**Final Term Project ‘Simulation and Modeling’**

**Topic: Motion Detector (Matlab)**

Date: 24th June, 2020

**Table of Contents**

Title page…………………………………………………………...……………………………01

Table of Contents………………………………………………………………………………...02

Introduction………………………………………………………………………………………04

1. Connection To A Webcam…………………………………………………………………...04
2. Preview Of Webcam…………………………………………………………………………05
3. Snapshot Of The Frame……………………………………………………………………...05
4. For Loop……………………………………………………………………………………...05
5. Frame Subtraction……………………………………………………………………………06
6. Size (x) Function……………………………………………………………………………..06
7. Imread………………………………………………………………………………………..07
8. Imshow VS imshowpair……………………………………………………………………...07
9. Function rgb2gray……………………………………………………………………………07
10. Function ‘im2bw’ VS ‘imbinarize’…………………………………………………………07
11. Tilde Sign (~)………………………………………………………………………………...08
12. Removing Small Objects From Binary Image……………………………………………….08
13. Connectivity And Label Function……………………………………………………………09
14. Regional Related Problems…………………………………………………………………..09
15. Variable ‘Locations’…………………………………………………………………………09
16. Hold Function………………………………………………………………………………..10
17. Showing Two Pictures……………………………………………………………………….10
18. Rectangle…………………………………………………………………………………….10
19. Concatenation Of Two Matrices……………………………………………………………..11
20. Plotting Centroid……………………………………………………………………………..11

Chart……………………………………………………………………………………………...12

Code Working……………………………………………………………………………………13

Code……………………………………………………………………………………………...16

Output……………………………………………………………………………………………18

Conclusion……………………………………………………………………………………….20

References………………………………………………………………………………………..21

**Introduction**

This report covers all the details about the project and how the code works. The most crucial part of the code was to connect the webcam to the software. Then I had to think how exactly we are going to find the Centroid and place a boundary around it. This code was achieved with the help of Lab Manuals given by the teacher and the rest of the data was found on the internet using Mathworks and different websites. The references can be found in the end of this document.

This document contains meaning of every new command, connectivity, details how a specific line of code is working or what exactly it is doing. Then there are the screenshots of code and the then output.

1. **Connection To A Webcam**
   1. One of the crucial parts of a motion detector is to connect the Webcam with the software MATLAB (which I am using to create this program). This was done by predefined function which makes objects of a web camera. What I did here is that I used the keyword ‘webcam’ and made an object of it called ‘cam’,
   2. The line ‘cam=webcam’ shows that an object of webcam is created and it connects to a single webcam.
   3. Since, I only have one camera connected to the system, I don’t have to use ‘webcam lists’. That functions shows all the cameras connected to the system.
   4. I cannot create multiple objects of a single camera, if I do so it gives an error.
   5. The name of the object can be anything. It is our choice.
2. **Preview Of Webcam**
   1. I can preview the webcam by a single command line ‘preview(cam)’.
   2. This function enables us to view the webcam feed in a new window. The window shows information about the feed. Such as, it displays the timestamp, video resolution of each frame, current frame rate, and the current status of the object.
   3. The working of this command line is to send the object of webcam into that function (in our case it is ‘cam’).
   4. The window of the feed will remain active as long as I don’t close it. I will show this at the end if the switch statement by using command ‘stoppreview’ or ‘closepreview’.
3. **Snapshot Of The Frame**
   1. It acquires a single frame of the webcam object.
   2. What I will be doing here is that we are going to store that single frame in ‘ReffFrame’ variable.
   3. The stored frame will always be a RGB i.e colorful image.
   4. I will perform this function two times in the program
4. **For Loop**
   1. Just like C++ it is used to repeat a specific code a specific number of times.
   2. The syntax is different from C++ obviously.
   3. In this ‘for’ loop, I will be storing another single frame of the webcam feed in a single variable called ‘CurrFrame ‘ using snapshot.
   4. The ‘for loop’ will be iterated for about a hundred times.
   5. To avoid the disruption in command window we use semicolon (;) in the end of the snapshot command.
   6. Once the ‘for loop’ has executed all the statements, it will end.
5. **Frame Subtraction**
   1. First, we will store a single frame in ‘RefFrame’ using snapshot function.
   2. Then the ‘for loop’ starts and we store a single new frame every time the ‘for loop’ iterates.
   3. We store new frame in ‘CurrFrame’.
   4. In the end, we subtract Current Frame from Reference Frame and store it into another variable i.e ‘x’ variable.
   5. What is happening here is that every pixel that differs from the reference image will be calculated.
6. **Size (x) Function**
   1. This function shows the size of ‘x’ variable.
   2. It’s answer may differ for different variable. This is because every type of variable has a different size.
   3. In our case, an image (colorful) is stored in it. So, it will display the combination of RGB that are used in the image.
7. **Imread**
   1. It loads an image in the software system.
   2. This function works by sending the complete file path in it, enclosed in single quotations symbol.
   3. This function is not used in the program because we already have image/frame loaded in our system i.e webcam.
8. **Imshow VS imshowpair**
   1. These functions display the image. The first one displays a single image in a new window, whereas the second one displays multiple images (from a pair two more). Since we have to show a comparison, we are going to use the second one.
   2. It works by passing a variable in it as arguments, in which the image is loaded in.
   3. This command will be used in our program only once.
9. **Function rgb2gray**
   1. This is a function which converts a true-color image in the gray scale format (not black n white, but gray).
   2. This is done by sending a true-color image to it.
   3. Here I will be storing the grayscale image in a variable.
10. **Function ‘im2bw’ VS ‘imbinarize’**
    1. This was the tricky part. At first, I tried to perform the Grayscale to Binary Image with ‘im2bw’ but it was not very efficient. So, I used another function (imbinarize) which was way more efficient and accurate than the previous/earlier one.
    2. Just as I have said that both of these functions convert a Gray scale image to Binary Image.
    3. These function converts image to Binary image by threshold (It is a technique which is used to classify the pixel value into two different categories which are foreground or background).
    4. The function I have used in the program uses ‘adaptive’ input argument. It’s meaning is not very clear to me, but I can see the difference in the images and calculation’s results.
11. **Tilde Sign (~)**
    1. Once we are done with conversion of the image from RGB to Binary Image (i.e Black and White).
    2. We will have a white background with black objects.
    3. We want to reverse that i.e. turn black to white and white to black
    4. In order to achieve that I will be using the ‘Tilde sign’.
    5. We store the reversed image in a new variable
12. **Removing Small Objects From Binary Image**
    1. Once I have our Binary Image I want to reduce the noise. This process is done by ‘bwareaopen’ function.
    2. We send an object and pixels value in it as arguments.
    3. It basically removes all the objects which are less than a certain amount of pixels. In our case, I have set the value to 1000 pixels. So, any object which is less than one thousand pixels will be removed.
13. **Connectivity And Label Function**
    1. As, I wanted to count the objects I used a function called ‘bwlabel’. It helped me in the connectivity and counting the objects
    2. Since, I am using 8th level of connectivity which means it will measure horizontals, verticals and diagonals when counting an object. Whereas, level 4 connectivity only check horizontal and vertical.
    3. It is also storing (‘Objectss’) and displaying number of objects.
14. **Regional Related Problems**
    1. It is known as ‘regionprops’ function.
    2. It helps in measure properties of image regions. Such as Eccentricity, Centroid, Bounding Box etc
    3. We pass an object and properties in the arguments for the function to process.
15. **Variable ‘Locations’**
    1. What the code is doing with this variable is that it is saving coordinates (x,y) and dimensions (width, height) in the variable ‘Locations’.
    2. This variable is basically a ‘struct field array’ and saves all the object’s data found in it.
16. **Hold Function**
    1. It retains (holds) current plot when we are adding new plots. In simpler words, it will pause for a moment.
    2. There are five types of Hold, but I have only used 2 of them i.e. ‘hold on’ and ‘hold off’.
17. **Showing Two Pictures** 
    1. To show a comparison or in simpler words show two picture side by side. I have implemented ‘imshowpair’ function.
    2. It worked beautifully, when the program displayed variable ‘x’ and ‘bw’ variables (images).
    3. I have also used the keyword ‘figure’ with it, which means it creates another window with default property values.
18. **Rectangle**
    1. This part took most of my time in this project. It has created the perfect rectangles to display detected objects.
    2. This part of code performs two things. Draw a rectangle and store the value of new rectangles.
    3. First, the ‘for loop’ starts and it will iterate to the number of locations it has detected. Then it stores the value of every ‘ith’ Bounding Box’s location (coordinates) in a variable ‘thisBB’. Afterwards, the code sends the 4 elements vector array in it. At last, we customize the bounding box according to our desire. In my case, I have chosen colour Magenta and width of the line for ‘3’.
19. **Concatenation Of Two Matrices**
    1. The word concatenation means addition or composing two items together. Here in this code we are concatenating two matrices.
20. **Plotting Centroid**
    1. It is obvious that we have found the Centroid’s location now we want to plot it on the picture.
    2. I have used the ‘CentroidLocation’ and passed its data in the plot function.
    3. The color of the cursor/range is Magenta.

Start

Infinite while loop starts

Run Detector or not Run Detector (Switch)

End

**NO**

Motion Detector Code starts

**YES**

***Code Working:***

The code starts from a ‘while’ loop which is an infinite loop and will keep running until it is stopped such as using break command. In this program, we ask the user choose (not enter) an option. This is asked with the help of a GUI (Graphical User Interface) menu. This menu is given two option whether to run or not run the motion detector. This menu returns the option’s number and it is stored into a variable. The variable is then sent into the ‘switch’ statements. We only have three cases in the ‘switch’ statement. So, there are basically only three possibilities. First, we have ‘Run Motion Detector’ this is stored in the case ‘1’ which was returned by the GUI menu. The code in that case is executed. Then we have case ‘2’ which says ‘Don’t run the Motion Detector’ and so the code is executed in that case. In the end, we have otherwise case which works same as to that of C++ default keyword.

The very first part of the code is connecting the webcam to the MATLAB software. So, I have connected my Laptop’s camera to the MATLAB software with help of a command. After that, I saved a frame in a variable and started the ‘for loop’ for about hundred more frames and saved the 100th frame in another variable. Then in the code we subtract both frames and store that value in another variable.

Since, we have acquired a new image that shows if a new object has been detected. The luminescent pixels are pixel that differs from the Reference Frame (image). Now, the main code is performed on this newly acquired image. The first task is to turn this image into RGB, then turn it into grayscale image. Once, it has been successfully converted to a gray scale image, we can further convert it to a Binary Image. The function I am using in this code to turn it into a Binary Image is ‘imbinarize’ and it uses adaptive thresholding method.

Once the image has been converted to a Binary Image we need to reverse it. By reverse I mean to turn white into black and black into white. This task is accomplished by a tilde sign. It will help us in detecting the objects. The white blobs will be the objects. As we can see, there will be a lot of noise i.e. scattered pixels. In order, to remove the noise we set a command in which any pixel which is less than one thousand will be removed. So, by doing this we can achieve a perfect noise less image.

Now, the next task is to count all the white blobs i.e. objects in the Binary Image. This is done by ‘bwlabel’ function. All the objects that are detected by this function will be stored in the ‘Objectss’ variable. I am using 8th level of connectivity which provides my program to find the objects in all possible way. This is done by calculating the blobs horizontally, vertically and diagonally.

The code has now detected all the objects found in the image. Now, the task is to find the Centroid and put a mark on it and draw a Boundary on every object the program has found. This is accomplished by a function called ‘regionprops’ which is abbreviation for Regional related Properties. This data will be stored in the ‘Locations’ struct field array. If it detects only one object then it is called struct field and not an array. The data is in the form of locations and measurements/dimensions.

Now, we are going to show the preview of two images, but since we are using ‘hold on’ command, it will be paused momentarily (until ‘hold off’ comes). Moving on from this a ‘for’ loop is started which will contain few command regarding the boundary i.e. Bounding Box. In this loop, we are storing every ith location/dimension of every Bounding Box in a variable ‘thisBB’. After that, we are calling rectangle draw function and passing it the the four vector in it and storing all the data of every rectangle in ‘r’ variable. Later on, I have changed the color and width of the rectangle to ‘3’ and Magenta color.

The last step of the case ‘1’ is concatenation which means addition. We are adding two matrices i.e. all the locations of the Centroid and storing it. Later on, we plot every single Centroid in their respective places. Then the ‘hold off’ command is displayed and so is the preview of the two pictures. Then we close the preview command and disable the camera. This concludes the first case of the switch statement.

At the start of the while loop I made a variable ‘Check’ and gave it a value of 0. This will be used to get out of the while loop in simpler work break the loop and terminate the program. Once the user presses option 2 i.e. Don’t run Motion detector we will enter second case. In this case, the program will display few messages and set the value of variable ‘Check’ to 1. When the value if one, the ‘If’ statement will become true and is executed. It holds a special command called ‘break’ which stops any loop or statement e.g. for loop, while loop.

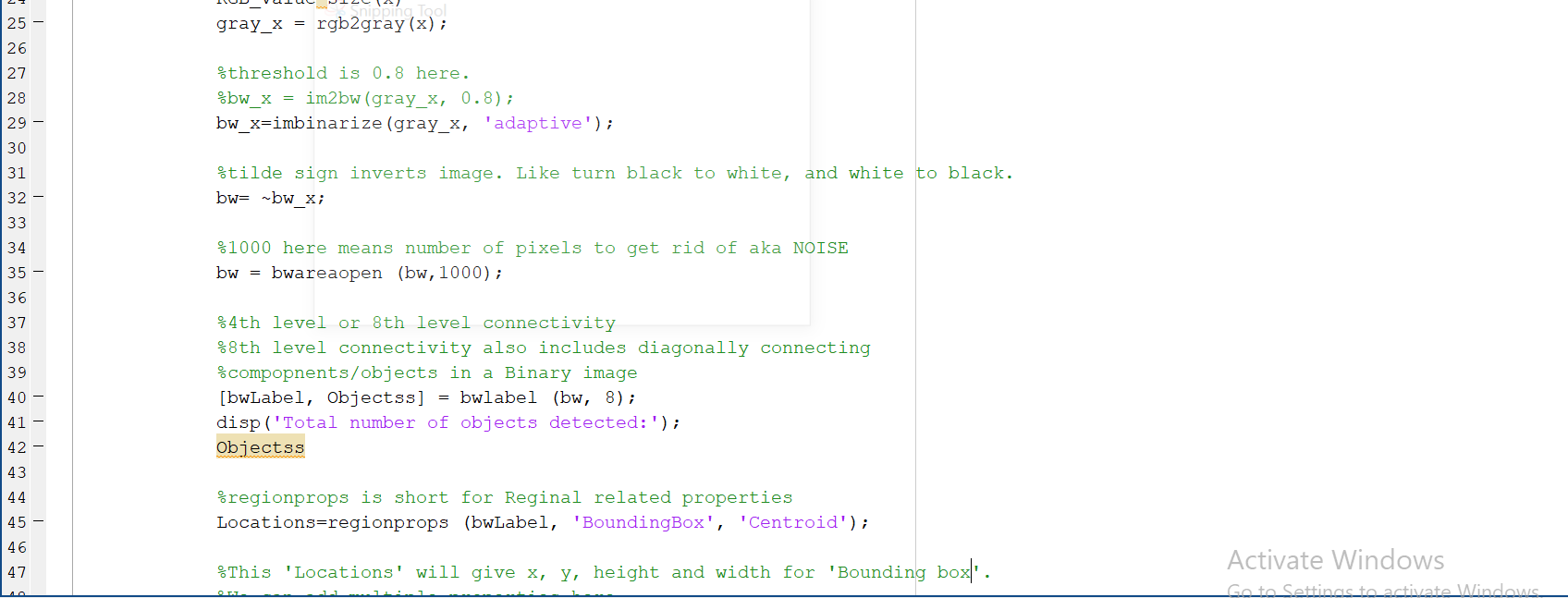
Once it is executed, the program will end. If it is not true then it will not be executed. If the ‘if’ statement is not execute a message is displayed saying that it is going to ask you for the menu again which means the while loop is again iterated.

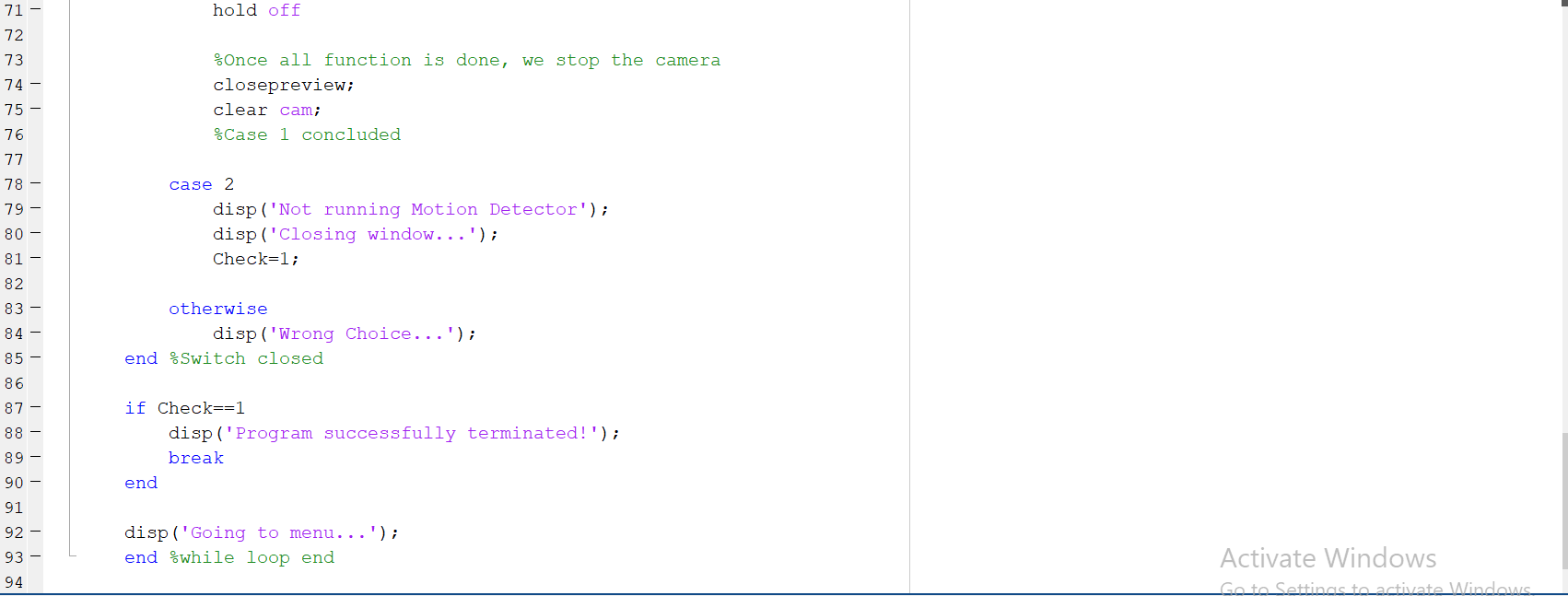
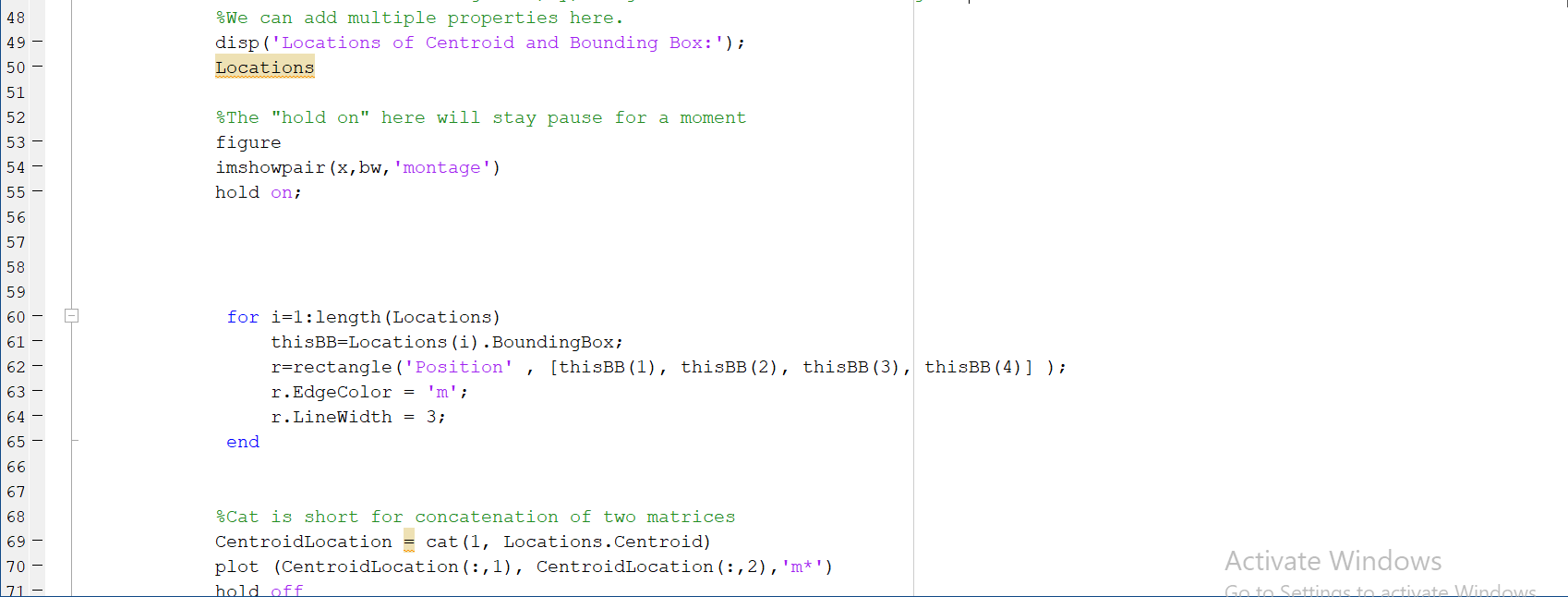
This concludes my program and my report.

***Code:***

I have also attached the code pasted on the notepad (.txt) file, but I am still attaching the screenshots for the code here.

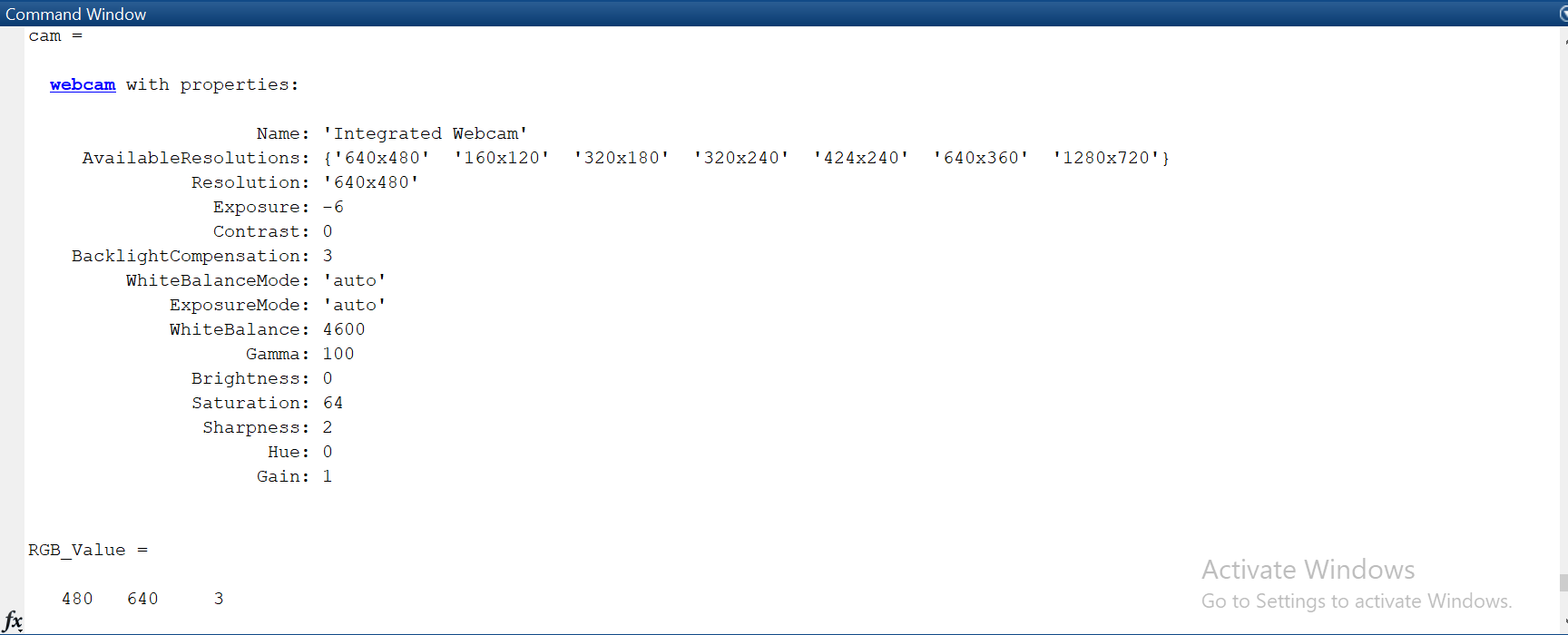


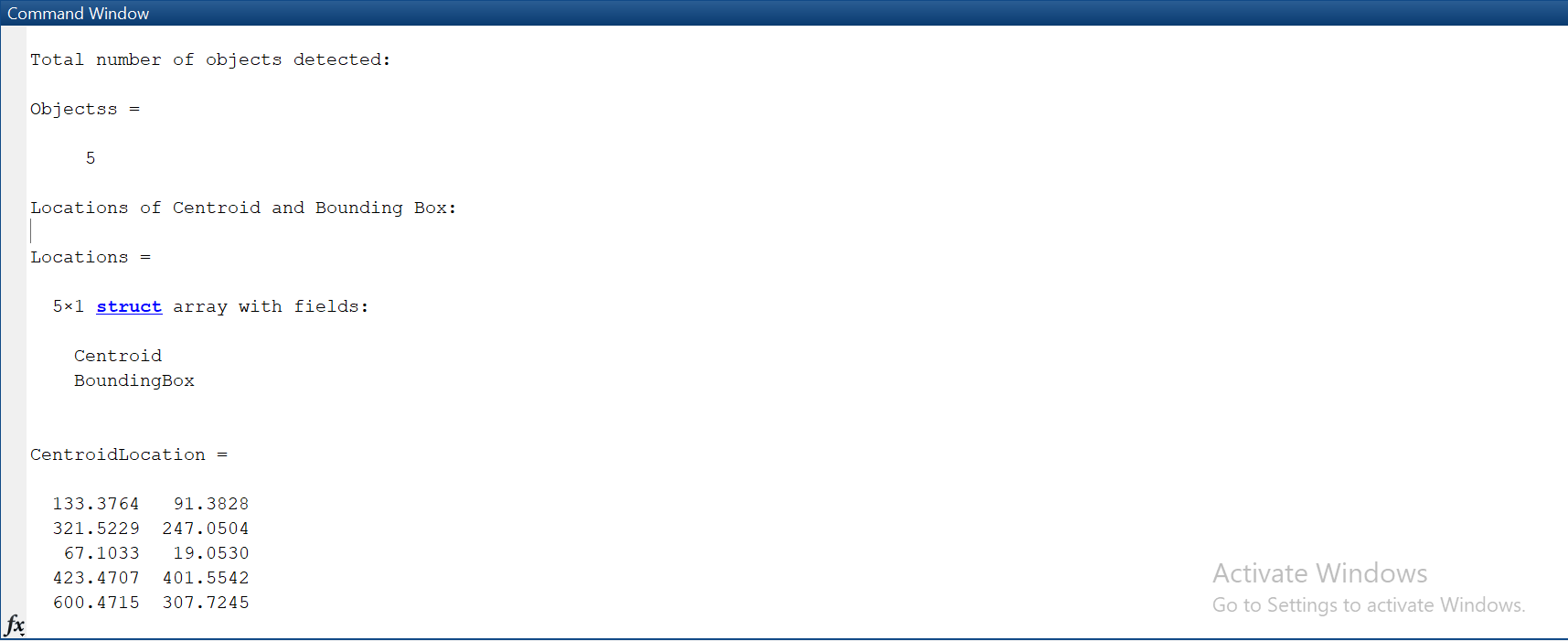


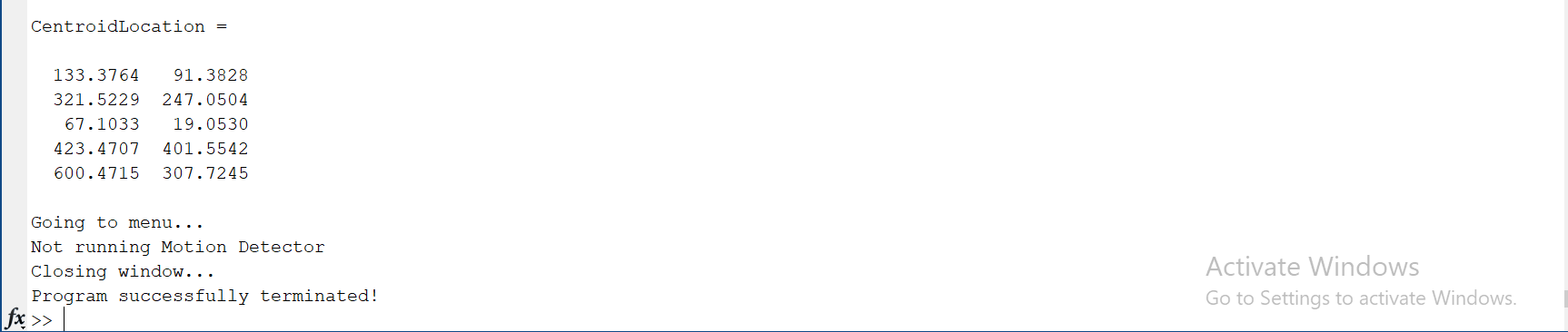


***Output:***

To get this output, I have used two different objects. One is hollow and the second one is a whole object. The motion detector has detected both the objects. It has even detected a small space made by my hand and though that it is an object.









***Conclusion:***

It was a very interesting yet challenging project. It took me a long time to complete this. I had to read others codes online and do reverse engineering on it. I also had to read loads and loads of books to find the meaning of every single command. This project has taught me how to analyze and adapt to different circumstances such as different syntax, working with something entirely unknown.

***References***:

<https://www.mathworks.com/help/matlab/ref/while.html?s_tid=srchtitle>

<https://www.mathworks.com/help/matlab/ref/for.html?s_tid=srchtitle>

<https://www.mathworks.com/help/images/ref/imshowpair.html>

<https://www.mathworks.com/help/images/ref/bwlabel.html>

<https://www.tutorialspoint.com/matlab/matlab_commands.htm>

<https://www.mccormick.northwestern.edu/documents/students/undergraduate/introduction-to-matlab.pdf>

<https://books.goalkicker.com/MATLABBook/>

<https://www.programming-book.com/download/?file=16067>

<https://www.mathworks.com/help/supportpkg/usbwebcams/ug/webcam.html>

<https://www.mathworks.com/help/supportpkg/usbwebcams/ug/webcam.preview.html>

<https://www.mathworks.com/help/imaq/snapshot.html>

<https://www.mathworks.com/help/matlab/ref/size.html>

<https://www.mathworks.com/help/images/ref/imbinarize.html>

<https://www.mathworks.com/help/images/ref/imshowpair.html>

<https://www.mathworks.com/help/images/ref/bwareaopen.html>

<https://www.mathworks.com/help/matlab/matlab_prog/matlab-operators-and-special-characters.html>

<https://www.mathworks.com/help/images/ref/regionprops.html>

<https://www.mathworks.com/help/matlab/ref/hold.html>

<https://www.mathworks.com/help/matlab/ref/rectangle.html>

<https://www.mathworks.com/help/matlab/ref/plot.html>

<https://www.mathworks.com/help/matlab/ref/cat.html>