OBJECT'S RGB COLOR EXTRACTION

IQBAL ZULKARNAIN BIN ABDUL AZIZ

This thesis is submitted as partial of requirements for the award of the Bachelor of Electrical Engineering (Power System)

Faculty of Electrical & Electronic Engineering
Universiti Malaysia Pahang

NOVEMBER 2007

"I hereby acknowledge that the scope and quality of this thesis is qualified for the award of the Bachelor Degree of Electrical Engineering (Power Systems)"

Signature : _____

Name : MOHD RAZALI BIN DAUD

Date : 27 NOVEMBER 2007

Signature : _____

Author : IQBAL ZULKARNAIN BIN ABDUL AZIZ

Date : 27 NOVEMBER 2007

ACKNOWLEDGEMENTS

Thank to my Lord, ALLAH S.W.T for blessing me and also give me strength and guidance to complete this project.

Special thanks to my Mother Retipah Hanum and my father Abdul Aziz and also my family members for understanding my work and patient until this project completed.

Special thanks also to my supervisor Mr. Mohd Razali for giving me support and guidance and also supervision in finishing this project.

I would like to thank all my colleagues for giving me idea and support from start of this project until it completed.

Lastly I would like to thanks to my beloved girl friend, Munirah to be patient and caring to me until this project complete

ABSTRACT

Color Blindness, or Color Vision Deficiency, is an eye condition where a person is not able to distinguish certain colors or shades of colors to some degree. As we know, color blind people facing problems that related to color recognizing in their daily life. Due to this problem, the purpose of this project is to help color blind people to recognize the object color around them to make their life better by extracting RGB color in the image. This project is focusing on software development. The color image that need to be recognize will be captured by a camera and processed using MATLAB Image Processing Toolbox software that have been programmed. The method for analyzing the image consist of RGB color pixel statistic to check the major color in the image and RGB color extraction for color definition that used in result display as color reference. The result will be displayed on monitor screen by wording in popup message box.

.

ABSTRAK

Buta warna atau rabun warna, ialah kondisi dimana seseorang tidak dapat membezakan sesetengah warna atau pembayang warna. Seperti yang kita tahu, golongan buta warna ini banyak mengahadapi masalah dalam mengenal warna dalam kehidupan mereka. Merujuk kepada masalah ini, projek ini bertujuan untuk membantu golongan buta warna untuk mengenalpasti warna objek disekeliling bagi memudahkan hidup mereka dengan cara mengekstrak warna RGB dalam sesuatu gambaran. Projek ini menfokus pada pembinaan perisian. Gambar berwarna yang hendak dikenalpasti diambil dengan mengunakan kamera dan diproses mengunakan perisian 'MATLAB Image Processing Toolbox' yang telah diprogramkan. Kaedah yang digunakan untuk menganalisis gambar adalah statistik pixel warna RGB untuk menentukan warna dominan dalam gambar dan kaedah mengekstrak warna RGB untuk dijadikan sebagai warna definasi yang digunakan dalam keputusan sebagai warna rujukan. Keputusannya akan dipamirkan pada skrin komputer dalam tetingkap kotak pesanan dengan perkataan.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.2 Color Blindness

Color Blindness, or Color Vision Deficiency, is an eye condition where a person is not able to distinguish certain colors or shades of colors to some degree. Color Blindness does not mean that a person can only see black and white. A person with color blindness is able to see different color, however they are not able to see some colors due to deficiencies in the eyes.

Color blindness is a hereditary condition but can also be caused by eye diseases, damage to the retina and macula, and aging or when the lens is darkened over time from a cataract. Although there is no absolute treatment for hereditary color blindness, there are methods, techniques, and special glasses that may help people with color blindness differentiate different colors but not truly see them.

There are four type of color blindness; Deuteranamolous (red and green color confiusion), Protanamolous (red and bluish-green color confiusion), Tritanamonolous (blue and yellow color confusion), Achromatopsia (only seeing black, white and grey). As stated above, color blind people facing many problems regarding abilities to choose products, to notice information in advertising and on packaging, and to operate effectively in store settings as consumer.

There are four type of color blindness; Deuteranamolous (red and green color confiusion), Protanamolous (red and bluish-green color confiusion), Tritanamonolous (blue and yellow color confusion), Achromatopsia (only seeing black, white and grey). As state above, color blind people faced many problems regarding abilities to choose products, to notice information in advertising and on packaging, and to operate effectively in store settings as consumer. Besides that, traffic lights are generally dependent upon color and position recognition; such signals sometimes vary between horizontal and vertical rows of lights, which can cause confusion. The color cue could be supplemented by a uniform system of stripes or shapes embedded in the lens of the signal

Therefore, this project intention is to help color blind people to recognize the object color. It is because there many difficulties to them such as in driving, they cannot recognize traffic light and emergency signal so they cannot get the driving license. They also faced problems in judging quality of goods that they want to buy by color. Besides that, they faced many problems that related to color recognizing in their daily life.

So, by developing the program that able to recognize colors maybe can help color blind people in their daily life. Hopefully this project will be used to invent mobile sensor that can help color blind people for their daily life

1.2 Problem Statement (Research Question)

This study aims to seek the following research question:

i. How to determine the color in RGB?
ii. How to process the captured image?
iii. How to define and differentiate major color in an image?
iv. How to develop GUI to make this system is user friendly to be operated

1.3 Research Objectives

i. To identify the variables that determined the pixel color.
ii. To identify the pixel value of captured image.
iii. To identify the method of color extraction
iv. To develop Graphical User Interface (GUI)

1.4 Significance of Study (Main Objectives)

Objective of this project are;

i. To study the concept of image processing.
   For this project, this is the important aspect that should be consider and understand before start running the project and also the main target of this project.

ii. To develop a program that able to recognize an object color by using MATLAB and Image Processing Toolbox.
   Therefore, the project will be focus on developing a program that able to recognize an object color by using MATLAB and Image Processing Toolbox as image processing tools.

1.5 Scope of Project

This project only covers the software development. The image that been used is manually corrected image or taken from any corrected image. For result, the archive color will be displayed on monitor screen in MATLAB command window by wording written in M-File command window. This project is about color recognition, that define the color in the image by refer to color range in the data base.

Thus, the focuses of this project are:

i. MATLAB simulation of Image Processing Toolbox and Simulink
ii. Construct the software programming to archive the expected result

CHAPTER 2

LITERITURE REVIEW

2.1 Color Space

A color space is defined as a model for representing color in terms of intensity values[1]. Typically, a color space defines a one- to four-dimensional space. A color component, or a color channel, is one of the dimensions. A color dimensional space (i.e. one dimension per pixel) represents the gray-scale space. The following two models are commonly used in color image retrieval system.

Most image formats such as JPEG, BMP, GIF, use the RGB colour space to store information [7]. The RGB colour space is defined as a unit cube with red, green, and blue axes. Thus, a vector with three co-ordinates represents the colour in this space. When all three coordinates are set to zero the colour perceived is black. When all three coordinates are set to 1 the colour perceived is white [7]. The other colour spaces operate in a similar fashion but with a different perception.

2.1.1 RGB

The RGB color model is composed of the primary colors Red, Green, and Blue. This system defines the color model that is used in most color CRT monitors and color raster graphics. They are considered the "additive primaries" since the colors are added together to produce the desired color. The RGB model uses the cartesian coordinate system as shown in Figure 4.1 (a). Notice the diagonal from (0,0,0) black to (1,1,1) white which represents the grey-scale. Figure 4.1 (b) is a view of the RGB color model looking down from "White" to origin.

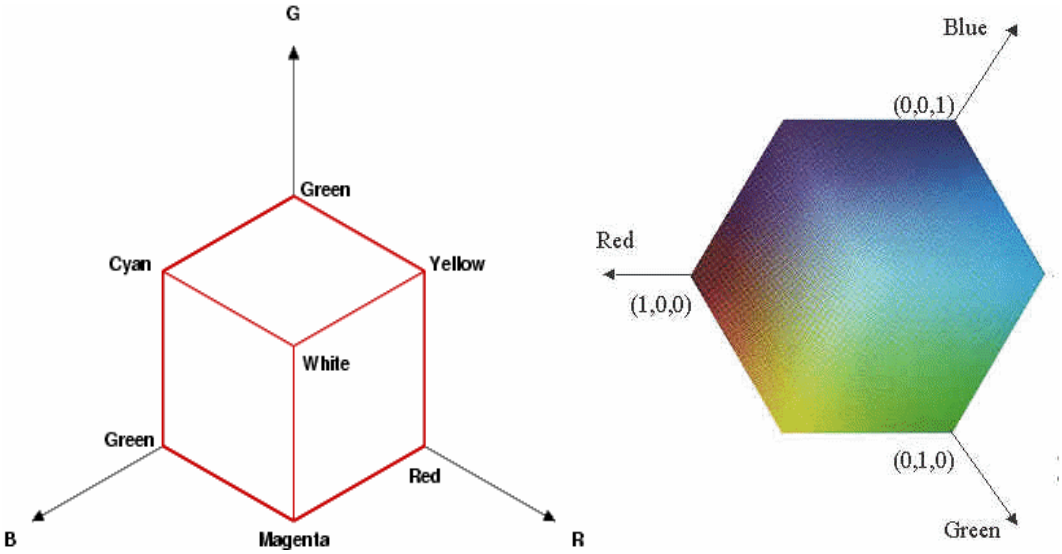


Figure 1.1 (a) RGB coordinates system (b) RGB color model

2.2 Color Range Definition Selection

The color range definition selection will be applied as to choose pixel color for data base for color recognition system. The color range definition can be done by referring to the RBG Hex triplet chart and Color Hexagon in the **Appendix B.**

2.3 Image Processing Toolbox

Image Processing Toolbox is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image processing operations, including

i. Spatial image transformations
ii. Morphological operations
iii. Neighborhood and block operations
iv. Linear filtering and filter design
v. Transforms
vi. Image analysis and enhancement
vii. Image registration
viii. Deblurring
ix. Region of interest operations

Many of the toolbox functions are MATLAB M-files, a series of MATLAB statements that implement specialized image processing algorithms. The capabilities of Image Processing Toolbox can be extend by writing M-files, or by using the toolbox in combination with other toolboxes, such as Signal Processing Toolbox and Wavelet Toolbox.

2.4 Color Extraction

2.4.1 Color Histogram Extraction

The color histogram for an image is constructed by counting the number of pixels of each color. The developments of the extraction algorithms follow a similar progression: selection of a color space, quantization of the color space, computation of histograms, derivation of the histogram distance function, identification of indexing shortcuts. First there was the issue of how much we would quantize the number of bins in a histogram [3].

Besides that, its main advantage is fast to manipulate, store and compare and insensitive to rotation and scale. On the other hand, it is also quite unreliable as it is sensitive to even small changes in the scene of the image. In color image processing, the histogram consists of three components, respect to the three components of the color space.

There are several difficulties with histogram based retrieval. The first of these is the high dimensionality of the color histograms. Even with drastic quantization of the color space, the image histogram feature spaces can occupy over 100 dimensions in real valued space. This high dimensionality ensures that methods of feature reduction, pre-filtering and hierarchical indexing must be implemented. The large dimensionality also increases the complexity and computation of the distance function. It particularly complicates `cross' distance functions that include the perceptual distance between histogram bins [2].

2.4.2 RGB Pixel Extraction

The object color can be can be extract by using impixel in MATLAB image processing toolbox. impixel returns the red, green, and blue color values of specified image pixels [4]. impixel can displays the input of image and specify the pixels with the right-click mouse and specify region in the image by placing the coordinates. Besides that, impixel works with indexed, intensity, and RGB images [4]. It also returns pixel values as RGB triplets, regardless of the image type.

CHAPTER 3

METHODOLOGY

3.1 Operation Concept

This project consist 3 major parts which are camera sensor, image processing and the result display. The equipment that used in this project included the digital camera, computer monitor screen and MATLAB image processing software that will perform the extraction process. The digital camera used to capture the image of the object because it support RGB format to produce input signal to the controller of the process.

Then, by pointing the cursor and right-clicking on the object, pixel data will be extracted from the captured image. After that, pixel data from image will be compared to pixel data in data base. This process can be done by programming the MATLAB Image Processing Toolbox software. Lastly, if the pixel data that have been extracted from the image match data base, archive color will be displayed by wording in popup message box on monitor screen. Figure 3.1 shows the flow chart of operation concept process.

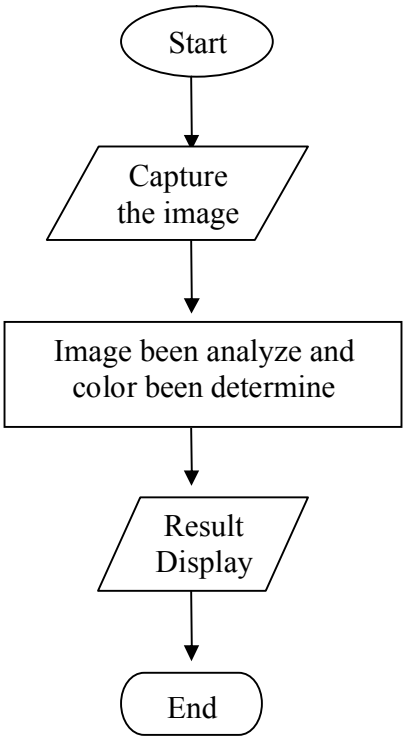


Figure 3.1: Operation Concept Flow Chart

3.2 Color Extraction

The image that needed to be processed must use appropriate method according to the processing concept. For this project, RGB color extraction concept is applied to analyze and determine the color of the capture image. There are many kind of method of RGB color extraction, such as color histogram, pixel statistic and pixel value analysis. The color histogram and pixel statistic methods are not suitable for this project because the both methods analyze overall color in the image.

Therefore, by using pixel value analysis concept, specific color can be extracted at needed point from the image and also it more accurate. This method included in MATLAB image processing toolbox by 'impixel' command. This command returns the red, green, and blue color values of specified image pixels. So, by this command the desired color point that needed to extract can be extract by right-clicking on the image.

3.3 Color Range Definition

The color range definition is the important aspects that need to be consider. Usually colors are defined in three dimensional color spaces. These could either be *RGB* (Red, Green, and Blue), *HSV* (Hue, Saturation, and Value) or *HSB* (Hue, Saturation, and Brightness). Most image formats such as JPEG, BMP, GIF, use the RGB color space to store information [7]. The RGB color space is defined as a unit cube with red, green, and blue in pixel value. Thus, a vector with three values in pixel represents the color in image.

Color image define by it pixel value in RGB. Color that been define by referring to the RGB Hex Triplet Chart and Hexagon Color that included in appendix. Color definition can be defined according to the needs. Table 3.1 shown the RGB color truth table for eight basic color.

Table 3.1: RGB color truth table

RGB Value	Color
[1 1 0]	Yellow
[1 0 1]	Magenta
[0 1 1]	Cyan
[1 0 0]	Red
[0 1 0]	Green
[0 0 1]	Blue
[1 1 1]	White
[0 0 0]	Black

Table 3.2: Customize RGB color table

RGB Value	Color
r>=153 g>=153 b<=102	Yellow
r>=255 g<=102 b>=255	Magenta
R<=102 g>=153 b>=153	Cyan
r>=153 g<=102 b<=102	Red
R<=102 g>=153 b<=102	Green
R<=102 g<=102 b>=153	Blue
r>=255 g>=255 b>=255	White
r<=0 g<=0 b<=0	Black

Table 3.2 shown the example of customize RGB color table, the color definition customize depend on the data selecting program usage. The color range must be defined to act as data base for color detection in image. Moreover, Color data base is used in image processing for determined the color in the captured image.

3.4 Color Decision Making

After color been extract from the captured image, it pixel will be compare to the data base for result display. Figure 3.3 shown flow chart of red, green and blue color recognition. This is the example of color decision making concept for red, green and blue color. By pointing the cursor and right-click on the object, pixel data will be extracted. Then, pixel data from image will be compared to pixel data in data base. If pixel data from captured image match data from data base, the result will be display. It shows the basic concept of color decision making. This concept is applied in this project for result display.

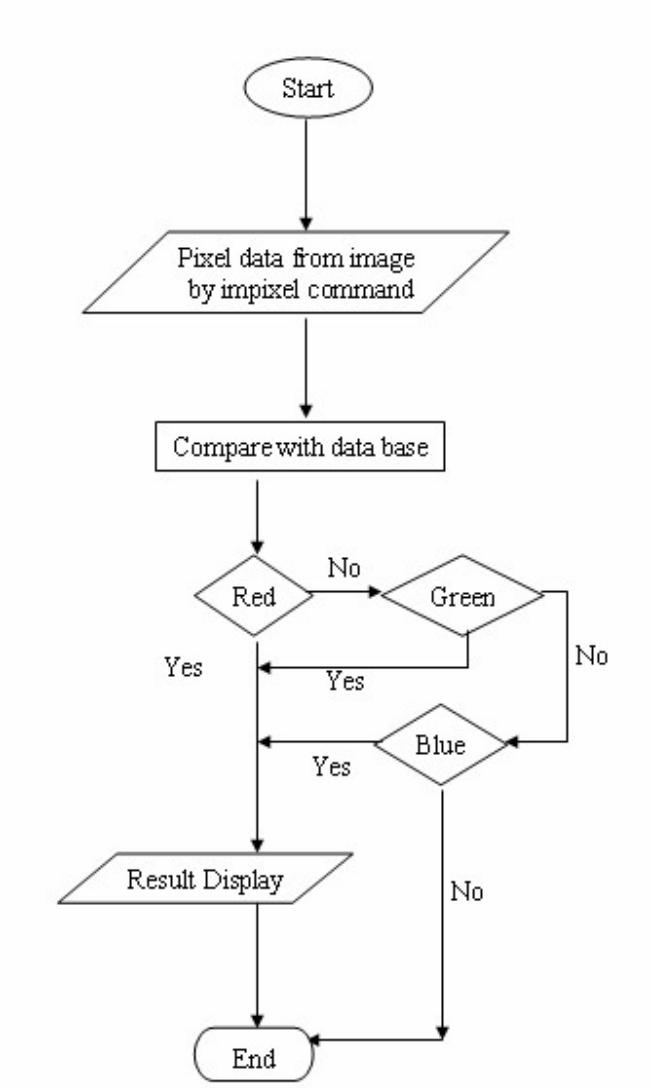


Figure 3.2: Flow Chart of Red, Green and Blue Color Recognition

3.5 MATLAB

3.5.1 Image Processing ToolBox

Image processing is the main process of this project. It used for processing the image in this project, for instance; read the image, analyze image data and produced the result. Since all the processes that need to be done are included in MATLAB Image Processing Toolbox, the process can be done easily. Figure 3.4 shows the simple sample of image processing programming for this project. Besides that, this sample program maybe contains some errors. It is because it only used for explanation for better understanding the basic concept that is used in image processing.

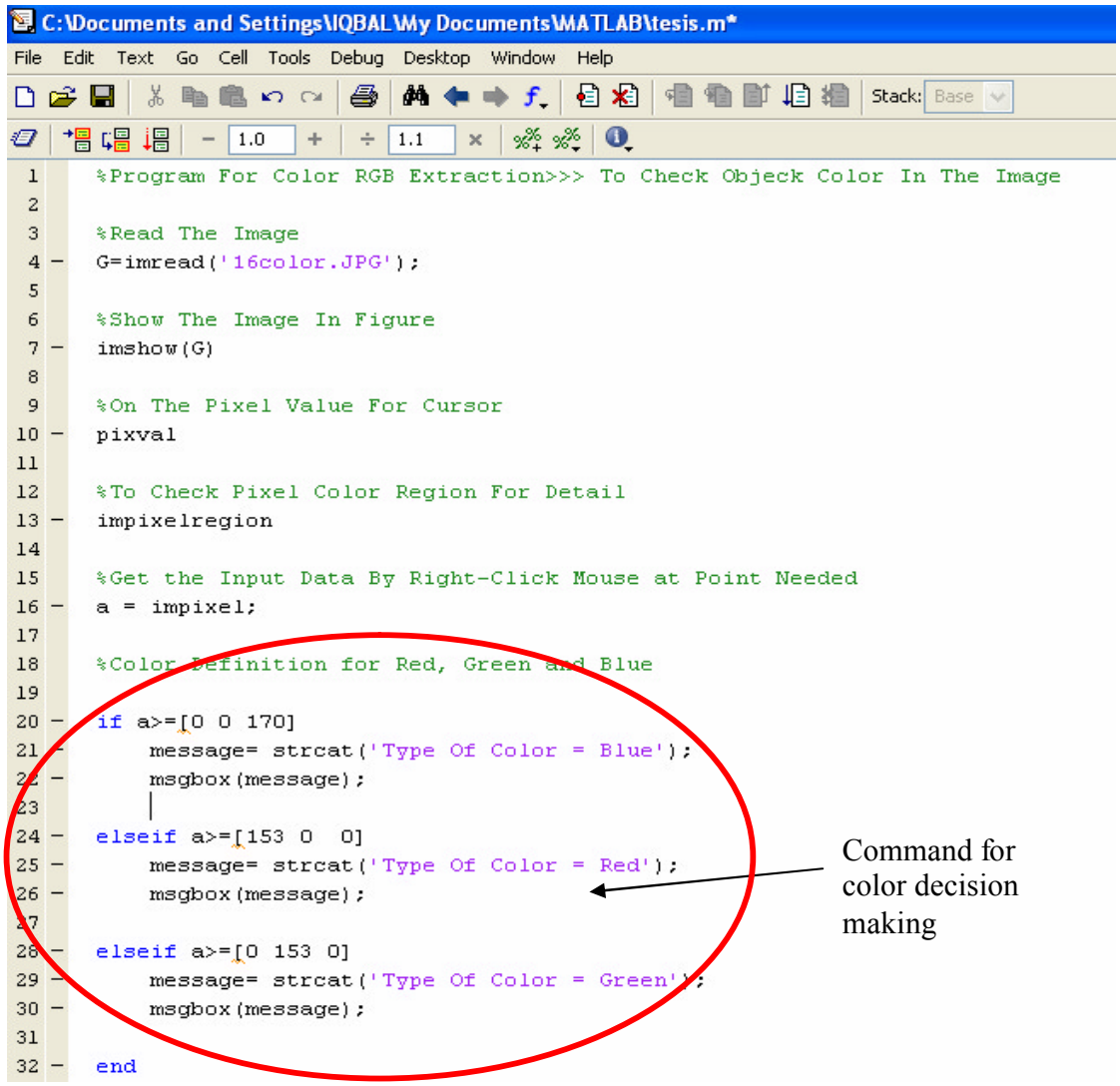


Figure 3.3: Simple command for image processing

According to the figure above, it shows sample of command that is used for image processing. The image had been read by 'imread' command, and then the image is displayed in the image processing toolbox window as a figure by 'imshow' command. The 'pixval' command in turn on the interactive display of information about image pixels in the current figure. Thus, the pixel of the figure can be checked by pointing the cursor at the image. Figure 3.5 shows the loaded image with its pixel value information on.

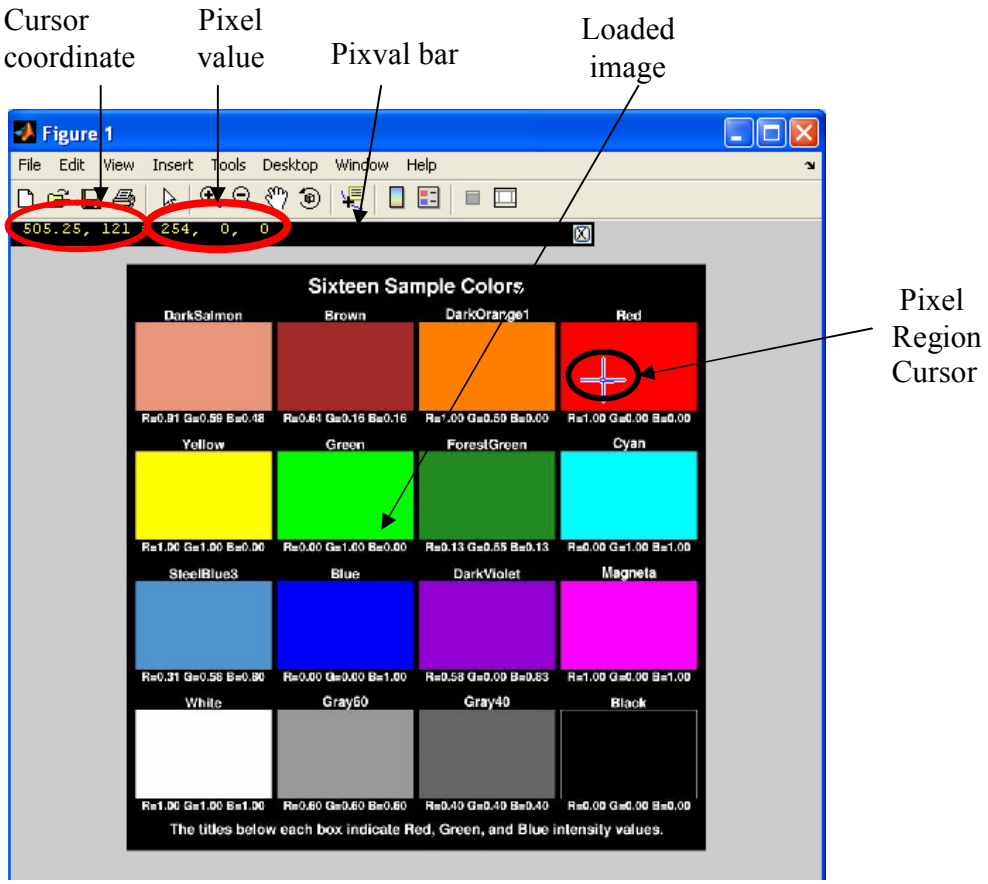


Figure 3.4: Loaded image with its pixel information on

Next, the 'impixelregion' it used for checking the details of the image pixel. Besides that, it included the zooming feature which are zoom in and zoom out. It can be done by left-click on zoom in or zoom out icon in pixel region tool bar. The zoom in feature can be zooming until see a pixel of the image. Furthermore, to check different region in the image, it can be done by dragging the pixel region cursor as shown in Figure 3.6. The Figure 3.7 shows pixel region window.

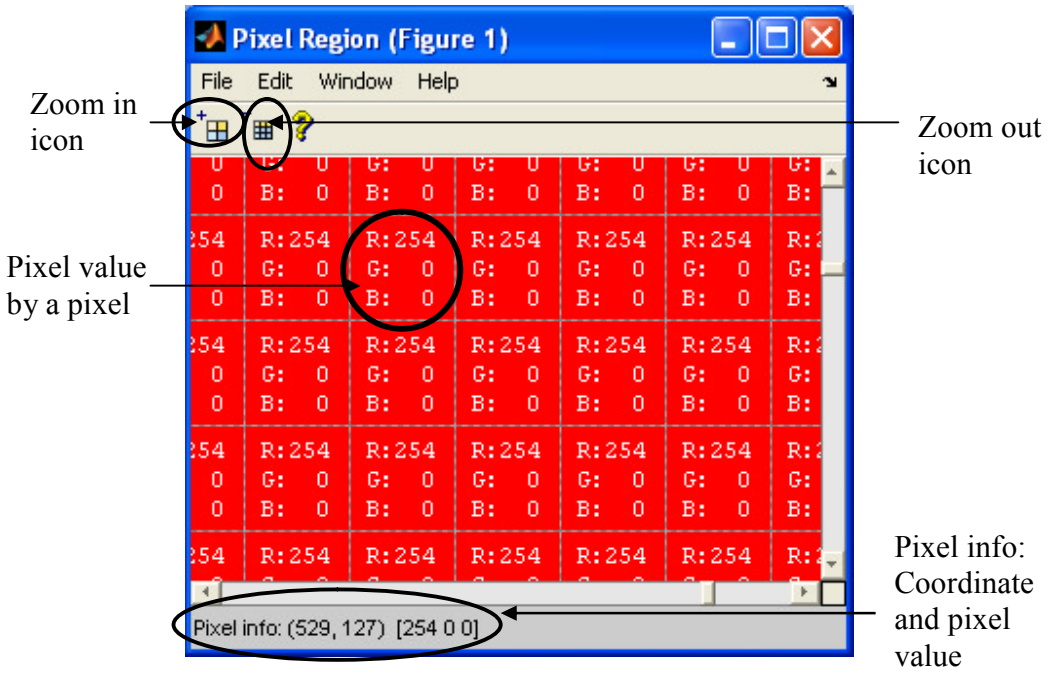


Figure 3.5: Pixel Region window

Moreover, the 'a=impixel' command is used to extract the pixel value from the image by right-clicking on the image. The pixel values that have been extracted from the image will be compared with pixel data base in color decision making process. If the extracted pixel values match the pixel values in data base, the result will be show as shown in Figure 3.9. The function of command that is used in program can be refer by Table 3.3 but for more detail please refer to MATLAB Help.

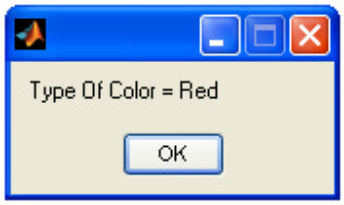


Figure 3.6: Message box of result display

Table 3.3: Command and function used in program

Command	Function
A= imread(filename,*fmt*)	Reads a grayscale or color image from the file specified by the string filename and file type(fmt).
imshow	Calls imread or dicomread to read the image from the file and display the image.
pixval	Turns on interactive display of information about image pixels in the current figure. It also installs a black bar at the bottom of the figure, which displays the (x,y) coordinates for whatever pixel the cursor is currently over and the color information for that pixel.
impixelregion	Creates a Pixel Region display tool associated with the image displayed in the current figure
impixel	Extract image color and returns the red, green, and blue color values of specified image pixels
msgbox(*message*)	creates a message dialog box that automatically wraps *message* to fit an appropriately sized figure

3.5.2 Simulink

Figure 3.6 shows the Simulink block diagram for pixel statistic in the image. By using this block diagram, RGB image data will be process and the pixel statistic graph will be display. Purpose of this block diagram is to check the major color in the image (red, green, blue)

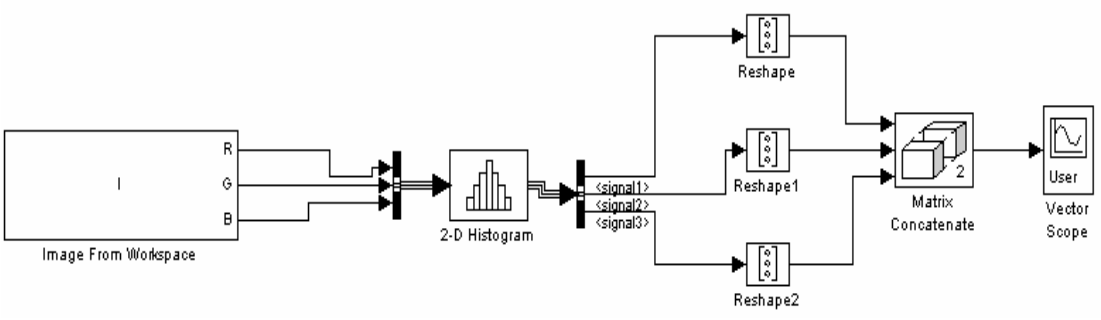


Figure 3.7: Simulink block diagram for pixel statistic

3.5.3 Design of Graphical User Interface (GUI)

A Graphical User Interface (GUI) is a graphical display that contains devices, or components, that enable a user to perform interactive tasks [1]. By using GUI, user does not have to type command at the command line in M-File. So, user will not be confuse with complex command and easily can run the application by using GUI. There are few components in GUI such as menus, toolbars, push buttons, radio buttons, list boxes, sliders and many more. In MATLAB, a GUI can also display data in tabular form or as plots, and can group related components [1].

The Graphical User Interface was constructed using MATLAB GUIDE or Graphical User Interface Design Environment. Using the layout tools provided by GUIDE, I designed the following graphical user interface Figure 3.7 shows the GUI design for color extraction application.

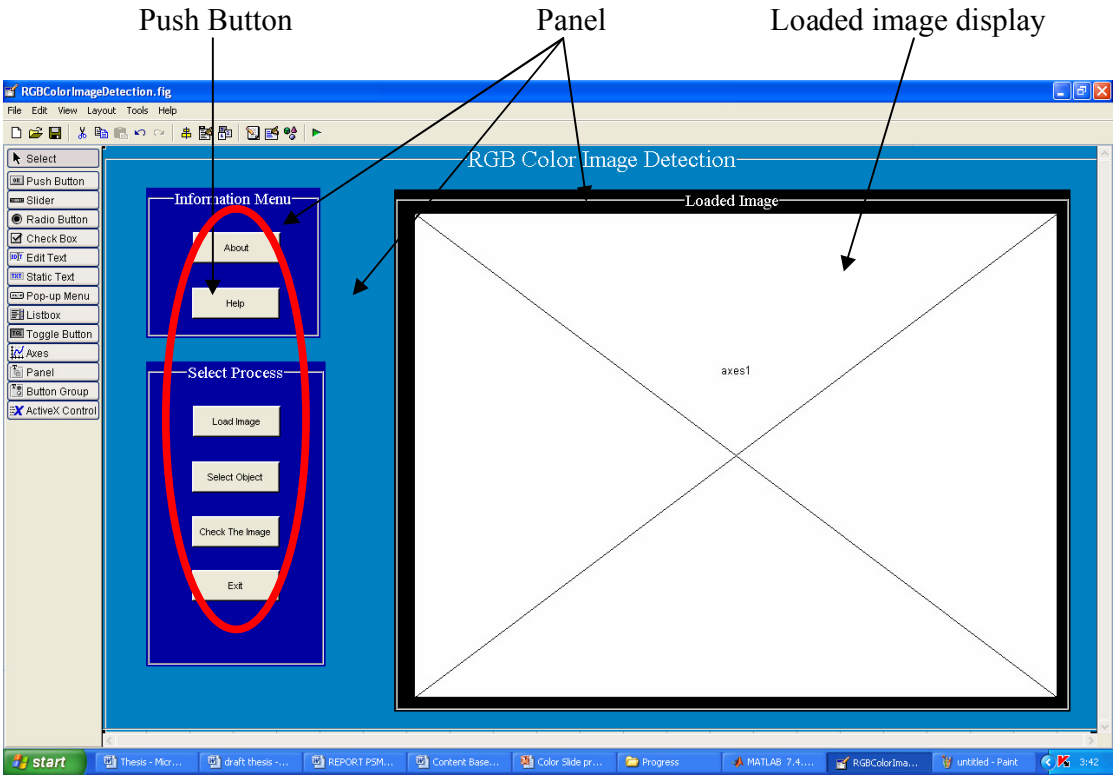


Figure 3.8: GUI Design, RGBColorImageDetection.fig

Table 3.4: The selected GUI component and their function

Components	Function
About button	Explain about this software
Help button	Explain how to use this software
Load image button	Load image from files
Select object button	Select object in loaded image window for extraction
Check the image button	To check detail of image pixel by region
Axes1	Display the loaded image from files
Exit button	To end or close the window
Panel	To arrange GUI components into groups. Each panel had a title and various borders.

3.5.3.1 Form Window

The important step in creating GUI is forming the window. So, all of the components can be placed on it. There are few step need to be follow to design the GUI:

Firstly, start GUIDE by clicking tool bar in MATLAB window as shown in figure below;

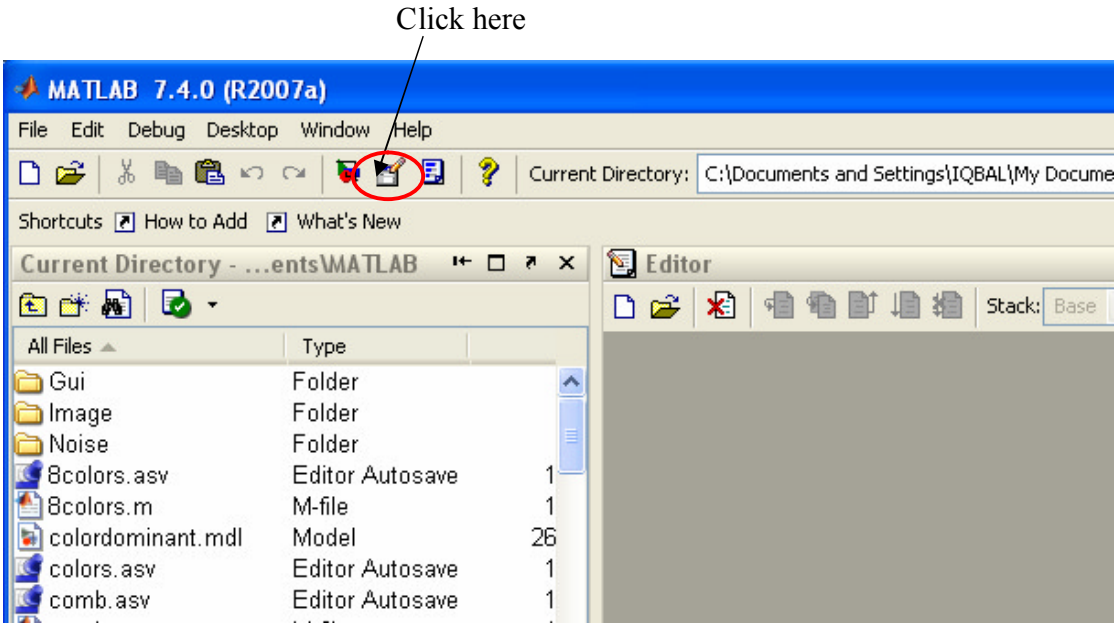


Figure 3.9: MATLAB Window

Then, the GUIDE quick start window will be pop-up as shown in figure 3.9. In the Quick Start window, select the Blank GUI (Default) template and click OK to display the blank GUI in the Layout Editor, as shown in the following figure.

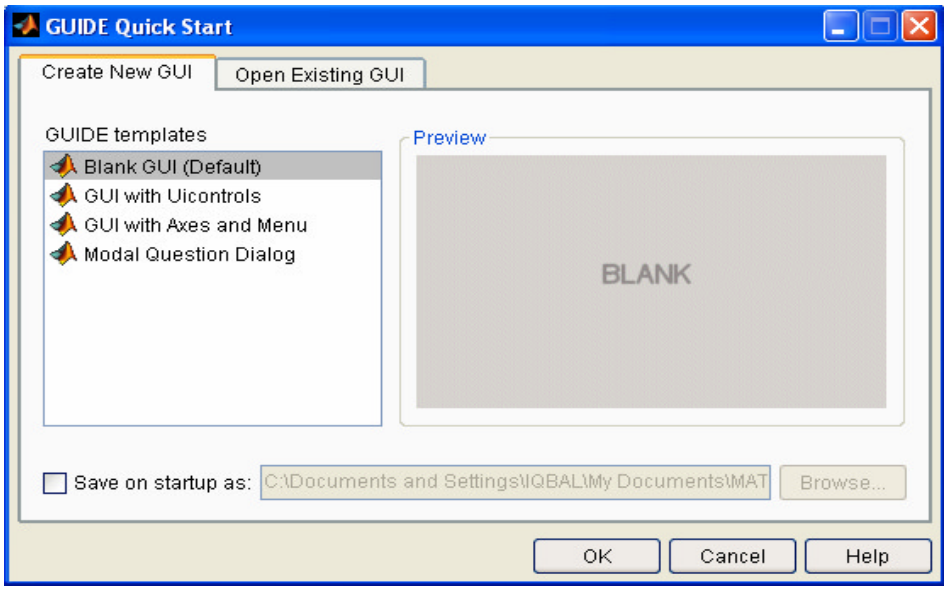


Figure 3.10: Guide Quick Start Window

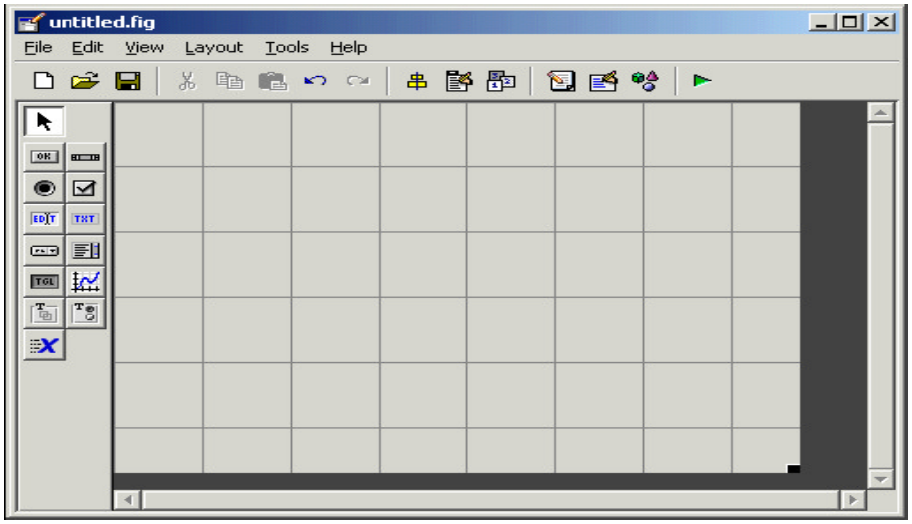
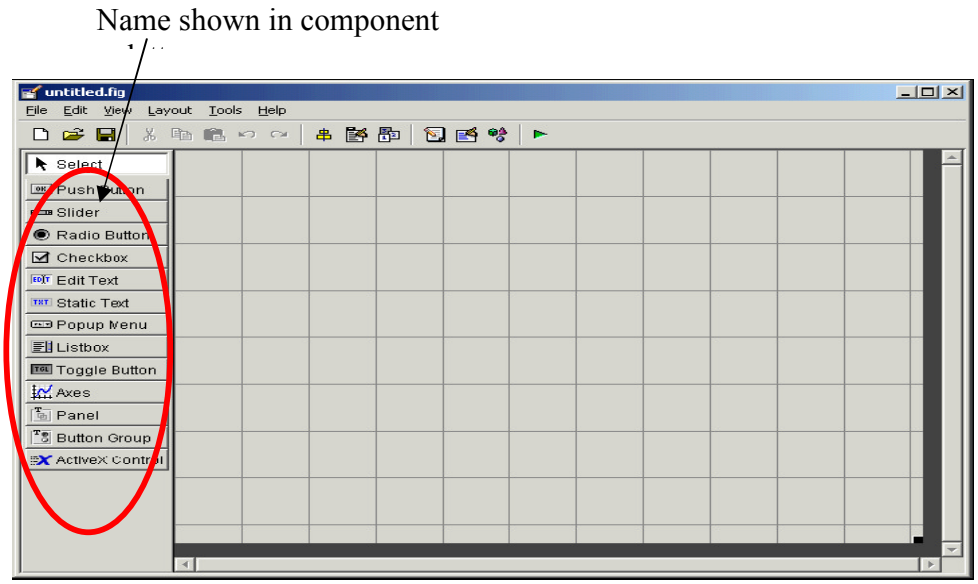


Figure 3.11: Layout Editor

Next, to make thing easier the names of the GUI component in the component pallet must be display. It is because starter will not recognize the symbol in component pallet. To display the component pallet, Select Preferences from the MATLAB File menu. Then select GUIDE > Show names in component palette, and click OK. The Layout Editor then appears as shown in the figure below;



.

Figure 3.12: Layout editor with names in command palette

After that, size of the GUI can be set by resizing the grid area in the Layout Editor by clicking lower-right corner and drag it according to the needs as shown in Figure 3.12.

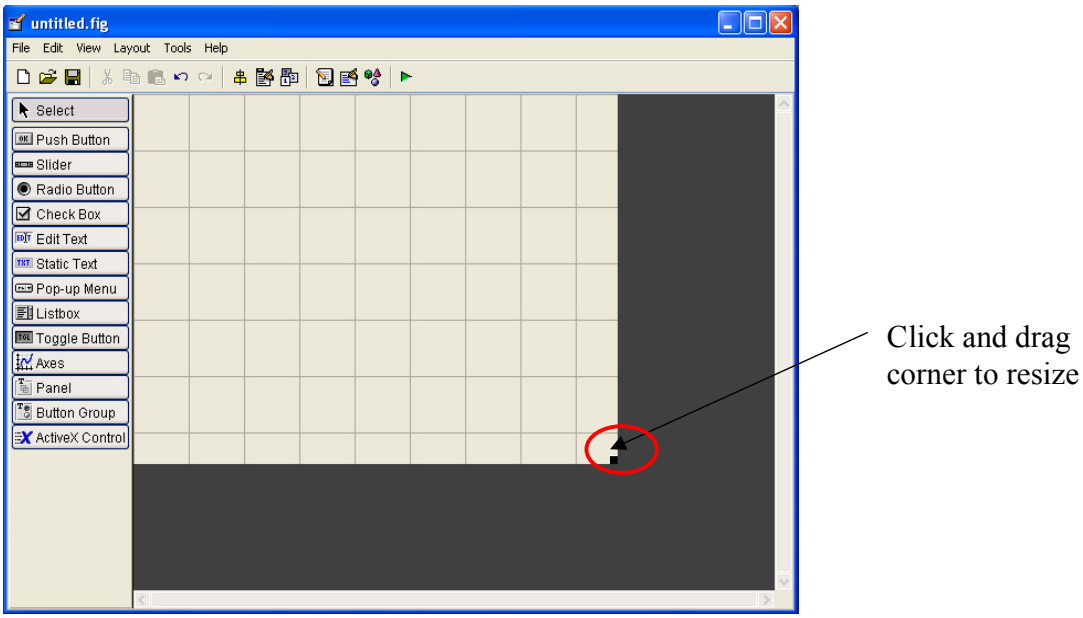


Figure 3.13: Resizing Layout Editor

3.5.3.2 Panel

The common use of panel is to arrange GUI component into group by visually grouping related button, controls or panels to make the user interface easy to understand. A panel can have boarder and a title. User interface controls, axes, push button and other panels can be place in panel. If the panel moves, all components on it will also move and maintain their positions on the panel.

Firstly, select panel from component palette at the left of the layout editor and drag it into the layout area as shown in Figure 3.13.

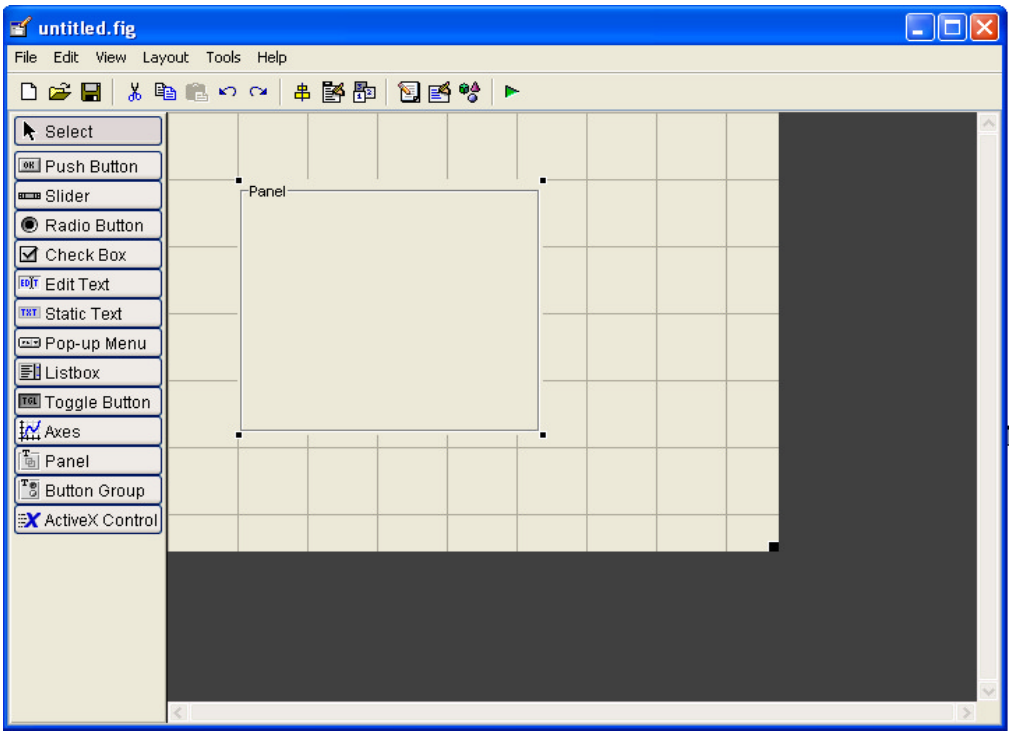


Figure 3.14: Panel Inserted in Layout

The feature in panel can be change by right-click on the panel, and then select property inspector. The features can be change, by changing the property in property inspector as shown in Figure 3.14. there are four panel were create in the proposed system which are 'RGB Color Image Detection' panel act as main window, 'Information Menu' panel to place menu push button, 'Select Process' panel to place process push button, and lastly 'Loaded Image' panel to place axes as shown in figure .

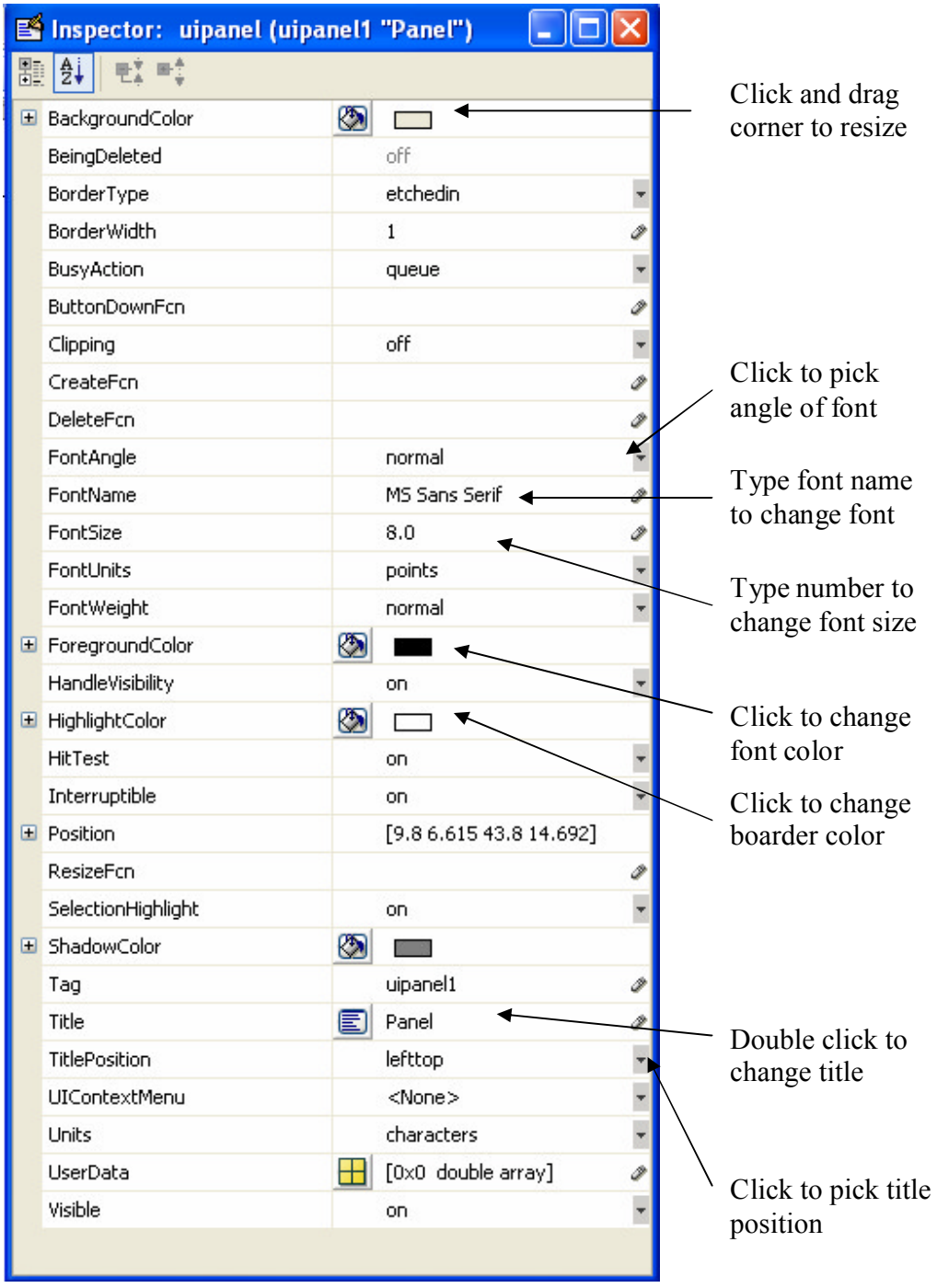


Figure 3.15: Property Inspector of Panel

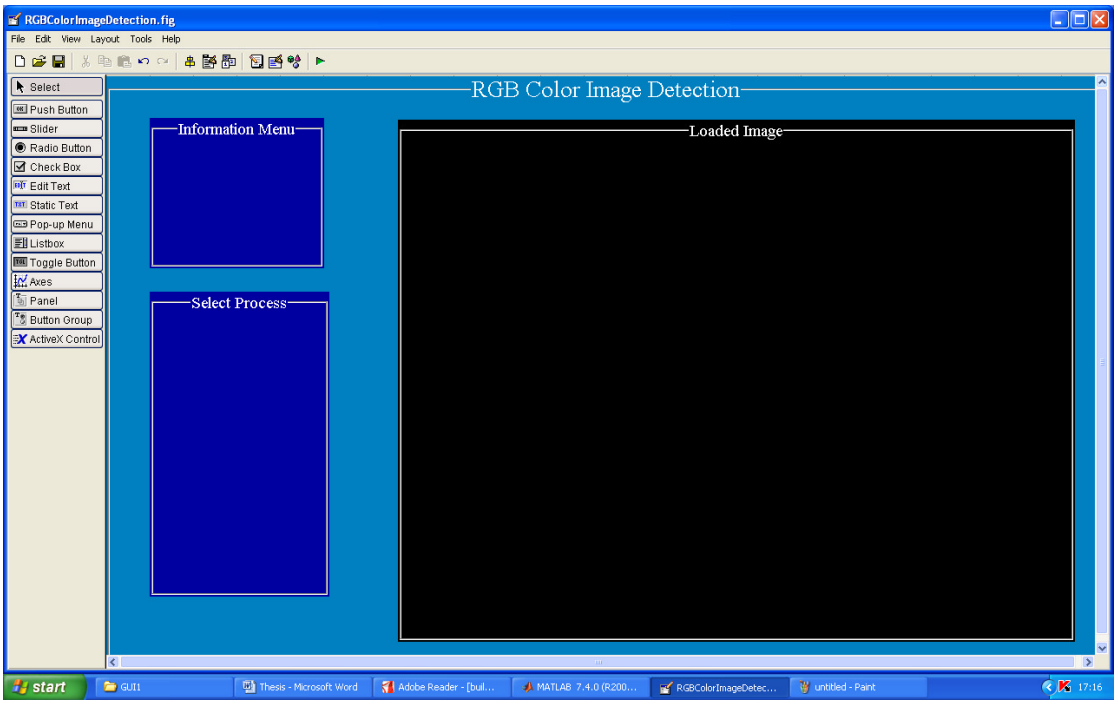


Figure 3.16: Four Panel Inserted in the Layout

3.5.3.3 Push Button and Axes

There are six push button and a Axes that were created in the proposed system. Two of the push buttons are placed on Information Menu panel. These push button are used to link to other window to act as information window. The balance of the push button are used in Select Process panel, that been used as process button. The Axes is placed on the Loaded Image panel. The Axes is used for display the loaded image from files.

Firstly, select push buttons from component pallet and then placed it all on as shown in Figure 3.16. Next, select Axes from component pallet and place it on Loaded Image panel. Push button and Axes can be resized by dragging their boarder using the cursor.

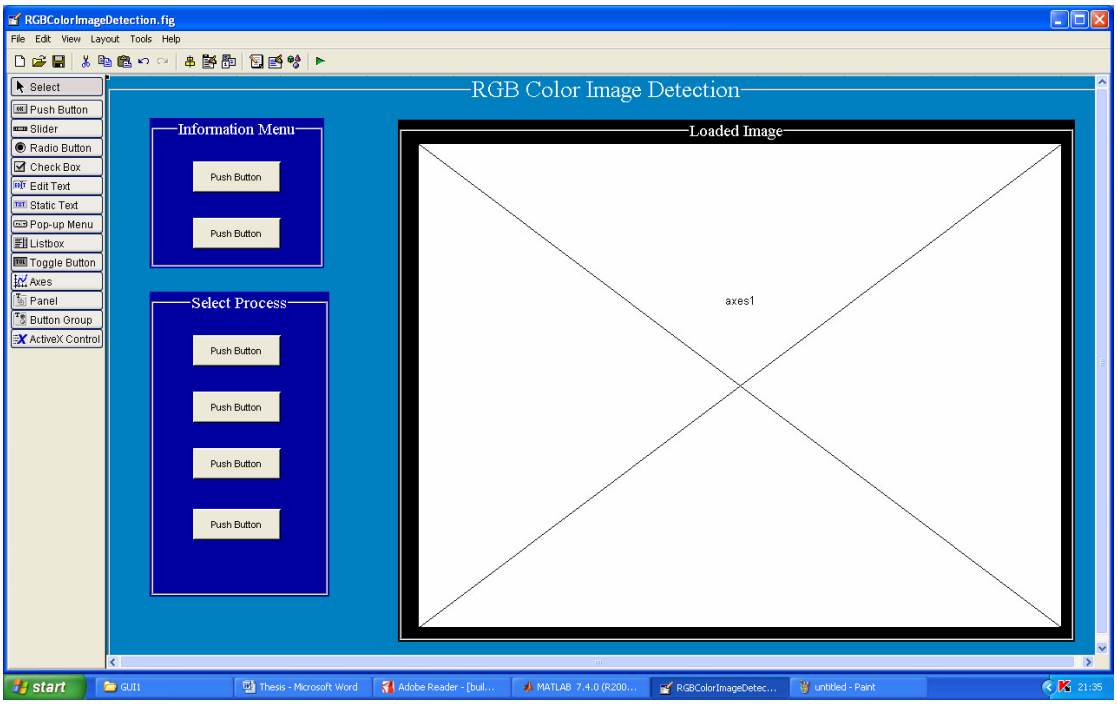


Figure 3.17: Push Button and Axes inserted in the Layout

To rename the push button, right-click the cursor on the push button in the layout and then select property inspector. In the property inspector, select string and rename ‘Push Button’. Thus, all the push buttons have been rename as follow in Figure 3.7. The last step in designing this GUI is programs all the push button according to it function. The first steps to program the push button is save the current layout, by left-clicking the cursor on ‘File’ in menu bar and then select ‘Save As’. Next, right-clicking on the push button and the on the layout select View Callbacks > Callback**.**

Therefore, the push button can be program by assigning a set of command that is written inside a desired command button's code in M-file command window. Figure 3.17 shows the 'Exit' push button program in M-file command window that is use to close the software application.

Program for the button

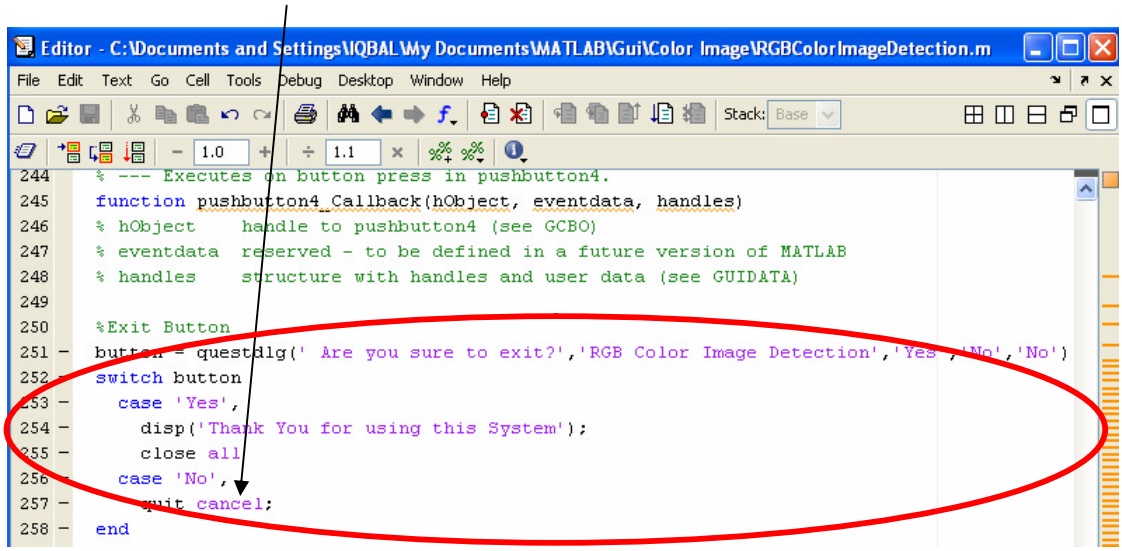


Figure 3.18: The 'Exit' button code window

3.6 Hardware Equipment

3.6.1 W550i Sony Ericsson

The digital camera that been used for this project is W550i Sony Ericssons handset. It is multipurpose handset that includes digital camera that takes both video and photographs. This camera has most function that a normal digital camera has. The camera can capture image at three different pixel resolution 1280 by 1024, 640 by 480, 160 by 120. The image that been capture will be save as JPEG image format.

Besides that, the usage of this camera is very basic. Like all other digital camera, first need to set its functional mode to 'camera'. Then, setup the resolution, shoot mode and zoom in/zoom out setting, the camera will be able to start capture image. All of the captured image will be saved in this handset memory. The captured image file can be transfer by using blue tooth, infrared and USB cable to the computer.

3.6.2 Computer LCD screen display

For the result display, I will use my own laptop acer Aspire 5540. It is because extracting process will be generated by MATLAB software in that laptop. Besides that, the result will be shown in MATLAB window on computer screen display.

3.7 Overall Operation

The GUI manages to help user and act as friendly user interface to run this project. The figure below shows the complete window for RGB Color Image Detection. There consist 3 major parts in this window, information menu, select process menu and loaded image window.

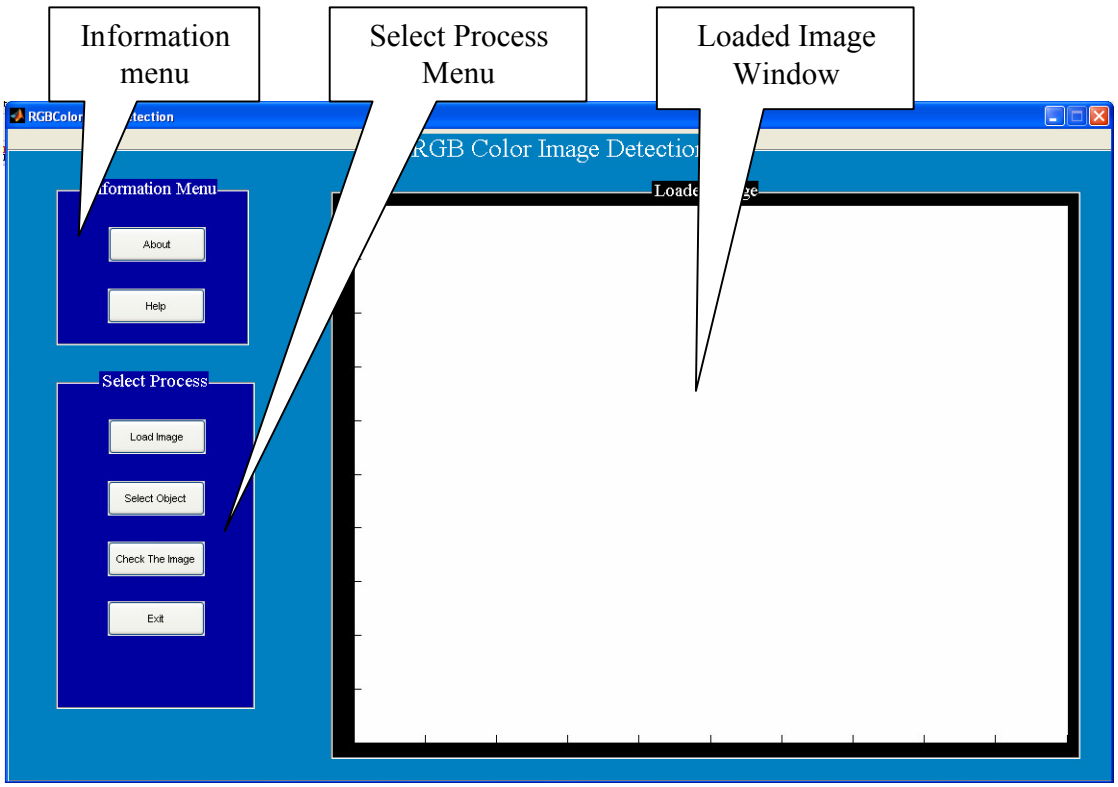


Figure 3.19: Complete window for RGB Color Image Detection

In the information menu consist of two push buttons, which are 'About' and 'Help' push button. This push button function is to link to other window. About window, display the information of this software and Help window display step and information to use this software. When push button of this menu is push, it will be link to other window. Below are the windows;

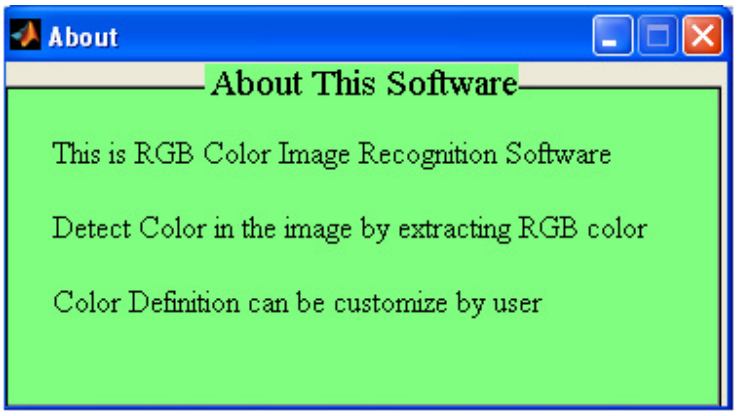


Figure 3.20: About window

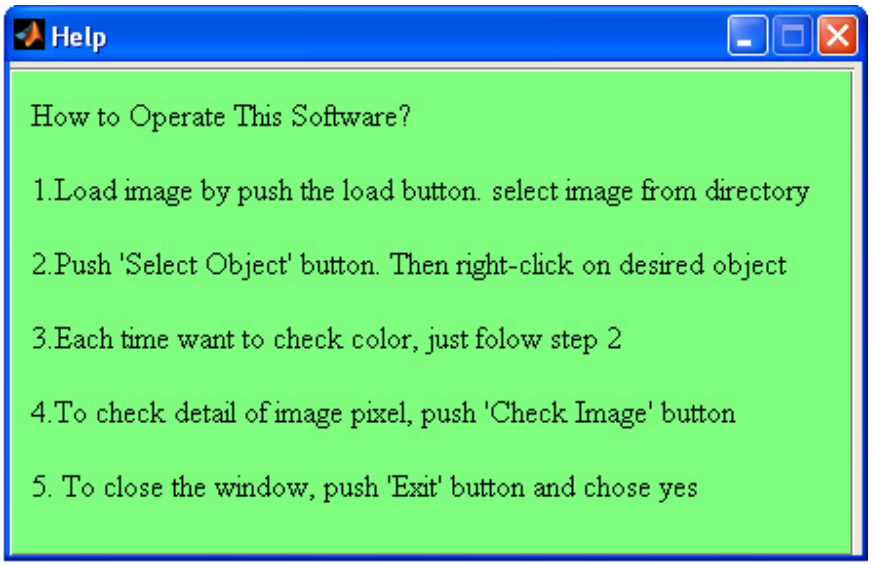


Figure 3.21: Help window

Furthermore, by understanding the method to operate this software, it can be start by loading an image from file as shown in Figure 3.21. This software only support JPG file type because this project only focusing on image from digital camera, that save image file in that type. Besides that, it also to avoid error when loading the image that's why it's been programmed only to load image with JPG file.

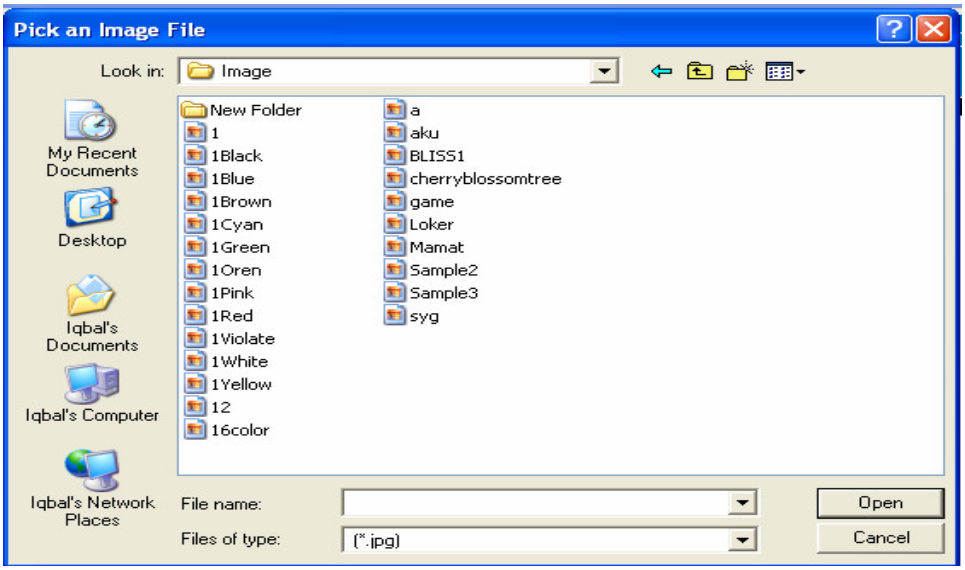


Figure 3.22: Image file window

After that, to detect color of the loaded image 'Select Object' push button need to be push. Then, right-clicking on the object to detected the color of the object. If the pixel data that have been extracted from the image match data base, the message box will popup. Figure 3.22 shows the loaded image and the result of color detection. In that figure, the object color that been detect is hair of the person in the image and the color define the hair is light black. Each time want to detect the object color, select push button need to be push.

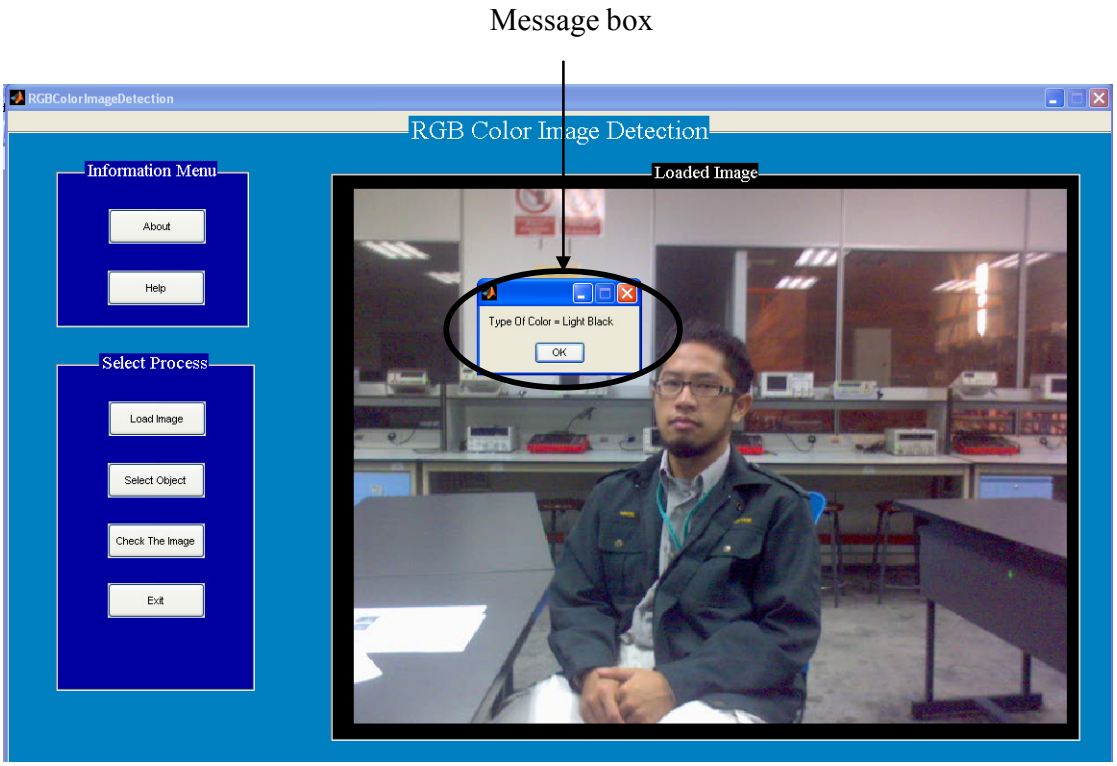


Figure 3.23: Example of detected object color

Furthermore, the 'Check the Image' push button is used to check pixel detail of the image. Lastly, the 'Exit' button is used to end and closed this project window. The 'Exit window' shows in Figure 3.23. The flow chart of the overall process as shown in Figure 3.24

Figure 3.24: Exit window

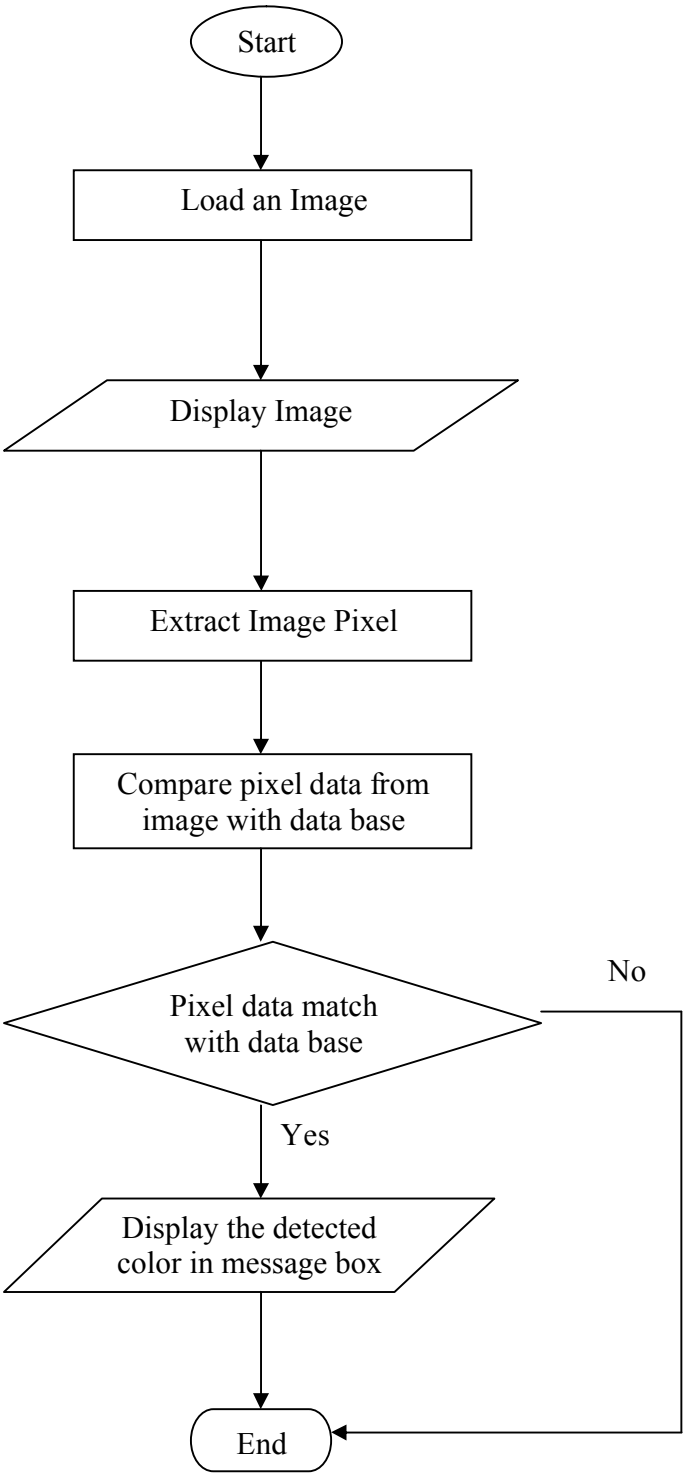


Figure 3.25: Flow chart of the overall operation

CHAPTER 4

RESULT AND DISCUSSION

4.1 Introduction

In this chapter, result and discussion of the color extraction will be discussed. The elements that need to be discussed are color extraction process in image processing and GUI of the project. Thus, this chapter will be discus the following process;

i. Development of Graphical User Interface (GUI).
ii. Variables to determine the pixel color.
iii. Pixel values of captured image.
iv. Method of color extraction.

4.2 GUI System

The GUI that had been designed for this system manages to help user and act as friendly user interface to run this project. The complete window for RGB Color Image Detection consist 3 major parts, information menu, select process menu and loaded image window. User can easily understand the operation of the system by clicking on Help button in information menu and read the instruction.

This GUI for this system is easy to understand, because all the button names are straight forward to it function. The details of the method to operate this system have been discus in the last chapter. Figure below shows the GUI design for this system.

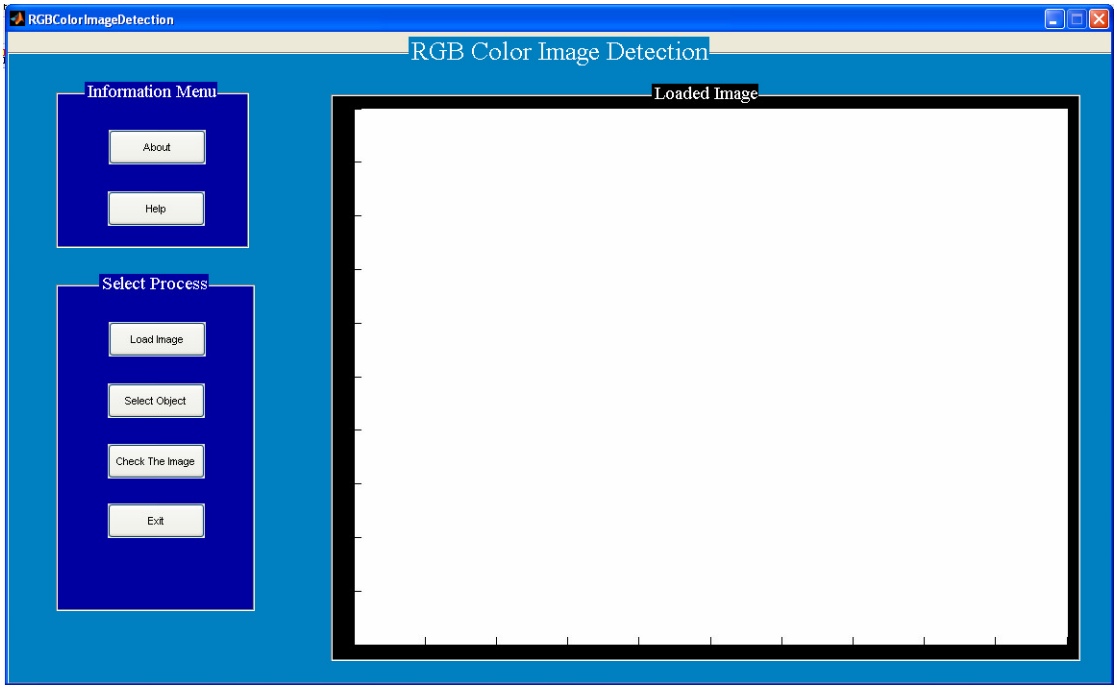


Figure 4.1: GUI of RGB Color Image Detection

4.3 Image Processing Test

The image processing test done by using MATLAB image processing toolbox and applied to this systems. Image processing test have been by using four different image and object. Figure 4.2 to 4.5 shows the tested image.

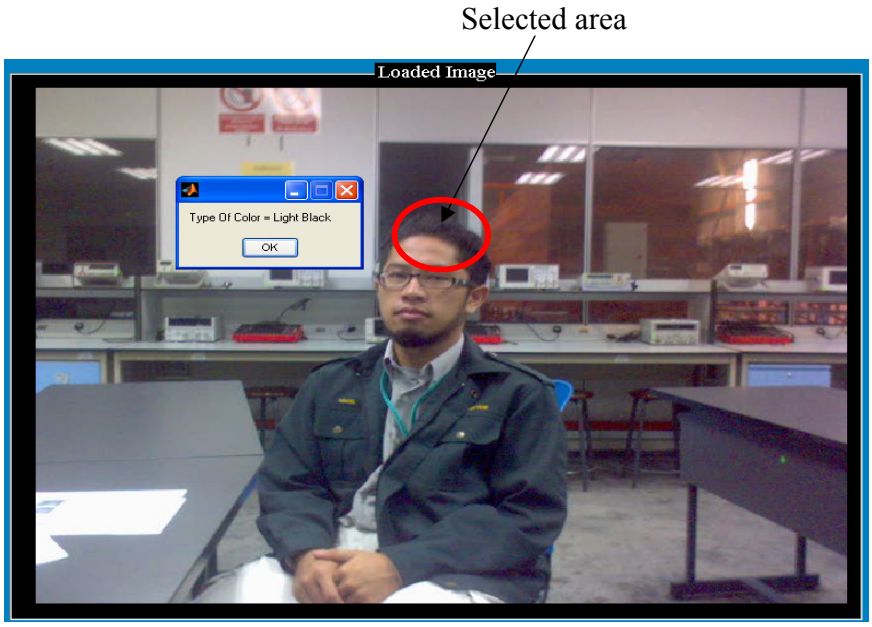


Figure 4.2: Test image 1

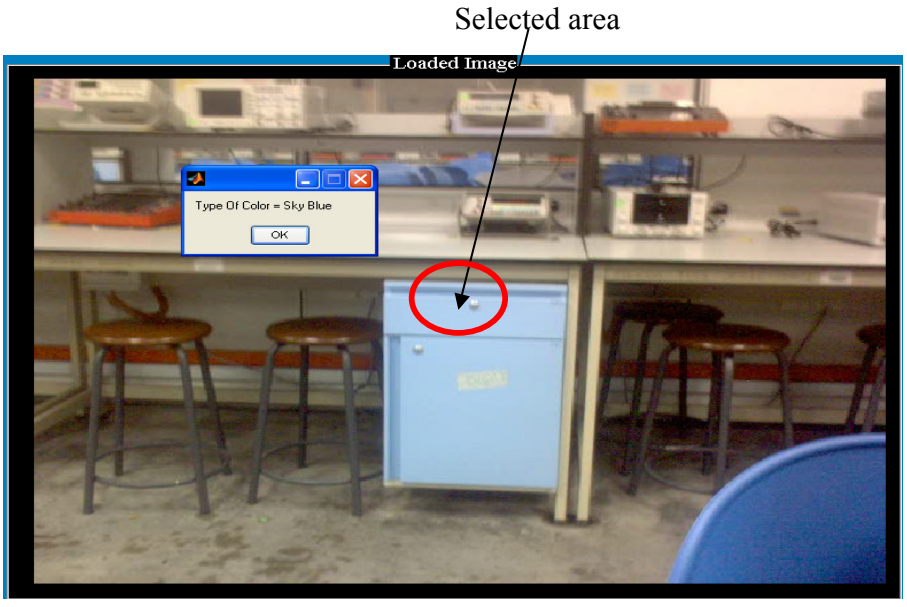


Figure 4.3: Test image 2

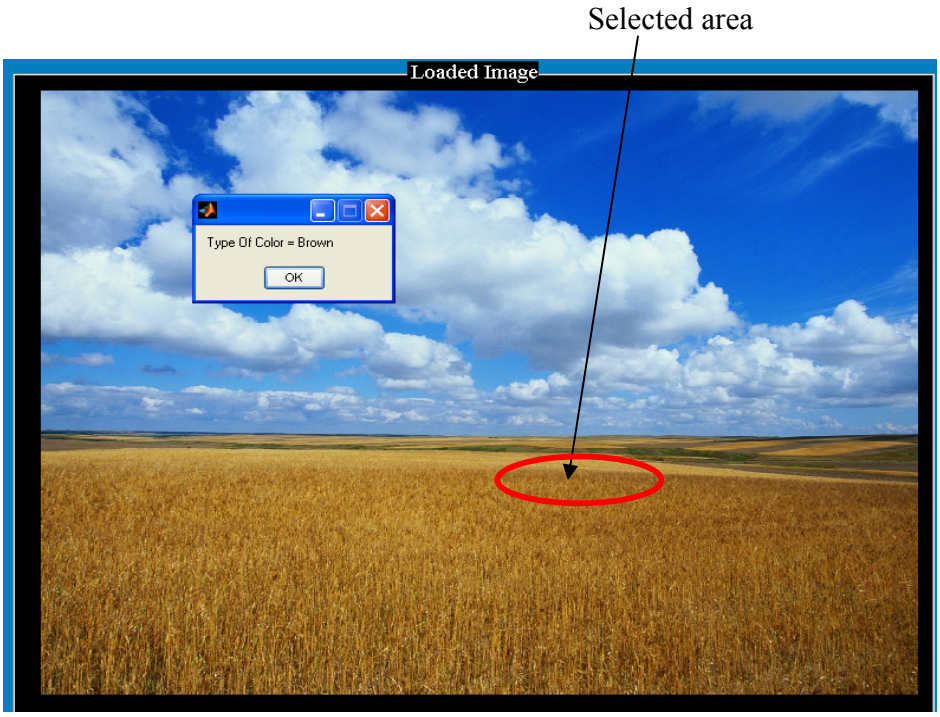


Figure 4.4: Test image 3

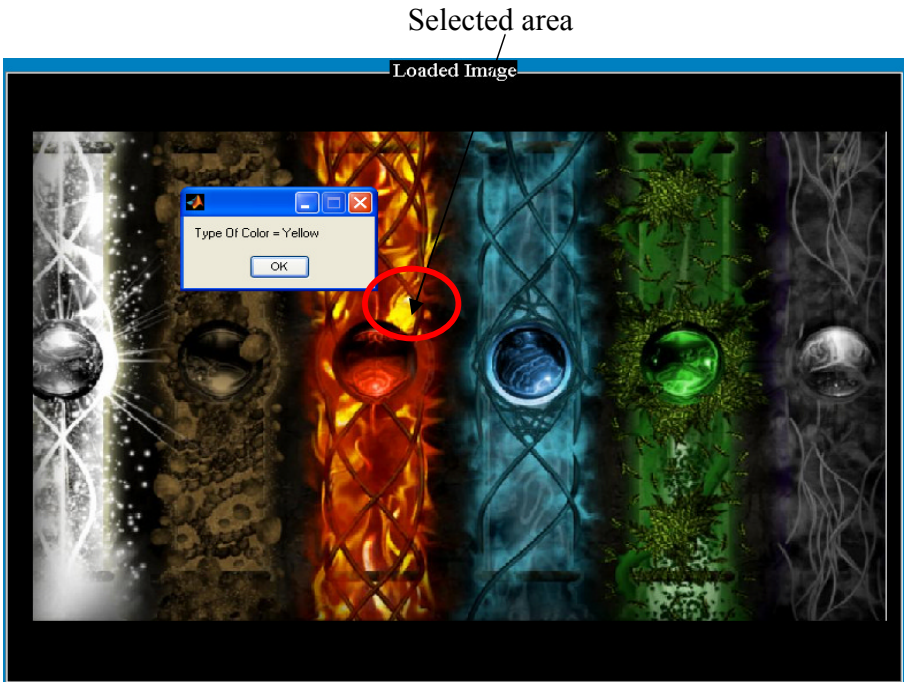


Figure 4.5: Test image 4

All four of the test image have been successful been extracted. So, that's proved the program are able to determined object RGB color and tell the user the color of the object by message box on computer screen display.

4.4 Summary of Result

The RGB color extraction system was successfully tested and developed. The system are able to extract RGB pixel data from the object color image and recognizing its color by referring to the data base by using image processing toolbox in MATLAB. Besides that, GUI able functioning as user friendly interface to user managed and operates the system.

4.5 Discussion

Pixel data that have been extracted will bee extracted as metric [R G B] which are R is values for red color, G is values of green color and B is values for blue color. In this system there are certain pixel data that conflict for instance object Brown color with input pixel [164 42 41], the result will be recognize as Dark Yellow color it is because the data base of Dark Yellow <= [200 200 80] and Brown >= [175 40 40] have conflict. So, to overcome this problem color need to be defined accurately because their range of pixel color really near and to avoid error in the program. To get the accurate result, data base for color definition must be create accurately and the color must be define not by its range but its true values.

Thus, if used the exact pixel value for color data base it will be lot of color data need to be define and it almost impossible to define all the color. Therefore, it is possible if the pixels data base must be create according to the set of colors that need to be detect in the system. Besides that, the type of the color can be programmed according to the needs of the systems.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

The systems are able to process the capture image, determine the color in RGB, define and differentiate color in the image. Furthermore, the image processing toolbox in MATLAB was proved that can be applied in this system for processing the image. Besides that, by applying GUI in this system, system can be manage and operate easily and also user friendly. Thus, this software can be applied to help color blind people to recognize object color.

5.2 Recommendation

This project also can be done by using Visual Basic and Neural Network for image processing. But it depends on the application of the system design. The development of this system can be apply as camera sensor in factory that use color in their product, such as in fabric factory to check the color of the dyed fabric. For further development, this system can be applied for real-time application for more advance technology. Besides that, this system concept can be applied in the advance mobile phone that can act as mobile sensor.

Furthermore, this system concept is suitable for color recognition in the paint factory which needs to have color recognition system for quality inspection of their product. This system also can be prototype for color recognition system for future development.

REFERENCES

[1] James Z. Wang, "Integrated Region-Based Image Retrieval", Boston, Kluwer Academic Publishers, 2001

[2] J. R. Smith and S.-F. Chang. "Automated image retrieval using color and texture", Technical Report CU/CTR 408-95-14, Columbia University, July 1995

[3] Rami Al-Tayeche and Ahmed Khalil. “CBIR: Content Based Image Retrieval”, Department of Systems and Computer Engineering Faculty of Engineering, Carleton University, April 4 2003

[4] “Image Processing Toolbox 5 User Guide”, MATLAB

[5] 20 March 2007, Citing Internet Source URL. Douglas R. Jacobson, *RGB Hex Triplet Color Chart*. http://homepage.mac.com/jakesan/DHP

[6] 6 March 2007, Citing Internet Source URL. J.P. Eakins, *Pattern Recog. 35 (2002) 3*. http://scien.stanford.edu/class/psyc[3]

[7] H.J. Zimmerman, in: Fuzzy Sets, Decision Making and Expert Systems, Kluwer Academic Publishers, Boston, MA, 1987.

[8] 22 March 2007, Citing Internet Source URL. *Color Blindness.* http://www.eyecaresources.com/conditions/

[9] 1 April 2007, Citing Internet Source URL. *Image Processing Using MATLAB*. http://www.triindia.co.in/resources/?p=47

APPENDIX A

SELECTED PROGRAMMING CODE

```
function varargout = RGBColorImageDetection(varargin)
% RGBCOLORIMAGEDETECTION M-file for RGBColorImageDetection.fig
%      RGBCOLORIMAGEDETECTION, by itself, creates a new
RGBCOLORIMAGEDETECTION or raises the existing
%      singleton*.
%
%      H = RGBCOLORIMAGEDETECTION returns the handle to a new
RGBCOLORIMAGEDETECTION or the handle to
%      the existing singleton*.
%
%      RGBCOLORIMAGEDETECTION('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in RGBCOLORIMAGEDETECTION.M with the
given input arguments.
%
%      RGBCOLORIMAGEDETECTION('Property','Value',...) creates a new
RGBCOLORIMAGEDETECTION or raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before RGBColorImageDetection_OpeningFunction
gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to
RGBColorImageDetection_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
RGBColorImageDetection

% Last Modified by GUIDE v2.5 17-Nov-2007 23:10:33

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn',
@RGBColorImageDetection_OpeningFcn, ...
                   'gui_OutputFcn',  @RGBColorImageDetection_OutputFcn,
...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before RGBColorImageDetection is made visible.
function RGBColorImageDetection_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to RGBColorImageDetection (see
VARARGIN)

% Choose default command line output for RGBColorImageDetection
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes RGBColorImageDetection wait for user response (see
UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = RGBColorImageDetection_OutputFcn(hObject,
eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[ColorImage, Interface] = uigetfile({'*.jpg'}, 'Pick an Image File');
G = imread([Interface, ColorImage]);
axes(handles.axes1);
imshow(G);

handles.G=G;
guidata(hObject,handles);
```

```
% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Get the Input Data By Right-Click Mouse at Point Needed
a = impixel;

%Color Definition Of The Image
if a==[255 127  0]
    message= strcat('Type Of Color = Dark Orange');
    msgbox(message);

elseif a==[128 64 0]
    message= strcat('Type Of Color = Chocolate');
    msgbox(message);

elseif a==[255 0  128]
    message= strcat('Type Of Color = Pink');
    msgbox(message);

elseif a==[234 150  122]
    message= strcat('Type Of Color = Dark Salmon');
    msgbox(message);


elseif a>=[240 240  240]
    message= strcat('Type Of Color = White');
    msgbox(message);

elseif a==[0 255 255]
    message= strcat('Type Of Color = Cyan');
    msgbox(message);

elseif a>=[255 0 254]
   message= strcat('Type Of Color = Magenta');
   msgbox(message);

elseif a>=[225 225 225]
   message= strcat('Type Of Color = Almost White');
   msgbox(message);

elseif a>=[220 150 150]
   message= strcat('Type Of Color = Cartoon Brown');
   msgbox(message);
```

```
elseif a>=[70 100 200]
    message= strcat('Type Of Color = Sky Blue');
    msgbox(message);

elseif a>=[160 150  140]
    message= strcat('Type Of Color = Light Gray');
    msgbox(message);

elseif a>=[163 40 40]
    message= strcat('Type Of Color = Brown');
    msgbox(message);

elseif a<=[30 30 30]
    message= strcat('Type Of Color = Black');
    msgbox(message);

elseif a<=[150 0 220]
    message= strcat('Type Of Color = Violate');
    msgbox(message);

elseif a<=[85 85 80]
    message= strcat('Type Of Color = Light Black');
    msgbox(message);

elseif a<=[90 70 65]
    message= strcat('Type Of Color = Dark Brown');
    msgbox(message);

elseif a<=[100 70 65]
    message= strcat('Type Of Color = Dark Brown');
    msgbox(message);

elseif a<=[40 140 40]
    message= strcat('Type Of Color = Dark Green');
    msgbox(message);

elseif a<=[150 150 50]
    message= strcat('Type Of Color = Light Yellow');
    msgbox(message);

elseif a<=[120 120  120]
    message= strcat('Type Of Color = Dark Gray');
    msgbox(message);

elseif a<=[80 80 170]
    message= strcat('Type Of Color = Dark Blue');
    msgbox(message);

elseif a<=[150 120 120]
    message= strcat('Type Of Color = Light Brown');
    msgbox(message);

elseif a<=[153 153  153]
```

```
    message= strcat('Type Of Color = Gray');
    msgbox(message);

elseif a<=[150 0 220]
    message= strcat('Type Of Color = Violate');
    msgbox(message);


elseif a<=[200 70  70]
    message= strcat('Type Of Color = Dark Red');
    msgbox(message);



elseif a<=[200 200 80]
    message= strcat('Type Of Color = Dark Yellow');
    msgbox(message);

elseif a>=[0 0 170]
    message= strcat('Type Of Color = Blue');
    msgbox(message);

elseif a>=[160 160 0]
    message= strcat('Type Of Color = Yellow');
    msgbox(message);

elseif a>=[153 0  0]
    message= strcat('Type Of Color = Red');
    msgbox(message);

elseif a>=[0 153 0]
    message= strcat('Type Of Color = Green');
    msgbox(message);
end



% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

G=handles.G;

figure,imshow(G);

%On The Pixel Value For Cursor
pixval

%To Check Pixel Color Region For Detail
impixelregion
```

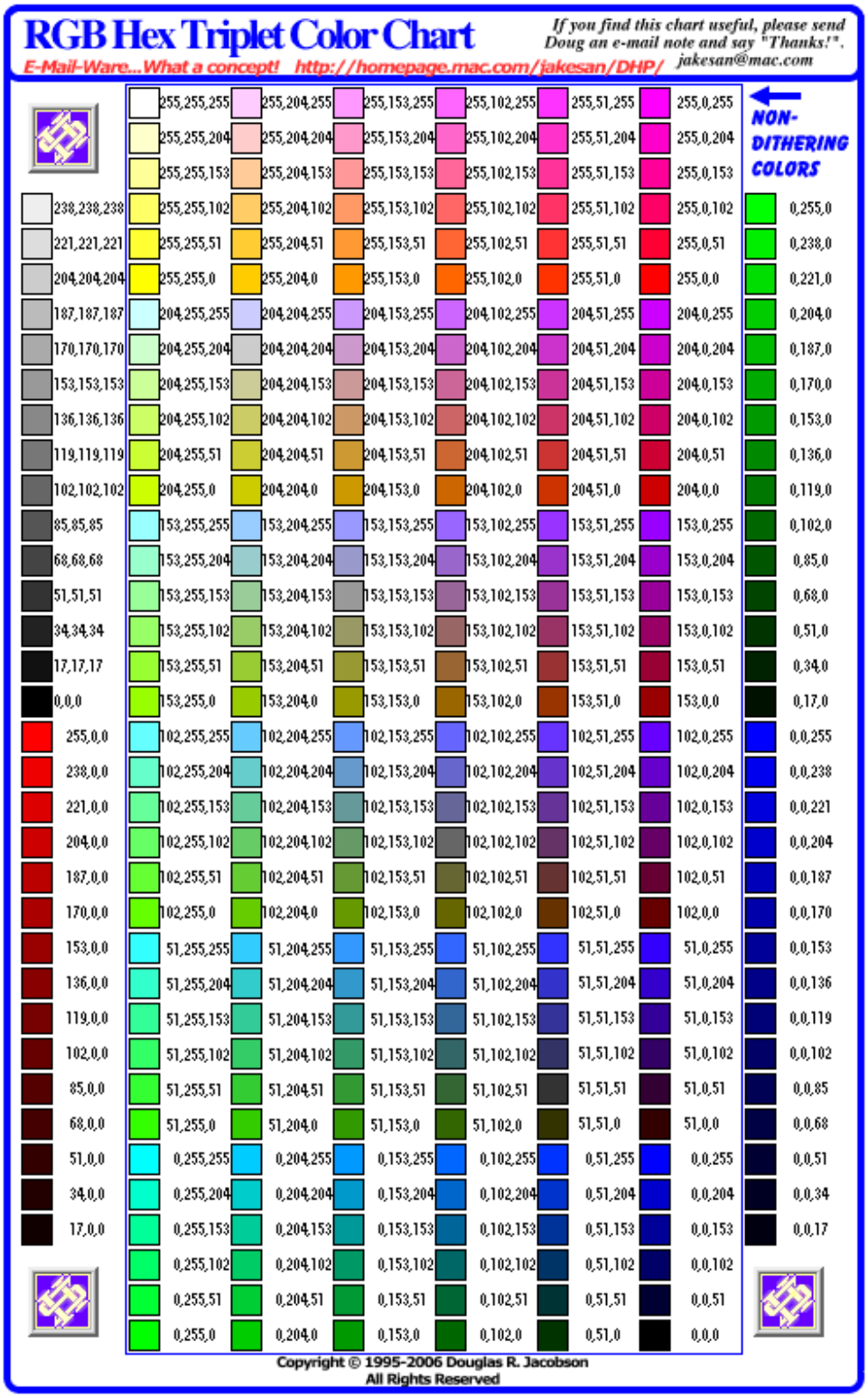
```
%Convert The Image For RGB Statistic In Simulink
I=im2double(G);
```

```
% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Exit Button
button = questdlg(' Are you sure to exit?','RGB Color Image
Detection','Yes','No','No');
switch button
  case 'Yes',
    disp('Thank You for using this System');
    close all
  case 'No',
    quit cancel;
end
```

```
% --- Executes on button press in pushbutton9.
function varargout=pushbutton9_Callback(h, eventdata, handles,varargin)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
figure (About)

% --- Executes on button press in pushbutton10.
function varargout=pushbutton10_Callback(h, eventdata,
handles,varargin)
% hObject    handle to pushbutton10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
figure (Help)
```

APPENDIX B

COLOR DEFINITION


RGB Hex Triplet Color Chart
If you find this chart useful, please send Doug an e-mail note and say "Thanks!".
E-Mail-Ware...What a concept! http://homepage.mac.com/jakesan/DHP/ jakesan@mac.com
NON-DITHERING COLORS
Copyright © 1995-2006 Douglas R. Jacobson
All Rights Reserved

Hexagon Color