	FACULTY : FTMK **UNIVERSITI TEKNIKAL MALAYSIA MELAKA**	
IMAGE PROCESSING AND PATTERN RECOGNITION FINAL PROJECT		
BTI 3313	**SEMESTER II**	**SESSION 2019/2020**
MINI PROJECT: **COUNTING OBJECT IN IMAGE**		
DATE OF SUBMISSION	17st JUNE 2020	
FACULTY & PROGRAM	FTMK (3 BITI)	
NAME OF GROUP MEMBERS & MATRIX NUMBER	**1. QUEK YAO JING (B031810136)** **2. TEOH KAH LUN (B031810069)**	
NAME OF LECTURER	DR. ASMALA BIN AHMAD	
PRESENTATION VIDEO	**https://youtu.be/zOap22-nI6Y**	

1.0 Introduction about the function

Object counting in an image is a task where we need to identify the number of items inside an image. In our project, we introduced a few phases in completing the task. The phases are read the image. This process required the use of imread() to read image files into an array.

Next, we will need to convert an image to grayscale, thresholding image into binary. To do the global thresholding, we can first view the object in histogram form and determine which one is the best value to a threshold. Good threshold value allows us to separated the background and images efficiently, which is a great help in the later finding the centroid. In our system, we implement a user-friendly interface sliding windows that range from 0 to 255 to allow easy segmentation of the image.

On top of that, we will apply the morphology opening and closing to clean up the images. Based on our observation, some images will perform better with only either one of the operations. Therefore, opening and closing also factor that affects the accuracy later in image counting. After the process, we will perform color inversion to invert the back and white pixels.

Lastly, we will need to segment out the boundaries of the previous inverted color image. We will then use a looping structure to loop over all the blob found inside the image. Besides, we will also need to find the centroid of the blob and append the number 1 to n based on the blob found.

In retrospect, we also have displayed the quantitative analysis details such as mean intensity, area, perimeter, centroid, and diameter. This information can be useful for the quantitative analysis required by the case study and rubrics.

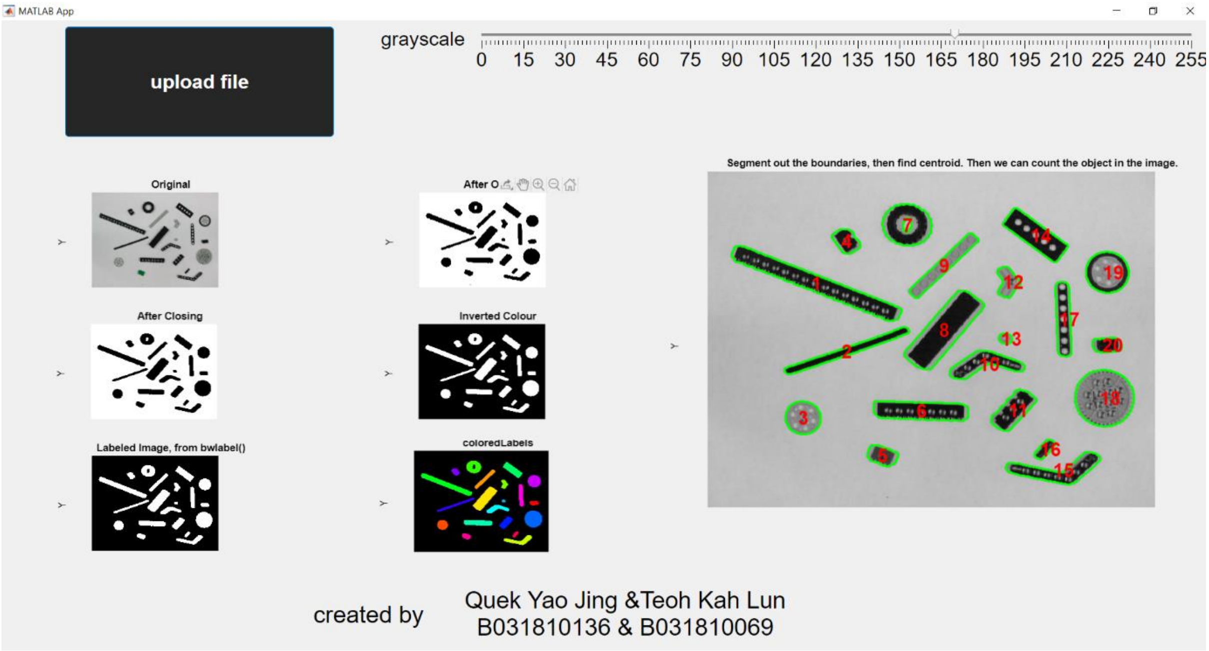
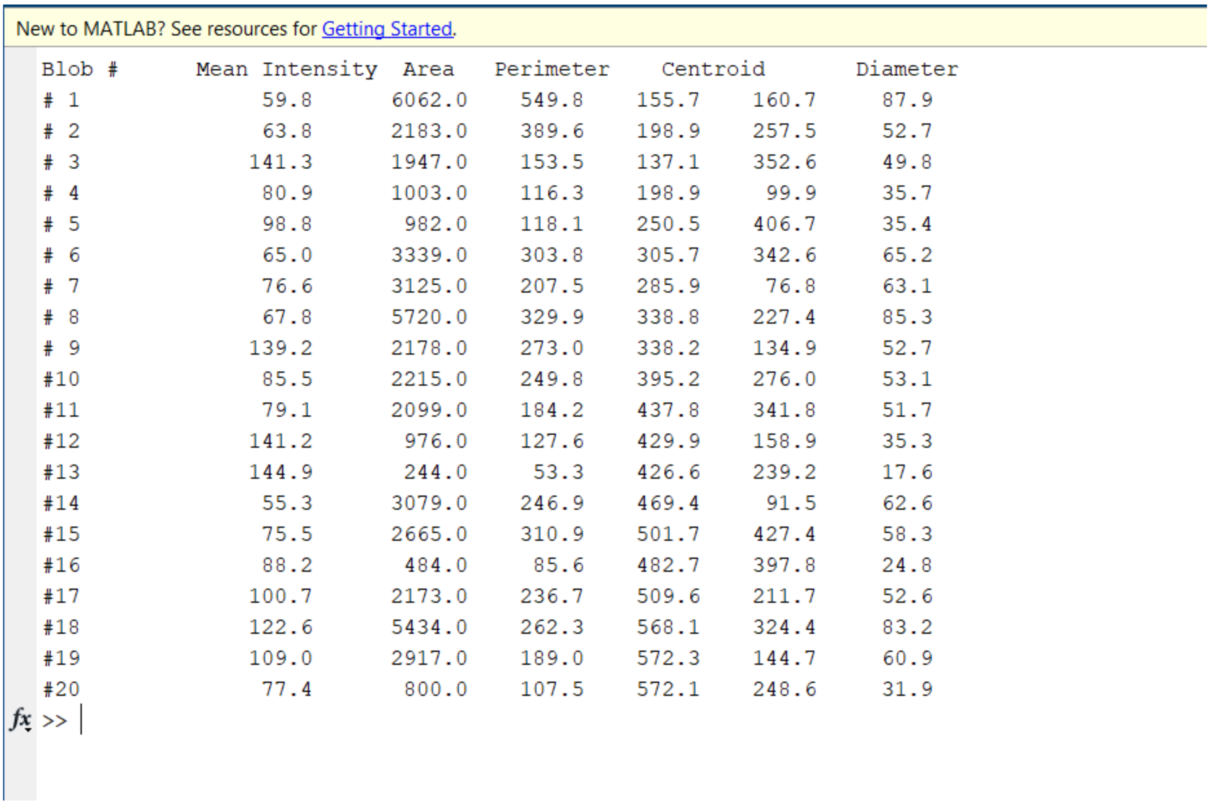


Figure 1.1 shows the Matlab app interface to count objects inside an image.



New to MATLAB? See resources for Getting Started.

```
Blob #      Mean Intensity   Area     Perimeter    Centroid          Diameter
# 1              59.8        6062.0     549.8     155.7     160.7      87.9
# 2              63.8        2183.0     389.6     198.9     257.5      52.7
# 3             141.3        1947.0     153.5     137.1     352.6      49.8
# 4              80.9        1003.0     116.3     198.9      99.9      35.7
# 5              98.8         982.0     118.1     250.5     406.7      35.4
# 6              65.0        3339.0     303.8     305.7     342.6      65.2
# 7              76.6        3125.0     207.5     285.9      76.8      63.1
# 8              67.8        5720.0     329.9     338.8     227.4      85.3
# 9             139.2        2178.0     273.0     338.2     134.9      52.7
#10              85.5        2215.0     249.8     395.2     276.0      53.1
#11              79.1        2099.0     184.2     437.8     341.8      51.7
#12             141.2         976.0     127.6     429.9     158.9      35.3
#13             144.9         244.0      53.3     426.6     239.2      17.6
#14              55.3        3079.0     246.9     469.4      91.5      62.6
#15              75.5        2665.0     310.9     501.7     427.4      58.3
#16              88.2         484.0      85.6     482.7     397.8      24.8
#17             100.7        2173.0     236.7     509.6     211.7      52.6
#18             122.6        5434.0     262.3     568.1     324.4      83.2
#19             109.0        2917.0     189.0     572.3     144.7      60.9
#20              77.4         800.0     107.5     572.1     248.6      31.9
fx >> |
```

Figure 1.2 shows the Matlab command window displays the quantitive analysis of the images.

2.0 How to use the application

Tested the application with below software and hardware:

Version : matlab 2019b

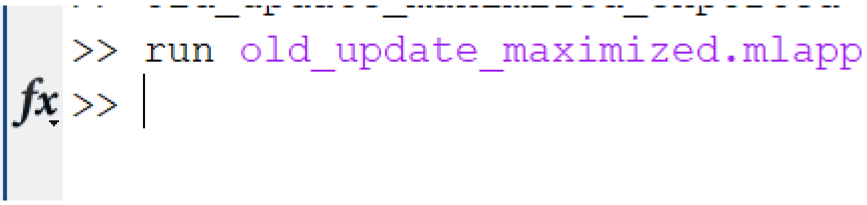
OS, window10

*Resoultion 1920 * 1080, display as 100% instead of 150%*

Graphic card : gtx 1050
processor : 8400H

Instruction of how to use the application

First, “ xxxx.m”, it will be open as Graphical User Interface mode. If interested in the source code, may open “xxxx.mlpp” as app designeer mode view. The program could be start with double click, F5 run or command window, in figure 1, with start <directory><file name>



```
>> run old_update_maximized.mlapp
>>
```

Figure 2.1 CLI commnad line

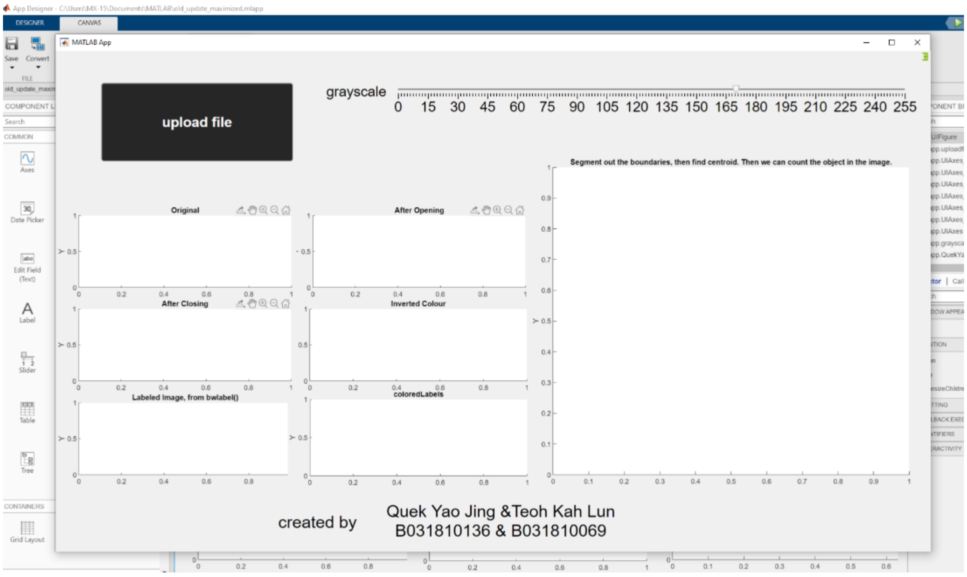


Figure 2.2 initial status of GUI

Second, GUI will be appear, as shown in figure2. Wait it load finish before maximized to prevent the icon or figure miss configurable.

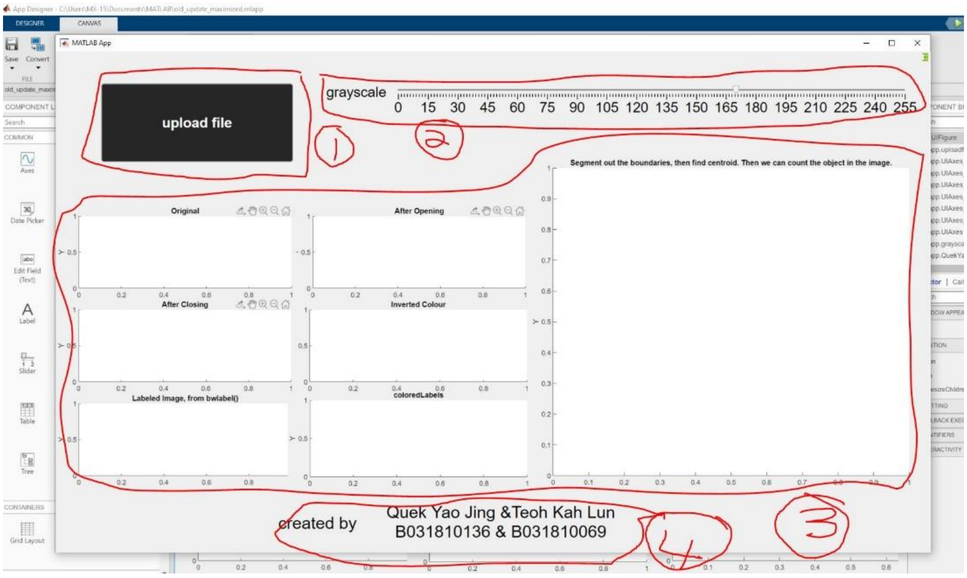
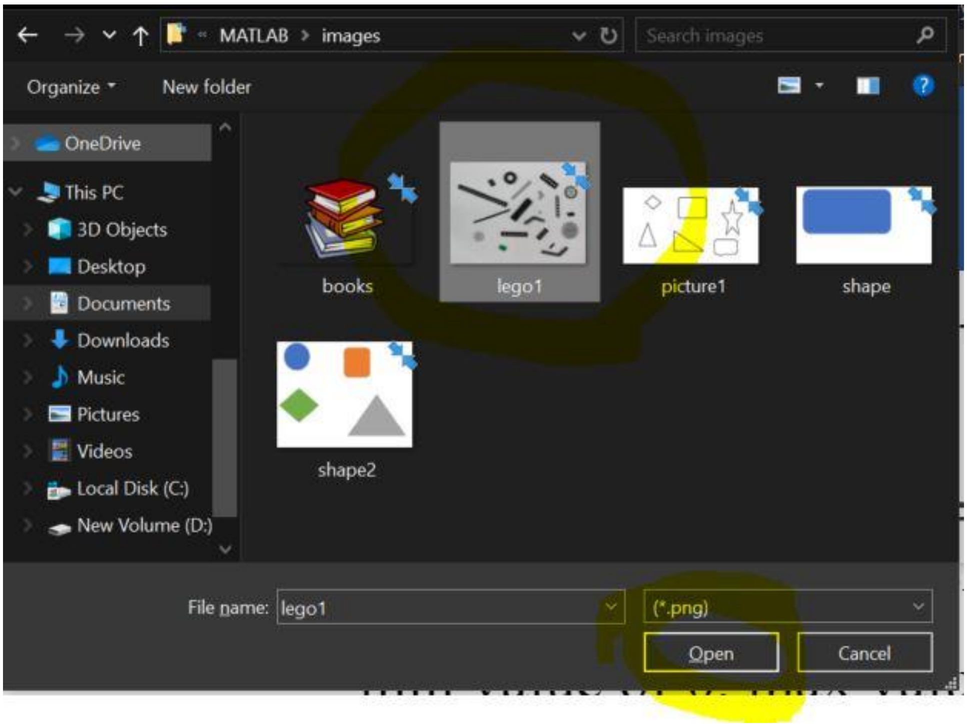


Figure 2.3 parts of our GUI application

Third, in figure 2.3, show the four part of the application, first part will be the upload image file button, mainly tested to support png and jpeg. Part 2 will be the slider input of grayscale value for thresholding of one dimension black and white color, default value 170, min value of 0, max value as 255. Third part will be the figure, to show the output of the application after load the image file, something just like subplot in the usual matlab script. Fourth part will be the credit, creator of this application.

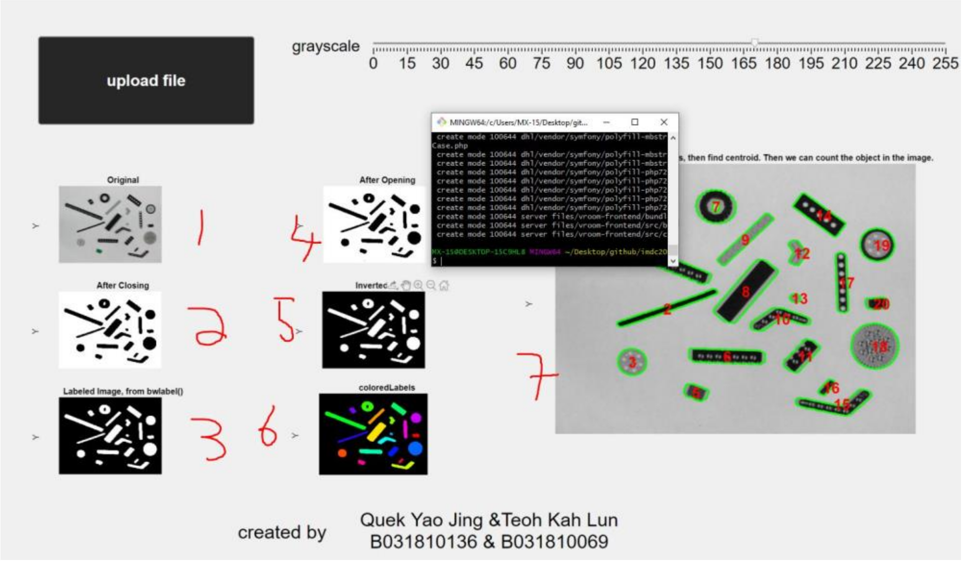


File name: lego1

(*.png)

Figure 2.4 shows example input of load image

Fourthly, start of the application, click upload file -> select a image to test from your local device. Then click open or keyboard <enter>. Just like figure 2.4.



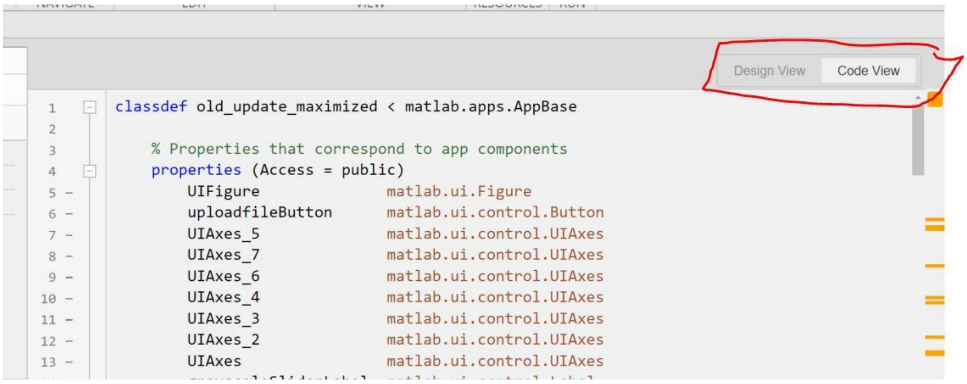
```
 create mode 100644 dhl/vendor/symfony/polyfill-mbstr
Case.php
 create mode 100644 dhl/vendor/symfony/polyfill-mbstr
 create mode 100644 dhl/vendor/symfony/polyfill-mbstr
 create mode 100644 dhl/vendor/symfony/polyfill-php72
 create mode 100644 dhl/vendor/symfony/polyfill-php72
 create mode 100644 dhl/vendor/symfony/polyfill-php72
 create mode 100644 dhl/vendor/symfony/polyfill-php72
 create mode 100644 dhl/vendor/symfony/polyfill-php72
 create mode 100644 server files/vroom-frontend/bundl
 create mode 100644 server files/vroom-frontend/src/b
 create mode 100644 server files/vroom-frontend/src/c

MX-15@DESKTOP-15C9HL8 MINGW64 ~/Desktop/github/imdc20
$ 
```

s, then find centroid. Then we can count the object in the image.

created by Quek Yao Jing &Teoh Kah Lun
B031810136 & B031810069

Figure 2.5 shows loaded image of the application interface

Fifthly, the application will take some times, it is quite fast, with GTX 1050 mobile (3gb) rouhgly, will be loaded in ~10 seconds with not fully loaded resources.



```
classdef old_update_maximized < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure              matlab.ui.Figure
        uploadfileButton      matlab.ui.control.Button
        UIAxes_5              matlab.ui.control.UIAxes
        UIAxes_7              matlab.ui.control.UIAxes
        UIAxes_6              matlab.ui.control.UIAxes
        UIAxes_4              matlab.ui.control.UIAxes
        UIAxes_3              matlab.ui.control.UIAxes
        UIAxes_2              matlab.ui.control.UIAxes
        UIAxes                matlab.ui.control.UIAxes
```

Figure 2.6 shows the view mode of gui application

Lastly. The source could be find in figure 2.6 code view, if the played file not in packed style.