Measuring the Diameter of an object using MATLAB Image Processing Toolbox

Introduction

Our main aim objective of this project is to write a MATLAB code 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.

Code

```
%% Import Image

clear;
clc;

obj = imread('circle.jpg');
imshow(obj)

%% Segment Image

%Divide image "obj" into its respective RGB intensities
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
figure(2)
level = 0.37;
bw2 = imbinarize(blue,level);
subplot(2,2,1); imshow(bw2); title('Blue plane thresholded');

%% Remove Noise

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

%Remove and blobs on the border of the image
clear = imclearborder(fill);
```

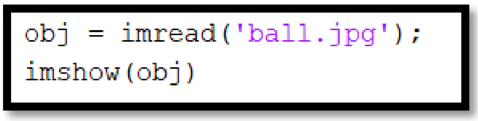
```
subplot(2,2,3); imshow(clear); title('Remove blobs on
border');

%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


%Show result
figure(3)
imshow(obj)
d = imdistline(gca); %Include a line to physically measure the
ball
api=iptgetapi(d);
%
pause();
%
dist=api.getDistance();
u=menu('Choose Measuring
unit','Pixels','Centimeters','Inches');
if(u==1)
    fprintf('The length of the object is: %0.2f Pixels',dist)
elseif(u==2)
    dist_cm=dist*0.0264583333;
    fprintf('The length of the object is: %0.2f CMs',dist_cm)
else
    dist_in=dist*0.01041666666667;
    fprintf('The length of the object is: %0.2f
Inches',dist_in)
end
```

Working & Screenshots

1. Importing Image

The command ‘imread’ reads the image and converts it into a 3-d matrix in the RGB color space. The image used in this project is ‘ball.jpg’ which is 583 x 822 pixel image. The ‘imread’ function converts this into a matrix that is 583x822x3 (Rows x Columns x RGB). The final dimension (RGB) corresponds to a red, green and blue intensity level.

The Command ‘imshow’ is used to view the produced image in a new window.



```
obj = imread('ball.jpg');
imshow(obj)
```

Figure 1 Import Code

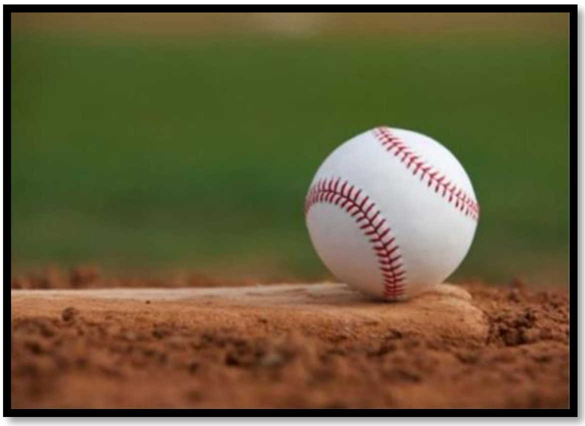
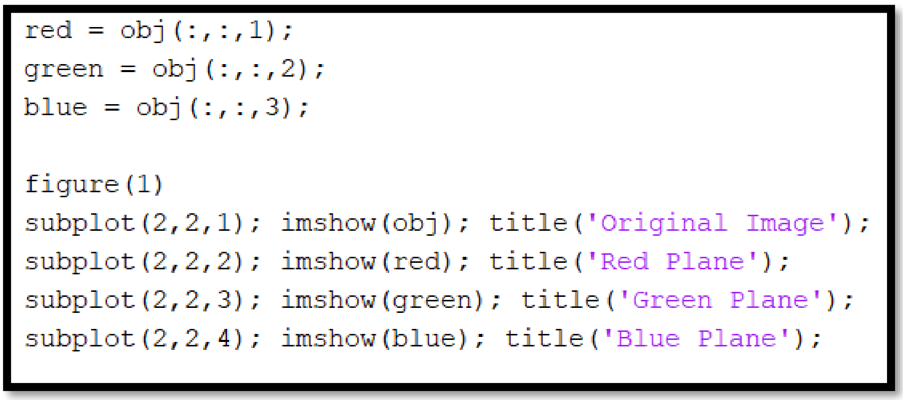


Figure 2 Import Output

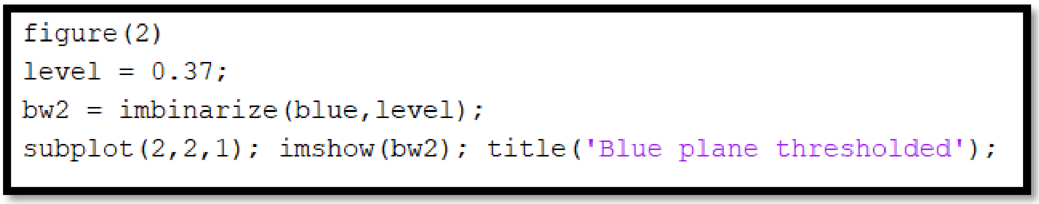
2. Segmenting image

We will segment the image into a binary image to differentiate 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. By visual comparison, the blue plane is the best choice to use for Image Thresholding because it provides the most contrast between the desired object (foreground) and the background. 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 will be set to a 1 (white) or 0 (black).



```
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');
```

Figure 3 Segment Code



```
figure(2)
level = 0.37;
bw2 = imbinarize(blue,level);
subplot(2,2,1); imshow(bw2); title('Blue plane thresholded');
```

Figure 4 Threshold Code

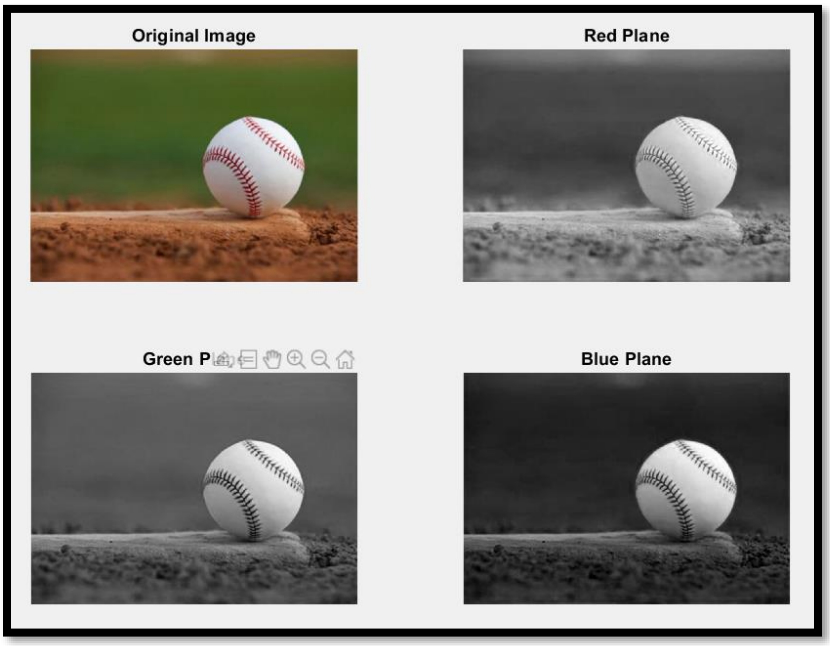
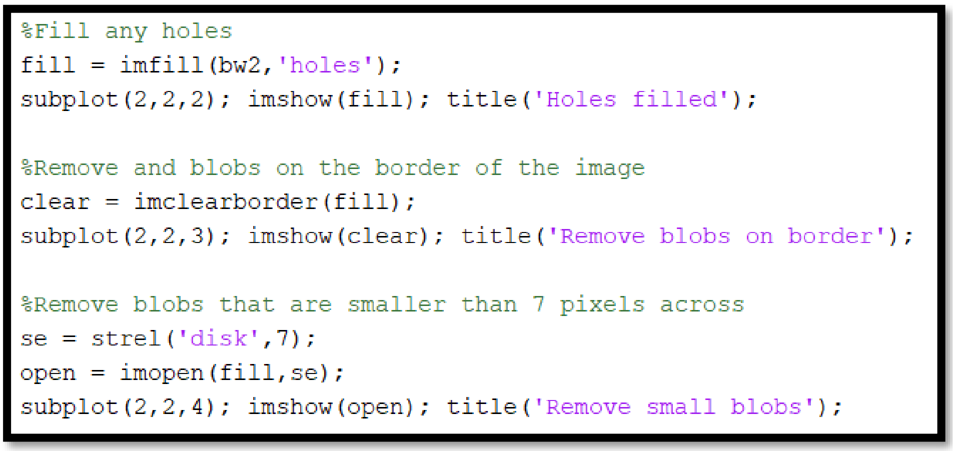


Figure 5 Segment Output

3. Remove Noise

There is quite a bit of noise as seen in the image. It is necessary to clean the image up significantly to improve the accuracy of our diameter measurement. Here, blobs are any collection of white pixels that touch to create a cohesive and distinct object.



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

%Remove and blobs on the border of the 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');
```

Figure 6 Remove Noise Code

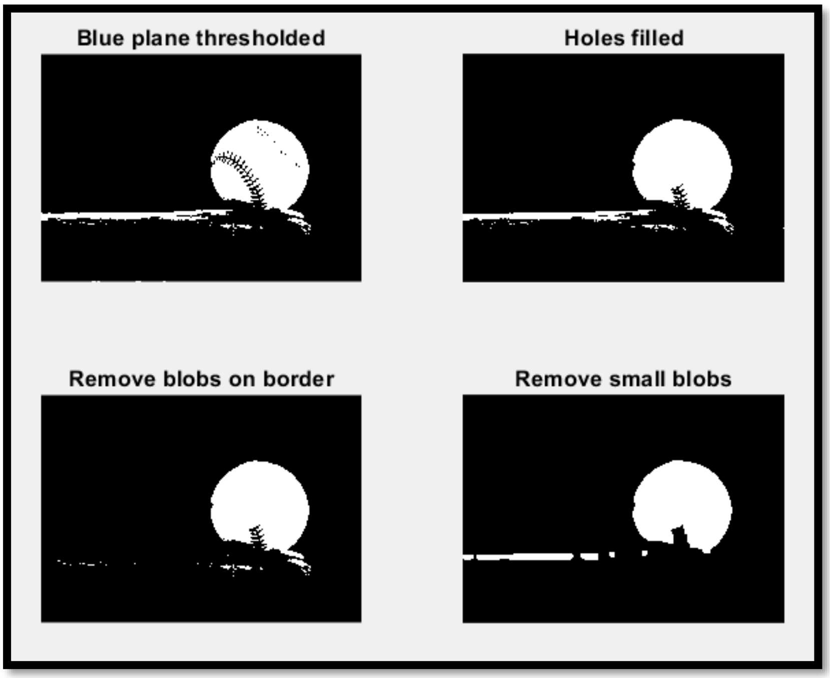
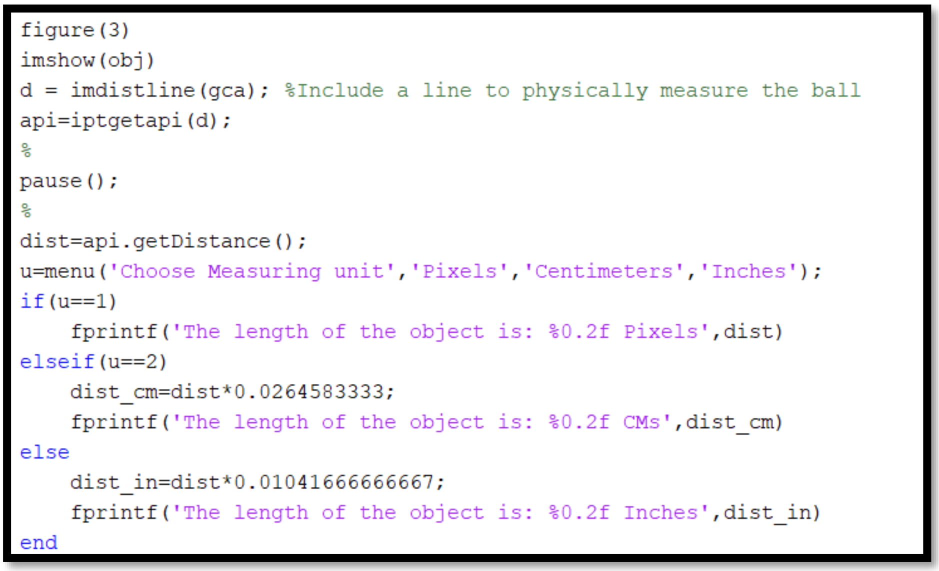


Figure 7 Remove Noise Output

4. Measuring Object

The command ‘imdistline’ will help us in manually verifying the diameter, and after clicking a button, a popup will ask the measuring unit in which we want our measurement to be displayed.



```
figure(3)
imshow(obj)
d = imdistline(gca); %Include a line to physically measure the ball
api=iptgetapi(d);
%
pause();
%
dist=api.getDistance();
u=menu('Choose Measuring unit','Pixels','Centimeters','Inches');
if(u==1)
    fprintf('The length of the object is: %0.2f Pixels',dist)
elseif(u==2)
    dist_cm=dist*0.0264583333;
    fprintf('The length of the object is: %0.2f CMs',dist_cm)
else
    dist_in=dist*0.01041666666667;
    fprintf('The length of the object is: %0.2f Inches',dist_in)
end
```

Figure 8 Measuring Code

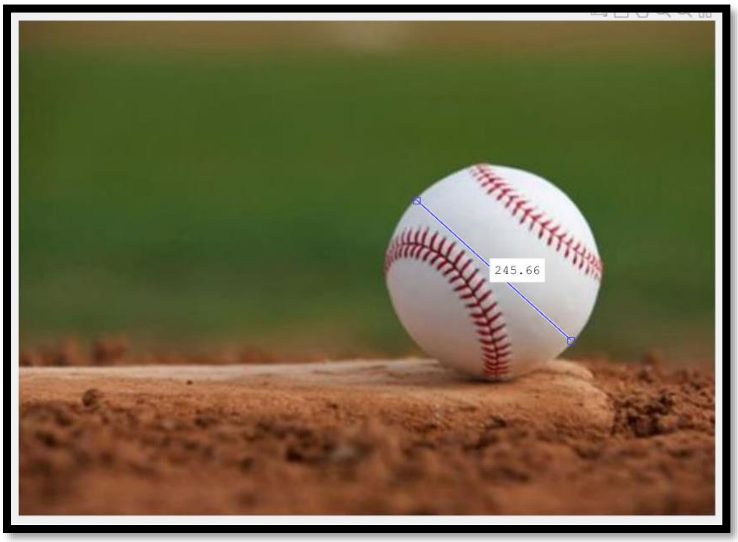


Figure 9 'imdistline' output

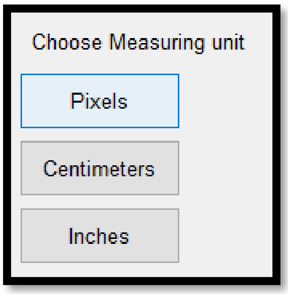
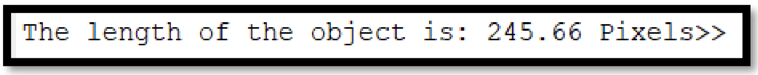


Figure 10 Measuring Unit Output

Result



```
The length of the object is: 245.66 Pixels>>
```

Figure 11 Result