

**DEPARTMENT OF  
ELECTRONICS AND COMMUNICATION ENGINEERING  
Faculty of Engineering and Technology  
SRM Institute of Science and Technology**

**MINI PROJECT REPORT  
ODD SEMESTER, 2020-2021**

**Lab Code & Sub Name** : 18ECC205J, Analog and Digital Communication

**Year & Semester** : 3<sup>RD</sup> Year, 5<sup>th</sup> Semester

**Project Title** : Image Processing Using MATLAB To Measure Objects

**Lab Supervisor** : **Mrs.Priyalakshmi.B**

**Team Members** :  
1.YAJNISH.M (RA1811004010291)  
2. NEIL MATHUR (RA1811004010292)  
3. ESHWAR VAMSI KRISHNA (RA1811004010293)

# **IMAGE PROCESSING USING MATLAB TO MEASURE OBJECTS IN AN IMAGE VIRTUALLY**

**KEYWORDS:** MATLAB, Image Processing, Measurement, Diameter, Toolbox

**AIM/OBJECTIVE:** 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 object's measurement (diameter).

**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.

## **SOFTWARE USED:**

MATLAB 2020

Image Processing Toolbox

## **HARDWARE REQUIREMENT:**

**Operating System:** windows XP or better, Mac OS X Lion or better

**Processors:** Intel or AMD x86 processor

**Disk Space:** 4 GB or better

**RAM:** 2048 MB at least recommended

**Graphics Card:** Hardware accelerated graphics card supporting OpenGL 3.3 with 1 GB GPU memory recommended.

## PROCESS:

### #1 IMPORTING AN IMAGE:

In the application section of the MATLAB software, Image Processing Tool Box is to be downloaded. Now a new MATLAB script file is created wherein the code is typed. The first few lines clear the workspace to remove any previous variables and clear the command window.

It is important to note that the Current Folder that is being worked 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 image used in this tutorial is *baseball.jpg (Figure 1)*, which is a 500 by 600 pixel image.

The *imread* function converts this into a matrix that is 500x600x3 (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.



**Figure 1; Original Image, baseball.jpg**

A screenshot of the MATLAB R2020a - academic use interface. The editor window shows a script named 'AddMiniProject.m' with the following code:

```
1 %% ADC MINI PROJECT
2 %% PROJECT TITLE: IMAGE PROCESSING USING MATLAB TO MEASURE OBJECTS WITHIN AN IMAGE VIRTUALLY
3 %% MEMBERS: YAJNISH (RA1811004010291), NEIL MATHUR (RA1811004010292), ESHWAR YAMSI KRISHNA (RA1811004010293)
4
5 %% importing of the image
6
7 clear;
8 clc;
9
10 obj = imread('baseball.jpg');
11 imshow(obj);
12
```

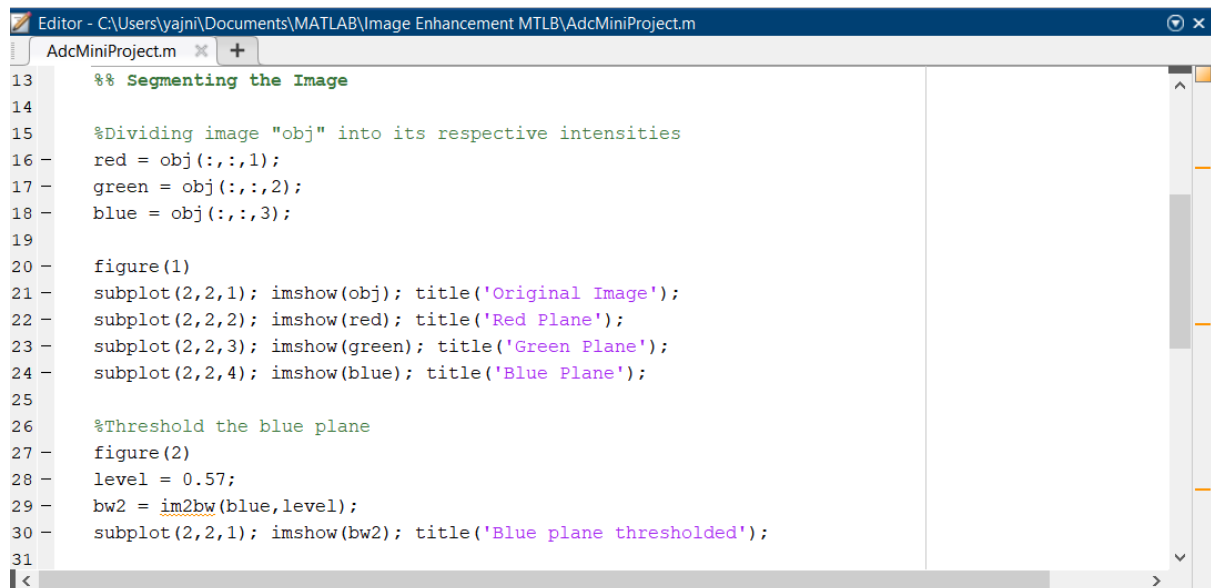
**Figure 2; Code to Import an Image**

## #2 SEGMENTING THE IMAGE

This process basically differentiates the background from the desired objects. 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.

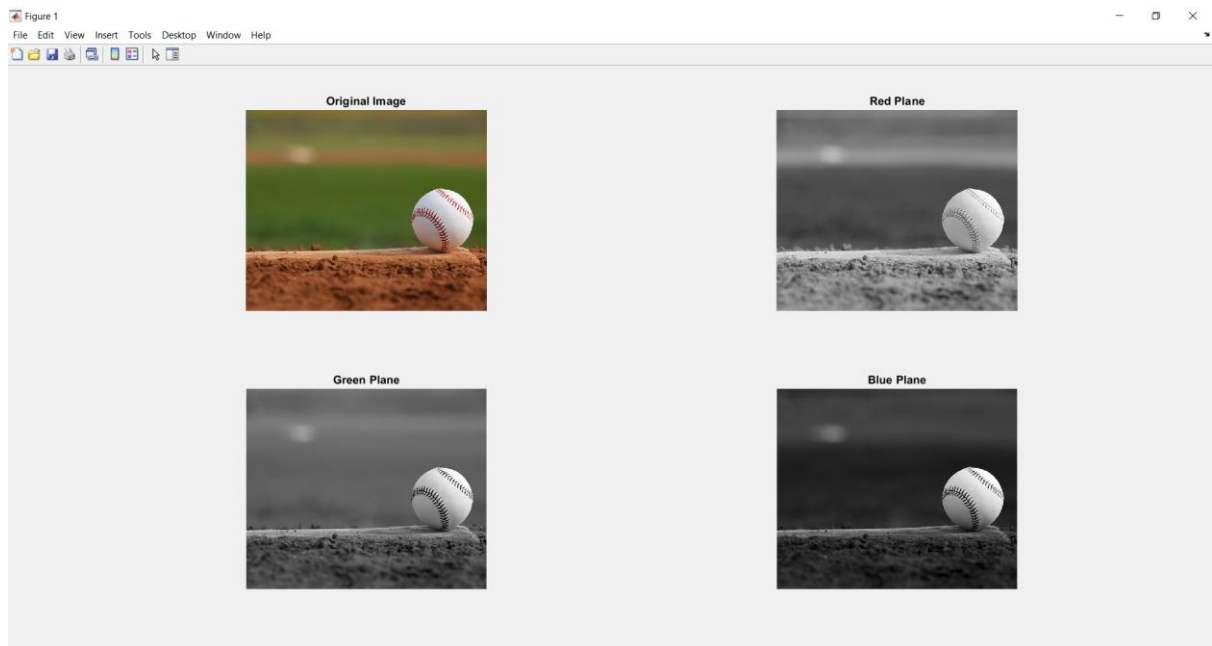
This is Color Based Image Segmentation. We can see from (*Figure 4*) that the blue plane is the best choice to use for Image Thresholding because it provides the most contrast between the foreground and the background.

Image Thresholding takes an intensity image and converts it into a binary image. A value between 0 and 1 determines which pixels (based on their value) will be set to a 1 (white) or 0 (black)). Now we can see that the image (Top-right of *Figure 5*) has been segmented between the object we desire to measure and the background.

A screenshot of a MATLAB Editor window titled 'Editor - C:\Users\yajni\Documents\MATLAB\Image Enhancement MTLB\AdcMiniProject.m'. The window contains a script named 'AdcMiniProject.m' with the following code:

```
13 %% Segmenting the Image
14
15 %Dividing image "obj" into its respective intensities
16 - red = obj(:,:,1);
17 - green = obj(:,:,2);
18 - blue = obj(:,:,3);
19
20 - figure(1)
21 - subplot(2,2,1); imshow(obj); title('Original Image');
22 - subplot(2,2,2); imshow(red); title('Red Plane');
23 - subplot(2,2,3); imshow(green); title('Green Plane');
24 - subplot(2,2,4); imshow(blue); title('Blue Plane');
25
26 %Threshold the blue plane
27 - figure(2)
28 - level = 0.57;
29 - bw2 = im2bw(blue,level);
30 - subplot(2,2,1); imshow(bw2); title('Blue plane thresholded');
31
```

**Figure 3; Code to segment an image**



**Figure 4; Result after Segmentation**

### #3 REMOVAL OF THE NOISE

As we can see from the top-left image in (*Figure 5*) there is a lot of “noise” and we need to clean the image up significantly to improve the accuracy of the diameter measurement. Blobs in this document are any collection of white pixels that touch to create a cohesive and distinct object. The presence of blobs drastically impact the measurement and thus they need to be removed before proceeding with the task.



**Figure 5; Removal of Noise**

```
Editor - C:\Users\yajni\Documents\MATLAB\Image Enhancement MTLB\AdcMiniProject.m
AdcMiniProject.m
28 - level = 0.57;
29 - bw2 = im2bw(blue,level);
30 - subplot(2,2,1); imshow(bw2); title('Blue plane thresholded');
31
32 %% Removing the Noise
33
34 %Fill any holes
35 - fill = imfill(bw2,'holes');
36 - subplot(2,2,2); imshow(fill); title('Holes filled');
37
38 %Remove any blobs on the border of the image
39 - clear = imclearborder(fill);
40 - subplot(2,2,3); imshow(clear); title('Remove blobs on border');
41
42 %Remove blobs that are small across
43 - se = strel('disk',7);
44 - open = imopen(fill,se);
45 - subplot(2,2,4); imshow(open); title('Remove small blobs');
46
```

**Figure 6; Code to remove the Noise**

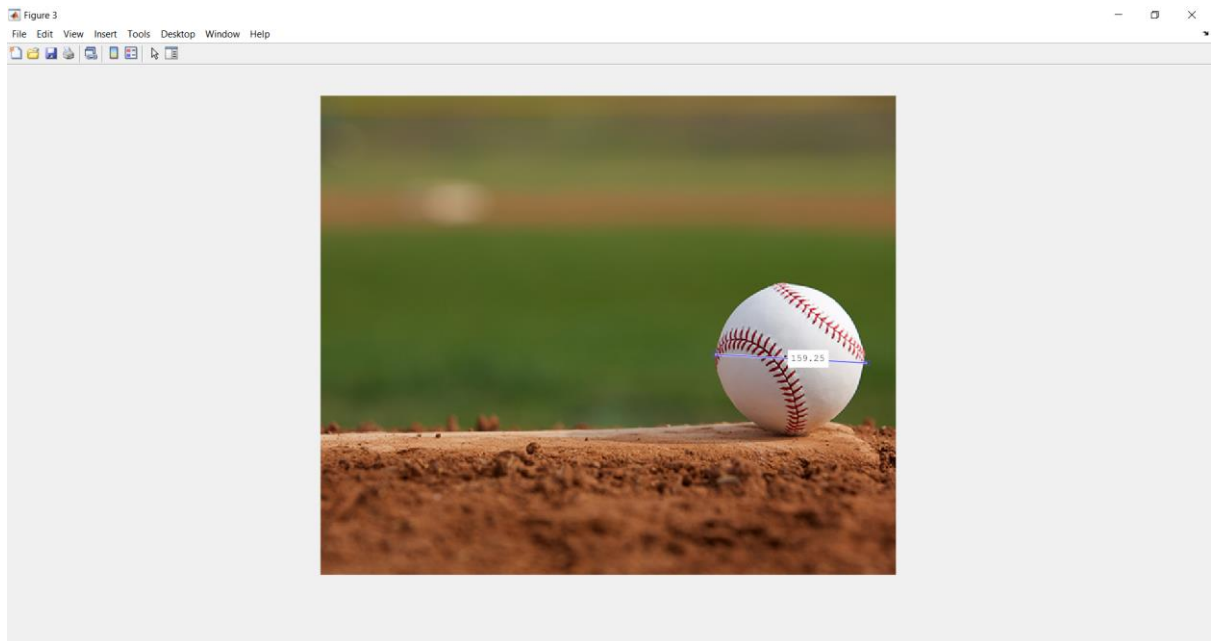
## #4 MEASURING THE IMAGE

The image in the bottom-right corner of (Figure 5) 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. This basically acts as a virtual scale in order to measure the desired object in the image.

```
Editor - C:\Users\yajni\Documents\MATLAB\Image Enhancement MTLB\AdcMiniProject.m
AdcMiniProject.m
37
38 %Remove any blobs on the border of the image
39 - clear = imclearborder(fill);
40 - subplot(2,2,3); imshow(clear); title('Remove blobs on border');
41
42 %Remove blobs that are small across
43 - se = strel('disk',7);
44 - open = imopen(fill,se);
45 - subplot(2,2,4); imshow(open); title('Remove small blobs');
46
47 %% Measuring the object diameter
48
49 - diameter = regionprops(open, 'MajorAxisLength')
50
51 %Show result
52 - figure(3)
53 - imshow(obj)
54 - d = imdistline; %Include a line to physically measure measure the ball
55
```

## Figure 7; Code for Measurement (Attached Above)

**RESULT:** The diameter is now displayed in the Command Window to be approximately 158.6 pixels across. This was verified in *Figure 8* by using the *imdistline* function in (*Figure 7*). As we can see between the two figures, the value calculated by the code was very close to the manual measurement in *Figure 8*.



**Figure 8: Original Image after Manual Measurement: (159.25)**

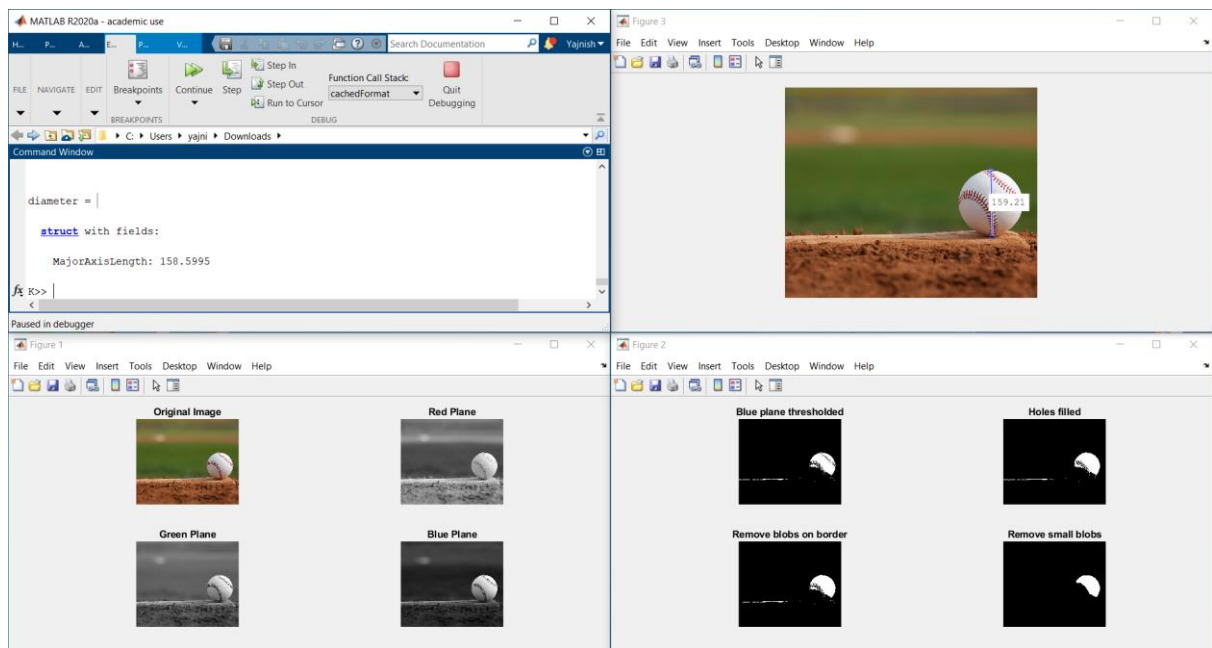
```
Command Window
44  42  41  40  39

diameter =
  struct with fields:
    MajorAxisLength: 158.5995

fx >>
```

**Figure 9: Diameter Output in the Command Window (158.60)**





**Figure 10; An overview of all the Outputs & Verification**

## CONCLUSION:

There are a multitude of options within the *regionprops* function that can output other measurements besides the *MajorAxisLength*. This technique of Image Processing is used or rather implemented in AR (Augmented Reality), mobile applications wherein we can measure the size of the desired objects instantly and identify what the object is as well.

## REFERENCES:

*Image Processing Made Easy*. Perf. Andy Thé. Mathworks. N.p., 15 Oct. 2014. Web. 07 Nov. 2014.