American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-7, Issue-9, pp-173-184
www.ajer.org
Research Paper Open Access

Some Of The Techniques For Images Processing Using Matlab

Ahmed Mohamed Ali Karrar, Professor: Jun Sun
School of Internet of Things & Engineering Jiangnan University - china
Corresponding Author: Ahmed Mohamed Ali Karrar

ABSTRACT- *In this paper, we presented a range of useful MATLAB applications for image processing and color detection, each of which is a useful graphical interface useful for those who are not familiar with the MATLAB software behind the Image Processor. The applications we provided (cropping the image, Rotate an Image, remove noise and remove it, RGB image histogram, get pixel values for the image, change pixel values for images, resize the mage, RGB format, Detect Regions of Texture in Images).*
MATLAB-based image processing is a very convenient and very easy platform to create an algorithm. The image is a matrix of pixel values. MATLAB considers all inputs as a matrix. For this reason, MATLAB provides an easy image processing tool where users can easily access and edit the value of each pixel of image matrices. In addition, there is a MATLAB treatment toolbox for this purpose.
KEYWORDS— *image processing, MATLAB, Image processing toolbox, GUI*

Date of Submission: 31-08-2018 Date of acceptance: 15-09-2018

I. INTRODUCTION

The basic data structure in MATLAB is the array, an ordered set of real or complex elements. This object is naturally suited to the representation of images, real-valued, ordered sets of color or intensity data. MATLAB stores most images as two-dimensional arrays (i.e., matrices), in which every component of the network compares to a solitary pixel in the showed image. (Pixel is gotten from picture component and as a rule, signifies a solitary speck on a PC show.) For instance, an image made out of 100 rows and 200 columns of various hued dabs would be put away in MATLAB as a 100x200 matrix. A few images, for example, RGB, require a three-dimensional array, where the primary plane in the third dimension represents the red pixel intensities, the second plane represents the green pixel intensities, and the third plane represents the blue pixel intensities.

For the most part, clients manage three kinds of image, consequently three distinct grids. High contrast or twofold image matrix comprises of just zero and one, one being the brighter portion and zero being the dark part. For the most part, images are 8bit and comparing the image matrix is 256x256. The grayscale image is likewise a 2-dimensional lattice with every component esteem differing from 0 to 256. Like the grayscale image, the RGB image can be indicated by the network with every pixel esteems shifting from 0 to 256. In the event of the RGB image, three separate networks for every red, green and blue segment cover to frame an RGB image of 256x256x3 dimension. Since we are present all around familiar with the image as a matrix, now any numerical activities can be performed on an image that should be possible with a matrix.[1]

1.2 Introduction to Image Processing Toolbox

Handling Toolbox is a gathering of capacities that expand the ability of the MATLAB® numeric registering condition. 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
- The region of interest operation

A large number of the tool compartment capacities are MATLAB M-records, the arrangement of MATLAB articulations that execute particular picture handling calculations. You can see the MATLAB code for these capacities utilizing the announcement:

type function_name You can broaden the capacities of the Image Processing Toolbox by composing your own particular M-files, or by utilizing the toolkit in the blend with different tool compartments, for example, the Signal Processing Toolbox and the Wavelet Toolbox.

You can do perform more confused activities on images. The picture preparing toolkit permits such controls as:

Direct visualization of images in MATLAB

Color space conversions (e.g. between RGB, HSV, L*a*b*, and so on)

Object grouping and data collection

Filtering and fast convolution

Fourier analysis of images

Image arithmetic

Morphological operations

and many others.[2]

image Processing Toolbox™ gives a far-reaching set of reference-standard calculations and work process applications for image preparing, examination, perception, and calculation improvement. You can perform image division, picture improvement, commotion decrease, geometric changes, image enlistment, and 3D image handling.

image Processing Toolbox applications let you computerize regular picture handling work processes. You can intuitively fragment image information, look at image enlistment methods, and bunch process vast datasets. Perception capacities and applications let you investigate images, 3D volumes, and recordings; modify differentiate; make histograms; and control areas of premium (ROIs).

You can quicken your calculations by running them on multicore processors and GPUs. Numerous toolkit capacities bolster C/C++ code age for work area prototyping and implanted vision system deployment.[3]

II. LITERATURE REVIEW

The fast advancement of computerized innovation has specifically affected the techniques in image-processing techniques and the implementation of survey image processing systems. This fundamental improvement has been moved from a mainframe system to a PC Desktop platform. The client currently can without much of a stretch perform and handle all kind operations, functions, and processing techniques ranging from the little scale to the huge scale in statistical operations.

Our point is to utilize the MATLAB programming platform [4] as an apparatus for change and creating of Image Processing programming bundle [5] [4]. Such techniques are cropping the image, Rotate an Image, remove noise and remove it, RGB image histogram, get pixel values for the image, change pixel values for images, resize the mage, RGB format, Detect Regions of Texture in Images. These techniques should capable of using GUI unit to display input image, output image and various click button for various image processing techniques as well as a description of the techniques so the user will earn effectively the application on how the image is analyzed. The number of image function packages for the image processing have increased over the few years.

M. Mansourpour, M.A. Rajabi, J.A.R. Blais proposed the Frost Filter technique for image preprocessing. This filter assumes multiplicative noise and stationary noise statistics [6]. A gradient-based adaptive median filter is used for removal of speckle noises in SAR images. This method is used to reduce/remove the speckle noise, preserves information, edges and spatial resolution and it was proposed by S.Manikandan, Chhabi Nigam, J P Vardhani and A.Vengadarajan [7]. The Wavelet Coefficient Shrinkage (WCS) filter is based on the use of Symmetric Daubechies (SD) wavelets [8]. The WCS filter developer by L. Gagnon and A. Jouan in 1997. Discrete Wavelet Transform (DWT) has been employed in order to preserve the high-frequency components of the image [9]. In order to achieve a sharper image, an intermediate stage for estimating the high-frequency subbands has been proposed by P. Karunakar, V. Praveen, and O. Ravi Kumar.

III. VISUALIZATION OF IMAGES IN MATLAB

Read and Display an Image:

We can read standard image documents by utilizing the imread function. The sort of information returned by imread relies up on the kind of image you are reading. For example, read image2.jpg by typing:

```
A = imread('image2.jpg');
```

which will stores image2.jpg in a matrix named A.

Presently show the image utilizing the imshow function. For example, type:

```
imshow(A);
```

[10]

3.1 Code Generation for Image Processing:

Certain Image Processing Toolbox functions have been empowered to create C code utilizing MATLAB Coder™. To utilize code generation with image processing functions, take after these steps:

• Write your MATLAB function or application as you would ordinarily, utilizing functions from the Image Processing Toolbox.

• Add the %#codegen compiler mandate to your MATLAB code.

• Open the MATLAB Coder application, create a project, and add your file to the project. Once in MATLAB Coder, you can check the preparation of your code for code generation. For a case, your code may contain functions that are not empowered for code generation.

Make any adjustments required for code generation.

• Generate code by clicking Generate on the Generate Code page of the MATLAB Coder application. You can create a MEX file, a mutual library, a dynamic library, or an executable.

Regardless of whether you tended to all preparation issues distinguished by MATLAB Coder, you may still, experience constructs issues. readiness check only looks at function dependencies.

When you endeavor to create code, MATLAB Coder may find coding designs that are not bolstered for code generation. View the error report and alter your MATLAB code until the point when you get a successful build. To produce code from MATLAB code that contains image processing functions, you should have the MATLAB Coder programming.

When working with generated code, take note of the accompanying:

•For some Image Processing Toolbox functions, code generation depends on a precompiled, platform-specific shared library.[11]

3.2 Some processes for image processing:

3.2.1 Crop an Image

Cropping is the removal of undesirable external regions from a photographic or illustrated image. The procedure, for the most part, comprises of the evacuation of a portion of the fringe regions of a picture, to enhance its surrounding, to change the perspective proportion, or to emphasize or separate the topic from its experience. The process of cropping is common to the photographic, film processing, broadcasting, graphic design, and printing businesses.

In general Cropping an image means making another image from a part of an original image. To crop an image using the Image Viewer, use the Crop Image tool.

Using the Crop Image Tool:

By default, if you close the Image Viewer, it does not save the modified image data. To save the cropped image, you can use the Save As option from the Image Viewer File menu to store the modified data in a file or use the Export to Workspace option to save the modified data in the workspace variable.

1.View an image in the Image Viewer.

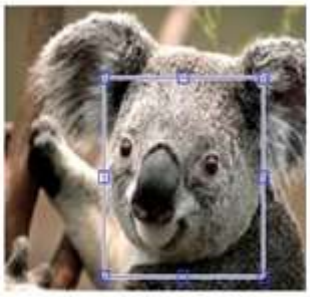
```
I = imread('koa.jpg');
imtool(I)
```

2.Start the Crop Image tool by clicking Crop Image in the Image Viewer toolbar or selecting Crop Image from the Image Viewer Tools menu. (Another option is to open a figure window with imshow and call imcrop from the command line.) When you move the pointer over the image, the pointer changes to cross hairs .

3. Define the rectangular crop region, by clicking and dragging the mouse over the image. You can fine-tune the crop rectangle by moving and resizing the crop rectangle using the mouse. Or, if you want to crop a different region, move to the new location and click and drag again. To zoom in or out on the image while the Crop Image tool is active, use Ctrl+Plus or Ctrl+Minus keys. Note that these are the Plus(+) and Minus(-) keys on the numeric keypad of your keyboard. The following figure shows a crop rectangle being defined using the Crop Image tool.

4. When you are finished defining the crop region, perform the crop operation. Double-click the left mouse button or right-click inside the region and select Crop Image from the context menu. The Image Viewer displays the cropped image. If you have other modular interactive tools open, they will update to show the newly cropped image.

5. To save the cropped image, use the Save as option or the Export to Workspace option on the Image Viewer File menu.[12]


3.2.2 Rotate an Image

To rotate an image, use the imrotate function. When you rotate an image, you assign the image to be rotated and the rotation angle, in degrees. If you assign a positive rotation angle, imrotate rotates the image counterclockwise; if you assign a negative rotation angle, imrotate rotates the image clockwise.

By default, the output image is large enough to include the entire original image. Pixels that fall outside the boundaries of the original image are set to 0 and appear as a black background in the output image. You can, however, specify that the output image be the same size as the input image, using the 'crop' argument.

By default, imrotate uses nearest-neighbor interpolation to determine the value of pixels in the output image, but you can specify other interpolation methods.

Read an image into the workspace.

I = imread('koa.jpg');

Rotate the image 35 degrees counterclockwise. In this example, specify bilinear interpolation.

J = imrotate(I,35,'bilinear');

Display the original image and the rotated image.

figure

imshowpair(I,J,'montage') [13]


3.2.3 Gray Conversion

Gray conversion is done mainly to convert a RGB image (three dimensional matrix) to gray scale (two dimensional matrix) having pixel values ranging from 0 to 255.

3.2.3.1 Noise Addition and Removal:

Various types of noise get added to an image when a snapshot is taken. In order to get rid of these noises various types of filters are used. To illustrate this authors have added a noise to an image externally and then applied various filters to get rid of it and evaluated the results. Since noises are two dimensional and RGB images are three dimensional, dimensional mismatch has to be avoided while adding the noises. For this reason RGB image is converted to gray image and then noise addition and removal is performed.

Noises can be of various types such as Poisson, Salt and pepper, Gaussian and Speckle. Median and adaptive filter are mainly in use. Figure 2 shows that Poisson noise has been added to the image in axis 1, and the noise added image is in axis 2 and after applying Adaptive filter to the image we get the filtered image as shown in axis 3.

3.2.3.2 Removing Noise from an Image

Digital images are prone to a variety of types of noise. Noise is the result of errors in the image acquisition process that result in pixel values that do not reflect the true intensities of the real scene. There are several ways that noise can be introduced into an image, depending on how the image is created. For example:

• If the image is scanned from a photograph made on film, the film grain is a source of noise. Noise can also be the result of damage to the film, or be introduced by the scanner itself.

• If the image is acquired directly in a digital format, the mechanism for gathering the data (such as a CCD detector) can introduce noise.

• Electronic transmission of image data can introduce noise.

You can use linear filtering to remove certain types of noise. Certain filters, such as averaging or Gaussian filters, are appropriate for this purpose. For example, an averaging filter is useful for removing grain noise from a photograph. Because each pixel gets set to the average of the pixels in its neighborhood, local variations caused by grain are reduced.

Median filtering is similar to using an averaging filter, in that each output pixel is set to an average of the pixel values in the neighborhood of the corresponding input pixel. However, with median filtering, the value of an output pixel is determined by the median of the neighborhood pixels, rather than the mean. The median is much less sensitive than the mean to extreme values (called outliers). Median filtering is therefore better able to remove these outliers without reducing the sharpness of the image.

The following example compares the use of a linear Gaussian filter and a median filter to remove salt and pepper noise for the same image:

2.10 Image Intensity Adjustment

Image intensity adjustment is used to improve an image, Read image1.jpg again

```
A = imread('koa.jpg');  Multiply the image pixels values by two.
E = A.*2;
figure
imshow(E);
```

3.2.4 RGB image histogram:

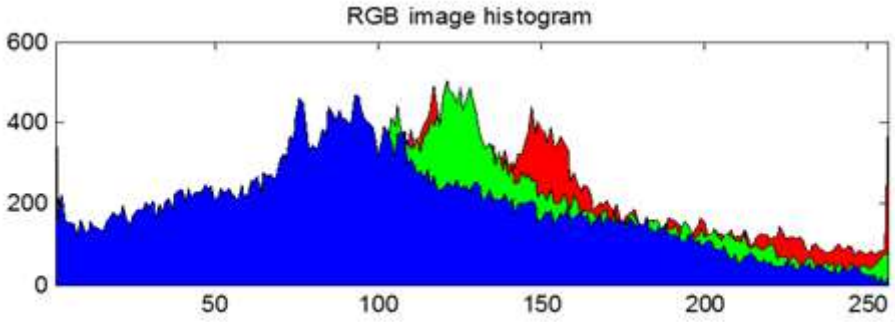
RGBHIST shows a histogram of the overall color balance of an image.Use the 'RGB' mode to get a quick sense of which colors are present in particular highlight, shadow, and midtone regions. The 'LAB' mode is more useful for determining the actual color balance of an image.

In image processing and photography, a color histogram is a representation of the distribution of colors in an image. For digital images, a color histogram represents the number of pixels that have colors in each of a fixed list of color ranges, that span the image's color space, the set of all possible colors.

The color histogram can be built for any kind of color space, although the term is more often used for three-dimensional spaces like RGB or HSV. For monochromatic images, the term intensity histogram may be used instead. For multi-spectral images, where each pixel is represented by an arbitrary number of measurements (for example, beyond the three measurements in RGB), the color histogram is N-dimensional, with N being the number of measurements taken. Each measurement has its own wavelength range of the light spectrum, some of which may be outside the visible spectrum.

If the set of possible color values is sufficiently small, each of those colors may be placed on a range by itself; then the histogram is merely the count of pixels that have each possible color. Most often, the space is divided into an appropriate number of ranges, often arranged as a regular grid, each containing many similar

color values. The color histogram may also be represented and displayed as a smooth function defined over the color space that approximates the pixel counts.



3.2.5 Get Pixel Information in Image:

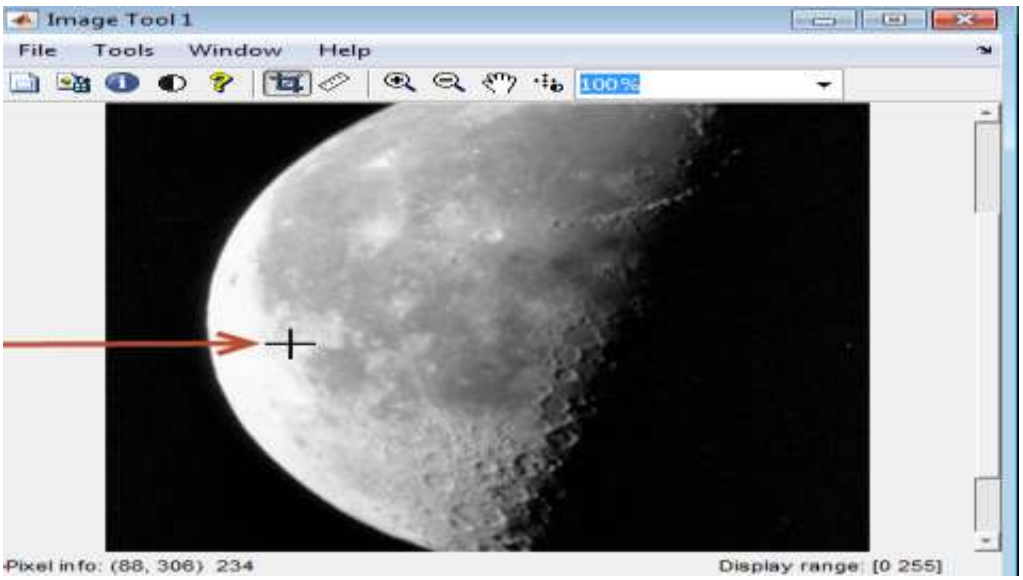
3.2.5.1 Determine Individual Pixel Values in Image Viewer

The Image Viewer displays information about the location and value of individual pixels in an image in the bottom left corner of the tool. (You can also obtain this information by opening a figure with imshow and then calling impixelinfo from the command line.) The pixel value and location information represent the pixel under the current location of the pointer. The Image Viewer updates this information as you move the pointer over the image.

For example, view an image in the Image Viewer.

imtool('moon.tif')

The following figure shows the Image Viewer with pixel location and value displayed in the Pixel Information tool.



*pixel under the pointer

*pixel info:(88,306) 234 (pixel location and intensity value)

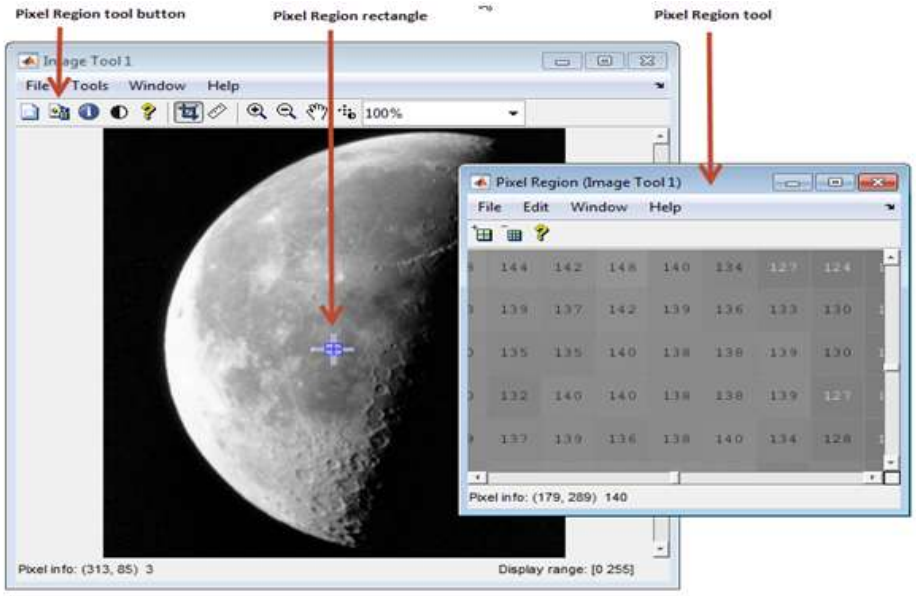
Saving the Pixel Value and Location Information

To save the pixel location and value information displayed, right-click a pixel in the image and choose the Copy pixel info option. The Image Viewer copies the x- and y-coordinates and the pixel value to the clipboard.

To paste this pixel information into the MATLAB® workspace or another application, right-click and select Paste from the context menu.

3.2.5.2 Determine Pixel Values in an Image Region

To view the values of pixels in a specific region of an image displayed in the Image Viewer, use the Pixel Region tool. The Pixel Region tool superimposes a rectangle, called the pixel region rectangle, over the image displayed in the Image Viewer. This rectangle defines the group of pixels that are displayed, in extreme close-up view, in the Pixel Region tool window. The following figure shows the Image Viewer with the Pixel Region tool. Note how the Pixel Region tool includes the value of each pixel in the display.



Selecting a Region

1-To start the Pixel Region tool, click the Pixel Region button in the Image Viewer toolbar or select the Pixel Region option from the Tools menu. (Another option is to open a figure using imshow and then call impixelregion from the command line.) The Image Viewer displays the pixel region rectangle in the center of the target image and opens the Pixel Region tool.

2-Using the mouse, position the pointer over the pixel region rectangle. The pointer changes to the fleur shape, ✣.

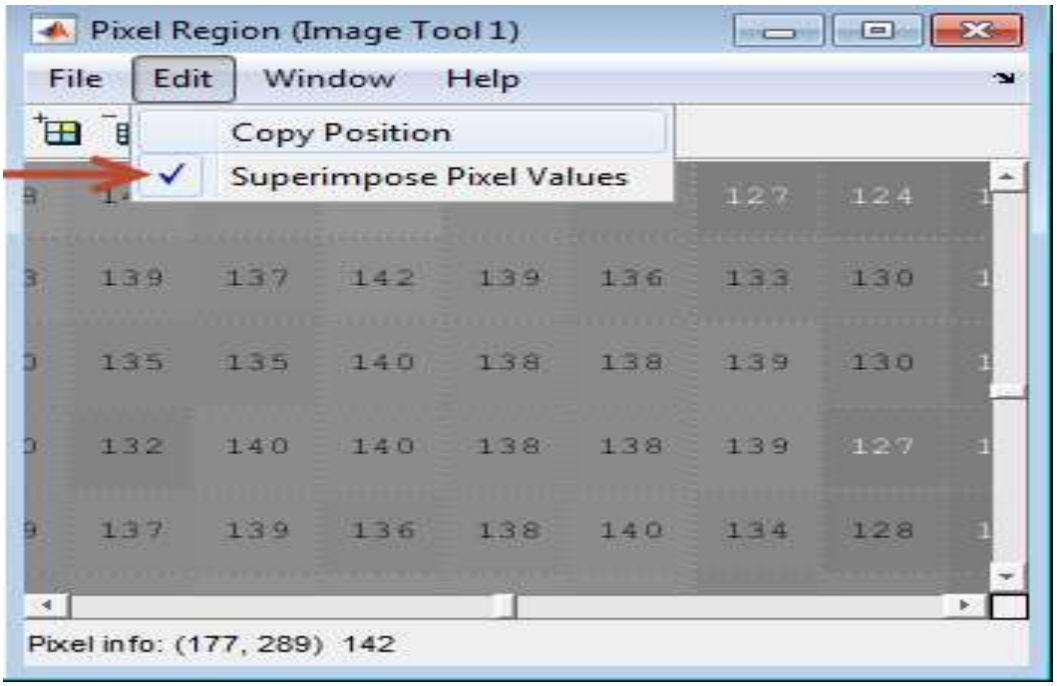
3-Click the left mouse button and drag the pixel region rectangle to any part of the image. As you move the pixel region rectangle over the image, the Pixel Region tool updates the pixel values displayed. You can also move the pixel region rectangle by moving the scroll bars in the Pixel Region tool window.

Customizing the View

To get a closer view of image pixels, use the zoom buttons on the Pixel Region tool toolbar. As you zoom in, the size of the pixels displayed in the Pixel Region tool increase and fewer pixels are visible. As you zoom out, the size of the pixels in the Pixel Region tool decrease and more pixels are visible. To change the number of pixels displayed in the tool, without changing the magnification, resize the Pixel Region tool using the mouse.

As you zoom in or out, note how the size of the pixel region rectangle changes according to the magnification. You can resize the pixel region rectangle using the mouse. Resizing the pixel region rectangle changes the magnification of pixels displayed in the Pixel Region tool.

If the magnification allows, the Pixel Region tool overlays each pixel with its numeric value. For RGB images, this information includes three numeric values, one for each band of the image. For indexed images, this information includes the index value and the associated RGB value. If you would rather not see the numeric values in the display, go to the Pixel Region tool Edit menu and clear the Superimpose Pixel Values option.



*Superimpose pixel values (Deselect to suppress pixel value display)

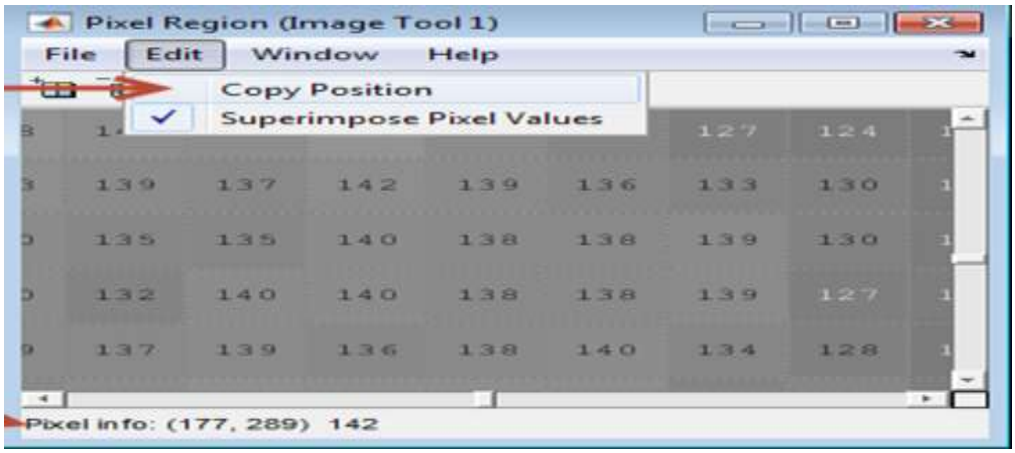
3.2.5.3 Determining the Location of the Pixel Region Rectangle

To determine the current location of the pixel region in the target image, you can use the pixel information given at the bottom of the tool. This information includes the x- and y-coordinates of pixels in the target image coordinate system. When you move the pixel region rectangle over the target image, the pixel information given at the bottom of the tool is not updated until you move the cursor back over the Pixel Region tool.

You can also retrieve the current position of the pixel region rectangle by selecting the Copy Position option from the Pixel Region tool Edit menu. This option copies the position information to the clipboard. The position information is a vector of the form [xmin ymin width height].

To paste this position vector into the MATLAB workspace or another application, right-click and select Paste from the context menu.

The following figure shows these components of the Pixel Region tool.



Copy position (get position of pixel Region rectangle)
*Pixel info:(177,289) 142 (location of pixel in target image)

Printing the View of the Image in the Pixel Region Tool

You can print the view of the image displayed in the Pixel Region tool. Select the Print to Figure option from the Pixel Region tool File menu. See Print Images for more information.

3.2.5.4 Determine Image Display Range in Image Viewer:

The Image Viewer provides information about the display range of pixels in a grayscale image. The display range is the value of the axes CLim property, which controls the mapping of image CData to the figure colormap. CLim is a two-element vector [cmin cmax] specifying the CData value to map to the first color in the colormap (cmin) and the CData value to map to the last color in the colormap (cmax). Data values in between are linearly scaled.

The Image Viewer displays this information in the Display Range tool at the bottom right corner of the window. The Image Viewer does not show the display range for indexed, truecolor, or binary images. (You can also obtain this information by opening a figure window with imshow and then calling imdisplayrange from the command line.)

For example, view an image in the Image Viewer.

imtool('moon.tif')

The following figure shows the Image Viewer displaying the image with display range information.

Display range tool

3.2.6 Changing Image Pixel Values:

You can change the values of specific image pixels. Type:

```
A(40:100,10:20,:) = 0;
Figure
imshow(A);
```

which changes the colors of the selected pixels into black color.

Now try:

```
A(40:100,10:20,:) = 255;
figure
imshow(A);
```

3.2.7 Image Resize:
Image resize is being used to resize the actual image to certain multiples. The syntax is 'imresize'.
MAGERESIZE able to resize an image. You can downsize and also upsize an image using scaling factor.

3.2.8 RGB Format:

The RGB model uses three primary colors, red, green and blue, This RGB panel is used to view the red, green and blue componenst of the image separately. It has already been mentioned that an RGB image is overlap of three two dimensional matrix.

the RGB color model is implemented in different ways, depending on the capabilities of the system used. By far the most common general-used incarnation as of 2006 is the 24-bit implementation, with 8 bits, or 256 discrete levels of color per channel. Any color space based on such a 24-bit RGB model is thus limited to a range of 256×256×256 ≈ 16.7 million colors. Some implementations use 16 bits per component for 48 bits total, resulting in the same gamut with a larger number of distinct colors. This is especially important when working with wide-gamut color spaces (where most of the more common colors are located relatively close together), or when a large number of digital filtering algorithms are used consecutively. The same principle applies for any color space based on the same color model, but implemented in different bit depths.

3.2.8.1 COLOR SPACE :

A color space is defined as a model for representing color in terms of intensity values.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.

3.2.8.2 RGB color space :

A RGB color space is any additive color space based on the RGB color model.[15] A particular RGB color space is defined by the three chromaticities of the red, green, and blue additive primaries, and can produce any chromaticity that is the triangle defined by those primary colors.[16] The complete specification of an RGB color space also requires a white point chromaticity and a gamma correction curve. As of 2007, sRGB is by far the most commonly used RGB color space.

3.2.9 Detect Regions of Texture in Images:
This example shows how to detect regions of texture in an image using the texture filter functions.

Read an image into the workspace and display it. In the figure, the background is smooth--there is very little variation in the gray-level values. In the foreground, the surface contours of the coins exhibit more texture. In this image, foreground pixels have more variability and thus higher range values.

IV. GETTING HELP IN MATLAB

For reference information about any of the functions, type in the MATLAB command window:
help functionname
For example:
help imread

V. CONCLUSION

To sum up, the outcome of this paper is a simple program that presents possibilities of Matlab and the GUIDE tool.

In addition to the practical part of the paper, the other goal was to learn new information from various areas related to the computer graphics and Matlab in general. The theory presented in this paper covers quite precisely the most important topics within the computer graphics, the great amount of time was given into getting to know the Image Processing Toolbox – one of the libraries of Matlab suite. Finding out what is the range of possible image transformations that can be done with this toolbox helped me to realize that Matlab is a great piece of software and may be widely used for many purposes.

The overall idea behind my paper was to show that Matlab is not only good for complicated and complex mathematical drawings but also provides a broad collection of regular image processing functions. Many programmers do not realize the full potential of Matlab and the Graphical User Interface tool that it provides.

Despite the outcome of my paper, including the image processing application, the topic stays open to the future modifications and development. Future work can be aimed to expand the set of applications than what has been used here.

REFERENCES

[1]. Image Processing Toolbox User's Guide COPYRIGHT 1993 - 2000 by The MathWorks, Inc.
[2]. Image Processing Toolbox User's Guide COPYRIGHT 1993 - 1998 by The MathWorks, Inc. All Rights Reserved.
[3]. https://ww2.mathworks.cn/help/images/index.html
[4]. J. Guerrero [2009], "Image Processing and Analysis with Matlab", International Conference on Electrical, Communications, and Computers, Cholula,Puebla, pp. xvi, 26-28 Feb. 2009.
[5]. Justyna Inglot [2012] ,"Advanced Image Processing with Matlab", in Bachelor"s Thesis Information Technology, Mikkeli University of Applied Sciences.
[6]. S.Manikandan, , Chhabi Nigam, J P Vardhani and A.Vengadarajan "Gradient based Adaptive Median filter for removal of Speckle noise inAirborne Synthetic Aperture Radar Images" ICEEA ,2011.
[7]. L. Gagnon and A. Jouan "Speckle Filtering of SAR Images - A Comparative Study between Complex-Wavelet-Based and Standard Filters"SPIE, 1997.
[8]. P. Karunakar, V. Praveen and O. Ravi Kumar"Discrete Wavelet Transform-Based Satellite Image Resolution Enhancement" Advance inElectronic and Electric Engineering, ISSN 2231-1297, Volume 3, Number 4, pp. 405-412, 2013.
[9]. Young GiByun, You Kyung Han, and Tae ByeongChae" A Multispectral Image Segmentation Approach for Object-based ImageClassification of High Resolution Satellite Imagery" KSCE,2012.
[10]. https://en.wikibooks.org/wiki/MATLAB_Programming/Advanced_Topics/Toolboxes_and_Extensions/Image_Processing_Toolbox#MATLAB's_image_arithmetic_functions
[11]. mage Processing Toolbox™ User's Guide© COPYRIGHT 1993–2016 by The MathWorks, Inc.
[12]. https://ww2.mathworks.cn/help/images/crop-image-using-image-viewer-app.html
[13]. https://ww2.mathworks.cn/help/images/rotate-an- image.html?searchHighlight=Rotate%20an%20Image&s_tid=doc_srchtitle

[14]. http://www.mathworks.com/help/images/get-pixel-information-in-image-viewer-app.html;jsessionid=ef023df738e14ed55272dad78e04
[15]. Poynton, Charles A. (2003). Digital Video and HDTV: Algorithms and Interfaces. Morgan Kaufmann. ISBN 1-55860-792-7.
[16]. Hunt, R. W. G (2004). The Reproduction of Colour (6th ed.). Chichester UK: Wiley–IS&T Series in Imaging Science and Technology. ISBN 0-470-02425-9.

Ahmed Mohamed Ali Karrar " Some Of The Techniques For Images Processing Using Matlab"American Journal of Engineering Research (AJER), vol. 7, no. 09, 2018, pp. 173-184