [Py(1) NewPy(1)], 'b')       %Electron 1

    plot([Px(2) NewPx(2)], [Py(2) NewPy(2)], 'g')       %Electron 2
    plot([Px(3) NewPx(3)], [Py(3) NewPy(3)], 'r')       %Electron 3
    plot([Px(4) NewPx(4)], [Py(4) NewPy(4)], 'c')       %Electron 4
    plot([Px(5) NewPx(5)], [Py(5) NewPy(5)], 'm')       %Electron 5
    plot([Px(6) NewPx(6)], [Py(6) NewPy(6)], 'y')       %Electron 6
    plot([Px(7) NewPx(7)], [Py(7) NewPy(7)], 'k')       %Electron 6
    hold on
    %Format plot
    title('Electron Paths');
    xlim([0 200e-9]);
    ylim ([0 100e-9]);
    pause(0.000001)

    %Update old positions
    Px=NewPx;
    Py=NewPy;

    %Calculate and plot temperature
    V2 = (Vx.^2)+(Vy.^2);
    V = sqrt(V2);

    Vm = mean(V2);
    TCalc = ((Vm)*(C.m_0*0.26))/2/C.kb;
    subplot(2,3,2)
    plot([loops-1 loops], [OGTemp TCalc],'b');
    TotalTemp = TotalTemp+TCalc;
    AverageTemp = (TotalTemp/loops);

    legend (strcat('Temp; Average Temp: ',
num2str(round(AverageTemp)),' k'));
    title ('Temperature (K)');
    xlim([0 time]);
    ylim ([0 600]);
    OGTemp = TCalc;
     hold on

     for yiter = 1:10
        ymax = yiter*10;
        ymin = ymax-10;
        for xiter = 1:20
            xmax = xiter*10;
            xmin = xmax-10;
            xming = Px >(xmin*1e-9);
            xmaxl = Px <(xmax*1e-9);
            ymming = Py >(ymin*1e-9);
            ymaxl = Py <(ymax*1e-9);
            IsLocated = bitand(bitand(xming, xmaxl), bitand(yming,
ymaxl));
            V22 = (Vx(IsLocated).^2)+(Vy(IsLocated).^2);
```

```
            Vmm = mean(V22);
            TCalc2 = ((Vmm)*(C.m_0*0.26))/2/C.kb;
            TempArray(xiter,yiter)= TCalc2;
        end
    end
    subplot(2,3,5)
    surf(transpose(TempArray));

    colorbar
        title ('Temperature Distribution');
        view(2)

end
```

The initial thermal velocity is equal to 1.870193e+05
The mean free path is equal to 3.740385e-08



Published with MATLAB® R2018b