|  |  |
| --- | --- |
| **Project Title: Using MATLAB to Measure the Diameter of an Object within an Image**  **Submitted By: F16CS06 | 22 | 38 | 58**  **Submitted To : Ma’am Aisha Zahid Junejo** |  |

**ACKNOWLEDGEMENT**

All praises & thanks to “Almighty Allah”, who has shown us the right path through every thick and thins.

Firstly, we would like to acknowledge the mighty presence of Allah, who enabled us to complete this Report. I would also like to express my deep gratitude to “Ma’am Aisha Zahid Junejo ”, who guided us & taught us impeccable ways to complete our desired goal. The assistance provided by other teachers helped us a lot. Also, the prayers of our parents could not be ignored as they have played a vital role in our success. The teamwork was executive and it also does help us in increasing the area of our thinking.

My special thanks are extended to our chairman “**Sir Sheeraz Memon**”, who assigned us, wonderful teachers, and is still pushing his effort to make our future better

**Regards: F16CS06|22|38|58**

## 

**TABLE OF CONTENTS:**

**ABSTRACT.......................................................................................................** 3

**INTRODUCTION...............................................................................................**3

**OBJECTIVE……………………………………………………………………………………………..3**

**LITERATURE REVIEW.....................................................................................3**

**TOOLS…………………………………………………………………………………………………..**

**WORKING METHODLOGY...............................................................................**

**RESULT...................................................................................................**8

**RECOMMENDATIONS**

**REFERENCES**

**Using MATLAB to Measure the Diameter of an Object within an Image**

**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.

**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.”. The Image Processing Toolbox is a collection of functions that extend the capabilities of the MATLAB’s numeric computing environment. The toolbox supports a wide range of image processing operations, including: – Geometric operations – Neighborhood and block operations – Linear filtering and filter design – Transforms – Image analysis and enhancement – Binary image operations – Region of interest operations 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.

**OBJECTIVE:**

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.

**LITERATURE REVIEW:**

**MATLAB:**

MATLAB is an abbreviation of MATrix LABoratory. It is a high performance language for technical computing”. MATLAB is a multi-paradigm numerical computing environment. A programming language developed by MathWorks. [Cleve Moler](https://en.wikipedia.org/wiki/Cleve_Moler), the chairman of the [computer science](https://en.wikipedia.org/wiki/Computer_science) department at the [University of New Mexico](https://en.wikipedia.org/wiki/University_of_New_Mexico), started developing MATLAB in the late 1970s. He designed it to give his students access to [LINPACK](https://en.wikipedia.org/wiki/LINPACK) and [EISPACK](https://en.wikipedia.org/wiki/EISPACK) without them having to learn [Fortran](https://en.wikipedia.org/wiki/Fortran). It soon spread to other universities and found a strong audience within the [applied mathematics](https://en.wikipedia.org/wiki/Applied_mathematics) community. [Jack Little](https://en.wikipedia.org/wiki/John_N._Little), an engineer, was exposed to it during a visit Moler made to [Stanford University](https://en.wikipedia.org/wiki/Stanford_University) in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in [C](https://en.wikipedia.org/wiki/C_(programming_language)) and founded [MathWorks](https://en.wikipedia.org/wiki/MathWorks) in 1984 to continue its development. These rewritten libraries were known as JACKPAC. In 2000, MATLAB was rewritten to use a newer set of libraries for matrix manipulation, [LAPACK](https://en.wikipedia.org/wiki/LAPACK). MATLAB was first adopted by researchers and practitioners in [control engineering](https://en.wikipedia.org/wiki/Control_engineering), Little's specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching of [linear algebra](https://en.wikipedia.org/wiki/Linear_algebra) and [numerical analysis](https://en.wikipedia.org/wiki/Numerical_analysis), and is popular amongst scientists involved in [image processing](https://en.wikipedia.org/wiki/Image_processing). MATLAB is a powerful engineering environment and language. It is a powerful tool in Engineering problem solving, data analysis, modeling and visualization. MATLAB is the vehicle used by this course to enhance Programming approach by solving problems. MATLAB is a programming platform designed specifically for engineers and scientists. The heart of MATLAB is the MATLAB language, a matrix-based language allowing the most natural expression of computational mathematics.

What can you do with MATLAB?

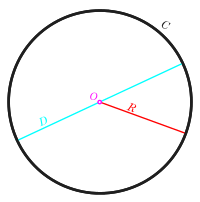
Using MATLAB, you can:

* Analyze data
* Develop algorithms
* Create models and applications

The language, apps, and built-in math functions enable you to quickly explore multiple approaches to arrive at a solution. MATLAB lets you take your ideas from research to production by deploying to enterprise applications and embedded devices, as well as integrating with Simulink and Model-Based Design.

**DIAMETER :**

In [geometry](https://en.wikipedia.org/wiki/Geometry), a **diameter** of a [circle](https://en.wikipedia.org/wiki/Circle) is any straight [line segment](https://en.wikipedia.org/wiki/Line_segment) that passes through the center of the circle and whose endpoints lie on the circle. It can also be defined as the longest [chord](https://en.wikipedia.org/wiki/Chord_(geometry)) of the circle. In more modern usage, the length of a diameter is also called the diameter. In this sense one speaks of *the* diameter rather than *a* diameter (which refers to the line itself), because all diameters of a circle or sphere have the same length, this being twice the [radius](https://en.wikipedia.org/wiki/Radius) **r**. {\displaystyle d=2r\quad \Rightarrow \quad r={\frac {d}{2}}.}For a [convex shape](https://en.wikipedia.org/wiki/Convex_set) in the plane, the diameter is defined to be the largest distance that can be formed between two opposite [parallel lines](https://en.wikipedia.org/wiki/Parallel_lines)tangent to its boundary, and the *width* is often defined to be the smallest such distance. Both quantities can be calculated efficiently using [rotating calipers](https://en.wikipedia.org/wiki/Rotating_calipers).[[1]](https://en.wikipedia.org/wiki/Diameter#cite_note-1) For a [curve of constant width](https://en.wikipedia.org/wiki/Curve_of_constant_width) such as the [Reuleaux triangle](https://en.wikipedia.org/wiki/Reuleaux_triangle" \o "Reuleaux triangle), the width and diameter are the same because all such pairs of parallel tangent lines have the same distance. For an [ellipse](https://en.wikipedia.org/wiki/Ellipse), the standard terminology is different. A diameter of an ellipse is any [chord](https://en.wikipedia.org/wiki/Chord_(geometry)) passing through the center of the ellipse.[[2]](https://en.wikipedia.org/wiki/Diameter#cite_note-2) For example, [conjugate diameters](https://en.wikipedia.org/wiki/Conjugate_diameters) have the property that a tangent line to the ellipse at the endpoint of one of them is parallel to the other one. The longest diameter is called the [major axis](https://en.wikipedia.org/wiki/Major_axis). The word "diameter" is derived from [Greek](https://en.wikipedia.org/wiki/Ancient_Greek) διάμετρος (*diametros*), "diameter of a circle", from διά (*dia*), "across, through" and μέτρον (*metron*), "measure".[[3]](https://en.wikipedia.org/wiki/Diameter#cite_note-3) It is often abbreviated **DIA**, **dia**, **d**, or **⌀**.

{\displaystyle d=2r\quad \Rightarrow \quad r={\frac {d}{2}}.}[](https://en.wikipedia.org/wiki/File:Circle-withsegments.svg)

The definitions given above are only valid for circles, spheres and convex shapes. However, they are special cases of a more general definition that is valid for any kind of *n*-dimensional convex or non-convex object, such as a [hypercube](https://en.wikipedia.org/wiki/Hypercube) or a set of scattered points. The **diameter** of a [subset](https://en.wikipedia.org/wiki/Subset) of a [metric space](https://en.wikipedia.org/wiki/Metric_space) is the [least upper bound](https://en.wikipedia.org/wiki/Supremum) of the set of all distances between pairs of points in the subset. So, if *A* is the subset, the diameter is

[sup](https://en.wikipedia.org/wiki/Supremum) { d(*x*, *y*) | *x*, *y* ∈ *A* } .

If the [distance function](https://en.wikipedia.org/wiki/Distance_function) d is viewed here as having [codomain](https://en.wikipedia.org/wiki/Codomain) **R** (the set of all [real numbers](https://en.wikipedia.org/wiki/Real_number)), this implies that the diameter of the [empty set](https://en.wikipedia.org/wiki/Empty_set) (the case *A* = ∅) equals −∞ ([negative infinity](https://en.wikipedia.org/wiki/Negative_infinity)). Some authors prefer to treat the empty set as a special case, assigning it a diameter equal to 0,[[4]](https://en.wikipedia.org/wiki/Diameter#cite_note-4) which corresponds to taking the codomain of d to be the set of nonnegative reals. For any solid object or set of scattered points in n-dimensional [Euclidean space](https://en.wikipedia.org/wiki/Euclidean_space), the diameter of the object or set is the same as the diameter of its [convex hull](https://en.wikipedia.org/wiki/Convex_hull). In medical [parlance](https://en.wikipedia.org/wiki/Idiom#Parlance) concerning a lesion or in geology concerning a rock, the diameter of an object is the supremum of the set of all distances between pairs of points in the object. In [differential geometry](https://en.wikipedia.org/wiki/Differential_geometry), the diameter is an important global [Riemannian](https://en.wikipedia.org/wiki/Riemannian_geometry) [invariant](https://en.wikipedia.org/wiki/Invariant_(mathematics)). In plane geometry, a diameter of a [conic section](https://en.wikipedia.org/wiki/Conic_section) is typically defined as any chord which passes through the conic's center; such diameters are not necessarily of uniform length, except in the case of the circle, which has [eccentricity](https://en.wikipedia.org/wiki/Eccentricity_(mathematics)) *e* = 0.

**TOOLS:**

* **HARDWARE TOOLS:**
* Windows XP or better Or Mac OS X Lion or better
* Intel or AMD x86 processor
* 4 GB or better disk space
* 2048 MB RAM at least recommended
* Hardware accelerated graphics card supporting OpenGL 3.3
* 1 GB GPU memory recommended
* **SOFTWARE TOOLS:**
* **MATLAB 2016a:**

MATLAB is a high-level language and interactive environment for computer computation, visualization, and programming. MATLAB is a programming platform designed specifically for engineers and scientists. The heart of MATLAB is the MATLAB language, a matrix-based language allowing the most natural expression of computational mathematics

* **IMAGE PROCESSING TOOLBOX:**

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.”. The Image Processing Toolbox is a collection of functions that extend the capabilities of the MATLAB’s numeric computing environment. The toolbox supports a wide range of image processing operations, including: – Geometric operations – Neighborhood and block operations – Linear filtering and filter design – Transforms – Image analysis and enhancement – Binary image operations – Region of interest operations 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

**WORKING:**

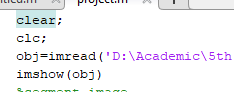
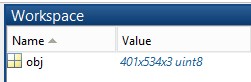
**METHODOLOGY:**

**Step#01: Import Image**

Open the MATLAB software and in the application section; download the Image Processing Tool Box. Create a new MATLAB script file. Refer to *Figure 2* to begin adding code to import the desired image to measure into the MATLAB workspace. 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 image used in this tutorial is *ball.jpg (Figure 1),* which is a 534 by 401 pixel image. The *imread* function converts this into a matrix that is 401x534x3 (Rows x Columns x RGB). The final dimension (RGB) corresponds to a red, green and blue intensity level. Use *imshow* to view the produced image in a new window

****

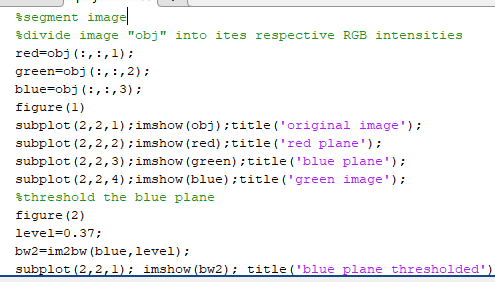
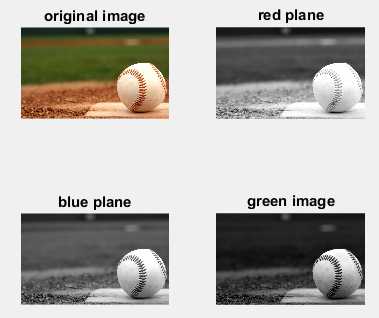
*Figure 1; Original Image, ball.jpg*

*Figure 2; Code to Import an Image Figure 3; obj variable in workspace*

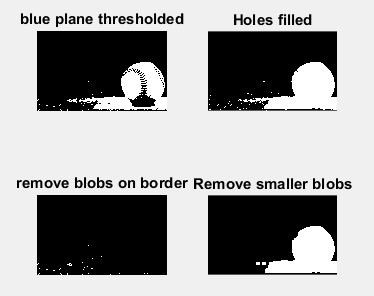
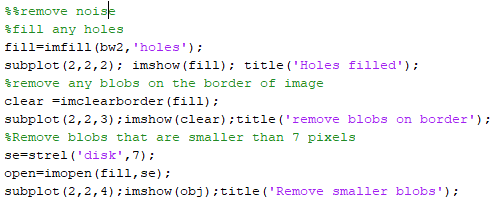
**Step#02:Segment Image**

Follow the code in *Figure 4* below to 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. The RGB triplet is a three-element row vector whose elements specify the intensities of the red, green, and blue components of the color. You can see from *Figure 5* that 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 (See line 25). A value between 0 and 1 determines which pixels (based on their value) will be set to a 1 (white) or 0 (black)). To choose 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. *Figure 5* shows the result of the Image Thresholding at 0.37. You can see that the image (Top-right of *Figure* 6) has been segmented between the object we desire to measure and the background. Here to convert image in a binary image **“im2bw “** is used. [BW](https://www.mathworks.com/help/images/ref/im2bw.html#d120e80904) = im2w([I](https://www.mathworks.com/help/images/ref/im2bw.html" \l "d120e80650),[level](https://www.mathworks.com/help/images/ref/im2bw.html#d120e80850)) converts the grayscale image I to binary image BW, by replacing all pixels in the input image with luminance greater than level with the value 1 (white) and replacing all other pixels with the value 0 (black).

 *Figure 4; Code to Segment the Image* Figure 5; color thresholding

**Step#03: Segmentation continued(Remove Noise)**

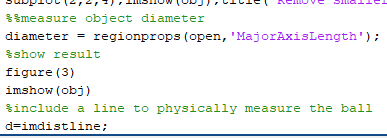
As you can see from the top-left image in *Figure 6* there is quite a bit of “noise” and we need to clean the image up significantly to improve the accuracy of our diameter measurement. Refer to *Figures 6 & 7* on the procedures taken to clean up the image and provide a more uniform blob to analyze. Blobs in this document are any collection of white pixels that touch to create a cohesive and distinct object.



*Figure 7; Code to Remove Noise Figure 6; clean up image*

**Step#04:Measuring Image**

The image in the bottom-right corner of *Figure 6* 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 45 *(Figure 8)* with a semi-colon, the diameter is displayed in the Command Window *(Figure 10)*.



*Figure 8; Code to Measure the Object Diameter*

**RESULT:**

The diameter is now displayed in the Command Window to be approx. 170 pixels across. This was verified in *Figure 9* by using the *imdistline* function in line 50 *(Figure 8).* As you can see between the two figures, the value calculated by the code was very close to the manual measurement in *Figure 9.*

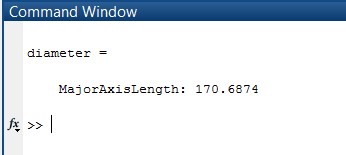
**** 

Figure 9; Original Image Manually Measure *Figure 10; Diameter Output in the Command Window*

**COMPLETE CODE:**

clear;

clc;

obj=imread('E:\MUET\Semester#5\Analogue and Digital Signal Processing (ADSP)\ADSP Project\title3.jpg');

imshow(obj)

%segment image

%divide image "obj" into ites 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('blue plane');

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

%threshold the blue plane

figure(2)

level=0.37;

bw2=im2bw(blue,level);

subplot(2,2,1); imshow(bw2); title('Blue Plane Thresholder');

%%remove noise

%fill any holes

fill=imfill(bw2,'Holes');

subplot(2,2,2); imshow(fill); title('Holes filled');

%remove any blobs on the 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

se=strel('disk',7);

open=imopen(fill,se);

subplot(2,2,4);imshow(obj);title('Remove smaller blobs');

%%measure object diameter

diameter = regionprops(open,'MajorAxisLength');

%show result

figure(3)

imshow(obj)

%include a line to physically measure the ball

d=imdistline;

**RECOMMENDATIONS:**

There are a multitude of options within the *regionprops* function that can output other measurements besides the *MajorAxisLength*. Type *Help regionprops* in the Command Window to get more information about the function. *Help* can be used with any function to obtain more information. Provide comments for each section of code as demonstrated. Comments provide a quick explanation of each section of code that helps other users debug and understand the code.

**REFERENCES:**

**1]**[https://in.mathworks.com/help/images/detect-and-measure-circular-objects-in-an image.html;jsessionid=24186caeb1efe5fe56c047626c94](https://in.mathworks.com/help/images/detect-and-measure-circular-objects-in-an%20image.html;jsessionid=24186caeb1efe5fe56c047626c94)

2] Image Processing Made Easy. Perf. Andy Thé. Mathworks. N.p., 15 Oct. 2014. Web. 07 Nov. 2014. <http://www.mathworks.com/videos/image-processing-made-easy81718.html>.