Elec4700A
Assignment #1

Khaled AbouShaban
101042658

Due Date: Feb. 7th, 2021

Table of Contents

Part 1: Electron Modelling,

Q2) Calculating the mean free path,
vthev = 1.8702e+05 = 187 km/s
I = Vthev*0.2e-12 = 3.7404e-08 = 37.4 nm
Therefore, the mean free path is approximately 37.4 nm.

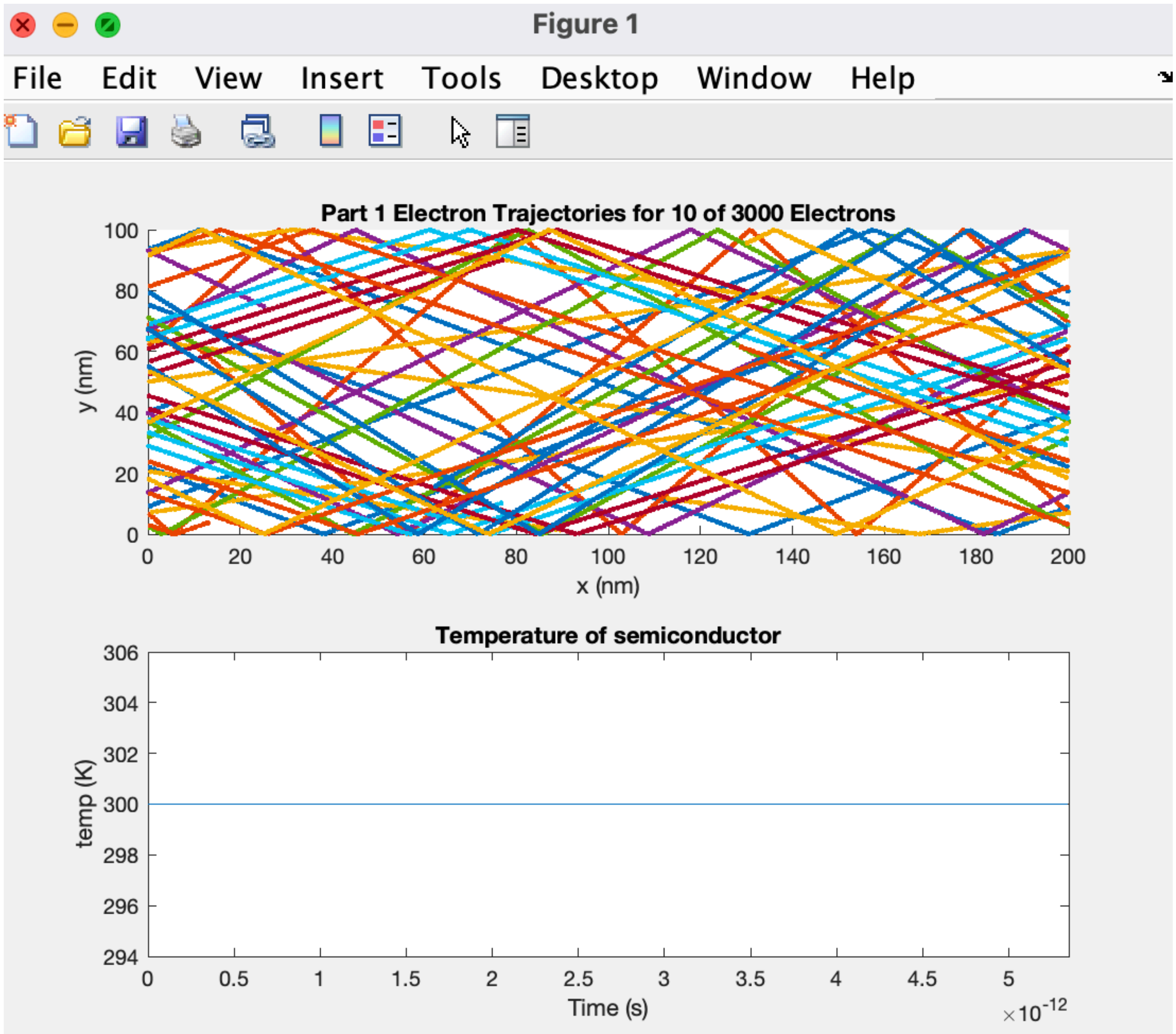
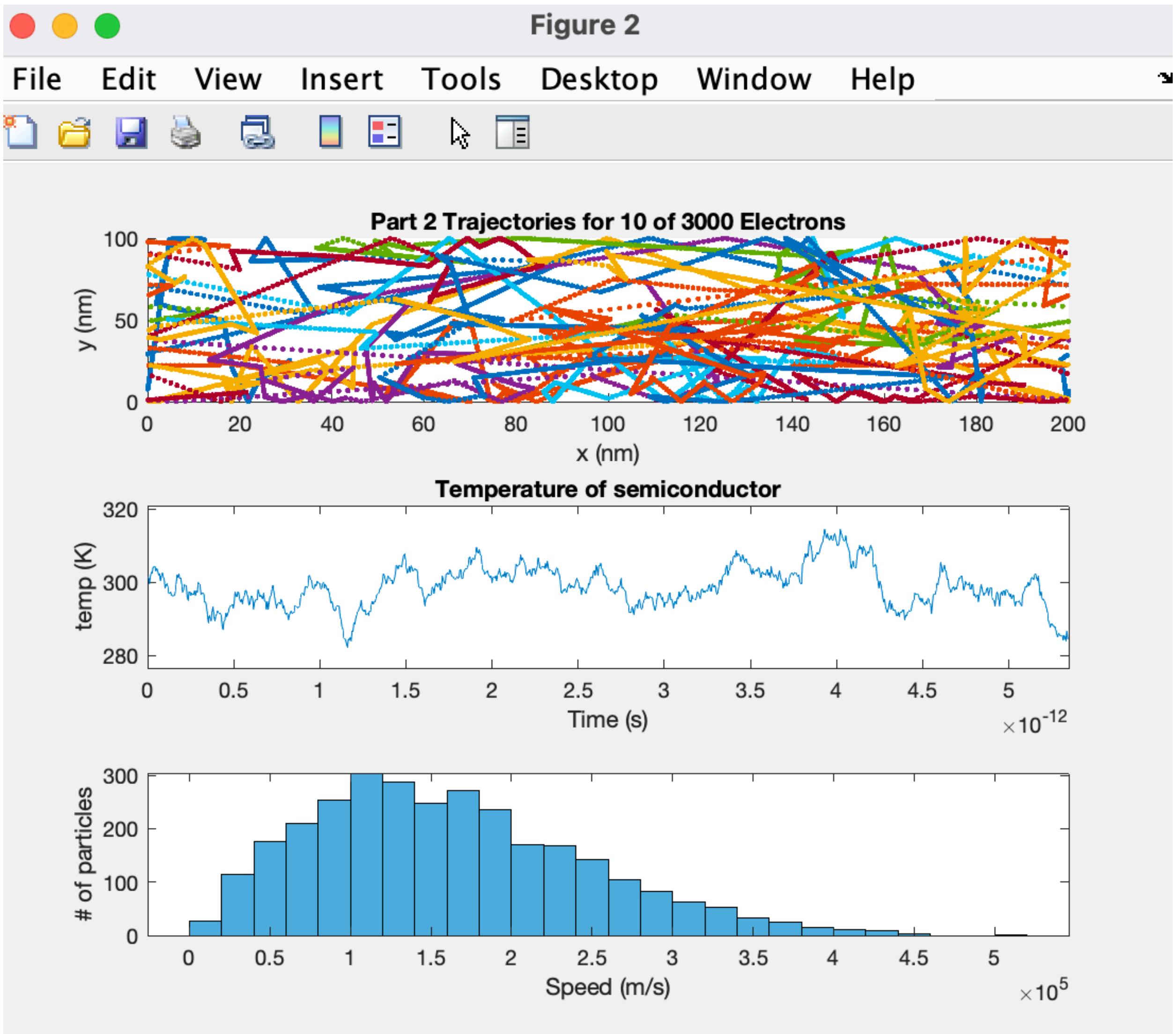


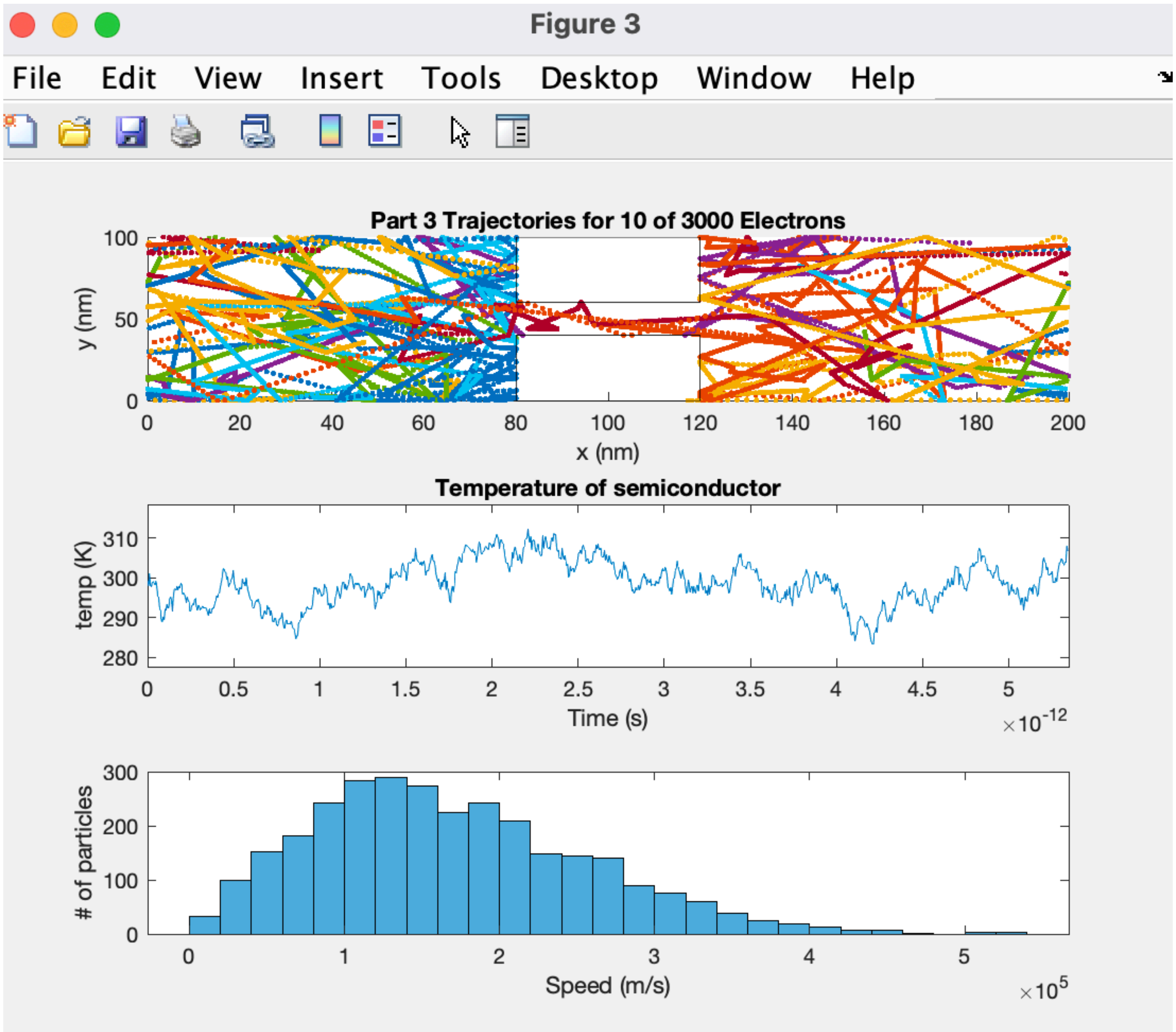
Figure 1
File Edit View Insert Tools Desktop Window Help
Part 1 Electron Trajectories for 10 of 3000 Electrons
y (nm)
x (nm)
0 20 40 60 80 100 120 140 160 180 200
0 20 40 60 80 100
Temperature of semiconductor
temp (K)
294 296 298 300 302 304 306
Time (s)
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5
×10^-12

Part 2: Collisions with Mean Free Path (MFP)

Q3) The average temperature fluctuates over time due to the scattering, but it maintains an average of 300 K over time as shown below.



Part 3: Enhancements,



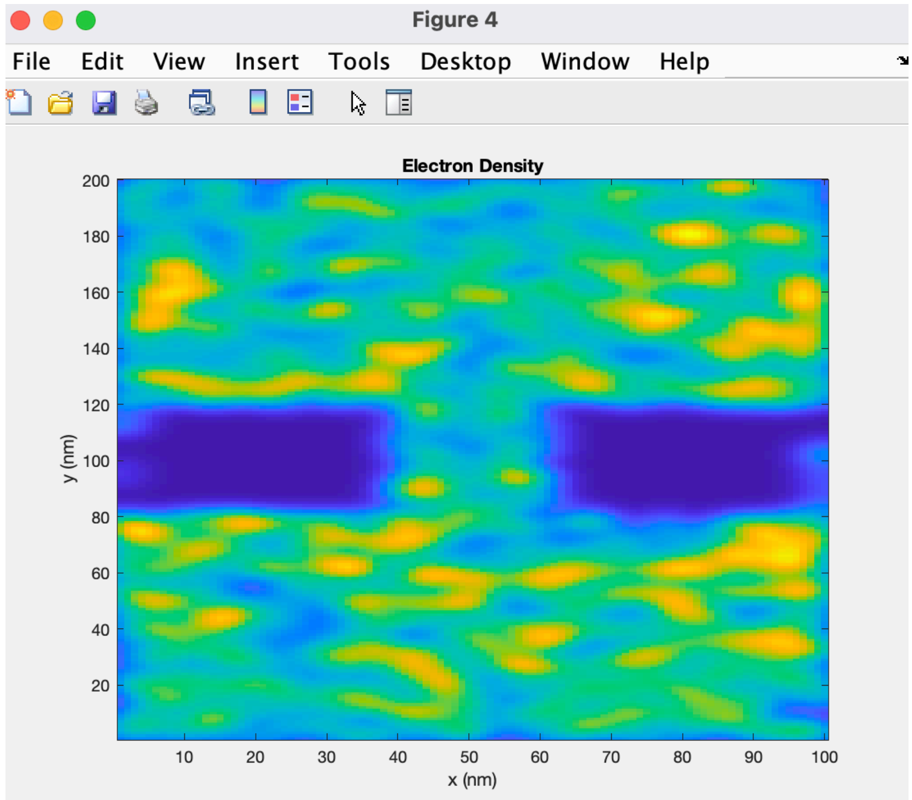


Figure 4
File Edit View Insert Tools Desktop Window Help
Electron Density
y (nm)
x (nm)
200
180
160
140
120
100
80
60
40
20
10
20
30
40
50
60
70
80
90
100

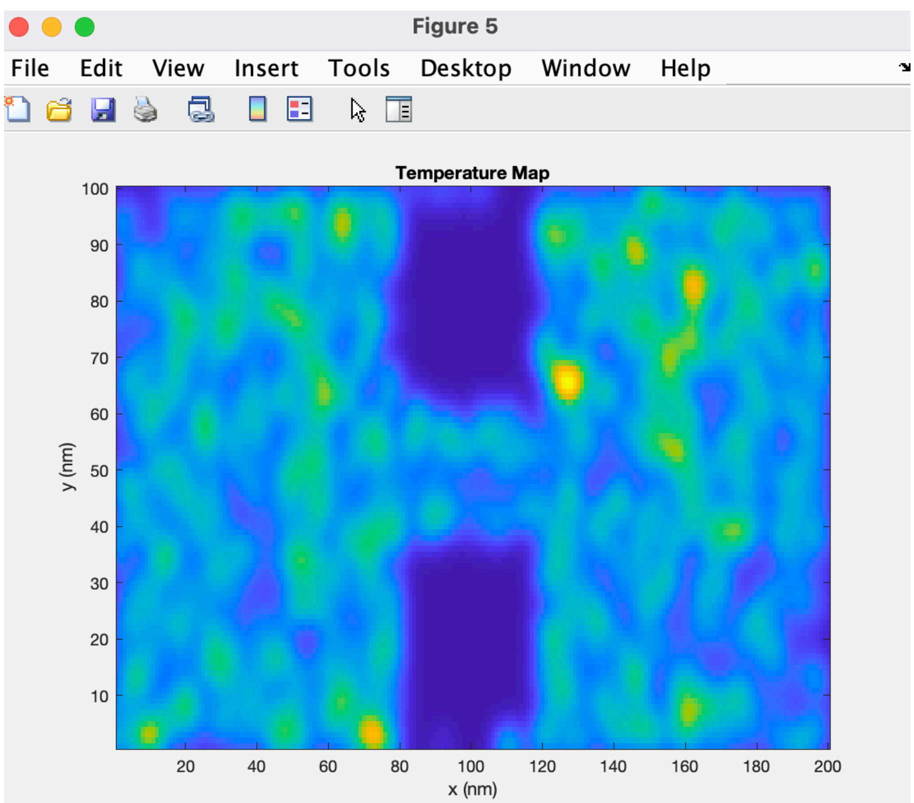


Figure 5
File Edit View Insert Tools Desktop Window Help
Temperature Map
y (nm)
x (nm)
100
90
80
70
60
50
40
30
20
10
20
40
60
80
100
120
140
160
180
200

Part 4: Code,

```
%Khaled AbouShaban 101042658
%Assignment 1 for Elec4700, Monte-Carlo Modeling of Electron Transport
%Part 1 Electron Modelling

clear all
close all

m0 = 9.10938356e-31; %electron mass
k = 1.38064852e-23; %Boltzmann's constant
T = 300; %Temp in Kelvin
m = 0.26*m0; %Effective mass of electron

vthev = sqrt(2*k*T/m)

height = 100e-9;
len = 200e-9;
popul_size = 3000;
popul_plot = 10;

time_step = height/vthev/100;

iters = 1000;

% Set value to 1 watch the motion
% Set value to 0 to see the plots
show_motion= 0;

% Each row maps electron with positions/velocities [x y vx vy]
state = zeros(popul_size, 4);
trajectories = zeros(iters, popul_plot*2);
temp = zeros(iters,1);
for i = 1:popul_size
    angle = rand*2*pi;
    state(i,:) = [len*rand height*rand vthev*cos(angle) vthev*sin(angle)];
end

for i = 1:iters
    state(:,1:2) = state(:,1:2) + time_step.*state(:,3:4);
    % Search for collisions along boundaries
    j = state(:,1) > len;
    state(j,1) = state(j,1) - len;
    j = state(:,1) < 0;
    state(j,1) = state(j,1) + len;
    j = state(:,2) > height;
    state(j,2) = 2*height - state(j,2);
    state(j,4) = -state(j,4);
    j = state(:,2) < 0;
    state(j,2) = -state(j,2);
    state(j,4) = -state(j,4);
    temp(i) = (sum(state(:,3).^2) + sum(state(:,4).^2))*m/k/2/popul_size;

    % Record trajectories
    for j=1:popul_plot
```

```
        trajectories(i, (2*j):(2*j+1)) = state(j, 1:2);
    end

    % Updating motion every 5 iters
    if show_motion && mod(i,5) == 0
        figure(1);
        subplot(2,1,1);
        hold off;
        plot(state(1:popul_plot,1)./1e-9,state(1:popul_plot,2)./1e-9, 'o');
        axis([0 len/1e-9 0 height/1e-9]);
        title(sprintf('Part 1 Electron Trajectories for %d of %d
Electrons',popul_plot, popul_size));
        xlabel('x(nm)');
        ylabel('y(nm)');
        if i > 1
            subplot(2,1,2);
            hold off;
            plot(time_step*(0:i-1), temp(1:i));
            axis([0 time_step*iters min(temp)*0.98
                max(temp)*1.02]);
            title('Temperature of semiconductor');
            xlabel('Time(s)');
            ylabel('Temperature(K)');
        end
        pause(0.05);
    end
end


% Plotting trajectories and subplot of temp
figure(1);
subplot(2,1,1);
title(sprintf('Part 1 Electron Trajectories for %d of %d
Electrons',popul_plot, popul_size));
xlabel('x (nm)');
ylabel('y (nm)');
axis([0 len/1e-9 0 height/1e-9]);
hold on;
for i=1:popul_plot
    plot(trajectories(:,i*2)./1e-9, trajectories(:,i*2+1)./1e-9, '.');
end
if(~show_motion)
    subplot(2,1,2);
    hold off;
    plot(time_step*(0:iters-1), temp);
    axis([0 time_step*iters min(temp)*0.98 max(temp)*1.02]);
    title('Temperature of semiconductor');
    xlabel('Time (s)');
    ylabel('temp (K)');
end



%_______________________________________________________________________


%
```

```matlab
%PART 2 Collisions with Mean Free Path (MFP)
%
scatter_p = 1 - exp(-time_step/0.2e-12)

pdf_v = makedist('Normal', 'mu', 0, 'sigma', sqrt(k*T/m));
for i = 1:popul_size
    angle = rand*2*pi;
    state(i,:) = [len*rand height*rand random(pdf_v) random(pdf_v)];
end
avg_v = sqrt(sum(state(:,3).^2)/popul_size + sum(state(:,4).^2)/popul_size)

for i = 1:iters

    %Updating positions
    state(:,1:2) = state(:,1:2) + time_step.*state(:,3:4);
    j = state(:,1) > len;
    state(j,1) = state(j,1) - len;
    j = state(:,1) < 0;
    state(j,1) = state(j,1) + len;
    j = state(:,2) > height;
    state(j,2) = 2*height - state(j,2);
    state(j,4) = -state(j,4);
    j = state(:,2) < 0;
    state(j,2) = -state(j,2);
    state(j,4) = -state(j,4);
    % Scatter particles
    j = rand(popul_size, 1) < scatter_p;
    state(j,3:4) = random(pdf_v, [sum(j),2]);

    % Recording temp
    temp(i) = (sum(state(:,3).^2) + sum(state(:,4).^2))*m/k/2/ popul_size;

    % Record positions
    for j=1:popul_plot
        trajectories(i, (2*j):(2*j+1)) = state(j, 1:2);
    end

    % Updating the motion for 5 iters
    if show_motion && mod(i,5) == 0
        figure(2);
        subplot(3,1,1);
        hold off;
        plot(state(1:popul_plot,1)./1e-9, state(1:popul_plot,2)./1e-9, 'o');
        axis([0 len/1e-9 0 height/1e-9]);
        title(sprintf('Part 2 Trajectories for %d of %d Electrons',
popul_plot, popul_size));
        xlabel('x (nm)');
        ylabel('y (nm)');
        if i > 1
            subplot(3,1,2);
            hold off;
            plot(time_step*(0:i-1), temp(1:i));
            axis([0 time_step*iters min(temp)*0.98
                max(temp)*1.02]);
            title('Temperature of semiconductor');
            xlabel('Time (s)');
            ylabel('temp (K)');
```

```
        end
        % Show histogram of speeds
        subplot(3,1,3);
        v = sqrt(state(:,3).^2 + state(:,4).^2);
        title('Histogram of Electron Speeds');
        histogram(v);
        xlabel('Speed (m/s)');
        ylabel('# of particles');
        pause(0.05);
    end
end

% Show trajectories after the motion
figure(2);
subplot(3,1,1);
title(sprintf('Part 2 Trajectories for %d of %d Electrons',popul_plot,
popul_size));
xlabel('x (nm)');
ylabel('y (nm)');
axis([0 len/1e-9 0 height/1e-9]);
hold on;
for i=1:popul_plot
    plot(trajectories(:,i*2)./1e-9, trajectories(:,i*2+1)./1e-9, '.');
end

% Show temp plot/time
if(~show_motion)
    subplot(3,1,2);
    hold off;
    plot(time_step*(0:iters-1), temp);
    axis([0 time_step*iters min(temp)*0.98 max(temp)*1.02]);
    title('Temperature of semiconductor');
    xlabel('Time (s)');
    ylabel('temp (K)');
end
% Show speed histogram
subplot(3,1,3);
v = sqrt(state(:,3).^2 + state(:,4).^2);
title('Histogram of Electron Speeds');
histogram(v);
xlabel('Speed (m/s)');
ylabel('# of particles');


%_________________________________________________________________________

%
%Part 3 Enhancments
%

specular_upper = 0;
specular_lower = 0;
boxes = 1e-9.*[80 120 0 40; 80 120 60 100];
boxes_specular = [0 1];
% Generating the initial popul
for i = 1:popul_size
    angle = rand*2*pi;
```

```
    state(i,:) = [len*rand height*rand random(pdf_v) random(pdf_v)];
    % Check for Box Contents
    while(for_box(state(i,1:2), boxes))
        state(i,1:2) = [len*rand height*rand];
    end
end

for i = 1:iters
    state(:,1:2) = state(:,1:2) + time_step.*state(:,3:4);
    j = state(:,1) > len;
    state(j,1) = state(j,1) - len;
    j = state(:,1) < 0;
    state(j,1) = state(j,1) + len;
    j = state(:,2) > height;
    if(specular_upper)
        state(j,2) = 2*height - state(j,2);
        state(j,4) = -state(j,4);
    else
        % The electrons bouncing off at a random angle
        state(j,2) = height;
        v = sqrt(state(j,3).^2 + state(j,4).^2);
        angle = rand([sum(j),1])*2*pi;
        state(j,3) = v.*cos(angle);
        state(j,4) = -abs(v.*sin(angle));
    end
    j = state(:,2) < 0;
    if(specular_lower)
        state(j,2) = -state(j,2);
        state(j,4) = -state(j,4);
    else % Diffusive
        % The electron bounces off at a random angle
        state(j,2) = 0;
        v = sqrt(state(j,3).^2 + state(j,4).^2);
        angle = rand([sum(j),1])*2*pi;
        state(j,3) = v.*cos(angle);
        state(j,4) = abs(v.*sin(angle));
    end

    % Moving electrons to their positions after entering box
    for j=1:popul_size
        box_num = for_box(state(j,1:2), boxes);
        while(box_num ~= 0)

            % Finding the electron collision side
            distance_x = 0;
            updated_x = 0;
            if(state(j,3) > 0)
                distance_x = state(j,1) - boxes(box_num,1);
                updated_x = boxes(box_num,1);
            else
                distance_x = boxes(box_num,2) - state(j,1);
                updated_x = boxes(box_num,2);
            end
            distance_y = 0;
            updated_y = 0;
            if(state(j,4) > 0)
                distance_y = state(j,2) - boxes(box_num, 3);
                updated_y = boxes(box_num, 3);
```

```
        else
            distance_y = boxes(box_num, 4) - state(j,2);
            updated_y = boxes(box_num, 4);
        end

        if(distance_x < distance_y)
            state(j,1) = updated_x;
            if(~boxes_specular(box_num))
                sgn = -sign(state(j,3));
                v = sqrt(state(j,3).^2 + state(j,4).^2);
                angle = rand()*2*pi;
                state(j,3) = sgn.*abs(v.*cos(angle));
                state(j,4) = v.*sin(angle);
            else
                state(j,3) = -state(j,3);
            end

        else
            state(j,2) = updated_y;
            if(~boxes_specular(box_num))
                sgn = -sign(state(j,4));
                v = sqrt(state(j,3).^2 + state(j,4).^2);
                angle = rand()*2*pi;
                state(j,3) = v.*cos(angle);
                state(j,4) = sgn.*abs(v.*sin(angle));
            else
                state(j,4) = -state(j,4);
            end
        end
        box_num = for_box(state(j,1:2), boxes);
    end
end

% Scatter particles
j = rand(popul_size, 1) < scatter_p;
state(j,3:4) = random(pdf_v, [sum(j),2]);

% Record the temp
temp(i) = (sum(state(:,3).^2) + sum(state(:,4).^2))*m/k/2/popul_size;

% Record positions
for j=1:popul_plot
    trajectories(i, (2*j):(2*j+1)) = state(j, 1:2);
end

% Update the motion for 5 iters
if show_motion && mod(i,5) == 0
    figure(3);
    subplot(3,1,1);
    hold off;
    plot(state(1:popul_plot,1)./1e-9,state(1:popul_plot,2)./1e-9, 'o');
    hold on;

    % Plotting the boxes
    for j=1:size(boxes,1)
        plot([boxes(j, 1) boxes(j, 1) boxes(j, 2) boxes(j, 2)
```

```
                boxes(j, 1)]./1e-9,[boxes(j, 3) boxes(j, 4) boxes(j, 4)
boxes(j, 3)
                boxes(j, 3)]./1e-9, 'k-');
        end

        axis([0 len/1e-9 0 height/1e-9]);
        title(sprintf('Part 3 Trajectories for %d of %d
Electrons',popul_plot, popul_size));
        xlabel('x(nm)');
        ylabel('y(nm)');
        if i > 1
            subplot(3,1,2);
            hold off;
            plot(time_step*(0:i-1), temp(1:i));
            axis([0 time_step*iters min(temp(1:i))*0.98
                max(temp)*1.02]);
            title('Temperature of semiconductor');
            xlabel('Time(s)');
            ylabel('Temperature(K)');
        end

        subplot(3,1,3);
        v = sqrt(state(:,3).^2 + state(:,4).^2);
        title('Electron Speeds Histogram');
        histogram(v);
        xlabel('Speed(m/s)');
        ylabel('# of particles');
        pause(0.05);
    end
end

% Showing trajectories after the motion is finished
figure(3);
subplot(3,1,1);
title(sprintf('Part 3 Trajectories for %d of %d Electrons', popul_plot,
popul_size));
xlabel('x (nm)');
ylabel('y (nm)');
axis([0 len/1e-9 0 height/1e-9]);
hold on;
for i=1:popul_plot
    plot(trajectories(:,i*2)./1e-9, trajectories(:,i*2+1)./1e-9, '.');
end

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

% Plotting temp
if(~show_motion)
    subplot(3,1,2);
    hold off;
    plot(time_step*(0:iters-1), temp);
    axis([0 time_step*iters min(temp)*0.98 max(temp)*1.02]);
    title('Temperature of semiconductor');
```

```
    xlabel('Time (s)');
    ylabel('temp (K)');
end
subplot(3,1,3);
v = sqrt(state(:,3).^2 + state(:,4).^2);
title('Histogram of Electron Speeds');
histogram(v);
xlabel('Speed (m/s)');
ylabel('# of particles');

%Temp map
density = hist3(state(:,1:2),[200 100]);

% Smoothes out the density map
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(4);
imagesc(conv2(density,f,'same'));
set(gca,'YDir','normal');
title('Electron Density');
xlabel('x (nm)');
ylabel('y (nm)');
temp_sum_x = zeros(ceil(len/1e-9),ceil(height/1e-9));
temp_sum_y = zeros(ceil(len/1e-9),ceil(height/1e-9));
temp_num = zeros(ceil(len/1e-9),ceil(height/1e-9));

% velocities
for i=1:popul_size
    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

    % Adds velocity to count
    temp_sum_y(x,y) = temp_sum_y(x,y) + state(i,3)^2;
    temp_sum_x(x,y) = temp_sum_x(x,y) + state(i,4)^2;
    temp_num(x,y) = temp_num(x,y) + 1;
end
temp = (temp_sum_x + temp_sum_y).*m./k./2./temp_num;
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(5);
imagesc(conv2(temp,f,'same'));
set(gca,'YDir','normal');
title('Temperature Map');
```

```
xlabel('x (nm)');
ylabel('y (nm)');
```