EE492 Senior Design II - Weekly Report 12

Group Number: May1634	Date: 3/24/16 - 3/31/16
Project Name: Studying cell behaviors in 3D microtissues using a LabChip	
Advisor: Long Que	
Client: Long Que	

The team

Role	Group Member
Group leader	Jonathan Yatckoske
Team Webmaster	Yaxiong Zhang
	Chun-Hao Lo
Team Communication Leader	Yuqian Hu
Team Key Concept Holder	Kaiyu Xu

Attendance (meeting date: Mar. 27th 2015)

Jonathan Yatckoske	In person
Chun-Hao Lo	In person
Yaxiong Zhang	In person
Kaiyu Xu	In person
Yuqian Hu	In person

Accomplishments of past week

1. Finish the PowerPoint for presentation.
2. Meet with Senior Design instructor. Give presentation and receive feed\back.

a. Fasten the speed of our presentation;
b. Include our challenges throughout the process of the presentation.
c. Get rid of the time-line.

3. Improve cell tracking code and GUI

Plan for coming week

1.Meet with our advisor and see if there is unsolved problem remains.

2. Improve our presentation. Prepare for the final presentation.

Pending issues

Plans for group meeting to rework on the final presentation before dead week.

Individual contributions

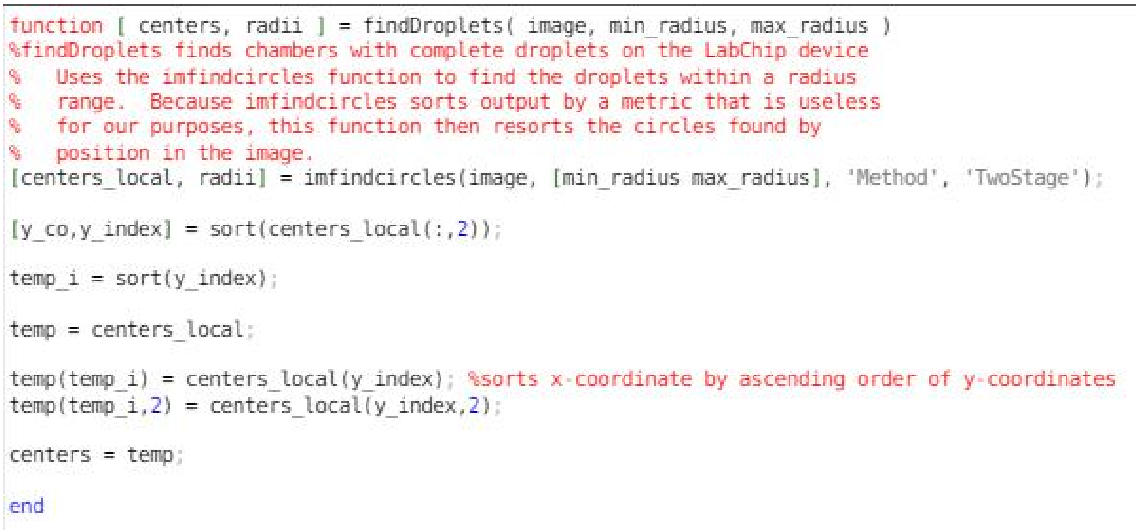
Jonathan Yatckoske	Work on coding; prepare for presentation
Chun-Hao Lo	website maintenance; code testing; prepare for presentation
Yaxiong Zhang	website maintenance; improve GUI; prepare for presentation
Kaiyu Xu	Take down meeting notes; prepare for presentation
Yuqian Hu	work on weekly report; prepare for presentation

Individual hourly contributions

Name	Week Hours	Cumulative Hours
Jonathan Yatckoske	6	81.5
Chun-Hao Lo	5	67.5
Yaxiong Zhang	5	67
Kaiyu Xu	2	36
Yuqian Hu	4	57.5

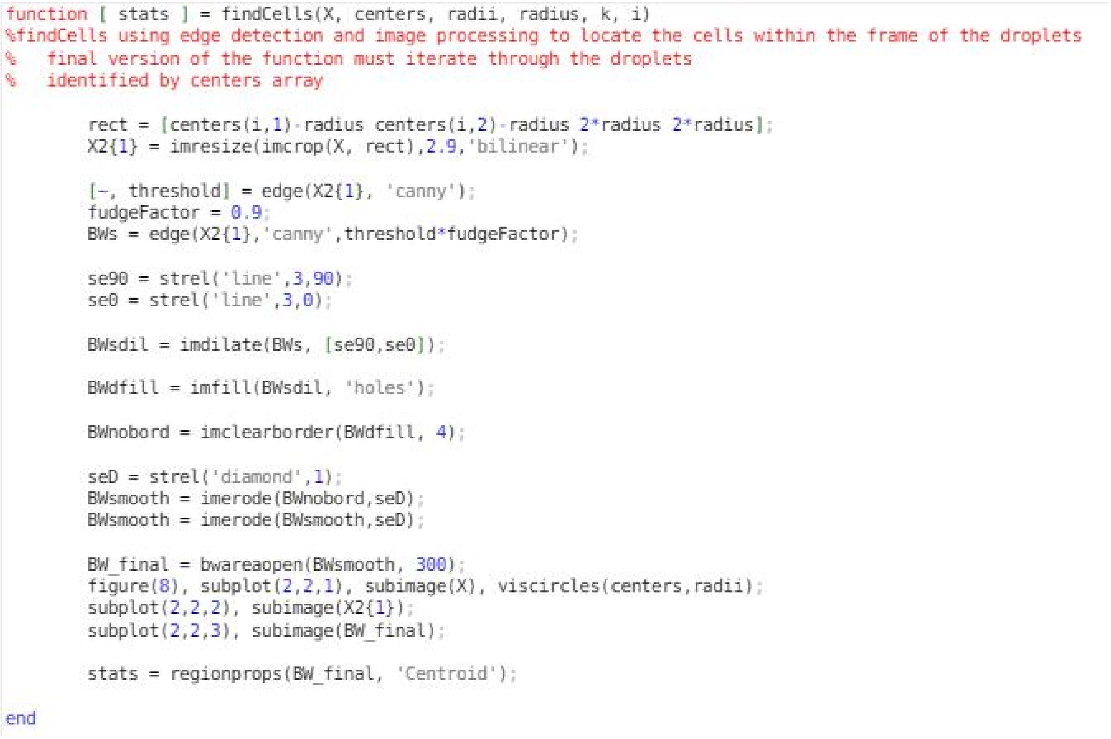
Appendix(Code)

1. findDroplets.m:



```
function [ centers, radii ] = findDroplets( image, min_radius, max_radius )
%findDroplets finds chambers with complete droplets on the LabChip device
%   Uses the imfindcircles function to find the droplets within a radius
%   range.  Because imfindcircles sorts output by a metric that is useless
%   for our purposes, this function then resorts the circles found by
%   position in the image.
[centers_local, radii] = imfindcircles(image, [min_radius max_radius], 'Method', 'TwoStage');

[y_co,y_index] = sort(centers_local(:,2));

temp_i = sort(y_index);

temp = centers_local;

temp(temp_i) = centers_local(y_index); %sorts x-coordinate by ascending order of y-coordinates
temp(temp_i,2) = centers_local(y_index,2);

centers = temp;

end
```

2. findCells.m

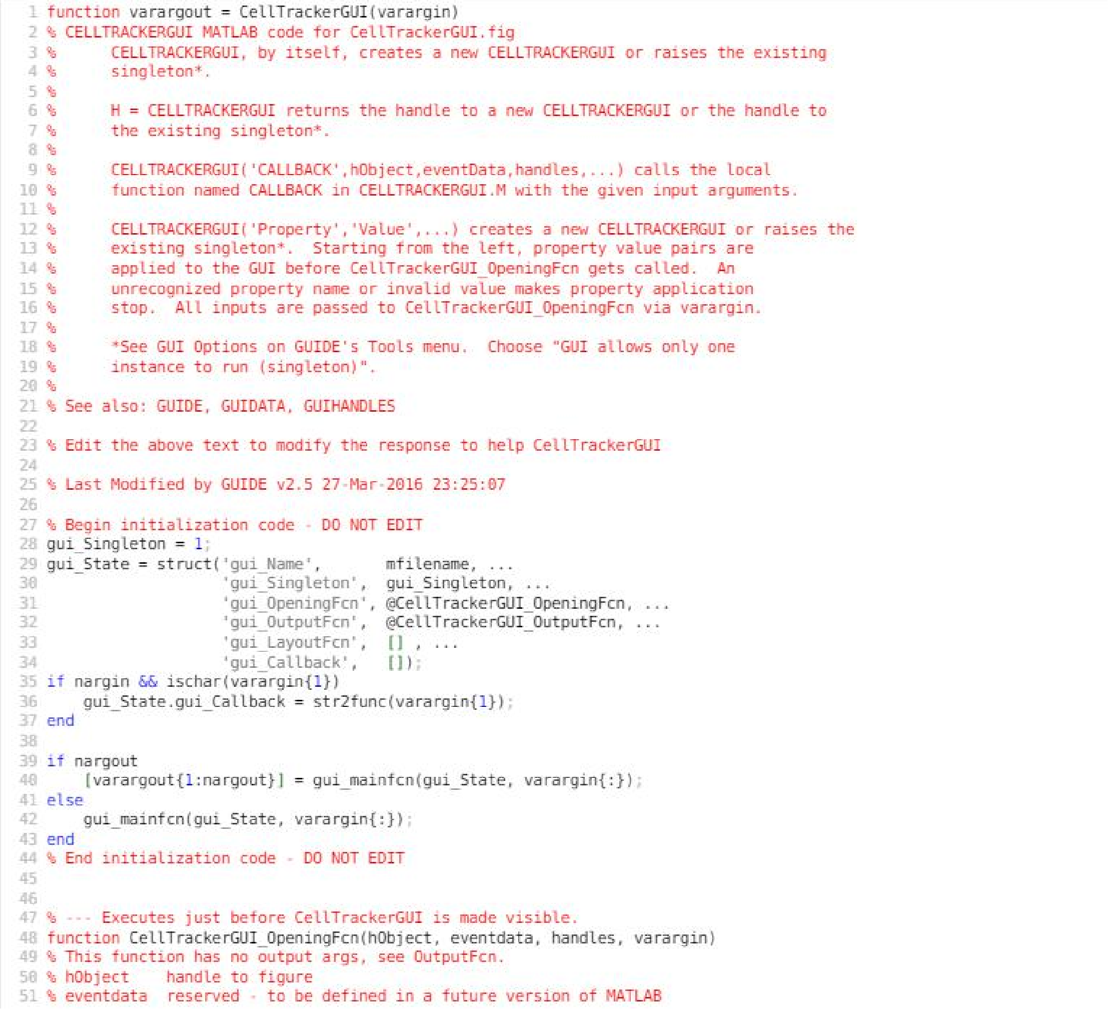


```
function [ stats ] = findCells(X, centers, radii, radius, k, i)
%findCells using edge detection and image processing to locate the cells within the frame of the droplets
%   final version of the function must iterate through the droplets
%   identified by centers array

        rect = [centers(i,1)-radius centers(i,2)-radius 2*radius 2*radius];
        X2{1} = imresize(imcrop(X, rect),2.9,'bilinear');

        [~, threshold] = edge(X2{1}, 'canny');
        fudgeFactor = 0.9;
        BWs = edge(X2{1},'canny',threshold*fudgeFactor);

        se90 = strel('line',3,90);
        se0 = strel('line',3,0);

        BWsdil = imdilate(BWs, [se90,se0]);

        BWdfill = imfill(BWsdil, 'holes');

        BWnobord = imclearborder(BWdfill, 4);

        seD = strel('diamond',1);
        BWsmooth = imerode(BWnobord,seD);
        BWsmooth = imerode(BWsmooth,seD);

        BW_final = bwareaopen(BWsmooth, 300);
        figure(8), subplot(2,2,1), subimage(X), viscircles(centers,radii);
        subplot(2,2,2), subimage(X2{1});
        subplot(2,2,3), subimage(BW_final);

        stats = regionprops(BW_final, 'Centroid');

end
```

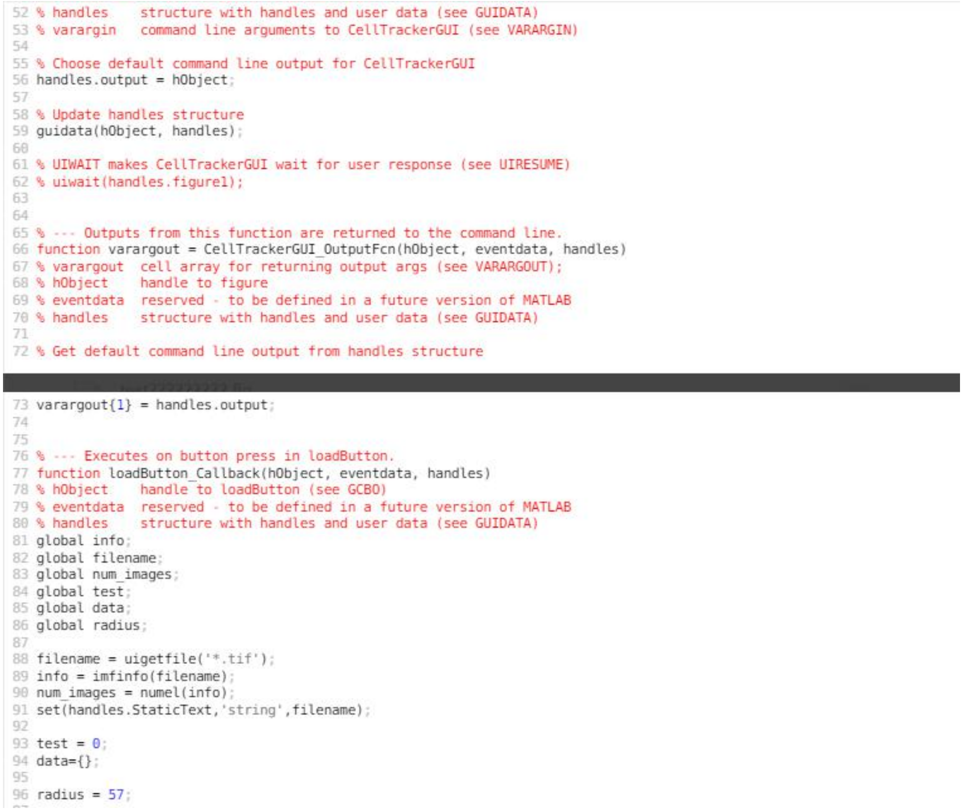
3. Code for cell-tracking:

```
close all
clear all

filename = uigetfile('*.tif');
info = imfinfo(filename);
num_images = numel(info);
bit_depth = info.BitDepth;
test = 0;
data={};

radius = 57;

for k = 1:num_images
    X = imread(filename, k);

    %if the image is RGB instead of grayscale, convert it before doing
    %anything else
    if bit_depth == 24
        X = rgb2gray(X);
    end

    %find the droplets
    [centers, radii] = findDroplets(X,40,120);

    %ignore any droplets with nothing in them
    %do this by deleting the rows from centers and radii that have no cells
    %found in the first frame
    %TODO:    write a function for this part

    %find the cells in remaining droplets
    if not(isempty(centers))
        for i = 1:length(centers)
            stats = findCells( X, centers, radii, radius, k, i);
            data{k,i} = stats;
        end
    else
        data{k} = [];
    end
end
```

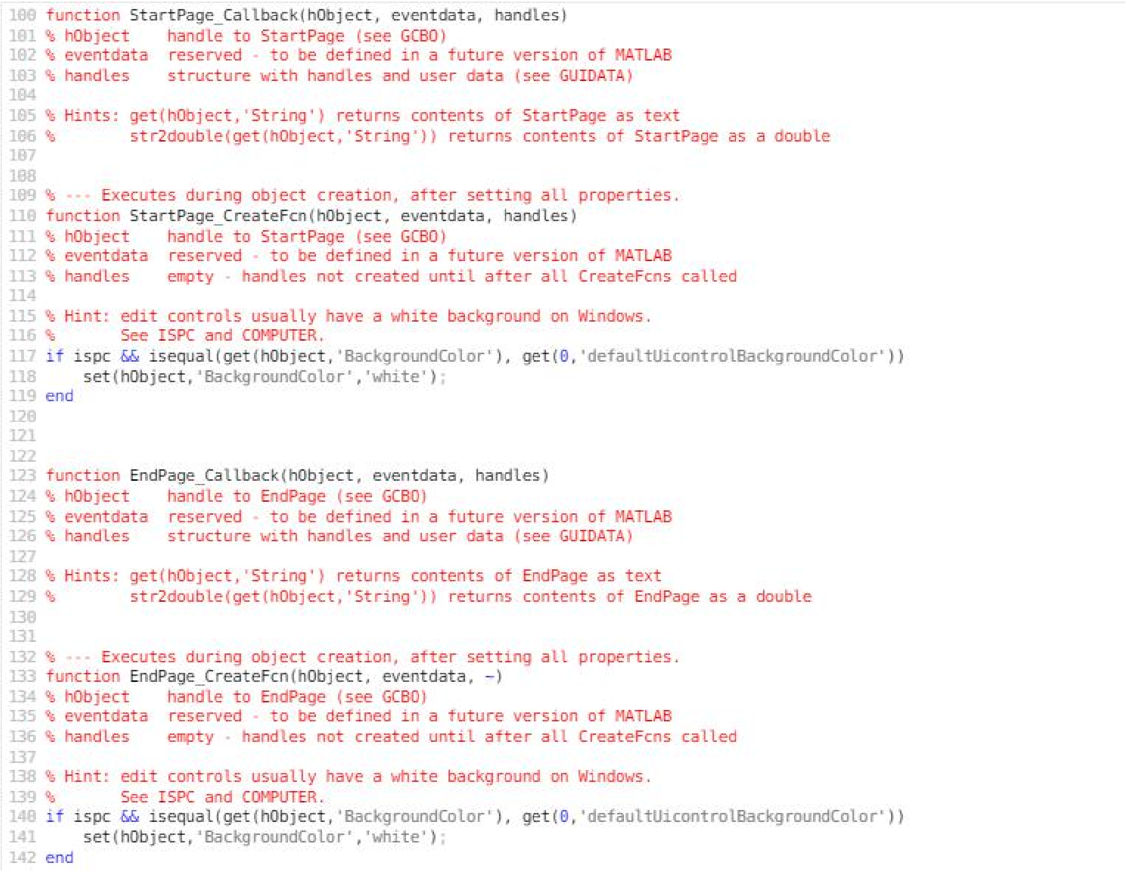
4.improved GUI code



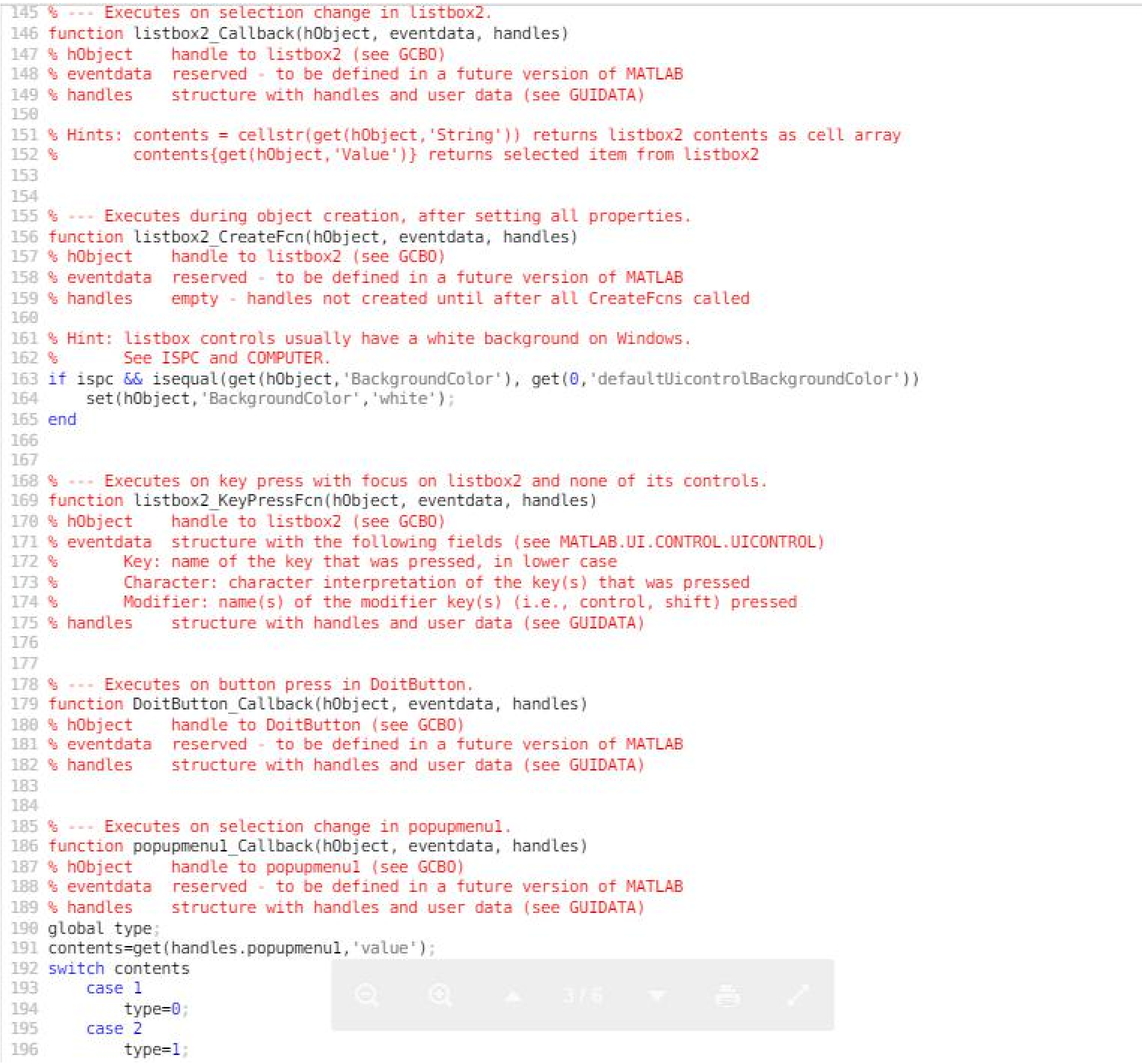
```
function varargout = CellTrackerGUI(varargin)
% CELLTRACKERGUI MATLAB code for CellTrackerGUI.fig
%      CELLTRACKERGUI, by itself, creates a new CELLTRACKERGUI or raises the existing
%      singleton*.
%
%      H = CELLTRACKERGUI returns the handle to a new CELLTRACKERGUI or the handle to
%      the existing singleton*.
%
%      CELLTRACKERGUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in CELLTRACKERGUI.M with the given input arguments.
%
%      CELLTRACKERGUI('Property','Value',...) creates a new CELLTRACKERGUI or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before CellTrackerGUI_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to CellTrackerGUI_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help CellTrackerGUI

% Last Modified by GUIDE v2.5 27-Mar-2016 23:25:07

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @CellTrackerGUI_OpeningFcn, ...
                   'gui_OutputFcn',  @CellTrackerGUI_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before CellTrackerGUI is made visible.
function CellTrackerGUI_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
```



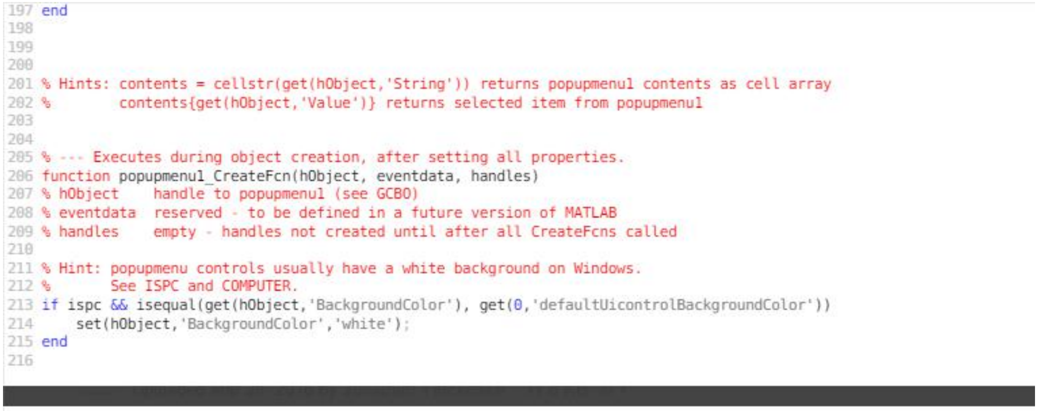
```
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to CellTrackerGUI (see VARARGIN)

% Choose default command line output for CellTrackerGUI
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes CellTrackerGUI wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = CellTrackerGUI_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in loadButton.
function loadButton_Callback(hObject, eventdata, handles)
% hObject    handle to loadButton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global info;
global filename;
global num_images;
global test;
global data;
global radius;

filename = uigetfile('*.tif');
info = imfinfo(filename);
num_images = numel(info);
set(handles.StaticText,'string',filename);

test = 0;
data={};

radius = 57;
```



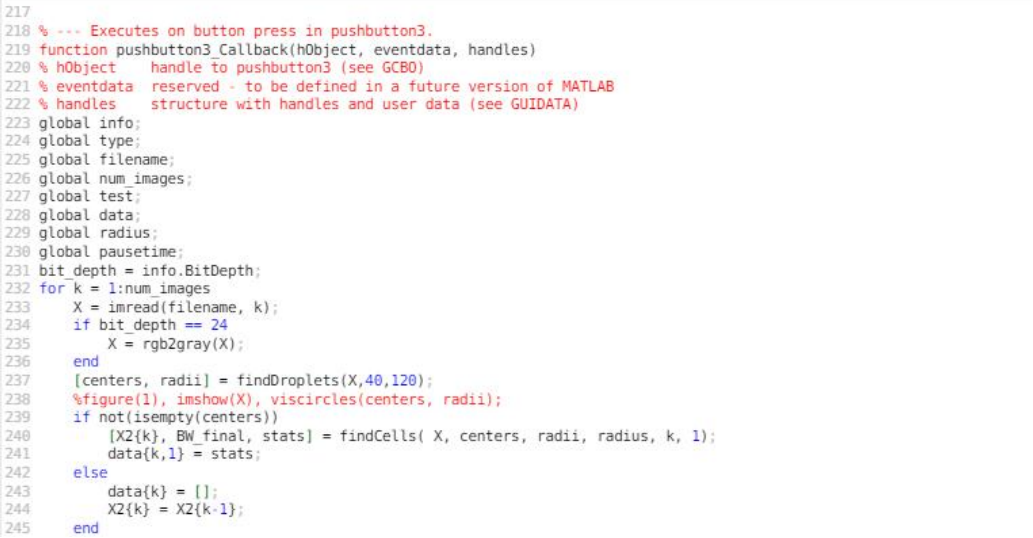
```
function StartPage_Callback(hObject, eventdata, handles)
% hObject    handle to StartPage (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of StartPage as text
%        str2double(get(hObject,'String')) returns contents of StartPage as a double


% --- Executes during object creation, after setting all properties.
function StartPage_CreateFcn(hObject, eventdata, handles)
% hObject    handle to StartPage (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function EndPage_Callback(hObject, eventdata, handles)
% hObject    handle to EndPage (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of EndPage as text
%        str2double(get(hObject,'String')) returns contents of EndPage as a double


% --- Executes during object creation, after setting all properties.
function EndPage_CreateFcn(hObject, eventdata, ~)
% hObject    handle to EndPage (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```



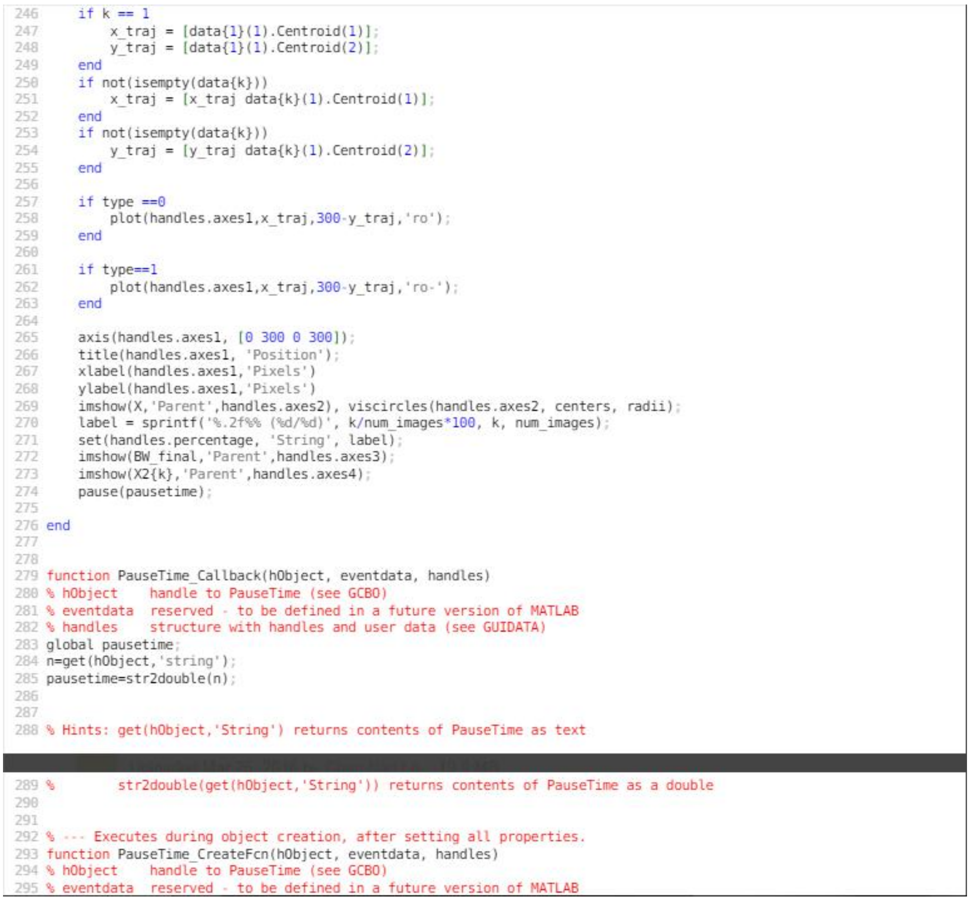
```
% --- Executes on selection change in listbox2.
function listbox2_Callback(hObject, eventdata, handles)
% hObject    handle to listbox2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listbox2 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from listbox2


% --- Executes during object creation, after setting all properties.
function listbox2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on key press with focus on listbox2 and none of its controls.
function listbox2_KeyPressFcn(hObject, eventdata, handles)
% hObject    handle to listbox2 (see GCBO)
% eventdata  structure with the following fields (see MATLAB.UI.CONTROL.UICONTROL)
%       Key: name of the key that was pressed, in lower case
%       Character: character interpretation of the key(s) that was pressed
%       Modifier: name(s) of the modifier key(s) (i.e., control, shift) pressed
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in DoitButton.
function DoitButton_Callback(hObject, eventdata, handles)
% hObject    handle to DoitButton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global type;
contents=get(handles.popupmenu1,'value');
switch contents
    case 1
        type=0;
    case 2
        type=1;
```



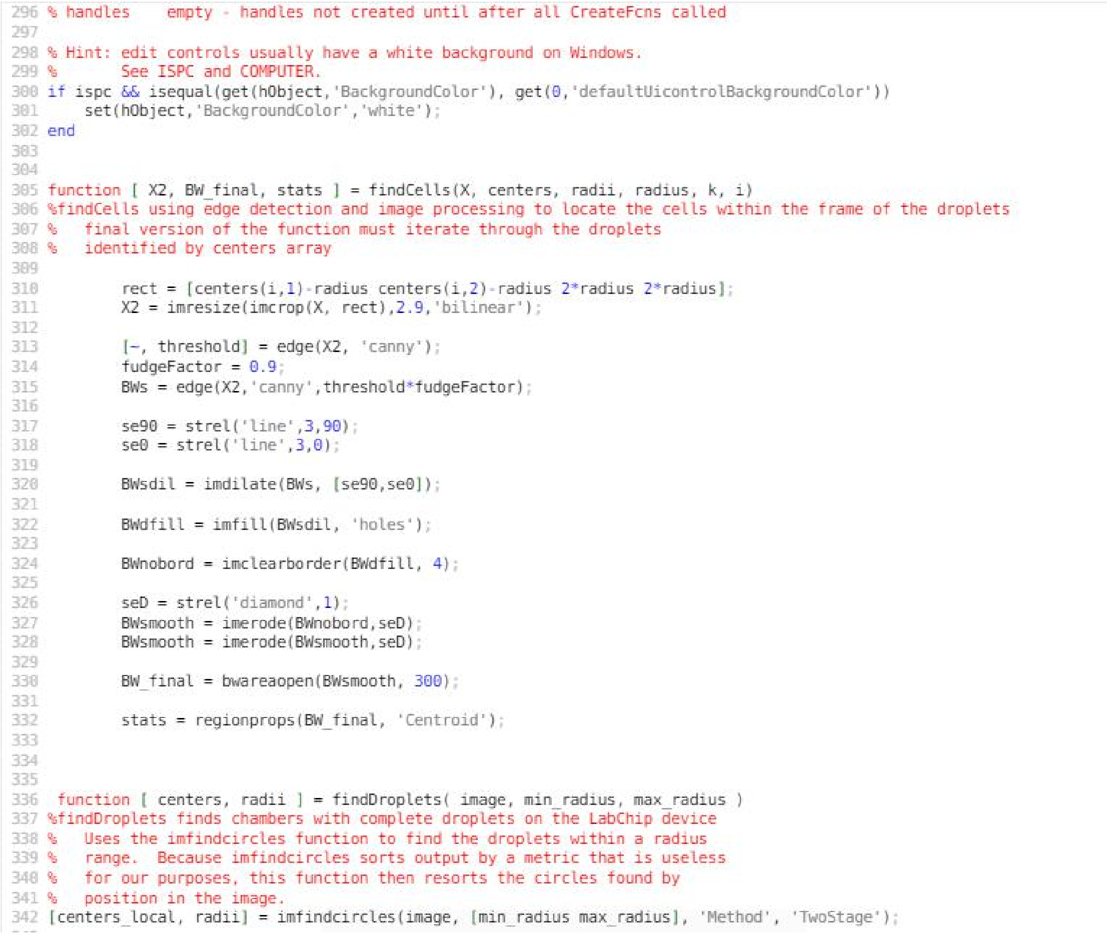
```
end


% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu1


% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```



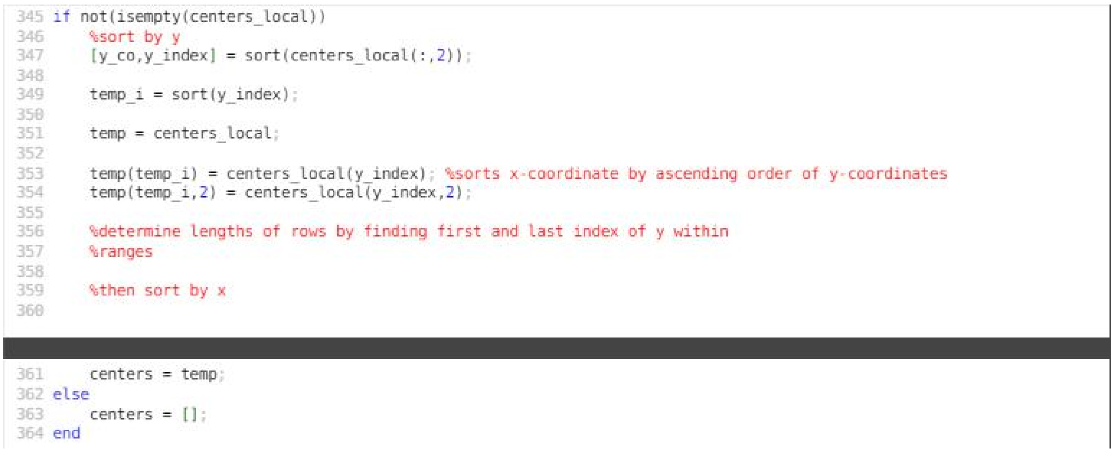
```

% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global info;
global type;
global filename;
global num_images;
global test;
global data;
global radius;
global pausetime;
bit_depth = info.BitDepth;
for k = 1:num_images
    X = imread(filename, k);
    if bit_depth == 24
        X = rgb2gray(X);
    end
    [centers, radii] = findDroplets(X,40,120);
    %figure(1), imshow(X), viscircles(centers, radii);
    if not(isempty(centers))
        [X2{k}, BW_final, stats] = findCells( X, centers, radii, radius, k, 1);
        data{k,1} = stats;
    else
        data{k} = [];
        X2{k} = X2{k-1};
    end
```



```
    if k == 1
        x_traj = [data{1}(1).Centroid(1)];
        y_traj = [data{1}(1).Centroid(2)];
    end
    if not(isempty(data{k}))
        x_traj = [x_traj data{k}(1).Centroid(1)];
    end
    if not(isempty(data{k}))
        y_traj = [y_traj data{k}(1).Centroid(2)];
    end

    if type ==0
        plot(handles.axes1,x_traj,300-y_traj,'ro');
    end

    if type==1
        plot(handles.axes1,x_traj,300-y_traj,'ro-');
    end

    axis(handles.axes1, [0 300 0 300]);
    title(handles.axes1, 'Position');
    xlabel(handles.axes1,'Pixels')
    ylabel(handles.axes1,'Pixels')
    imshow(X,'Parent',handles.axes2), viscircles(handles.axes2, centers, radii);
    label = sprintf('%.2f%% (%d/%d)', k/num_images*100, k, num_images);
    set(handles.percentage, 'String', label);
    imshow(BW_final,'Parent',handles.axes3);
    imshow(X2{k},'Parent',handles.axes4);
    pause(pausetime);

end


function PauseTime_Callback(hObject, eventdata, handles)
% hObject    handle to PauseTime (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global pausetime;
n=get(hObject,'string');
pausetime=str2double(n);


% Hints: get(hObject,'String') returns contents of PauseTime as text
%        str2double(get(hObject,'String')) returns contents of PauseTime as a double


% --- Executes during object creation, after setting all properties.
function PauseTime_CreateFcn(hObject, eventdata, handles)
% hObject    handle to PauseTime (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```



```
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function [ X2, BW_final, stats ] = findCells(X, centers, radii, radius, k, i)
%findCells using edge detection and image processing to locate the cells within the frame of the droplets
%   final version of the function must iterate through the droplets
%   identified by centers array

        rect = [centers(i,1)-radius centers(i,2)-radius 2*radius 2*radius];
        X2 = imresize(imcrop(X, rect),2.9,'bilinear');

        [~, threshold] = edge(X2, 'canny');
        fudgeFactor = 0.9;
        BWs = edge(X2,'canny',threshold*fudgeFactor);

        se90 = strel('line',3,90);
        se0 = strel('line',3,0);

        BWsdil = imdilate(BWs, [se90,se0]);

        BWdfill = imfill(BWsdil, 'holes');

        BWnobord = imclearborder(BWdfill, 4);

        seD = strel('diamond',1);
        BWsmooth = imerode(BWnobord,seD);
        BWsmooth = imerode(BWsmooth,seD);

        BW_final = bwareaopen(BWsmooth, 300);

        stats = regionprops(BW_final, 'Centroid');



 function [ centers, radii ] = findDroplets( image, min_radius, max_radius )
%findDroplets finds chambers with complete droplets on the LabChip device
%   Uses the imfindcircles function to find the droplets within a radius
%   range.  Because imfindcircles sorts output by a metric that is useless
%   for our purposes, this function then resorts the circles found by
%   position in the image.
[centers_local, radii] = imfindcircles(image, [min_radius max_radius], 'Method', 'TwoStage');
```



```
if not(isempty(centers_local))
    %sort by y
    [y_co,y_index] = sort(centers_local(:,2));

    temp_i = sort(y_index);

    temp = centers_local;

    temp(temp_i) = centers_local(y_index); %sorts x-coordinate by ascending order of y-coordinates
    temp(temp_i,2) = centers_local(y_index,2);

    %determine lengths of rows by finding first and last index of y within
    %ranges

    %then sort by x


    centers = temp;
else
    centers = [];
end
```