

Carleton University

Course: ELEC 4700 Modelling of Integrated Device

Assignment No: 1

Monte-Carlo Modeling of Electron Transport

Kwabena Gyasi Bawuah

101048814

Due: 02/7/2021

# Table of Contents

Introduction: .....	3
Electron Modelling: .....	3
Collisions with Mean Free Path (MFP):.....	8
Enhancements: .....	13
Conclusion: .....	23

## Introduction:

This report is for ELEC 4700 in response to the call for an assignment report. The assignment was to model carriers as a population of electrons in an N-type Si semiconductor crystal (Monte-Carlo model). These particles are then to be giving velocities using the Maxwell-Boltzmann distribution. Lastly, an enhancement is to put the system to test by adding a bottle neck boundary. This report will detail the results from the built simulation, observation of results, discussion of results, answers to specific questions asked in the assignment and conclusions derived. Samples of the code used to perfume these models will also be produced within the sections.

## Electron Modelling:

Few questions:

- a) What is the thermal velocity  $v_{th}$ ? Assume  $T = 300\text{ K}$ .

With  $m_n$  = effective mass =  $0.26 \times m_0$

$T = 300\text{ K}$

$k = 1.38 \times 10^{-23}$

$$v_{th} = \sqrt{\frac{(2 \times k \times T)}{m_n}}$$

$$= 1.8702 \times 10^5 \text{ m/s}$$

- b) If the mean time between collisions is  $\tau_{mn} = 0.2\text{ ps}$  what is the mean free path?

$$\text{MFP} = \tau_{mn} * v_{th} = 3.7404 \times 10^{-8} \text{ m}$$

c)

- i. A simulation of the particles was first created. An end picture of the the simulation is below:

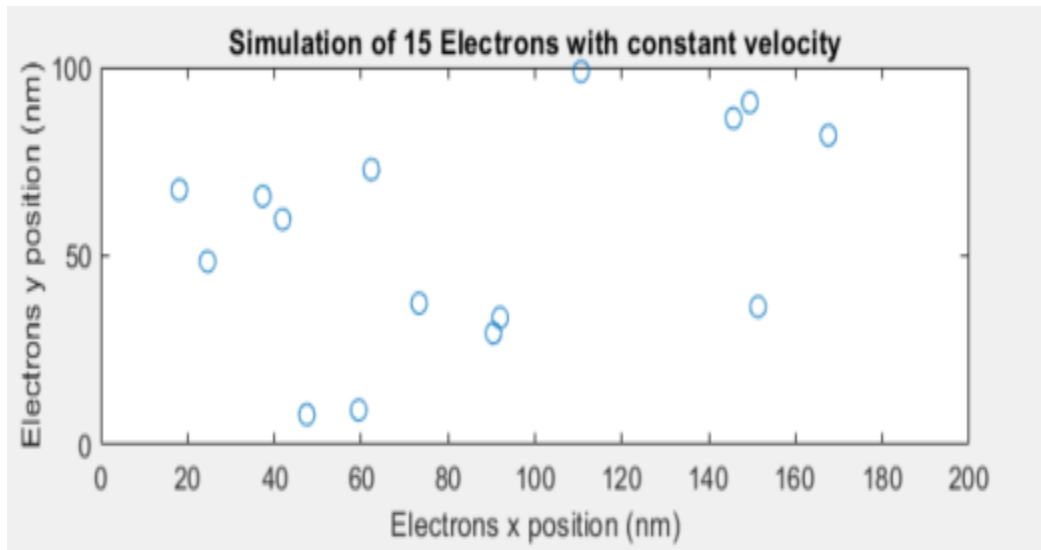


Figure 1: Plot of electrons in 2D.

2-D plot of particle trajectories:

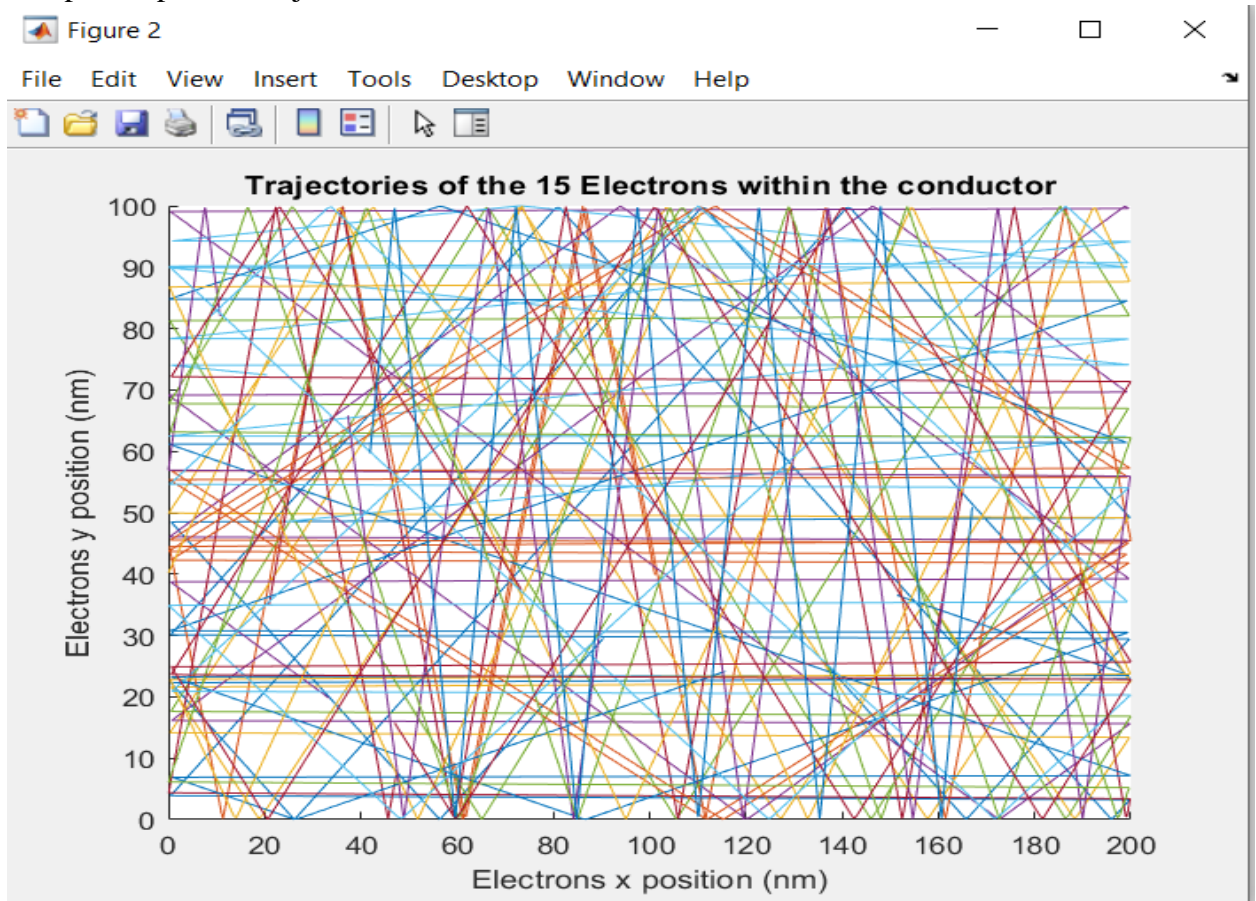


Figure 2: Plot of electrons trajectory

ii. Temperature plot

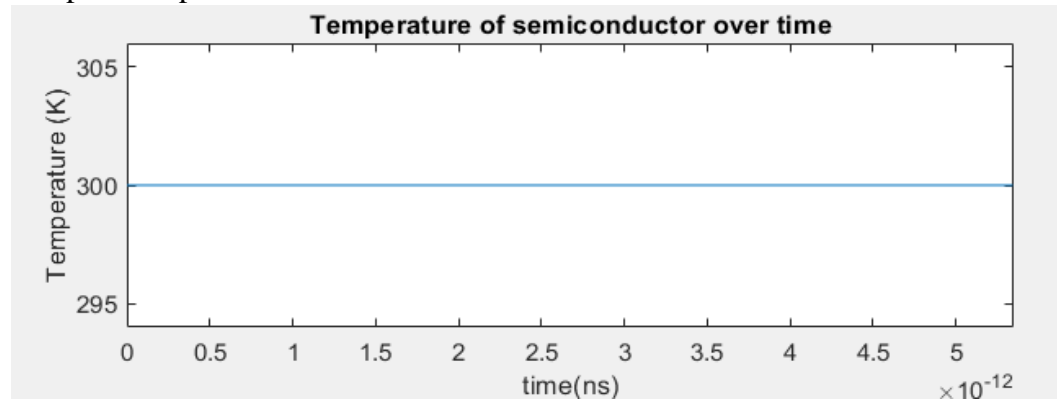


Figure 3: Plot of average temperature.

Code Used for Q1:

```
%Question 1
close all;
clc
%Kwabena Gyasi Bawuah
%101048814
%UNTITLED Summary of this function goes here
% Detailed explanation goes here
%electron spec
global C

    addpath ../geom2d/geom2d

    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 s^2
```

```

T = 300;
k = 1.38e-23;
mn = 0.26*C.m_0; %effective mass
tmn = 0.2e-12;    % Mean time between collisions

vth = sqrt((2*C.kb*T)/mn); % Thermal velocity

freepath = vth*tmn    % mean free path

ConductorL = 180e-9;
ConductorW = 80e-9;

dpoints = 5e4;
ecount = 15; %the number of electron to show on plot

detaT= ConductorW/vth/100;
sims = 1000;

%      Xpos = rand(1,ecount).*ConductorW;
%      Ypos = rand(1,ecount).*ConductorL;
traj=zeros(sims,ecount*2);
temp=zeros(sims, 1);

%to bring in the robotics system toolbox in order to
use state
%declare initial state
for i = 1: dpoints
    angle = rand*2*3.14;
    state(i,:) = [ConductorL*rand ConductorW*rand
vth*cos(angle) vth*sin(angle)];
end
%initial array of temp
temp(:,1) = 300;
%to iterate over 1000 time steps for plot points
for i = 1 :sims
state(:,1:2)=state(:,1:2)+detaT.*state(:,3:4);
%specifying the particles reactions at boundary
out = state(:,1) > ConductorL;
state(out,1) = state(out,1)-ConductorL;

out = state(:,2) < 0;
state(out,2) = -state(out,2);
state(out,4) = -state(out,4);

```

```

out = state(:,2)> ConductorW;
state(out,2)= 2 * ConductorW - state(out,2);
state(out,4)= -state(out,4);

out = state(:,1)< 0;
state(out,1)=state(out,1)+ ConductorL;

%iterating over array of visible electrons
for out = 1:ecount
    traj(i, (2 * out):(2 * out + 1)) = state(out, 1 :
2);
end

%plot on every 5 iteration steps
if mod(i,5)==0
    figure (1)
    subplot(2,1,1);
    plot(state(1:ecount,1)./1e-9,state(1:ecount,2)./1e-
9, 'o')
    xlim([0 ConductorL/1e-9])
    ylim([0 ConductorW/1e-9])
    xlabel('Electrons x position (nm)')
    ylabel('Electrons y position (nm)')
    title('Simulation of 15 Electrons with constant
velocity')

    subplot(2,1,2);
    plot(detaT*(0:i-1),temp(1:i));
    xlim([0 detaT*sims])
    ylim([min(temp)*0.98 max(temp)*1.02]);
    xlabel('time(s)');
    ylabel('Temperature (K)');
    title('Temperature of semiconductor over time');
end
end

%iterate over the visible particles and show thier
%paths
figure(2)
hold on;
xlim([0 ConductorL/1e-9]);
ylim([0 ConductorW/1e-9]);
xlabel('Electrons x position (nm)');
ylabel('Electrons y position (nm)');

```

```

title('Trajectories of the 15 Electrons within the
conductor');
for i = 1: ecount
plot(traj(:,i*2)./1e-9, traj(:,i*2+1)./1e-9, '-');
end

```

Collisions with Mean Free Path (MFP):

Histogram:

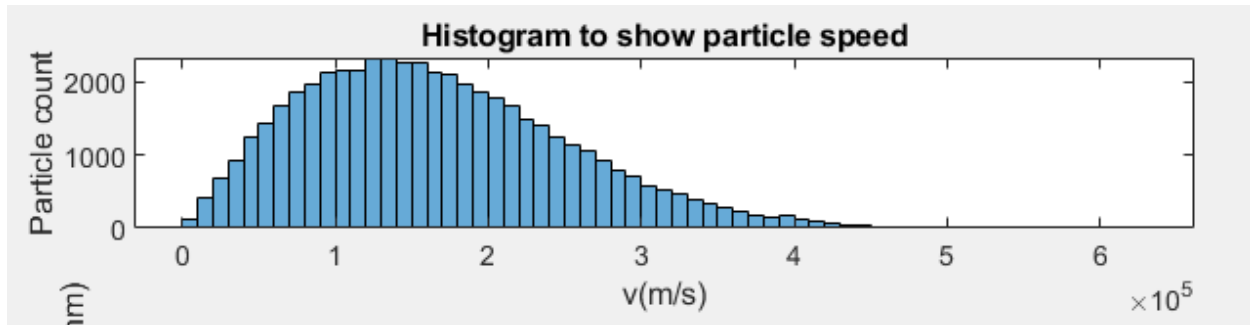


Figure 4: Histogram of average particle speed.

2-D plot of particle trajectories:

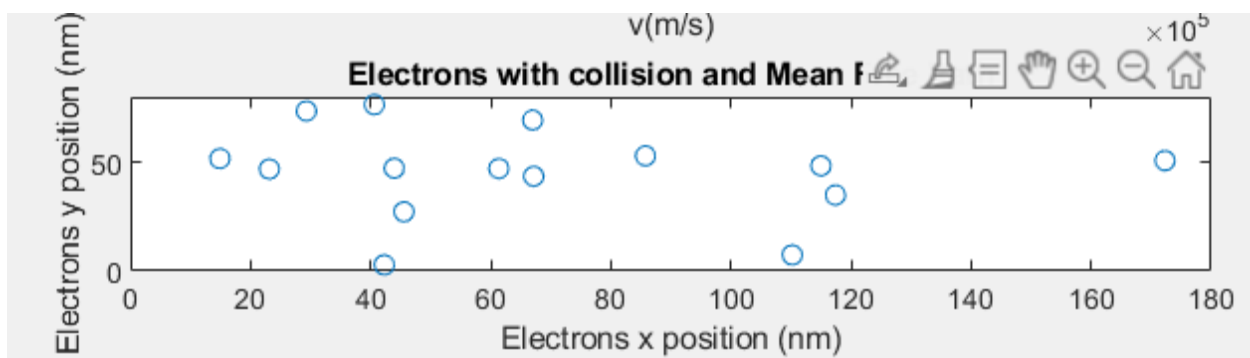


Figure 5: Plot of electrons in 2D.



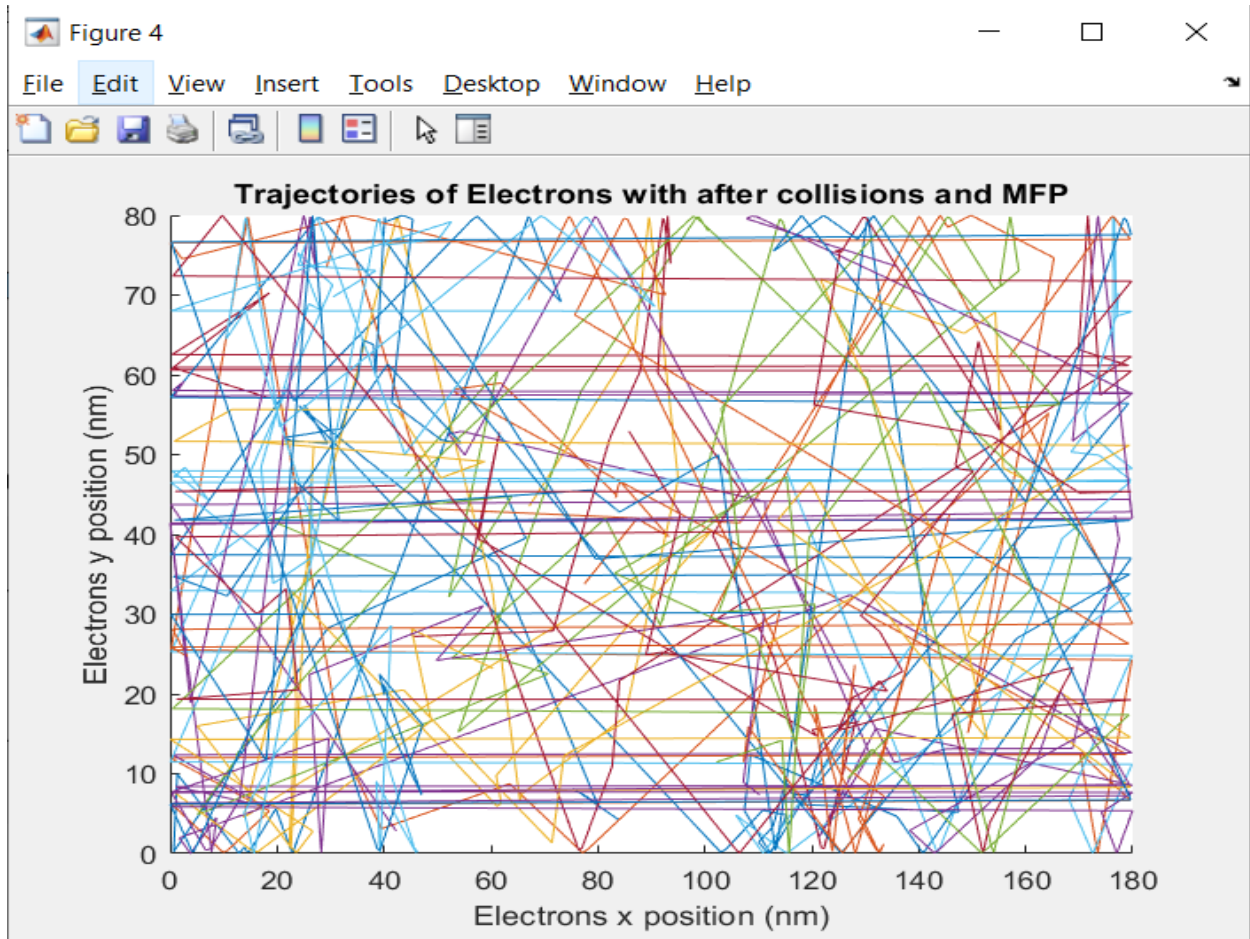


Figure 6: Plot of electrons trajectory.

Plot of temperature:

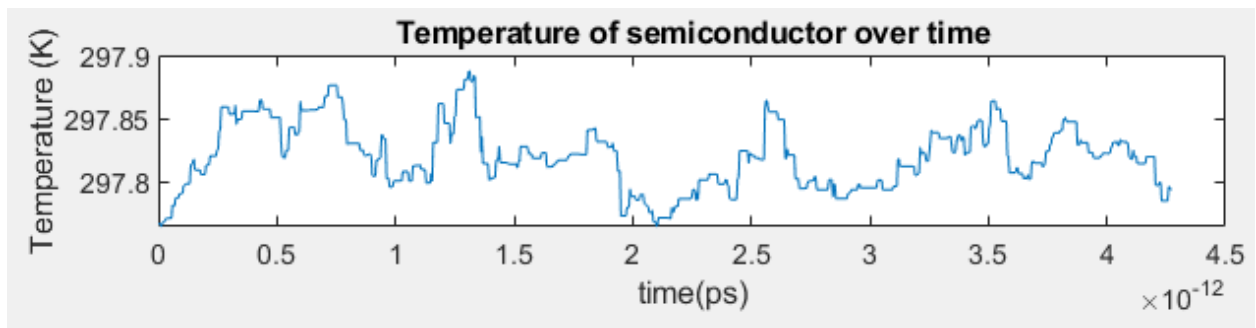


Figure 7: Plot of average temperature over time.

1. What happens to the average temperature over time?  
This averaged around the 297.8K temperature which is approximately 300K.
2. Measure the actual Mean Free Path and mean time between collisions to verify your model.  
 $=1.9006e-08$

$$\begin{aligned}
 T_{mn} &= \text{MFP} / v_{th} \\
 &= 1.9006\text{e-}08 / 1.8702 \times 10^5 \\
 &= 10 \times 10^{-12}
 \end{aligned}$$

Code Used:

```

%Question 2
close all;
clc
%Kwabena Gyasi Bawuah
%101048814
%UNTITLED Summary of this function goes here
% Detailed explanation goes here
%electron spec
global C

addpath ../geom2d/geom2d

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 s^2

T = 300;
k = 1.38e-23;
mn = 0.26*C.m_0; %effective mass
tmn = 0.2e-12; % Mean time between collisions

vth = sqrt((2*C.kb*T)/mn); % Thermal velocity

freepath = vth*tmn % mean free path

ConductorL = 180e-9;
ConductorW = 80e-9;

dpoints = 5e4;

```

```

ecount = 15; %the number of electron to show on plot

detaT= ConductorW/vth/100;
sims = 1000;

%     Xpos = rand(1,ecount).*ConductorW;
%     Ypos = rand(1,ecount).*ConductorL;
traj=zeros(sims,ecount*2);
temp=zeros(sims, 1);

temp(:,1)= 300;
%-----
%Collisions with Mean Free Path

Pscat = 1-exp(-detaT/tmn);
%to make a probability distribution with mu=0 and
sigma=vth
ProbDistr = makedist('Normal','mu', 0, 'sigma',
sqrt(C.kb*T/mn));
for i = 1: dpoints
    state(i,:)= [ConductorL*rand ConductorW*rand
random(ProbDistr) random(ProbDistr)];
end

%from part 1
for i = 1 :sims
state(:,1:2)=state(:,1:2)+detaT.*state(:,3:4);
%specifying the particles reactions at boundary
out = state(:,1)> ConductorL;
state(out,1) = state(out,1)-ConductorL;

out = state(:,2) < 0;
state(out,2) = -state(out,2);
state(out,4) = -state(out,4);

out = state(:,2)> ConductorW;
state(out,2)= 2 * ConductorW - state(out,2);
state(out,4)= -state(out,4);

out = state(:,1)< 0;
state(out,1)=state(out,1)+ ConductorL;

out = rand(ecount,1) < Pscat;
state(out,3:4)=random(ProbDistr,[sum(out),2]);

```

```

%varying temp
temp(i)=(sum(state(:,3).^2) +
sum(state(:,4).^2)).*mn/k/2/dpoints;
%iterating over array of visible electrons
for out = 1:ecount
    traj(i, (2 * out):(2 * out + 1)) = state(out, 1:2);
end
%plot on every 5 iteration steps
if mod(i,5)==0
    figure(3);
    subplot(3,1,1);
    part = sqrt(state(:,3).^2 + state(:,4).^2);
    xlim([0 7e5]);
    ylim([0 2000]);
    histogram(part);
    xlabel('v(m/s)');
    ylabel('Particle count');
    title('Histogram to show particle speed');

    subplot(3,1,2);
    plot(state(1:ecount,1)./1e-9,state(1:ecount,2)./1e-
9, 'o')
    xlim([0 ConductorL/1e-9])
    ylim([0 ConductorW/1e-9])
    xlabel('Electrons x position (nm)')
    ylabel('Electrons y position (nm)')
    title('Electrons with collision and Mean Free
Path')

    subplot(3,1,3);
    plot(detaT*(0:i-1),temp(1:i));
    xlabel('time(s)');
    ylabel('Temperature (K)');
    title('Temperature of semiconductor over time');

    end
end

%iterate over the visible particles and show thier
%paths
figure(4)
hold on;
xlim([0 ConductorL/1e-9]);

```

```

ylim([0 ConductorW/1e-9]);
xlabel('Electrons x position (nm)');
ylabel('Electrons y position (nm)');
title('Trajectories of Electrons with after collisions
and MFP');
for i = 1: ecount
plot(traj(:,i*2)./1e-9, traj(:,i*2+1)./1e-9, '-');
end

Vx=(vth/sqrt(2))*rand(ecount,1);
Vy=(vth/sqrt(2))*rand(ecount,1);
Vdis=sqrt(Vx.^2+Vy.^2);
Vavg = mean(Vdis);
MFP = Vavg*tmn;

```

Enhancements:

a) 2-D plot of particle trajectories

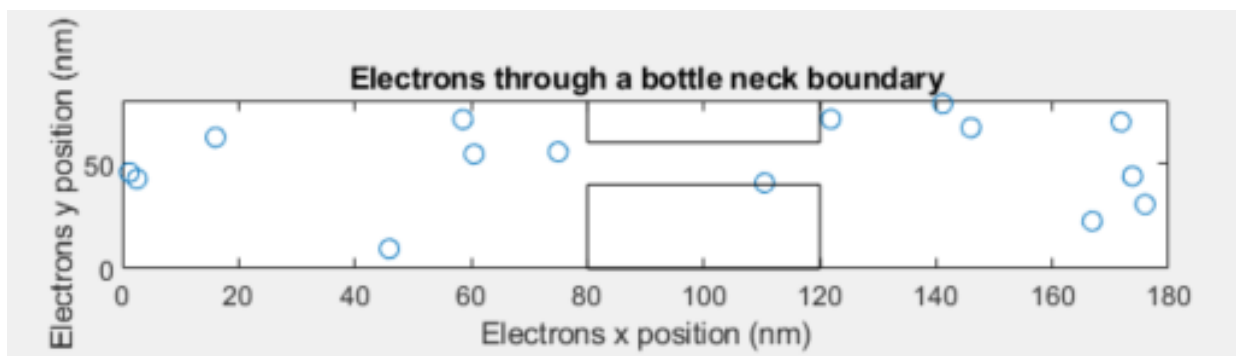


Figure 8: Plot of electrons in 2D through bottle neck boundary.

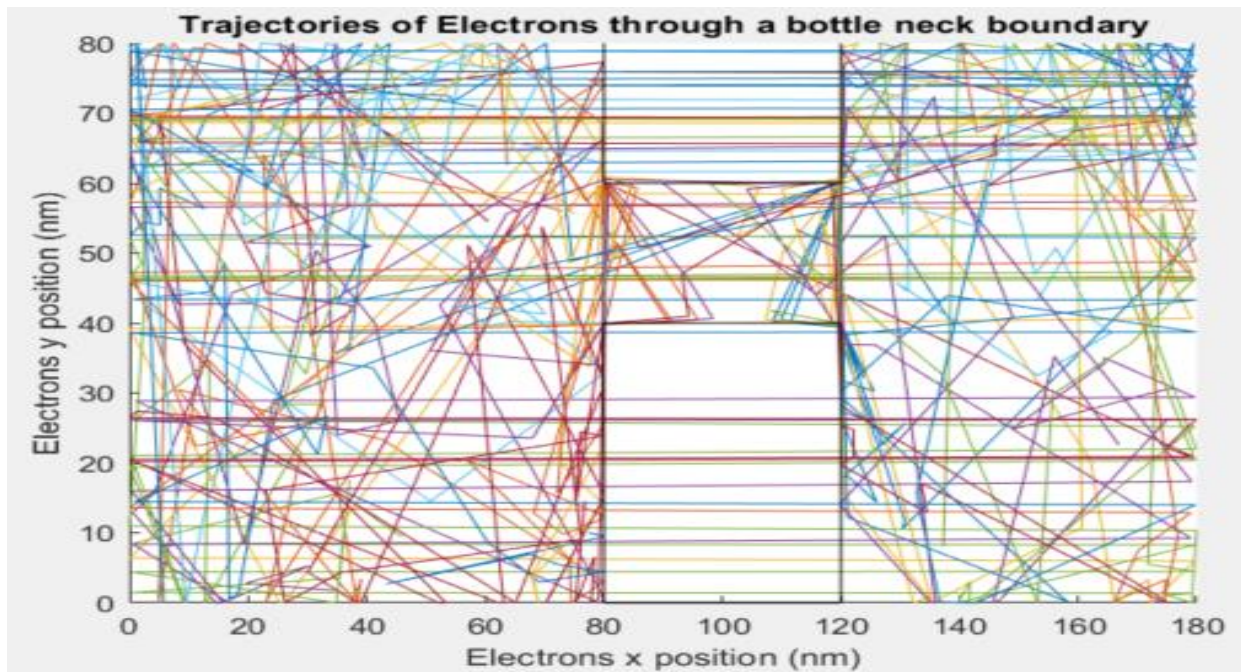


Figure 9: Plot of electrons trajectory.

b) Histogram showing particle speed and temperature:

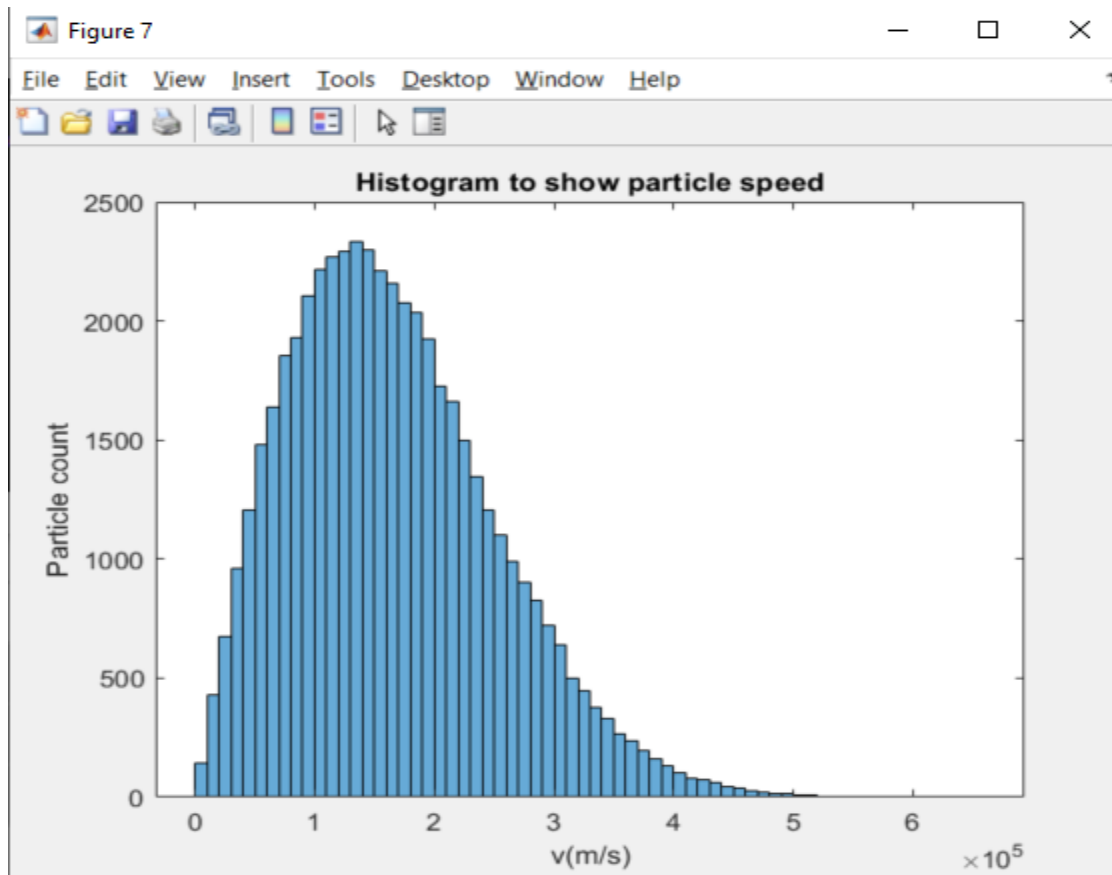


Figure 10: Histogram of particle speed.

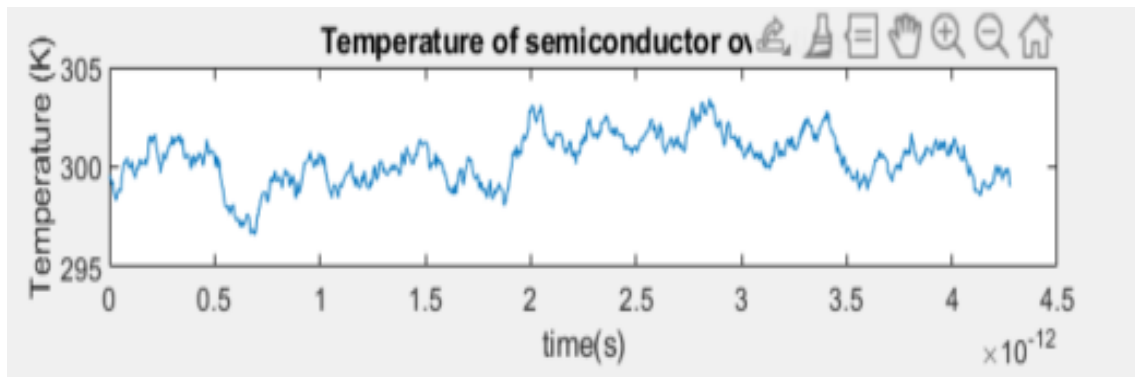


Figure 11: Plot of average temperature.

c) Electron density map

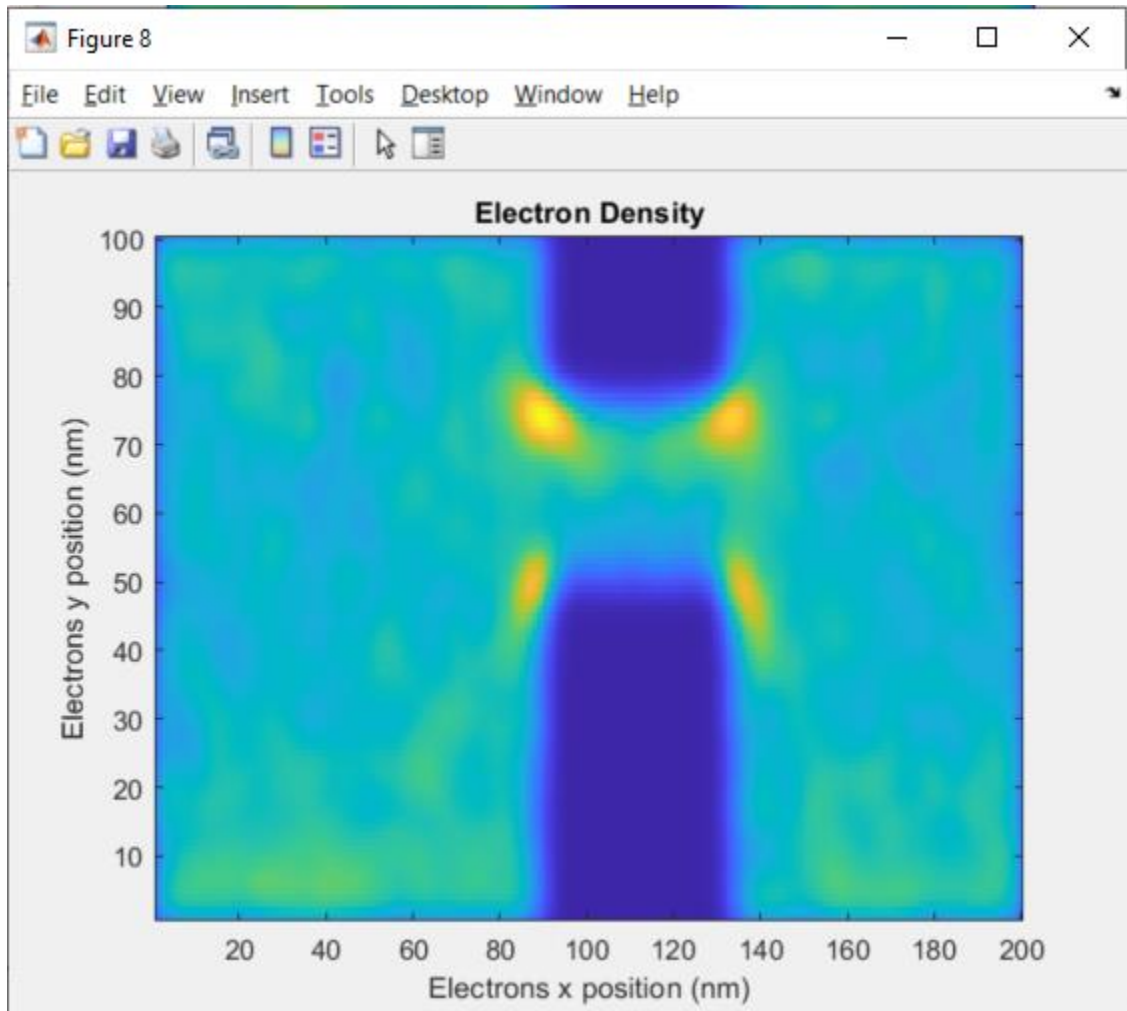


Figure 12: electron density plot.



d) Temperature map

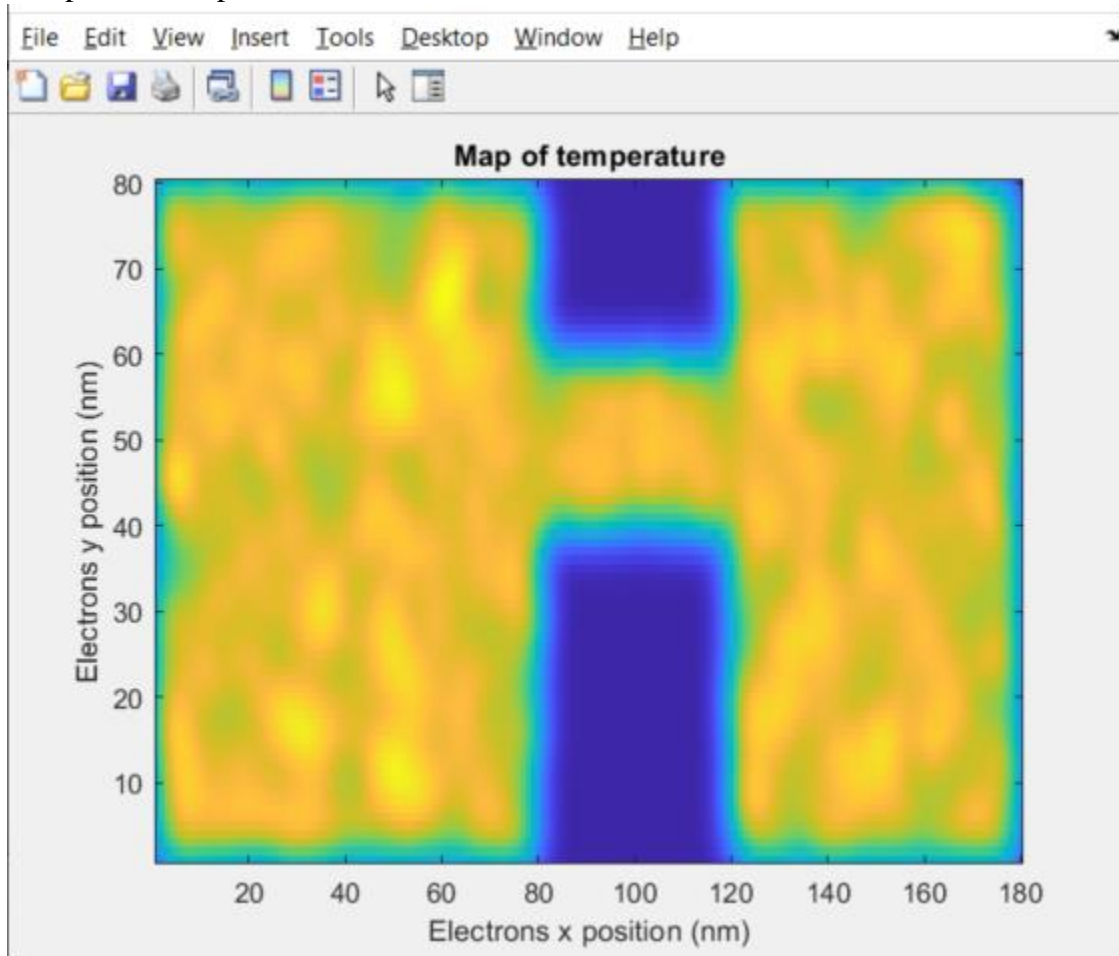


Figure 2: Map of temperature.

Code Used:

```
%Question 3
close all;
clc
%Kwabena Gyasi Bawuah
%101048814
%UNTITLED Summary of this function goes here
% Detailed explanation goes here
%electron spec
global C

addpath ../geom2d/geom2d

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

        T = 300;
        k = 1.38e-23;
        mn = 0.26*C.m_0; %effective mass
        tmn = 0.2e-12;    % Mean time between collisions

        vth = sqrt((2*C.kb*T)/mn); % Thermal velocity

        freepath = vth*tmn    % mean free path

        ConductorL = 180e-9;
        ConductorW = 80e-9;

        dpoints = 5e4;
        ecoun = 15; %the number of electron to show on plot

        detaT= ConductorW/vth/100;
        sims = 1000;

%       Xpos = rand(1,ecoun).*ConductorW;
%       Ypos = rand(1,ecoun).*ConductorL;
        traj=zeros(sims,ecoun*2);
        temp=zeros(sims, 1);

        temp(:,1)= 300;
        Pscat = 1-exp(-detaT/tmn);
        %-----
        ProbDistr = makedist('Normal','mu', 0, 'sigma',
sqrt(C.kb*T/mn));

        tspec = 0;
        bspec=0;
        boxes = 1e-9.*[80 120 0 40; 80 120 60 100];

```

```

specularbox = [0 1];
for i = 1: dpoints
    angle = rand*2*3.14;
    state(i,:) = [ConductorL*rand ConductorW*rand
random(ProbDistr) random(ProbDistr)];

    if (state(i,2)>60e-9 & (state(i,1)>80e-9 &
state(i,1)<120e-9)) | (state(i,2)< 40e-9 & (state(i,1)>80e-
9 & state(i,1)<120e-9))
        state(i,1:2) = [ConductorL*rand
ConductorW*rand];
    end

end

for i = 1:sims
    state(:,1:2) = state(:,1:2) + detaT.*state(:,3:4);

    out = state(:,1) > ConductorL;
    state(out,1) = state(out,1) - ConductorL;

    out = state(:,1) < 0;
    state(out,1) = state(out,1) + ConductorL;

    out = state(:,2) > ConductorW;

    if (tspec)
        state(out,2) = 2*ConductorW - state(out,2);
        state(out,4) = -state(out,4);
    else
        state(out,2) = ConductorW;
        part = sqrt(state(out,3).^2 + state(out,4).^2);
        angle = rand([sum(out),1])*2*3.14;
        state(out,3) = part.*cos(angle);
        state(out,4) = -abs(part.*sin(angle));
    end

    out = state(:,2) < 0;

    if (bspec)
        state(out,2) = -state(out,2);
        state(out,4) = -state(out,4);
    else
        state(out,2) = 0;
    end
end

```

```

    part = sqrt(state(out,3).^2 + state(out,4).^2);
    angle = rand([sum(out),1])*2*3.41;
    state(out,3) = part.*cos(angle);
    state(out,4) = abs(part.*sin(angle));
end

for out=1: dpoints
    if (state(out,2)>60e-9 & (state(out,1)>80e-9 &
state(out,1)<120e-9))
        boxNum = 1;
    elseif (state(out,2)< 40e-9 & (state(out,1)>80e-
9 & state(out,1)<120e-9))
        boxNum = 2;
    else
        boxNum = 0;
    end
    while(boxNum ~= 0)
        XDist = 0;
        newx = 0;
        if(state(out,3) > 0)
            XDist = state(out,1) - boxes(boxNum,1);
            newx = boxes(boxNum,1);
        else
            XDist = boxes(boxNum,2) - state(out,1);
            newx = boxes(boxNum,2);
        end

        yDist = 0;
        newy = 0;
        if(state(out,4) > 0)
            yDist = state(out,2) - boxes(boxNum,
3);
            newy = boxes(boxNum, 3);
        else
            yDist = boxes(boxNum, 4) -
state(out,2);
            newy = boxes(boxNum, 4);
        end

        if(XDist < yDist)
            state(out,1) = newx;
            if(~specularbox(boxNum))
                sgn = -sign(state(out,3));

```

```

                                part = sqrt(state(out,3).^2 +
state(out,4).^2);
                                angle = rand()*2*3.14;
                                state(out,3) =
sgn.*abs(part.*cos(angle));
                                state(out,4) = part.*sin(angle);
                                else
                                    state(out,3) = -state(out,3);
                                end
                                else
                                    state(out,2) = newy;
                                    if(~specularbox(boxNum))
                                        sgn = -sign(state(out,4));
                                        part = sqrt(state(out,3).^2 +
state(out,4).^2);
                                        angle = rand()*2*3.14;
                                        state(out,3) = part.*cos(angle);
                                        state(out,4) =
sgn.*abs(part.*sin(angle));
                                    else
                                        state(out,4) = -state(out,4);
                                    end
                                end
                                boxNum = 0;
                            end
                        end

out = rand(dpoints, 1) < Pscat;
state(out,3:4) = random(ProbDistr, [sum(out),2]);

temp(i) = (sum(state(:,3).^2) +
sum(state(:,4).^2))*mn/k/2/dpoints;

for out=1:ecount
    traj(i, (2*out):(2*out+1)) = state(out,1:2);
end

if mod(i,5) == 0
    figure(5);
    subplot(3,1,1);
    hold off;

```

```

        plot(state(1:ecount,1)./1e-9,
state(1:ecount,2)./1e-9, 'o');
        hold on;
        for ouut=1:size(boxes,1)
            plot([boxes(ouut, 1) boxes(ouut, 1) boxes(ouut,
2) boxes(ouut, 2) boxes(ouut, 1)]./1e-9,[boxes(ouut, 3)
boxes(ouut, 4) boxes(ouut, 4) boxes(ouut, 3) boxes(ouut,
3)]./1e-9, 'k-');
        end

        xlim([0 ConductorL/1e-9])
        ylim([0 ConductorW/1e-9])
        xlabel('Electrons x position (nm)')
        ylabel('Electrons y position (nm)')
        title('Electrons through a bottle neck boundary')

        subplot(3,1,2);
        plot(detaT*(0:i-1), temp(1:i));
        plot(detaT*(0:i-1),temp(1:i));
        xlabel('time(s)');
        ylabel('Temperature (K)');
        title('Temperature of semiconductor over time');

        subplot(3,1,3);
        part = sqrt(state(:,3).^2 + state(:,4).^2);
        xlim([0 7e5]);
        ylim([0 2000]);
        histogram(part);
        xlabel('v(m/s)');
        ylabel('Particle count');
        title('Histogram to show particle speed');
        end
    end

    figure(6)
    hold on;
    xlim([0 ConductorL/1e-9]);
    ylim([0 ConductorW/1e-9]);
    xlabel('Electrons x position (nm)');
    ylabel('Electrons y position (nm)');
    title('Trajectories of Electrons through a bottle neck
boundary');

```

```

    for i = 1: ecount
        plot(traj(:,i*2)./1e-9, traj(:,i*2+1)./1e-9, '-');
    end

    for out=1:size(boxes,1)
        plot([boxes(out, 1) boxes(out, 1) boxes(out, 2)
boxes(out, 2) boxes(out, 1)]./1e-9,...
        [boxes(out, 3) boxes(out, 4) boxes(out, 4)
boxes(out, 3) boxes(out, 3)]./1e-9, 'k-');
    end

    figure(7)
    part = sqrt(state(:,3).^2 + state(:,4).^2);
    xlim([0 7e5]);
    ylim([0 2000]);
    histogram(part);
    xlabel('v(m/s)');
    ylabel('Particle count');
    title('Histogram to show particle speed');

    density = hist3(state(:,1:2),[200 100])';

    N = 20;
    sigma = 3;
    [x y]=meshgrid(round(-N/2):round(N/2), round(-
N/2):round(N/2));
    f=exp(-x.^2/(2*sigma^2)-y.^2/(2*sigma^2));
    f=f./sum(f(:));
    figure(8);
    imagesc(conv2(density,f,'same'));
    set(gca,'YDir','normal');
    title('Electron Density');
    xlabel('Electrons x position (nm)');
    ylabel('Electrons y position (nm)');

    tempSumX = zeros(ceil(ConductorL/1e-
9),ceil(ConductorW/1e-9));
    tempSumY = zeros(ceil(ConductorL/1e-
9),ceil(ConductorW/1e-9));
    tempSum = zeros(ceil(ConductorL/1e-
9),ceil(ConductorW/1e-9));

    for i=1:dpoints

```

```

        x = floor(state(i,1)/1e-9);
        y = floor(state(i,2)/1e-9);
        if(x==0)
            x = 1;
        end
        if(y==0)
            y= 1;
        end

        tempSumY(x,y) = tempSumY(x,y) + state(i,3)^2;
        tempSumX(x,y) = tempSumX(x,y) + state(i,4)^2;
        tempSum(x,y) = tempSum(x,y) + 1;
    end

    temp = (tempSumX + tempSumY).*mn./k./2./tempSum;
    temp(isnan(temp)) = 0;
    temp = temp';

    N = 20;
    sigma = 3;
    [x y]=meshgrid(round(-N/2):round(N/2), round(-
N/2):round(N/2));
    f=exp(-x.^2/(2*sigma^2)-y.^2/(2*sigma^2));
    f=f./sum(f(:));
    figure(9);
    imagesc(conv2(temp,f,'same'));
    set(gca,'YDir','normal');
    title('Map of temperature');
    xlabel('Electrons x position (nm)');
    ylabel('Electrons y position (nm)');

```

## Conclusion:

The model of electrons of different speed and different bounds in an N-type Si semiconductor crystal was successfully constructed. The results from observations to be as expected. It is obvious that I failed to ensure the integrity of the boundaries of the bottle neck in figure 9. All questions where also as answers with the code used provided in the sections. The codes can be put together in a Matlab file and run for a complete simulation of the system.