ELEC 4700

ASSIGNMENT 3

MONTE-CARLO AND FINITE DIFFERENCE METHOD

Submitted by: Oritseserundede Eda (Albert)

Student Number: 100993421

Date Submitted: 21/03/2021

Introduction

In this Assignment, we combined the Monte-Carlo simulation with the Finite-Difference method.

Furthermore, to observe what happens when an electron experiences a push in a field.

Part 1

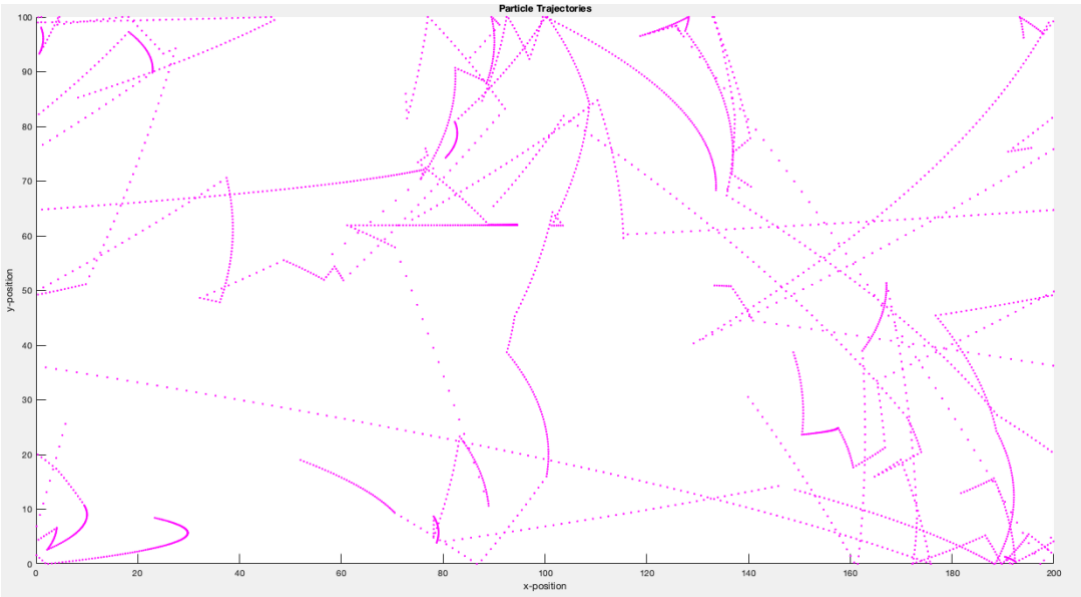

Figure 1: Particle trajectory (without bottleneck)

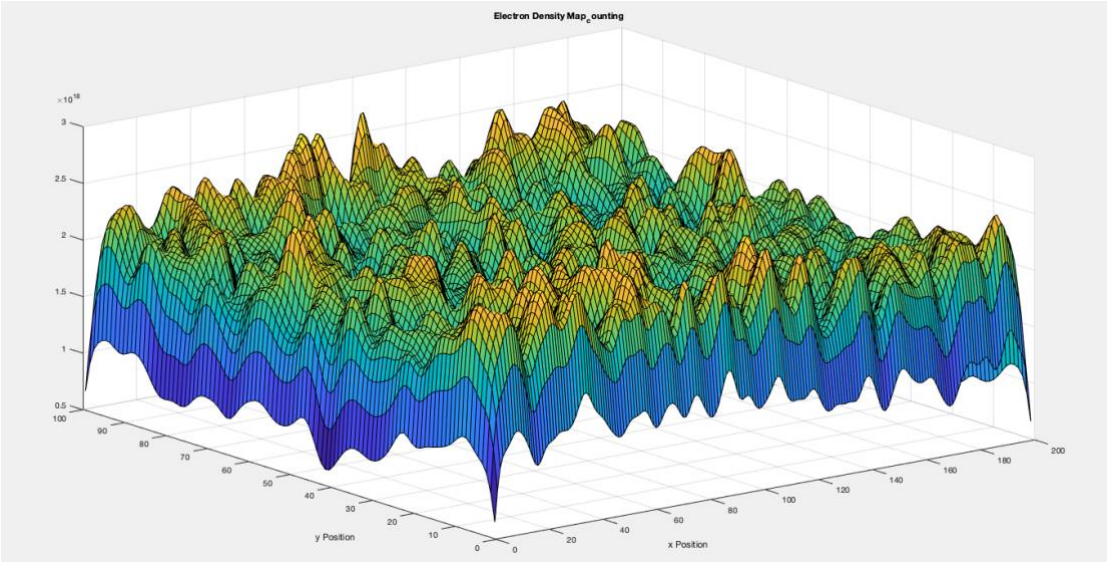

Figure 2: The Electron Density Mapping

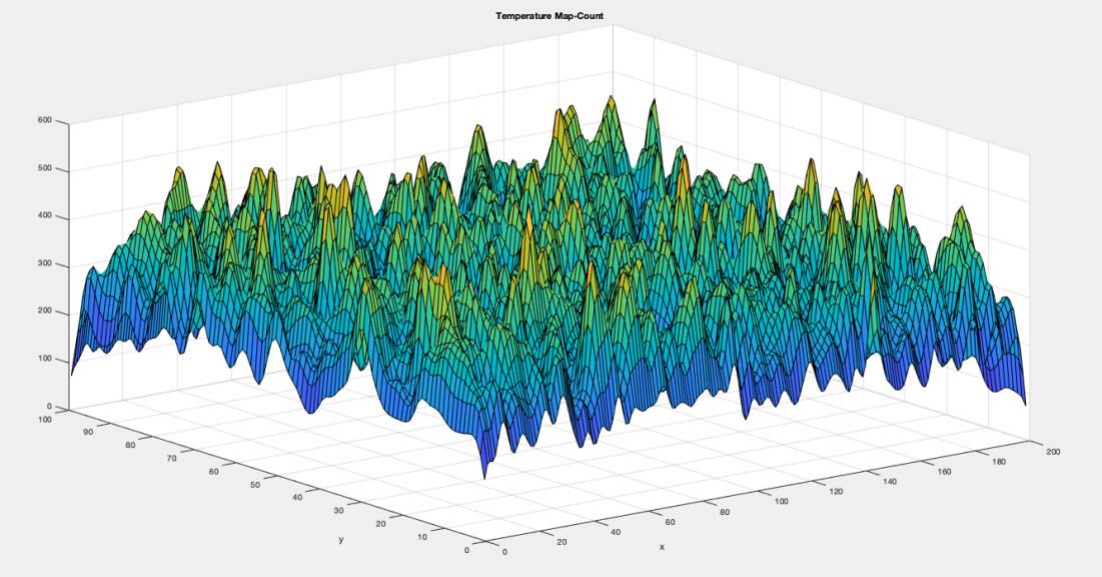

Figure 3: The Temperature Mapping Count

There is a linear relationship between the Electron drift current density and the average carrier velocity. The relationship was modelled through the equation:

$$J = V * n * q * Ny$$

Where:

- V is the average carrier velocity of the particles
- n is the electron charge concentrate, which was given as $10^{15}cm^{-2}$
- Q is the electron charge and Ny is the length in the y-boundary.

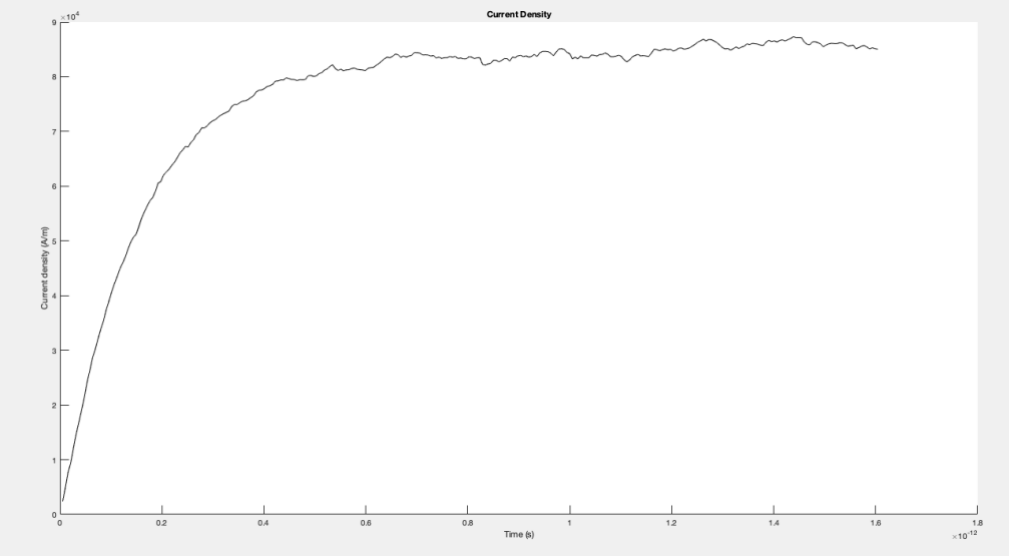

Figure 4: Current Density Vs Time

From the plot above, the current rise until it reaches its peak where saturation occurs. And this saturation in current is re-thermalizing, which involves a reset in the velocities of the particles that caused the scattering.

```
The Electric Field of the Charge is 500000.000000 V/m.
The Force on the charge is 8.010883e-14 N.
The acceleration of the Charge is 338234619754597888.000000 m/s^2.
```

Figure 5: The MATLAB command line showing the Force, Electric Field and Acceleration on the charge

Part 2

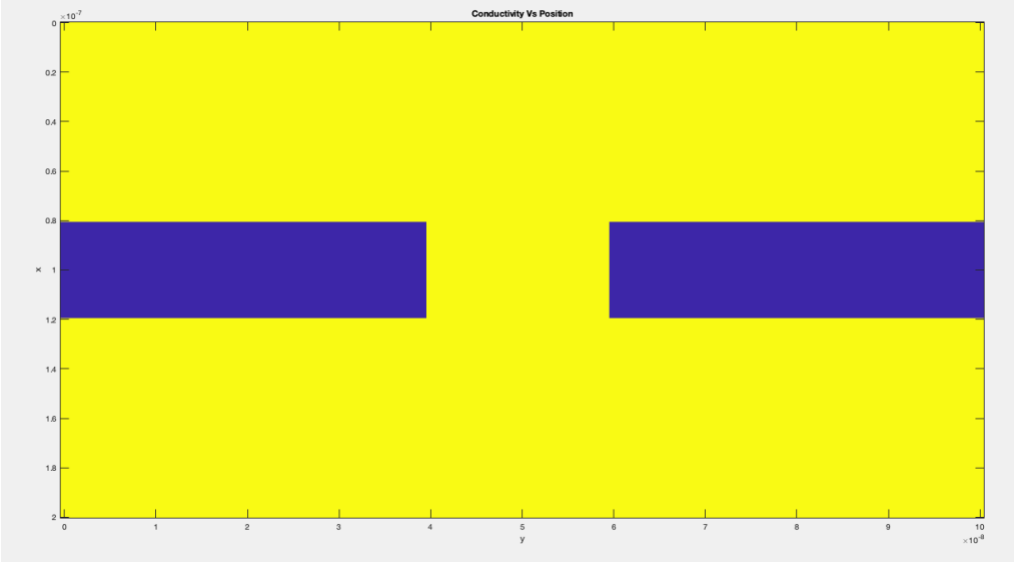

Figure 6: Conductivity Vs Position

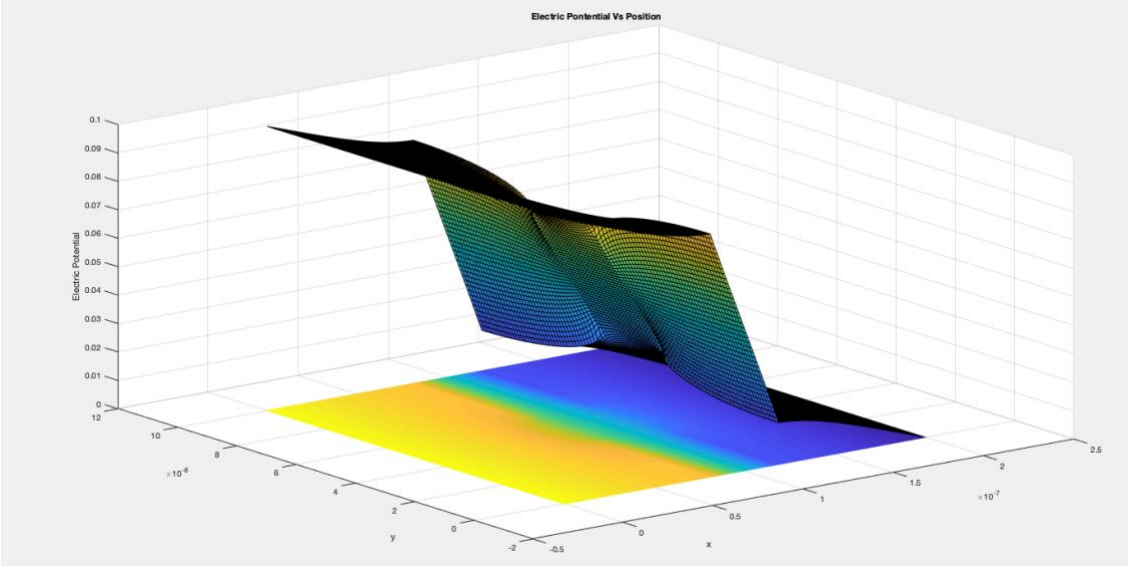

Figure 7: The Electric Potential Vs Position

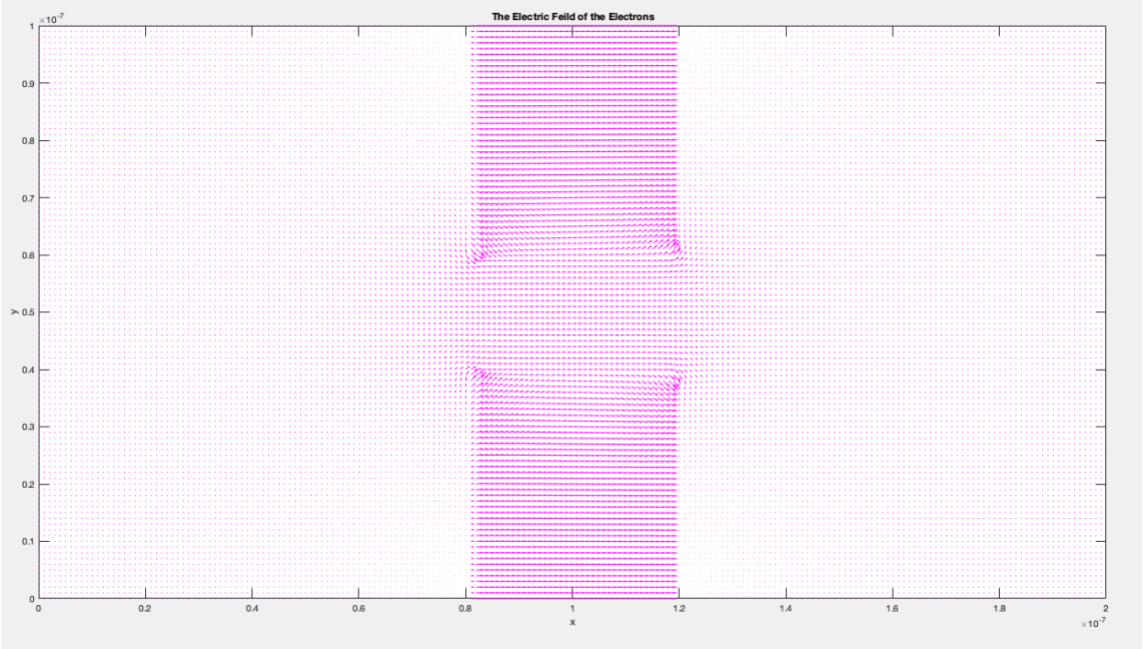

Figure 8: The Electric Field of the Electrons

Part 3:

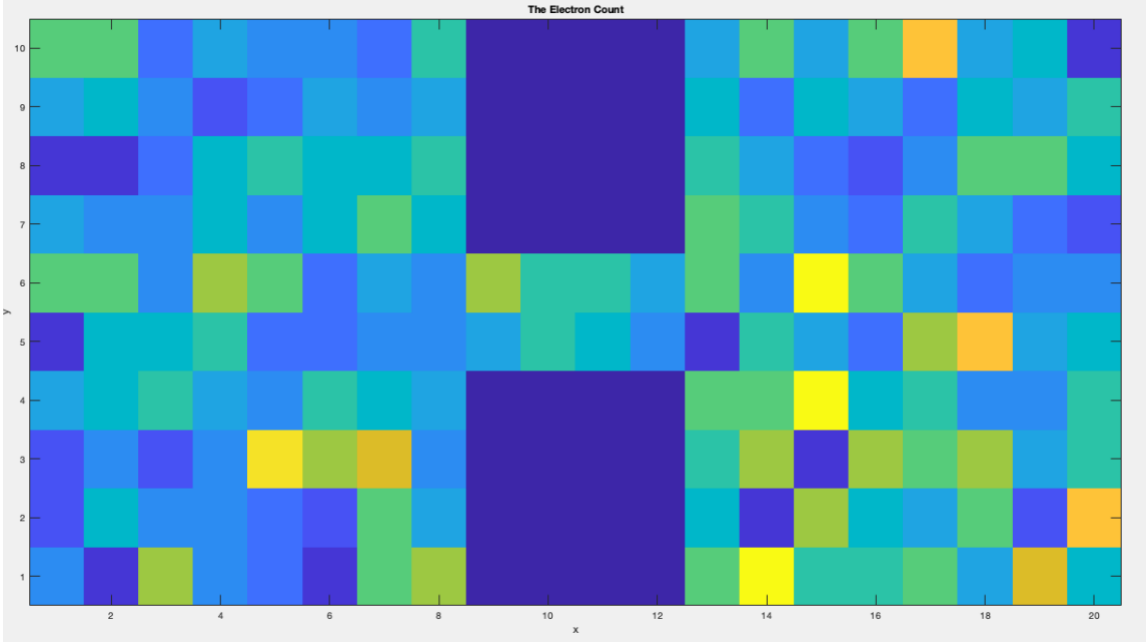

Figure 9: Density Mapping

From the above density mapping plot, it depicts where the electrons are most concentrated during the simulation.

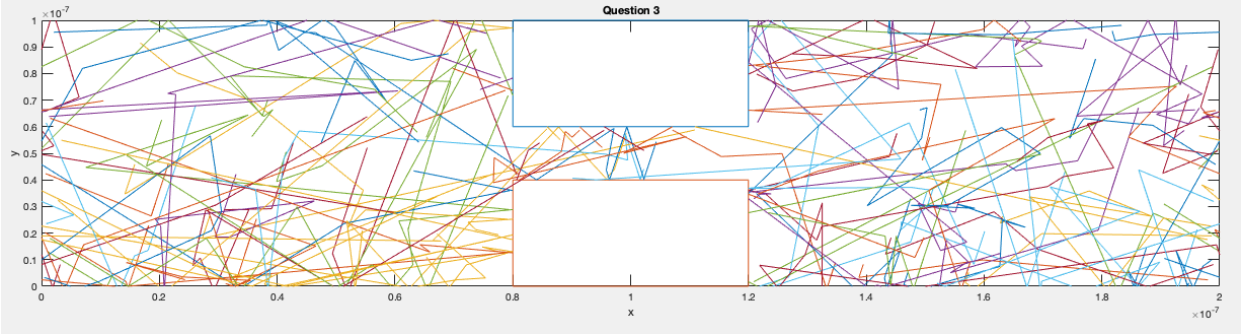

Figure 10: Particle trajectory

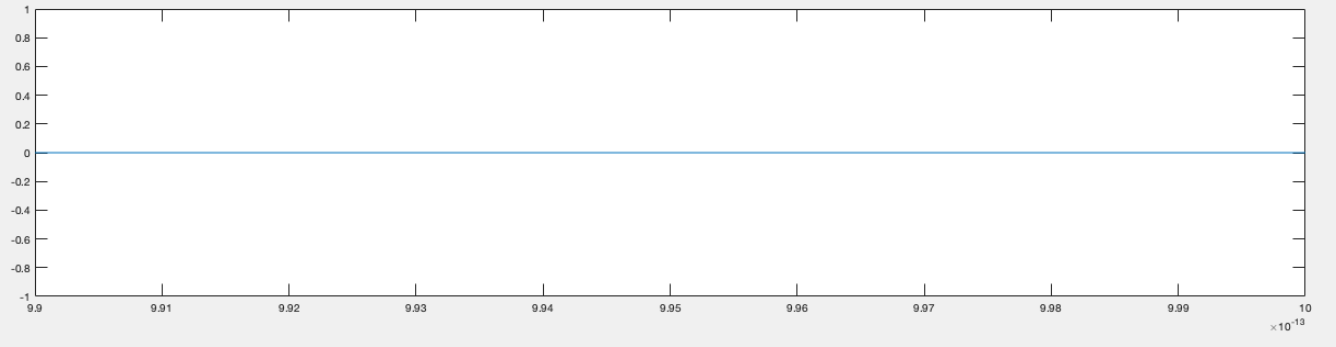

Figure 11: Temperature Vs Time Step

In an ideal situation, the Temperature vs Time step plot should display a fluctuation around a constant value.

The next step would be to make the simulation more accurate. Furthermore, this means to increase the number of electrons in the simulation, this would give a more accurate result. In addition, we could also finer mesh the simulation.

Conclusion

In conclusion, this Assignment 3 was a success. We were able to combine both simulations from Assignment 1 and 2, to observe what happens to the electrons. Furthermore, we observed the

relationship between the Electron drift, the Current density and the carrier velocity of the electrons.

Appendix

Question 1 and 2

```
% ELEC 4700
% Name: Oritseserundede Eda
% Student Number: 100993421
% Assignment 3
% Part 1
% We take the Monte-Carlo simulations done in Assignment 1 and we employ
% them withouth the Bottle-neck Constraints

clc
clear
set(0,'DefaultFigureWindowStyle','docked')


width = 200e-9; % width across the x-dimension
length = 100e-9; % length across the y-dimension
volt_x = 0.1; % voltage applied across the x-dimension
volt_y = 0; % voltage applied across the y-dimension
qcharge = -1.60217662e-19; % charge of electrons
charge_conc = 1e15*100^2; % Density of electrons
m0 = 9.10938356e-31; % the rest mass of the electrons
eff_mass = 0.26*m0; % the effective mass of electrons
tempt = 300; % the temperature in kelvin
b_constant = 1.38064852e-23; % the Boltzmann Constant
% The electrons thermal Velocity
therm_volt = sqrt(2 * b_constant * tempt / eff_mass);
% The Mean free path of the electrons
mfp = therm_volt*0.2e-12;
spec_tb = 0;
spec_bb = 0;
t_step = length/therm_volt/100;
iter = 300; % the number of iterations
p_size = 40000;
p_count = 10;
p_scat = 1 - exp(-t_step/0.2e-12);
vel = makedist('Normal', 'mu', 0, 'sigma', sqrt(b_constant*tempt/eff_mass));
```

```
display_m = 0;

% Calculating the Electric Field using the relationship between voltage
% and distance.
e_x = volt_x/width;
e_y = volt_y/length;
e_total = e_x + e_y;
fprintf('The Electric Field of the Charge is %f V/m.\n',e_total);
% The force on each electron is the sum of its individual components.
xforce = qcharge*e_x;
yforce = qcharge*e_y;
f_total = abs(xforce + yforce);
fprintf('The Force on the charge is %d N.\n',f_total);
% From the relationship f=ma to calculate the accelration of the particle.
acc = f_total/eff_mass;
fprintf('The acceleration of the Charge is %f m/s^2.\n',acc);

% Using the current formula J = vnqNy, to show the relationship
% between the electron drift with current density and the average carrier velocity
% to avoid MATLAB from performing complicated integrations.
vx = xforce*t_step/eff_mass;
vy = yforce*t_step/eff_mass;
vx = vx.*ones(p_size,1);
vy = vy.*ones(p_size,1);
pos = zeros(p_size, 4);
traj = zeros(iter, p_count*2);
temp_a = zeros(iter,1);
J = zeros(iter,2);

% Initializations for the positions for each particle
for i = 1:p_size
    theta = rand*2*pi;
    pos(i,:) = [width*rand length*rand random(vel) random(vel)];
end
tempt_plot = animatedline;
figure(2);
current_plot = animatedline;
title('Current Density');
xlabel('Time (s)');
ylabel('Current density (A/m)');

% Interations through the simulation for each of those particle positions
for i = 1:iter
    pos(:,3) = pos(:,3) + vx;
    pos(:,4) = pos(:,4) + vy;
    pos(:,1:2) = pos(:,1:2) + t_step.*pos(:,3:4);
```

```
j = pos(:,1) > width;
pos(j,1) = pos(j,1) - width;
j = pos(:,1) < 0;
pos(j,1) = pos(j,1) + width;
j = pos(:,2) > length;

if(spec_tb)
    pos(j,2) = 2*length - pos(j,2);
    pos(j,4) = -pos(j,4);
else
    pos(j,2) = length;
    v = sqrt(pos(j,3).^2 + pos(j,4).^2);
    theta = rand([sum(j),1])*2*pi;
    pos(j,3) = v.*cos(theta);
    pos(j,4) = -abs(v.*sin(theta));
end

j = pos(:,2) < 0;

if(spec_bb)
    pos(j,2) = -pos(j,2);
    pos(j,4) = -pos(j,4);
else
    pos(j,2) = 0;
    v = sqrt(pos(j,3).^2 + pos(j,4).^2);
    theta = rand([sum(j),1])*2*pi;
    pos(j,3) = v.*cos(theta);
    pos(j,4) = abs(v.*sin(theta));
end

j = rand(p_size, 1) < p_scat;
pos(j,3:4) = random(vel, [sum(j),2]);
temp_a(i) = (sum(pos(:,3).^2) + sum(pos(:,4).^2))*eff_mass/b_constant/2/p_size;

for j=1:p_count
    traj(i, (2*j):(2*j+1)) = pos(j, 1:2);
end

J(i, 1) = qcharge.*charge_conc.*mean(pos(:,3));
J(i, 2) = qcharge.*charge_conc.*mean(pos(:,4));
addpoints(tempt_plot, t_step.*i, temp_a(i));
addpoints(current_plot, t_step.*i, J(i,1));
if(display_m && mod(i,5) == 0)
    figure(1);
    hold off;
    plot(pos(1:p_count,1)./1e-9, pos(1:p_count,2)./1e-9, 'o');
```

```
        axis([0 width/1e-9 0 length/1e-9]);
        hold on;
        title('Particle Trajectories');
        xlabel('x-position');
        ylabel('y-position');
        pause(0.05);
    end
end

figure(1);
title('Particle Trajectories');
xlabel('x-position');
ylabel('y-position');
axis([0 width/1e-9 0 length/1e-9]);
hold on;

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

charge_conc = hist3(pos(:,1:2),[200 100])';
N = 20;
sigma = 1.5;
[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(3);
charge_conc = conv2(charge_conc,f,'same');
charge_conc = charge_conc/(length./size(charge_conc,1)*width./size(charge_conc,2));
surf(conv2(charge_conc,f,'same'));
title('Electron Density Map_counting');
xlabel('x Position');
ylabel('y Position');
sum_x = zeros(ceil(width/1e-9),ceil(length/1e-9));
sum_y = zeros(ceil(width/1e-9),ceil(length/1e-9));
temp_num = zeros(ceil(width/1e-9),ceil(length/1e-9));

% Electron Velocity
for i=1:p_size
    x = floor(pos(i,1)/1e-9);
    y = floor(pos(i,2)/1e-9);
    if(x==0)
        x = 1;
    end
    if(y==0)
        y= 1;
```

```
    end
    sum_y(x,y) = sum_y(x,y) + pos(i,3)^2;
    sum_x(x,y) = sum_x(x,y) + pos(i,4)^2;
    temp_num(x,y) = temp_num(x,y) + 1;
end
temp_a = (sum_x + sum_y).*eff_mass./b_constant./2./temp_num;
temp_a(isnan(temp_a)) = 0;
temp_a = temp_a';
N = 20;
sigma = 1.5;
[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(4);
surf(conv2(temp_a,f,'Same'));
title('Temperature Map-Count');
xlabel('x');
ylabel('y');



% Part 2:
% From the Finite Difference Method in Assignment 2, the Electric Field was
% calculated and then the Monte-Carlo bottlenecks are introduced to the
% field.

clc
clear
set(0,'DefaultFigureWindowStyle','docked')

length2 = 200e-9;
width2 = 100e-9;
length_box = 40e-9;
width_box = 40e-9;
meshspace = 1e-9;
num_x = round(length2/meshspace + 1);
num_y = round(width2/meshspace + 1);
outside_conduct = 1;
inside_conduct = 1e-2;
conduct_mapcount = zeros(num_x,num_y);

for i = 1:num_x
  for j = 1:num_y
     if (i-1)>0.5*(length2-length_box)/meshspace&&(i-
1)<0.5*(length2+length_box)/meshspace&&((j-1)<width_box/meshspace||(j-1)>(width2-
width_box)/meshspace)
```

```
            conduct_mapcount(i,j) = inside_conduct;
        else
            conduct_mapcount(i,j) = outside_conduct;
        end
    end
end

figure(5)
imagesc([0 width2],[0 length2],conduct_mapcount);
xlabel('y')
ylabel('x')
title('Conductivity Vs Position')

% The G and B Matrices
G = sparse(num_x*num_y);
B = zeros(1,num_x*num_y);
for i = 1:num_x
    for j = 1:num_y
        n = j +(i-1)*num_y;
        n1 = j + (i - 1) * length2;
        nxm1 = j + ((i-1) - 1) * length2;
        nxp1 = j + ((i+1) - 1) * length2;
        nym1 = (j-1) + (i - 1) * length2;
        nyp1 = (j+1) + (i - 1) * length2;

        if i == 1
        n1 = j + (i - 1) * length2;
        nxm1 = j + ((i-1) - 1) * length2;
        nxp1 = j + ((i+1) - 1) * length2;
        nym1 = (j-1) + (i - 1) * length2;
        nyp1 = (j+1) + (i - 1) * length2;
            G(n,n) = 1;
            B(n) = 0.1;

        elseif i == num_x
        n1 = j + (i - 1) * length2;
        nxm1 = j + ((i-1) - 1) * length2;
        nxp1 = j + ((i+1) - 1) * length2;
        nym1 = (j-1) + (i - 1) * length2;
        nyp1 = (j+1) + (i - 1) * length2;
        G(n,n) = 1;


        elseif j == 1
        n1 = j + (i - 1) * length2;
        nxm1 = j + ((i-1) - 1) * length2;
```

```
nxp1 = j + ((i+1) - 1) * length2;
nym1 = (j-1) + (i - 1) * length2;
nyp1 = (j+1) + (i - 1) * length2;
nxm = j + (i-2)*num_y;
nxp = j + i*num_y;
nyp = j+1 + (i-1)*num_y;
rxm = (conduct_mapcount(i,j) + conduct_mapcount(i-1,j))/2;
rxp = (conduct_mapcount(i,j) + conduct_mapcount(i+1,j))/2;
ryp = (conduct_mapcount(i,j) + conduct_mapcount(i,j+1))/2;

    G(n,n) = -(rxm + rxp + ryp);
    G(n,nxm) = rxm;
    G(n,nxp) = rxp;
    G(n,nyp) = ryp;


elseif j == num_y
    nxm = j + (i-2)*num_y;
    nxp = j + i*num_y;
    nym = j-1 + (i-1)*num_y;
    n1 = j + (i - 1) * length2;
    nxm1 = j + ((i-1) - 1) * length2;
    nxp1 = j + ((i+1) - 1) * length2;
    nym1 = (j-1) + (i - 1) * length2;
    nyp1 = (j+1) + (i - 1) * length2;

    rxm = (conduct_mapcount(i,j) + conduct_mapcount(i-1,j))/2;
    rxp = (conduct_mapcount(i,j) + conduct_mapcount(i+1,j))/2;
    rym = (conduct_mapcount(i,j) + conduct_mapcount(i,j-1))/2;

    G(n,n) = -(rxm + rxp + rym);
    G(n,nxm) = rxm;
    G(n,nxp) = rxp;
    G(n,nym) = rym;

else
    nxm = j + (i-2)*num_y;
    nxp = j + i*num_y;
    nym = j-1 + (i-1)*num_y;
    nyp = j+1 + (i-1)*num_y;
    n1 = j + (i - 1) * length2;
nxm1 = j + ((i-1) - 1) * length2;
nxp1 = j + ((i+1) - 1) * length2;
nym1 = (j-1) + (i - 1) * length2;
nyp1 = (j+1) + (i - 1) * length2;
    rxm = (conduct_mapcount(i,j) + conduct_mapcount(i-1,j))/2;
```

```
            rxp = (conduct_mapcount(i,j) + conduct_mapcount(i+1,j))/2;
            ryp = (conduct_mapcount(i,j) + conduct_mapcount(i,j+1))/2;
            rym = (conduct_mapcount(i,j) + conduct_mapcount(i,j-1))/2;

            G(n,n) = -(rxm + rxp + rym + ryp);
            G(n,nxm) = rxm;
            G(n,nxp) = rxp;
            G(n,nym) = rym;
            G(n,nyp) = ryp;

        end
    end
end
V = G\B';
Voltage_map = zeros(num_x,num_y);
for i = 1:num_x
    for j = 1:num_y
        n = j +(i-1)*num_y;
        Voltage_map(i,j) = V(n);
    end
end
[X, Y] = meshgrid(0:meshspace:length2,0:meshspace:width2);
figure(6)
surf(X',Y',Voltage_map)
hold on
imagesc([0 length2],[0 width2],Voltage_map')
xlabel('x')
ylabel('y')
zlabel('Electric Potential')
title('Electric Pontential Vs Position')
hold off

[e_y, e_x] = gradient(Voltage_map,meshspace);
e_x = -e_x;
e_y = -e_y;

figure(7)
quiver(X',Y',e_x,e_y, 'm')
xlim([0 length2])
ylim([0 width2])
xlabel('x')
ylabel('y')
title('The Electric Feild of the Electrons')
```

Question 3

```
% ELEC 4700
% Name: Oritseserundede Eda
% Student Number: 100993421
% Assignment 3
% Part 3
% In this part, we use the coupled simulations for the device and
% trajectory investigations

clc
clear
set(0,'DefaultFigureWindowStyle','docked')

global C
global Vtotal Vx Vy x y
global Ecount
Ecount =1000;
C.mo = 9.10938215e-31;
C.k = 1.3806504e-23;
qcharge = -1.60217662e-19;
temp =300;
eff_mass = 0.26*C.mo;
length = 200e-9;
width = 100e-9;
therm_volt = sqrt((2*C.k*temp)/eff_mass);
t_step = 10e-15;
frame = 100*t_step;
x = zeros(Ecount, 2);
y = zeros(Ecount, 2);
temp = zeros(1,2);
Time = 0;
visible_ecount = 50;
tmn = 0.2e-12;
PScat = 1 - exp(-t_step/tmn);
V_Histogram = zeros(Ecount, 1);
b_x = [80e-9 80e-9 120e-9 120e-9 80e-9];

b_y1 = [100e-9 60e-9 60e-9 100e-9 100e-9];
b_y2 = [40e-9 0 0 40e-9 40e-9];
spec = true;
inside_box = true;
s_map = 10e-9;
d_map = zeros(width/s_map, length/s_map);
t_map = zeros(width/s_map, length/s_map);

wid_x = 30;
len_y = 20;
```

```
change_x = length/wid_x;
change_y = width/len_y;
outside_conduct = 1;
inside_conduct = 01e-2;
conductivity = zeros(wid_x,len_y);
G = sparse (wid_x*len_y, wid_x*len_y);
V = zeros(1, wid_x*len_y);
volt_x = 0.1;

for i = 1:wid_x
   for j = 1:len_y
      n = j + (i - 1)*len_y;
      nxm = j + ((i-1) - 1)*len_y;
      nxp = j + ((i+1) - 1)*len_y;
      nym = (j-1) + (i - 1)*len_y;
      nyp = (j+1) + (i - 1)*len_y;
      if (i > (0.3*wid_x) || i < (0.6*wid_x)) && (j > (0.6*len_y) || j < (0.3*len_y))
         conductivity(i,j) = inside_conduct;
      else
         conductivity(i,j) = outside_conduct;
      end

   end
end
for i = 1:wid_x
   for j = 1:len_y
      n = j + (i - 1)*len_y;
      nxm = j + ((i-1) - 1)*len_y;
      nxp = j + ((i+1) - 1)*len_y;
      nym = (j-1) + (i - 1)*len_y;
      nyp = (j+1) + (i - 1)*len_y;
      if (i == 1)
      n = j + (i - 1)*len_y;
      nxm = j + ((i-1) - 1)*len_y;
      nxp = j + ((i+1) - 1)*len_y;
      nym = (j-1) + (i - 1)*len_y;
      nyp = (j+1) + (i - 1)*len_y;
         V(n) = volt_x;
         G(n,n) = 1;
      elseif (i == wid_x)
      n = j + (i - 1)*len_y;
      nxm = j + ((i-1) - 1)*len_y;
      nxp = j + ((i+1) - 1)*len_y;
      nym = (j-1) + (i - 1)*len_y;
      nyp = (j+1) + (i - 1)*len_y;
         V(n) = 0;
```

```
            G(n,n) =1;
        elseif (j == 1)
        n = j + (i - 1)*len_y;
        nxm = j + ((i-1) - 1)*len_y;
        nxp = j + ((i+1) - 1)*len_y;
        nym = (j-1) + (i - 1)*len_y;
        nyp = (j+1) + (i - 1)*len_y;
            G(n,n) = -((conductivity(i,j) + conductivity(i-1,j))/2) - ((conductivity(i,j) +
conductivity(i+1,j))/2) - ((conductivity(i,j) + conductivity(i,j+1))/2);
            G(n, nxm) = ((conductivity(i,j) + conductivity(i-1,j))/2);
            G(n,nxp) = ((conductivity(i,j) + conductivity(i+1,j))/2);
            G(n, nyp) = ((conductivity(i,j) + conductivity(i,j+1))/2);
        elseif (j == len_y)
        n = j + (i - 1)*len_y;
        nxm = j + ((i-1) - 1)*len_y;
        nxp = j + ((i+1) - 1)*len_y;
        nym = (j-1) + (i - 1)*len_y;
        nyp = (j+1) + (i - 1)*len_y;
            G(n,n) = -((conductivity(i,j) + conductivity(i-1,j))/2) - ((conductivity(i,j) +
conductivity(i+1,j))/2) - ((conductivity(i,j) + conductivity(i,j-1))/2);
            G(n,nxm) = ((conductivity(i,j) + conductivity(i-1,j))/2);
            G(n,nxp) = ((conductivity(i,j) + conductivity(i+1,j))/2);
            G(n,nym) = ((conductivity(i,j) + conductivity(i,j-1))/2);
        else
        n = j + (i - 1)*len_y;
        nxm = j + ((i-1) - 1)*len_y;
        nxp = j + ((i+1) - 1)*len_y;
        nym = (j-1) + (i - 1)*len_y;
        nyp = (j+1) + (i - 1)*len_y;
            G(n,n) = -((conductivity(i,j) + conductivity(i-1,j))/2) - ((conductivity(i,j) +
conductivity(i+1,j))/2) - ((conductivity(i,j) + conductivity(i,j-1))/2) - ((conductivity(i,j) +
conductivity(i,j+1))/2);
            G(n,nxm) = ((conductivity(i,j) + conductivity(i-1,j))/2);
            G(n,nxp) = ((conductivity(i,j) + conductivity(i+1,j))/2);
            G(n,nym) = ((conductivity(i,j) + conductivity(i,j-1))/2);
            G(n,nyp) = ((conductivity(i,j) + conductivity(i,j+1))/2);
        end
    end
end

soln = G\V';
surf = zeros(wid_x,len_y);
for i = 1:wid_x
    for j = 1:len_y
        n = j + (i - 1)*len_y;
        nxm = j + ((i-1) - 1) * len_y;
```

```
        nxp = j + ((i+1) - 1) * len_y;
        nym = (j-1) + (i - 1) * len_y;
        nyp = (j+1) + (i - 1) * len_y;
        surf(i,j) = soln(n);
    end
end
[E_x, E_y] = gradient(-surf);
Force_x = qcharge*E_x;
Force_y = qcharge*E_y;
Acceleration_x = Force_x /eff_mass;
Acceleration_y = Force_y /eff_mass;

for i = 1:Ecount
    x(i,1) = rand()*200e-9;
    y(i,1) = rand()*100e-9;
    inside_box = true;
    while inside_box == true
        if (x(i) >= 40e-9 && x(i) <= 120e-9) && (y(i) >= 60e-9 ||...
                y(i) <= 40e-9)
            x(i,1) = rand * 200e-9;
            y(i,1) = rand * 100e-9;
        else
            inside_box = false;
        end
    end

end

for i = 1:Ecount

Vx(1:Ecount) = therm_volt * randn;
Vy(1:Ecount) = therm_volt * randn;
end

figure(8)
subplot(2,1,1);
plot(b_x, b_y1, b_x, b_y2)
axis([0 length 0 width]);
title('Question 3');
xlabel('x');
ylabel('y');
hold on;

while Time < frame
    subplot(2,1,1)
    for j = 1:Ecount
```

```
leaking = true;
if PScat> rand
        Vx(j) = therm_volt * randn;
        Vy(j) = therm_volt * randn;
end
x_index = round((x(j,2)/length) * 30);
y_index = round((y(j,2)/width)*20);
if x_index < 1
    x_index = 1;
elseif x_index > 30
        x_index = 30;
end
if y_index < 1
    y_index = 1;
elseif y_index > 20
    y_index = 20;
end

Vx(j) =  Vx(j) + Acceleration_x(x_index,y_index)*t_step;
Vy(j) =  Vy(j) + Acceleration_y(x_index,y_index)*t_step;
x(j,2) = x(j,1);
y(j,2) = y(j,1);
x(j,1) = x(j,1) + (t_step * Vx(j));
y(j,1) = y(j,1) + (t_step * Vy(j));

if (x(j,1) >= 80e-9 && x(j,1) <= 120e-9) && y(j,1) >= 60e-9

        if y(j,2) < 60e-9
            Vy(j) = -Vy(j);
            y(j,1) = 60e-9;
            y(j,2) = 60e-9;

        elseif x(j,2) < 80e-9
            Vx(j) = -Vx(j);
            x(j,1) = 80e-9;
            x(j,2) = 80e-9;

        elseif x(j,2) > 120e-9
            Vx(j) = -Vx(j);
            x(j,1) = 120e-9;
            x(j,2) = 120e-9;
        end

    if spec == true

        x(j,1) = x(j,2) + Vx(j)*t_step;
```

```
        y(j,1) = y(j,2) + Vy(j)*t_step;
    else

     Vx(j) = therm_volt * randn;
     Vy(j) = therm_volt * randn;

     while leaking == true
        if(x(j,2) < 80e-9 && Vx(j) >= 0) || ...
              (x(j,2) > 120e-9 && Vx(j) <= 0) || ...
              (y(j,2) < 60e-9 && Vy(j) >= 0)
           Vx(j) = therm_volt * randn;
           Vy(j) = therm_volt * randn;
        else
           leaking = false;
        end
     end
     x(j,1) = x(j,2) + Vx(j)*t_step;
     y(j,1) = y(j,2) + Vy(j)*t_step;
    end
end
if (x(j,1) >= 80e-9 && x(j,1) <= 120e-9) && y(j,1) <= 40e-9
       if y(j,2) > 40e-9
          Vy(j) = -Vy(j);
          y(j,1) = 40e-9;
          y(j,2) = 40e-9;
       elseif x(j,2) < 80e-9
          Vx(j) = -Vx(j);
          x(j,1) = 80e-9;
          x(j,2) = 80e-9;
       elseif x(j,2) > 120e-9
          Vx(j) = -Vx(j);
          x(j,1) = 120e-9;
          x(j,2) = 120e-9;
       end
    if spec == true
       x(j,1) = x(j,2) + Vx(j)*t_step;
       y(j,1) = y(j,2) + Vy(j)*t_step;
    else
     Vx(j) = therm_volt * randn;
     Vy(j) = therm_volt * randn;
     while leaking == true
        if(x(j,2) < 80e-9 && Vx(j) >= 0) || ...
              (x(j,2) > 120e-9 && Vx(j) <= 0) || ...
              (y(j,2) > 40e-9 && Vy(j) <= 0)
           Vx(j) = therm_volt * randn;
           Vy(j) = therm_volt * randn;
```

```
                else
                    leaking = false;
                end
            end
            x(j,1) = x(j,2) + Vx(j)*t_step;
            y(j,1) = y(j,2) + Vy(j)*t_step;
            end
        end
        if x(j,1) > length
            x(j,2) = 0;
            x(j,1) = t_step * Vx(j);
        end
        if x(j,1) < 0
            x(j,2) = length;
            x(j,1) = x(j,2) + (t_step * Vx(j));
        end
        if y(j,1) > width || y(j,1) < 0
            Vy(j) = -Vy(j);
        end
        XPlot = [x(j,2) x(j,1)];
        YPlot = [y(j,2) y(j,1)];
        if j < visible_ecount
        plot(XPlot,YPlot);
        end

      VTotal = sqrt(Vx(j)^2 + Vy(j)^2);
    end

    average_temp = temp(1,2)/Ecount;
    temp_plot = [temp(1,1) average_temp];
    time_plot = [(Time - t_step) Time];
    subplot(2,1,2);
    plot(time_plot, temp_plot);
    temp(1,1) = average_temp;
    average_temp = 0;
    temp(1,2) = 0;
    pause(1e-19)
    Time = Time + t_step;
end
for i = 1:(length/s_map)
    for j = 1:(width/s_map)
        for m = 1:Ecount
            if(x(m,1) > s_map*(i -1)) && ...
                    (x(m,1) < s_map*(i)) && ...
                    (y(m,1) > s_map*(j - 1)) && ...
                    (y(m,1) < s_map*(j))
```

```
            Vtotal(m) = sqrt(Vx(m)^2 + Vy(m)^2);

            d_map(j, i) = d_map(j, i) + 1;
            t_map(j, i) = t_map(j,i) + ...
                (eff_mass*Vtotal(m)^2)/(2*C.k);
        end
        t_map(j,i) = t_map(j,i)/d_map(j,i);
    end
  end
end
figure(9)
imagesc(d_map)
title('The Density-Mapping of all electrons in the frame')
xlabel('x');
ylabel('y');
set(gca, 'Ydir', 'Normal')
title('The Electron Count')
```