

Using MATLAB to measure the diameter of an object within an image

18AIS101J – Introduction to MATLAB for Artificial Intelligence

Submitted by

Siddharth Ghosh -RA2011047010089

Pavan Shrinivas - RA2011047010091

Shreesh Upadhyay - RA2011047010125

Prakhar Chitransh - RA2011047010129

Ranjan Sharma - RA2011047010136

MINI PROJECT

BACHELOR OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR- 603 203

February 2021

BONAFIDE CERTIFICATE

Certified to be the project report titled “**MACHINE LEARNING MODELS TO MEASURE THE DIAMETER OF AN OBJECT WITHIN AN IMAGE**” is the Bonafide work done by Siddharth Ghosh (Reg No: RA2011047010089), Pavan Shrinivas (Reg No: RA2011047010091), Shreesh Upadhyay (Reg No: RA2011047010125), Prakhar Chitransh (Reg No: RA2011047010129) and Ranjan Sharma (Reg No. RA2011047010136) of CSE B Tech Degree course in the theory **18AIS101J – Introduction to MATLAB for Artificial Intelligence** during the academic year 2020-2021.

Signature of the Faculty

Dr. S Thanga Revathi

AP/CSE

Date of Submission: 21/02/2021

ABSTRACT

Measuring objects within an image or frame can be an important capability for many applications where computer vision is required instead of making physical measurements. This application note will cover a basic step-by-step algorithm for isolating a desired object and measuring its diameter.

Through this application note you will be able to write a MATLAB script file to import an image, segment the image in order to isolate the desired object from its background and then use the MATLAB functions that come with the Image Processing Toolbox to determine the objects diameter. It is assumed in this Application Note that the reader has a basic knowledge of MATLAB.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	3
1.	INTRODUCTION	5
2.	PROJECT DESCRIPTION	6 & 7
3.	CODE	8 & 9
4.	SCREENSHOTS	10 - 13
	CONCLUSION	14
	REFERENCES	15

INTRODUCTION

“MATLAB is a high-level language and interactive environment for computer computation, visualization, and programming. Image Processing Toolbox is an application available for use in MATLAB, which provides a comprehensive set of reference-standard algorithms, functions, and apps for image processing, analysis, visualization, and algorithm development.” Using these tools provides a fast and convenient way to process and analyze images without the need for advanced knowledge of a complex coding language.

PROJECT DESCRIPTION

Importing the image:

Open the MATLAB software and in the application section; download the Image Processing Tool Box. Create a new MATLAB script file. The first few lines clear the workspace to remove any previous variables and clear the command window. It is important that the Current Folder that you are working out of be the folder that contains both the script file and image. The command 'imread' reads an image and converts it into a "3-dimensional" matrix in the RGB color space. The 'imread' function converts this into a matrix (Rows x Columns x RGB). The final dimension (RGB) corresponds to a red, green and blue intensity level. 'imshow' is used to view the produced image in a new window.

Segmenting the image:

The first step taken is to divide the image into three images based on the intensities of each red, green and blue component within the image. This is Color Based Image Segmentation. Image Thresholding takes an intensity image and converts it into a binary image based on the level desired. A value between 0 and 1 determines which pixels (based on their value) will be set to a 1 (white) or 0 (black)). For choosing the best value suited for your application, right-click on the value and at the top of the menu and select "Increment Value and Run Section". Set the increment value to 0.01 and choose the best value at which to threshold. Image Thresholding at 0.37 is seen in the output. We can see that the image has been segmented between the object we desire to measure and the background.

Segmentation continued (to remove noise):

There is quite a bit of ‘noise’ and we need to clean the image up significantly to improve the accuracy of our diameter measurement. Procedures taken to clean up the image and provide a more uniform blob to analyze are seen in the output. Blobs in this document are any collection of white pixels that touch to create a cohesive and distinct object.

Measuring the image:

The final image is the result of all image segmentation and cleanup procedures to provide one distinct and cohesive blob, which represents the ball in the original image. Having the original image in a binary form such as this will make it easy for other functions built into MATLAB to quickly analyze the region and a host of different information. The ‘regionprops’ function is the tool that will provide the ‘MajorAxisLength’ of the blob in the image. As you can see, by not suppressing line 39 with a semi-colon, the diameter is displayed in the Command Window.

Result:

The diameter is now displayed in the Command Window to be approximately 177 pixels across. This was verified by using the ‘imdistline’ function. As you can see, the value calculated by the code was very close to the manual measurement.

CODE

```
%Importing image
```

```
clc;
```

```
clear all;
```

```
obj = imread('ball3.jfif');
```

```
imshow(obj)
```

```
%Dividing image 'obj' into its RGB intensities (giving filters to the image)
```

```
red = obj(:,:,1);
```

```
green = obj(:,:,2);
```

```
blue = obj(:,:,3);
```

```
figure(1)
```

```
subplot(2,2,1); imshow(obj); title('Original image');
```

```
subplot(2,2,2); imshow(red); title('Red plane');
```

```
subplot(2,2,3); imshow(green); title('Green plane');
```

```
subplot(2,2,4); imshow(blue); title('Blue plane');
```

```
%Threshold the blue plane of image
```

```
figure(2)
```

```
level = 0.37;
```



```
bw2 = imbinarize(blue,level);
subplot(2,2,1); imshow(bw2); title('Blue plane thresholded');

%Fill holes
fill = imfill(bw2,'holes');
subplot(2,2,2); imshow(fill); title('Holes filled');

%Remove blobs on border of image
clear = imclearborder(fill);
subplot(2,2,3); imshow(clear); title('Remove blobs on border');

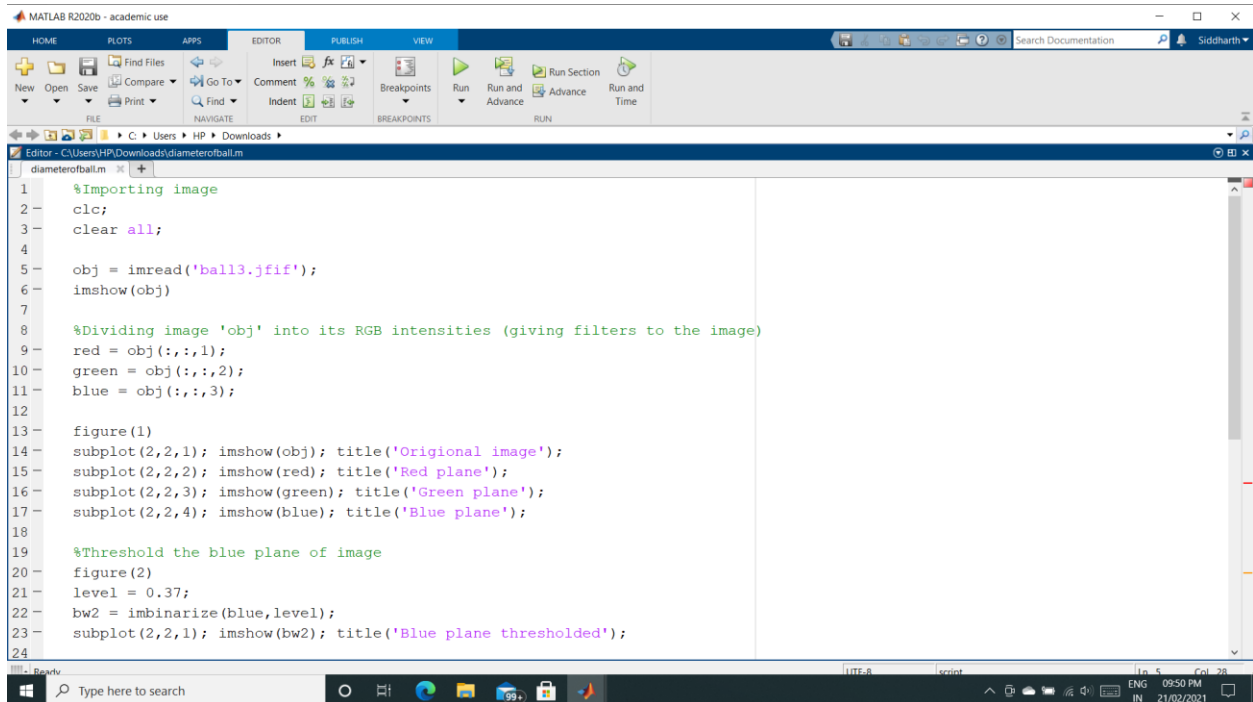
%Remove blobs that are smaller than 7 pixels across
se = strel('disk',7);
open = imopen(fill,se);
subplot(2,2,4); imshow(open); title('Remove small blobs');

%Measure object diameter
diameter = regionprops(open,'MajorAxisLength')

%show result
figure(3)
imshow(obj)
d = imdistline; %Includes line to physically measure the object
```

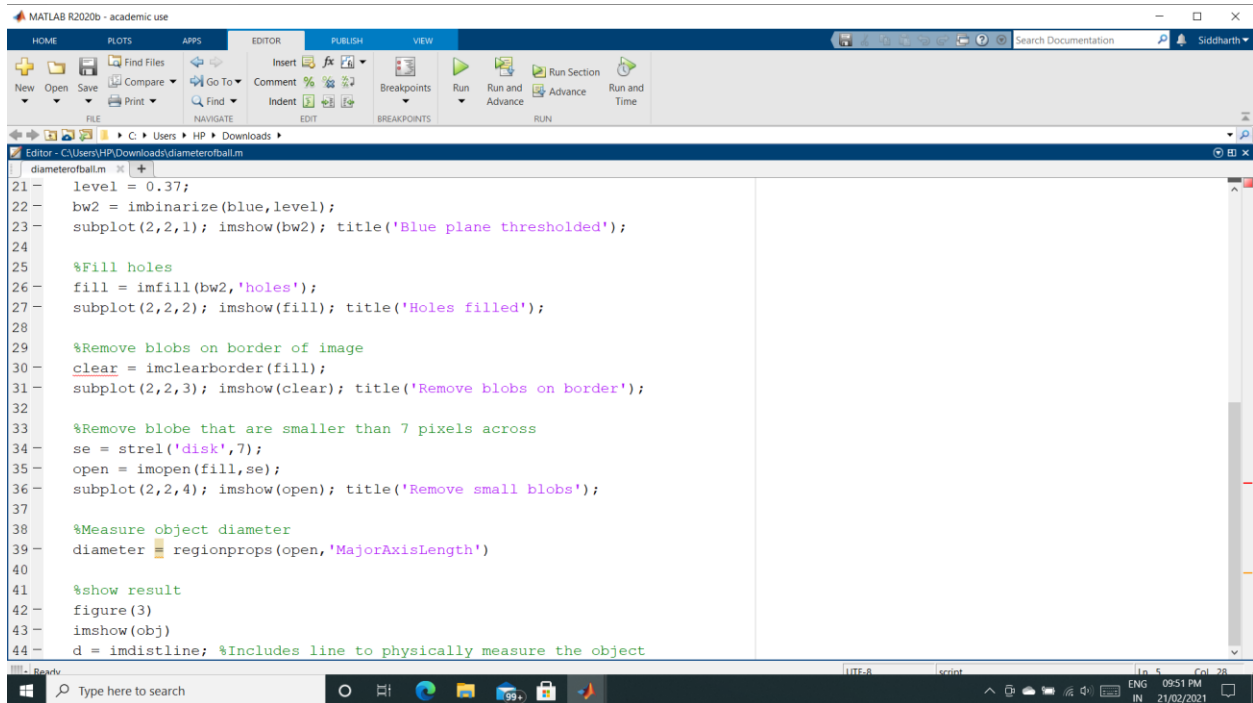
SCREENSHOTS

Screenshots of code:



Editor - C:\Users\HP\Downloads\diameterofball.m

```
1 %Importing image
2 clc;
3 clear all;
4
5 obj = imread('ball3.jfif');
6 imshow(obj)
7
8 %Dividing image 'obj' into its RGB intensities (giving filters to the image)
9 red = obj(:,:,1);
10 green = obj(:,:,2);
11 blue = obj(:,:,3);
12
13 figure(1)
14 subplot(2,2,1); imshow(obj); title('Original image');
15 subplot(2,2,2); imshow(red); title('Red plane');
16 subplot(2,2,3); imshow(green); title('Green plane');
17 subplot(2,2,4); imshow(blue); title('Blue plane');
18
19 %Threshold the blue plane of image
20 figure(2)
21 level = 0.37;
22 bw2 = imbinarize(blue,level);
23 subplot(2,2,1); imshow(bw2); title('Blue plane thresholded');
24
```

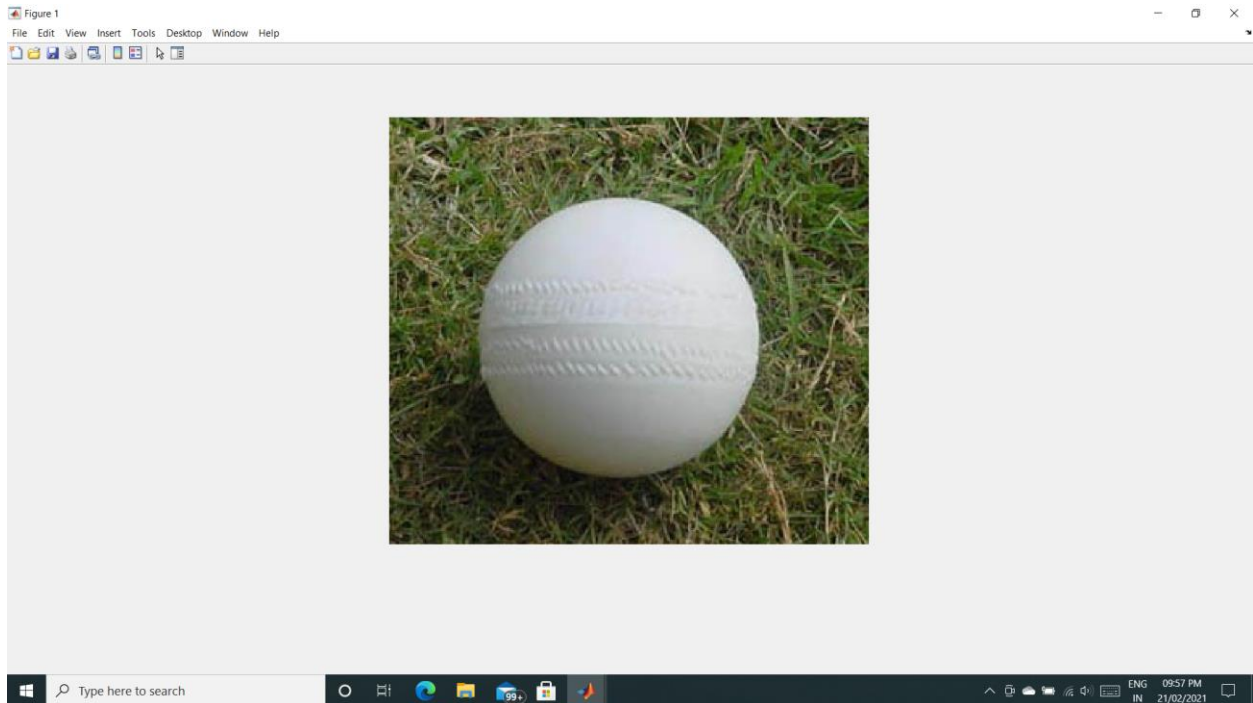


Editor - C:\Users\HP\Downloads\diameterofball.m

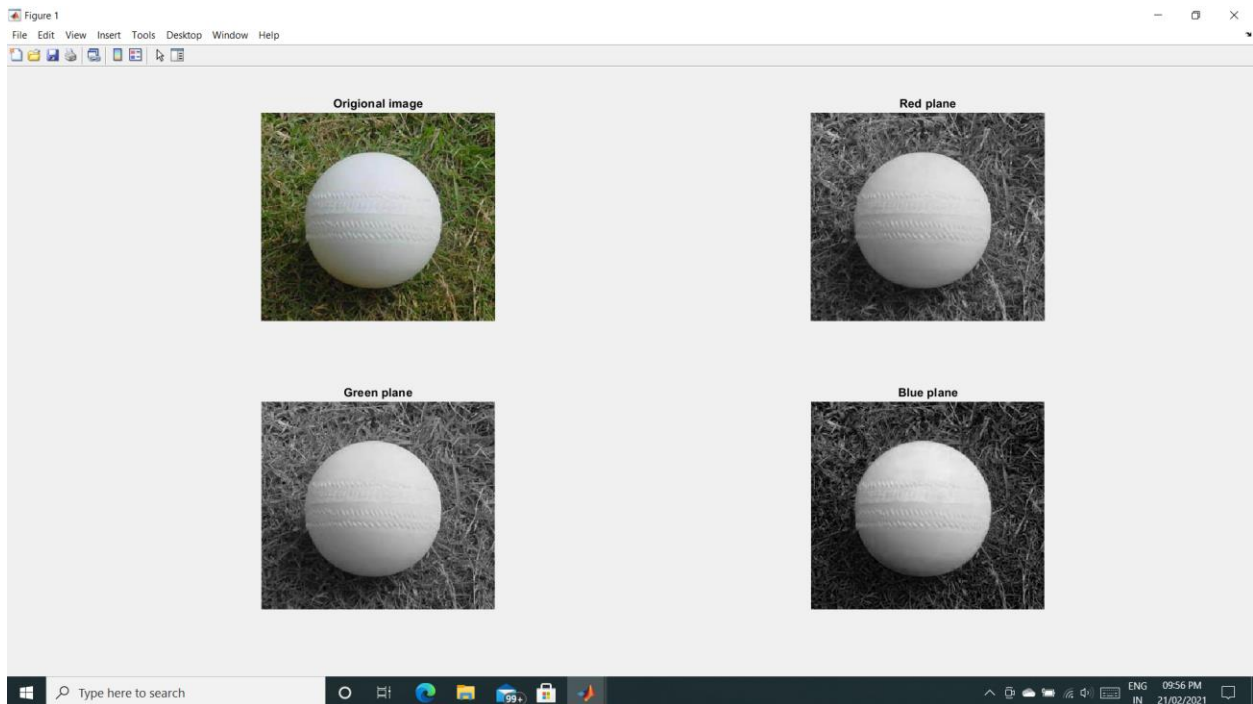
```
21 level = 0.37;
22 bw2 = imbinarize(blue,level);
23 subplot(2,2,1); imshow(bw2); title('Blue plane thresholded');
24
25 %Fill holes
26 fill = imfill(bw2,'holes');
27 subplot(2,2,2); imshow(fill); title('Holes filled');
28
29 %Remove blobs on border of image
30 clear = imclearborder(fill);
31 subplot(2,2,3); imshow(clear); title('Remove blobs on border');
32
33 %Remove blobe that are smaller than 7 pixels across
34 se = strel('disk',7);
35 open = imopen(fill,se);
36 subplot(2,2,4); imshow(open); title('Remove small blobs');
37
38 %Measure object diameter
39 diameter = regionprops(open,'MajorAxisLength')
40
41 %show result
42 figure(3)
43 imshow(obj)
44 d = imdistline; %Includes line to physically measure the object
```

Screenshots of outputs:

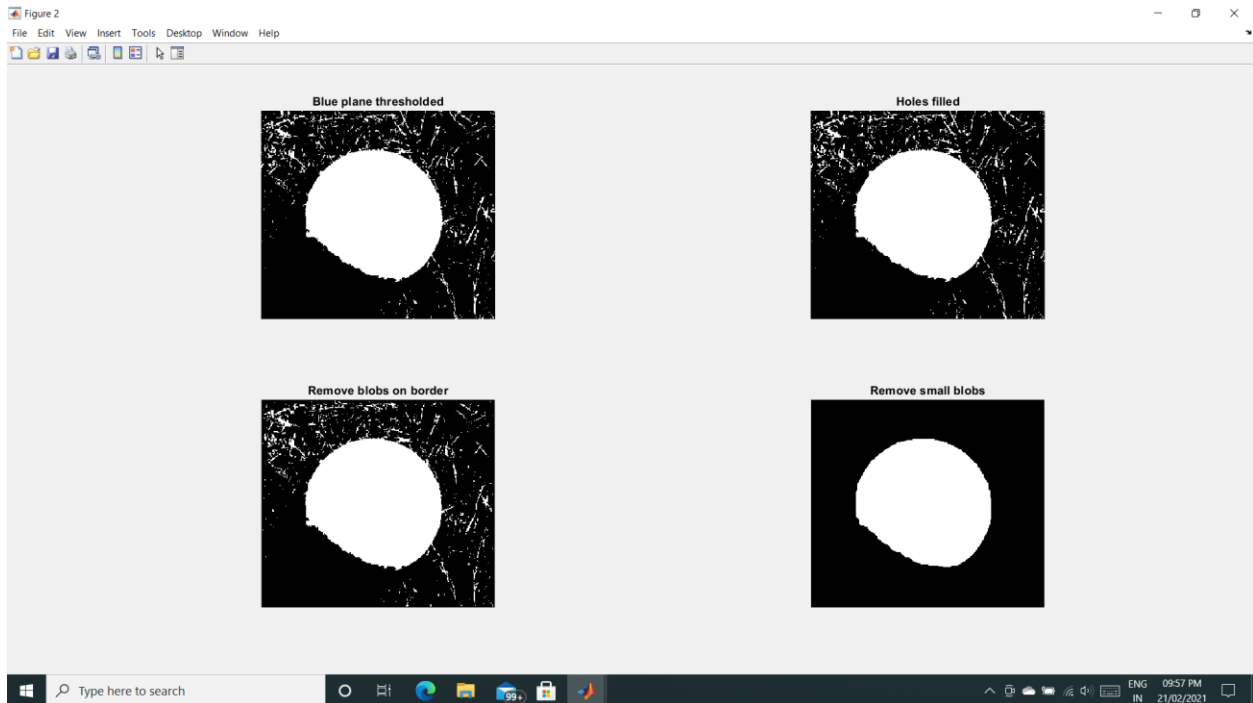
After importing the image:



After dividing the image into its RGB intensities:



After segmenting the images:



MajorAxisLength of image mentioned in command window:

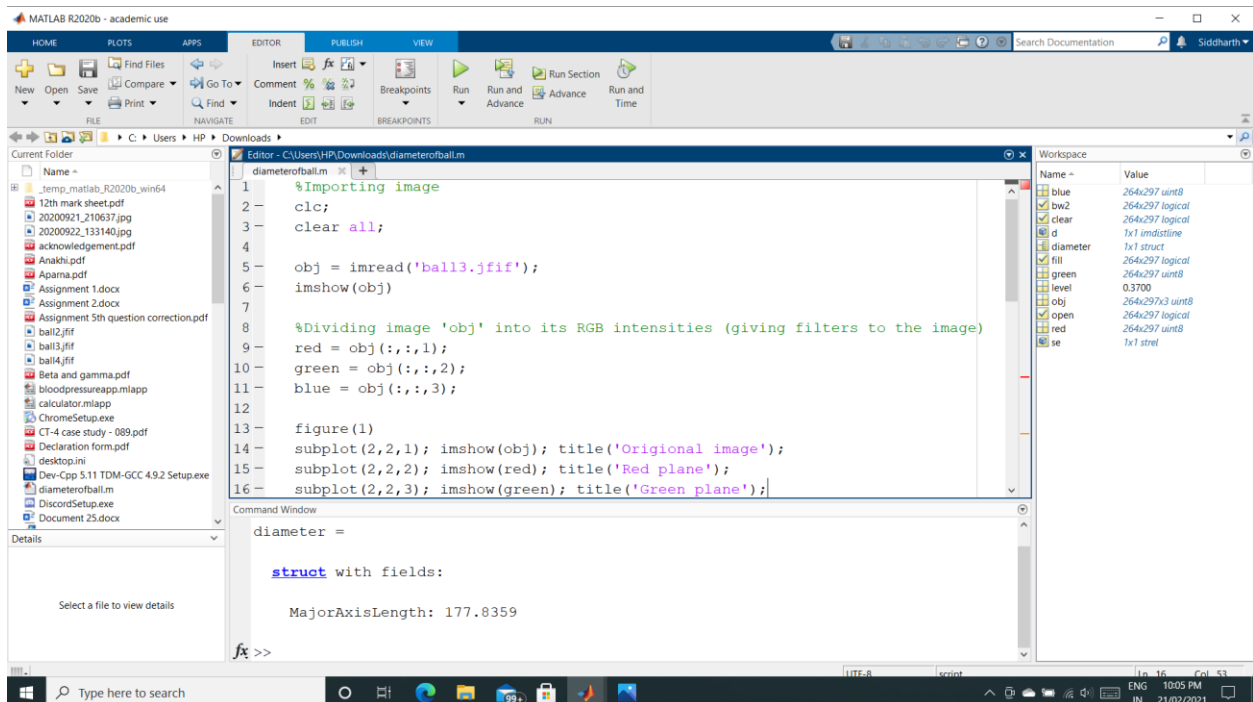
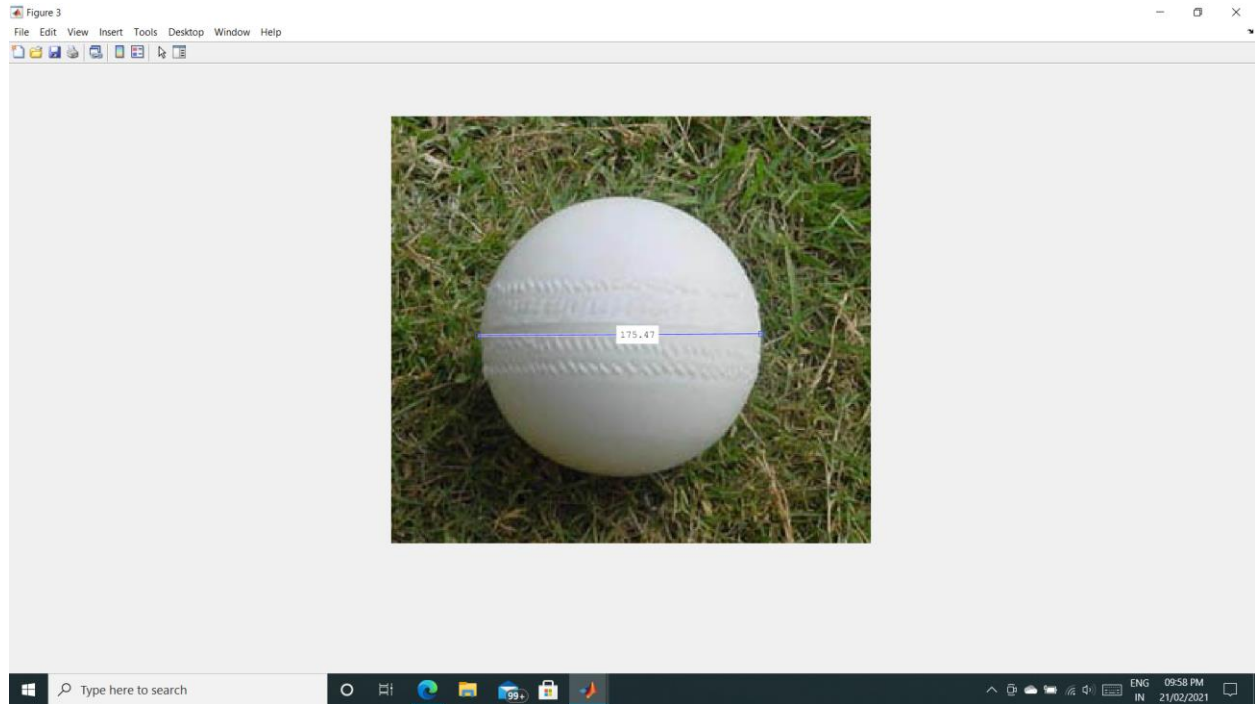


Image with indistline function used:



CONCLUSION

There are a multitude of options within the 'regionprops' function that can output other measurements besides the 'MajorAxisLength'.

However, one of the drawbacks are that not all 3-dimensional figures can be measured properly under all circumstances. 3-d images which have sides facing directly can be measured feasibly. This works best with spherical shapes.

REFERENCE

<https://in.mathworks.com/videos/image-processing-made-easy-81718.html>