Image processing system using MATLAB-based analytics

Gerald K. Ijemaru[1], Augustine O. Nwajana[2], Emmanuel U. Oleka[3], Richard I. Otuka[4], Isibor K. Ihianle[5], Solomon H. Ebenuwa[6], Emenike Raymond Obi[7]

[1]School of Science, Technology and Engineering, University of the Sunshine Coast, Australia
[2]School of Engineering, University of Greenwich, United Kingdom (UK)
[3]College of Engineering, North Carolina A & T State University, United States of America (USA)
[4]Department of Computing, University of East London, United Kingdom (UK)
[5]Department of Computer Science, Nottingham Trent University, United Kingdom (UK)
[6]School of Computing, University of Greenwich, United Kingdom (UK)
[7]RaySoft Asset Analytics, Regina, Saskatchewan, Canada

Article Info

Article history:

Received May 27, 2021
Revised Jul 31, 2021
Accepted Aug 29, 2021

Keywords:

Algorithms
Digital image processing
Discrete cosine transforms
Image acquisition
Image enhancement
Image segmentation
MATLAB

ABSTRACT

Owing to recent technological advancement, computers and other devices running several image editing applications can be further exploited for digital image processing operations. This paper evaluates various image processing techniques using matrix laboratory (MATLAB-based analytics). Compared to the conventional techniques, MATLAB gives several advantages for image processing. MATLAB-based technique provides easy debugging with extensive data analysis and visualization, easy implementation and algorithmic-testing without recompilation. Besides, MATLAB's computational codes can be enhanced and exploited to process and create simulations of both still and video images. Moreover, MATLAB codes are much concise compared to C++, thus making it easier for perusing and troubleshooting. MATLAB can handle errors prior to execution by proposing various ways to make the codes faster. The proposed technique enables advanced image processing operations such as image cropping/resizing, image denoising, blur removal, and image sharpening. The study aims at providing readers with the most recent MATLAB-based image processing application-tools. We also provide an empirical-based method using two-dimensional discrete cosine transform (2D-DCT) derived from its coefficients. Using the most recent algorithms running on MATLAB toolbox, we performed simulations to evaluate the performance of our proposed technique. The results largely present MATLAB as a veritable approach for image processing operations.

Corresponding Author:

Augustine O. Nwajana
School of Engineering, University of Greenwich
SE10 9LS, United Kingdom (UK)
Email: a.nwajana@ieee.org

1. INTRODUCTION

Recent technological advances have triggered a growing research interest in image processing techniques. Generally, image processing techniques are rapidly increasing among technologies, and form an important research area in Engineering and Computing disciplines. This technique involves operations on an image for the purpose of enhancing the quality of the image or extracting some useful and desirable information from the image. It describes a type of signal processing operation on an image in which the image (e.g. photo or video frame) is an input signal with output characteristics of the original/input image. Image processing has been described as a mathematical operation of a computer on a two-dimensional photo

[1], [2], and a compendium of signal processing operations on an image that transform it into a digital form for the purpose of quality enhancement and extraction of useful information from it. The above definition mathematically presents an image as a function of two real variables in the x and y coordinates which can be digitally processed with the brightness of the image represented by the amplitude of the function. The image data is digitally processed after being sampled and converted into a matrix of numbers which are quantized and digitally represented. Accordingly, captured image data must undergo some critical phases described in [3] and include image pre-processing, image enhancement and display, and extraction of useful information from the image data. Incidentally, image processing technique has become more relevant in our daily lives as it has great impact and applications in technical fields exploiting different types of electronic devices including computers, digital cameras, and mobile phones. Image data can take the form of grayscale image, index image, true color image, and binary image. A grayscale image depicts one whose pixel value is a single sample that indicates only the amount of light. This type of image consists essentially of shades of gray with variations ranging from black showing the weakest intensity to white showing the strongest intensity.

Although digital images can be produced by exploiting different image processing applications running in several physical devices (such as ultrasonic, x-ray devices, cameras, and electron microscopes), MATLAB-based image processing can be exploited to give the desired quality suitable for a two-dimensional image. MATLAB is a high-level programming language with desirable features that are fundamental to both the computing and engineering disciplines with algorithms that have potentials for a wide range of applications. Owing to the rapid growth of technology, many scientists and engineers currently have ubiquitous access to multimedia software and hardware devices running image processing applications including 3D visualization. However, majority of these are limited in applications such that they require the support of other applications for image processing operations. Thus, this study is aimed at bringing to the fore the potentials of some of the novel features inherent in MATLAB as a standalone application tool for signal processing operations including digital image processing.

This paper presents an image processing technique using MATLAB-based analytics. Compared to the traditional and state-of-the-art image processing techniques and applications, MATLAB gives several advantages. For instance, MATLAB provides easy implementation and algorithmic testing without recompilation [4] as well as easy debugging with extensive data analysis and visualization [5]. In addition, MATLAB's computational codes can easily be enhanced and exploited to process and create simulations of both still and video images [6]. Compared to c++, MATLAB's codes are much concise thus enabling easy perusing and troubleshooting operations. Perhaps, the most fascinating feature of MATLAB-based image processing technique is its ability to handle errors prior to execution by proposing various ways to make the codes faster. Besides, MATLAB provides easy development of computational codes [4], provides a large database of built-in algorithms [7]. Moreover, MATLAB performs mathematical and numerical manipulation for the implementation of image processing techniques, and is also useful for successful documentation step-by-step image processing operations. Furthermore, maintenance of numerical precision is possible through the enhancement process of image in MATLAB. Image scrutinization and testing are also possible because MATLAB provides a quicker access to the image processing source-codes in its toolbox. This paper studies an updated performance evaluation of MATLAB-based image processing applications using the latest MATLAB toolbox. The present study is meant to give readers and researchers a broader perspective of the latest image processing applications running in MATLAB. An empirical method of image processing with the 2D-DCT is also provided. Figure 1 is the block diagram of an image processing technique. Basically, three steps are involved in image processing which are stated as shown in [3], [8]:

- Image acquisition. This involves importation of the image using the image acquisition tools such as optical scanner or digital photography
- Image analysis and manipulation
- Outputting the result of the image analysis

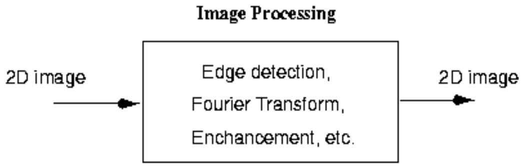


Figure 1. The block diagram of a 2D image processing technique

The rest of the paper is structured is being as. Some related works are presented in section 2, while section 3 gives an empirical method with 2D-DCT. Section 4 presents the system design and implementation and section 5 gives simulation results and analysis. Finally, the conclusion is presented in section 6.

2. REVIEW OF RELATED WORKS

In this section, some related works are reviewed. MATLAB has got good signal processing application tools in its toolbox that are designed to assist scientists and engineers to meet their goals. Some of the MATLAB-based application tools for image processing applications are hereby presented. Yang [9] proposed a MATLAB-based image processing for medical engineering application. He highlighted the importance of image data in medical engineering as the main source of information exchange and posited that although the medical engineering application of digital image processing attracts a huge cost, it always produces effective results by minimizing noise effects and enhancing the image quality. In medical engineering field, processed images carry much medical, and pathological information about a particular ailment. A software simulation analysis of an image recognition using a MATLAB-based technique is presented in [7]. The paper highlights methodological approach showing a MATLAB-based implementation of a software system for the analysis of image recognition. Buksh *et al.* [5] proposed a MATLAB-based image editing and color detection that exploits functions inherent in the MATLAB toolbox to implement various image processing applications. The use of MATLAB as a computing platform and backbone of emerging visual communication, suitable for developing, and testing a number of applications was emphasized in [10], where authors also suggested its use in the teaching of digital signal processing. According to the authors, MATLAB provides a GUI that enhances easy understanding of the concepts. Some insights into a MATLAB-based basic image processing technique such as edge detection, image restoration, segmentation and denoising are provided in [1], [11]. The authors suggest a digital image processing application that includes character recognition and feature extraction as well as remote sensing and computer vision. Duth and Deepa [12], Goel *et al.* [13], in authors suggest the use of RGB color model in the MATLAB picture toolbox for color detection and the implementation of the result of the algorithm as part of security architecturdhu [14] presented an analysis of a MATLAB-based image segmentation technique, which presented image segmentation as an essential step for image analysis, object representation, and image visualization to transform an image into the desirable format. Also, the various techniques employed in image segmentation and the analyses of their outputs using MATLAB software were also highlighted. Authors in Sahu and Dewangan [15], Mishra *et al.* [16] proposed fruit identification and classification using MATLAB-based image processing applications. This technique is essentially aimed at detecting good and bad fruits including identifying visibe defects in sizes and shapes of the fruits.

Sharma [17] presented the performance analysis of a MATLAB-based image processing algorithms applied in the biomedical sector. Their analysis showed the importance of biomedical imaging in the detection of disease as result of image processing. Some medical equipment that depend on image processing algorithms include ultrasound, MRI, and CT scan machine. Radiologists are those that make use of image processing algorithms to detect some diseases and abnormalities. The author further explained that MATLAB has proprietary tools used by image architects to design these algorithms. According to the author, image processing algorithms that are designed using MATLAB provide efficiency, accuracy, and flexibility and timing constraints. Abdullah *et al.* [18] highlighted a MATLAB-based digital image processing technique for video text extraction that specifically describes detection, tracking and recognition.

Ray *et al.* [19] proposed a MATLAB-based image processing technique for vehicle detection using a color-based approach. The main objective of this approach is to enable easy tracking of cars, check for road traffic safety or conduct traffic analysis [20], control traffic lights [21], and to enhance control over vehicular accidents. The use of MATLAB, SCILAB, and Raspberry Pi for image processing was discussed in [22], where authors proposed the use of edge detection technique to locate the boundaries of objects within images, and for image segmentation and data extraction. Other implementations using MATLAB-based technique for edge detection were studied. Some examples of edge detection techniques running on MATLAB include CANNY, SOBEL, and LAPLACIAN. To promote the growth and productivity of crop field, authors in [23]-[25] proposed a MATLAB-based image processing technique for plant disease detection and monitoring. Sing and Chetia, [24] investigated the use of MATLAB-based image processing technique for the detection of plant-leaf disease. Their technique took the form of image acquisition, processing, segmentation, feature extraction and classification. Details of image processing technique for fake note detection and for finger print recognition are presented in [26]-[28] respectively.

Koshy and Nair [29] proposed black box for accident analysis using MATLAB. The paper proposed a system that revealed the consciousness of the driver in non-real-time processing using MATLAB simulation of the image fetched from the black box device. The authors suggested three major ways of exploiting the system to include detection and recording of data, storing the recorded data in the black box and then analyzing the image data stored in black box using MATLAB. Several discussions were provided by different authors on image processing applications using MATLAB technique. Some of these include: parking space detection [30], printed circuit board (PCB) defect detection [31], edge detection [32], cell growth analysis [33], and grain counting [34]. Samarawickrama and Wickramasinghe in [35], authors used a MATLAB-based image processing technique to detect surface defect in ceramic tiles. Bansal *et al.* authors

[36] discussed image enhancement technique using MATLAB simulation-based approach. With this approach, authors distinguished two categories of image enhancement and concluded that the frequency domain approach presented a much easier implementation compared to the spatial domain approach.

Kumar and Kusagur [37] examined a MATLAB-based performance evaluation of various image denoising techniques on several images. The aim of the evaluation was to further highlight recent improvements in MATLAB-based image denoising techniques compared to the conventional noise filtering approaches. The evaluation showed improvements over the traditional techniques for image denoising. The study in [38], [39] investigated a MATLAB-based approach for edge detection and analysis of an image. Edge detection is a fundamental step in image processing operations utilizing several detection operators that perform image processing operations on an image. Details of the functions of each of the various edge detection techniques and operators (e.g. Sobel, Robert, Prewitt, Canny, and Laplacian of Gaussian) are also highlighted. The paper [39] notes that edge detection serves as a basis for several image processing operations including image enhancement, image tracking, image segmentation, and coding. Image segmentation is regarded as one of the major steps in image processing techniques and a difficult and complex aspect of image processing. Segmentation method is used to partition a digital image into multiple portions prior to analysis. The method is also used to differentiate various objects in an image and is required for the purpose of analysis. Major image segmentation techniques are discussed in [40]. A comprehensive review using a MATLAB-based approach is provided in [41].

3. EMPIRICAL-BASED MODEL WITH 2D-DCT

One of the most commonly used algorithms for signal processing applications is the two-dimensional discrete cosine transform (2D-DCT). This algorithm finds major application in data compression and forms the basis for the widely utilized JPEG-an image compression algorithm. The DCT has the property and potential to make even the most virtually significant information about an image to be focused using few coefficients. The DCT has extended coefficients that are significant for image recognition tasks including face recognition. The DCT empirical-based model is predicated on image transformation from the spatial domain to the frequency domain. This process involves sampling of an image by removing or filtering higher frequency coefficients of the samples. Since lower frequency coefficients are very critical in an image compared to the higher frequency coefficients, the DCT model discards the higher frequencies while the remainder of the frequency coefficients are quantized. This process sees to the reduction of the image data volume without compromising the quality of the image. The 2D-DCT of a C x D matrix A is depicted by:

$$K_{st} = \alpha_s \alpha_t \sum_{c=0}^{C-1} \sum_{d=0}^{D-1} \mathrm{A}_{mn} \cos\left(\frac{\pi(2c+1)s}{2C}\right) \cos\left(\frac{\pi(2d+1)t}{2D}\right) \tag{1}$$

$$0 \le s \le C-1; 0 \le t \le D-1;$$

The values K_{st} denote the DCT coefficients. The inverse transform of DCT given as 2D-IDCT is denoted by:

$$\mathrm{B}_{cd} = \sum_{s=0}^{C-1} \sum_{t=0}^{D-1} \alpha_s \alpha_t \; K_{st} \cos\left(\frac{\pi(2c+1)s}{2C}\right) \cos\left(\frac{\pi(2d+1)t}{2D}\right) \tag{2}$$

$$0 \le s \le C-1; \; 0 \le t \le D-1$$

The values α_s and α_t in (1) and (2) are given by:

$$\alpha_s = \begin{cases} \sqrt{\frac{1}{C}} \\ \sqrt{\frac{2}{C}} \end{cases}, s = 0; 1 \le s \le C-1 \tag{3}$$

$$\alpha_t = \begin{cases} \sqrt{\frac{1}{D}} \\ \sqrt{\frac{2}{D}}, \end{cases}, t = 0; 1 \le t \le D-1 \tag{4}$$

The proposed technique utilizes the DCT matrix inherent in the MATLAB-based image processing toolbox and is effective for small square inputs like image blocks of 8 x 8 pixels. A transform matrix K of C x D is denoted by:

$$K_{st} = \begin{cases} \sqrt{\frac{1}{C}} \\ \sqrt{\frac{2}{C}} \end{cases} \cos\left(\frac{\pi(2t+1)s}{2C}\right); s = 0, 1 \leq s \leq C-1; 0 \leq t \leq C-1 \tag{5}$$

A digital camera of dimension 1200 x 1600 pixels is used to capture face images. And captured image data have the following characteristic features:

- Common illumination
- Light background colour
- Frontal upright position
- Tilting and rotation tolerance up to 20 degrees

Firstly, image acquisition is employed whereby a digital camera is deployed to capture face images. Captured image data are pre-processed using adobe photoshop CS2. Then a MATLAB-based analytic is employed to perform interpolation to resize the pre-processed face image from 512 x 512 to 8 x 8 pixels. This design technique is employed for calculating the 2D-DCT image block of dimensions 8 x 8 pixels in which 8 out of the 64 DCT coefficients are used for masking while the remaining 56 coefficients are discarded. The next step is to perform image reconstruction by calculating the inverse transform (i.e. 2D-IDCT) of each block using the matrix transform method. Finally, an output image is obtained which represents both in size and form the original input image of size 8 x 8 pixels. The above description is represented in Figure 2 for a sample image.

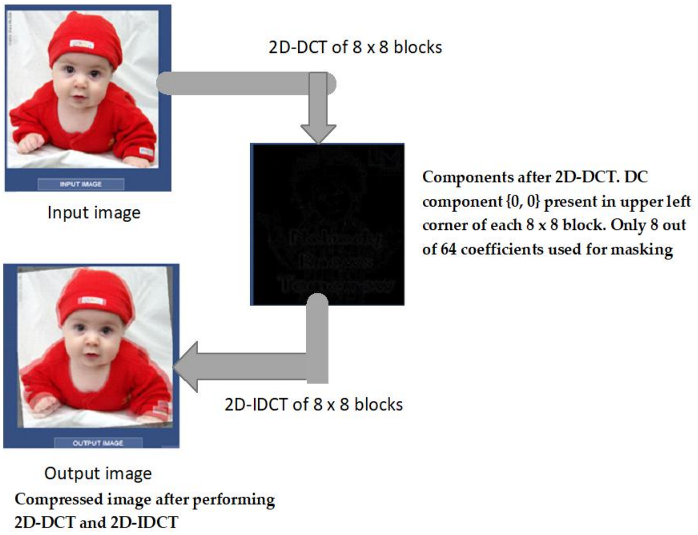

2D-DCT of 8 x 8 blocks
INPUT IMAGE
Input image
Components after 2D-DCT. DC component {0, 0} present in upper left corner of each 8 x 8 block. Only 8 out of 64 coefficients used for masking
OUTPUT IMAGE
2D-IDCT of 8 x 8 blocks
Output image
Compressed image after performing 2D-DCT and 2D-IDCT

Figure 2. A 2D-DCT computation of an image

4. SYSTEM DESIGN AND IMPLEMENTATION

This section describes the system design for the proposed image processing technique. The main features of the system are (1) workability and extensibility, (2) usability, and (3) generality.

4.1. System design methodology

The most common method of image acquisition in image processing task is conducted using a digital photography [42], usually a digital camera or a preloaded image from a memory source [12]. Image processing technique is used to operate some processes on a picture which includes an image enhancement or the removal of some functional data from the image. Image processing is a type of signal processing, where some operations are performed on the input image data to enhance the quality and extract useful information. Major image processing techniques include image enhancement, image restoration and compression. Digital image processing technique is very effective and notably extensive with wide area of application.

Image enhancement involves a process that enhances the quality of the original image including sharpening some of the features of the image to obtain a more useful graphic display suitable for analyzing the image data. The process consists of the following: gray-level adjustment, filtering, edge cropping and sharpening, interpolation and magnification, and adjustment of color. On the other hand, image restoration is the process that is meant for removal of image noise and estimation of the clean creative image. Blurred or misfocused images and noise are forms of images that can undergo image restoration to obtain the desired output image.

All operations are performed on RGB image. In the course of this operation, the RGB image is converted to gray-scale image, black & white image and binary image by some processes such as edge detection, noise removal, and addition. A three-dimensional (3D) matrix MxNx3 is used to represent RGB image format where the image resolution is a matrix of MxN showing the color picture of the image represented by each dimension of the RGB. The following segments of the picture (green, blue, and red) represent the input matrix. Figure 3 shows a color conversion for an image processing technique, while Figure 4 shows a block diagram of an image processing technique for the proposed system. Algorithm 1 is meant for color detection. MATLAB is used to achieve a great degree of transformation for a piecewise linear function using mat2gray () enhancement function. The basic codes are shown in Algorithm 2

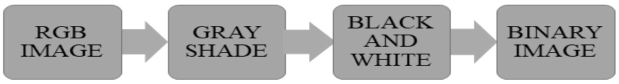


Figure 3. Color conversion for image processing techniques

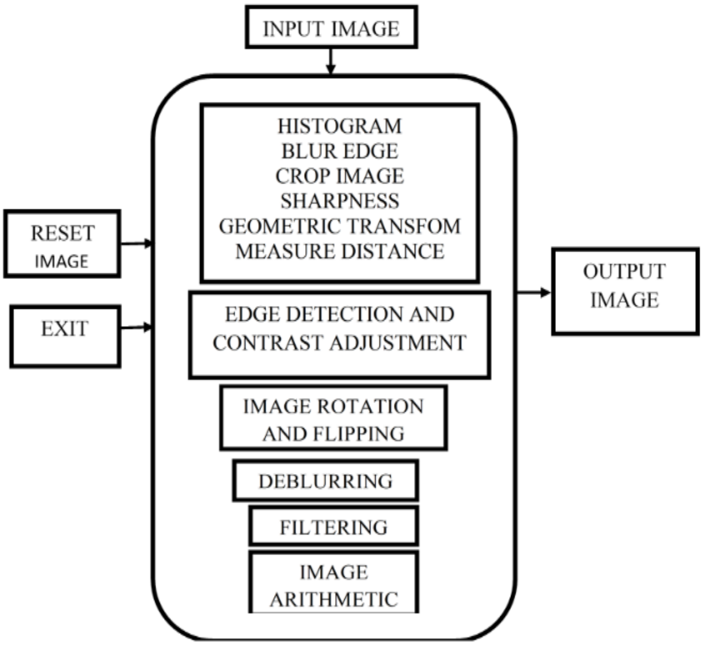


Figure 4. Block diagram of image processing techniques for this project software

Algorithm 1. An algorithm for detecting colors in an image

```
Algorithm1: An Algorithm for detecting colors in an image
    1: f(x) = input image; p(x) = color bands (Red, Blue, Green)
    2: Read f(x)
    3: Create p(x)f(x)
    4: Compute and plot
    5: Convert f(x) to gray-scale image
    6: Track Red color and subtract the red segments from step 5 to remove the blue
    segments in f(x)
    7: Apply median-filter to denoise
    8: Convert the gray-scale picture to a black & white image
    9: Remove pixels not up to 300px
    10: Select every segment in the picture
    11: Calculate the area, geometric center, minimum bounding rectangle (envelop) using
    region props
    12: Stop
```

Algorithm 2. Piecewise linear transformation using mat2gray () gray enhancement function

```
ALGORITHM2 : Piecewise Linear Transformation using mat2gray() gray enhancement function
    g1 = getimage(gco);
     figure
    imshow(g1)
    p0 = 0; q0 = 0;
    p1 = 30; q1 = 20
    p2 = 170; q2 = 220;
    p3 = 250; q3 = 250;
    figure, plot([p0, p1, p2, p3], [q0, q1, q2, q3])
    s1 = (q1 − q0)/(p1 − p0);
    t1 = q0 − s1 ∗ p0;
    s2 = (q2 − q1)/(p2 − p1);
    t2 = q1 − s2 ∗ p1;
    s3 = (q3 − q2)/(p3 − p2);
    t3 = q2 − s3 ∗ p2;
    [c, d] = size(g1);
    g2 = double(g1);
    for i = 1: c
     for j = 1: d
     P  = g2(i, j);
     q(i, j) = 0;
     if (p >= f1) & (p <= p2)
     q(i, j) = s1 ∗ p + t2;
    else if(p ≥ p2) & (p ≤ p3)
     q(i. j) = s3 ∗ p + t3;
     end
     end
     end
     figure
     imshow(mat2gray(g))
```

4.2. System functions and commands

In theory, the image is a two-dimensional (2D) linear function. Incidentally, sampling, and quantization of the image becomes critical, if the image must be digitally processed on a computer hardware. A 2D image that is uniformly sampled falls under the MxN samples, where it is an integer array. Therefore, a matrix description of the digital image is considered the most percipient and sample approach. The advantage of a MATLAB-based approach is its capability to process matrix operations. Hence, digital images can be processed using MATLAB-based analytics with convenience. The system supports various types of operations on an image and consists of some powerful functions including conversion functions to realize colour conversion and editing functions to perform geometric operations on an image. All these functions can be realized by simply writing some MATLAB code in M file based on MATLAB programming language. Table 1 shows some of the MATLAB programming codes and their functions used in the design, while Figure 5 shows the effect of grayscale image processing.

Table 1. Some MATLAB programming codes and their functions

MATLAB Command Codes	Function	MATLAB Command Codes	Function
Inread	Reads image from system graphics file to MATLAB environment	imshow	For displaying the image
Imhist	Displays the histogram of the image	imrotate	For image rotation
rgb2gray	Converts RGB image or color map to grayscale	im2bw	Converts image to binary image, based on threshold
Imadjust	For adjusting contrast of an image	imnoise	Adds noise to an image
Imfinfo	Displays information about graphics	imfilter	To reduce image noise
Imadd	For adding images	imcrop	For cropping image
imcomplement	To get the complement of an image	imdistline	To get the measurement of distance between an image
Flipdim	For flipping image	imwarp	For applying geometric transformation
Flipdim	For flipping image	imwarp	For applying geometric transformation

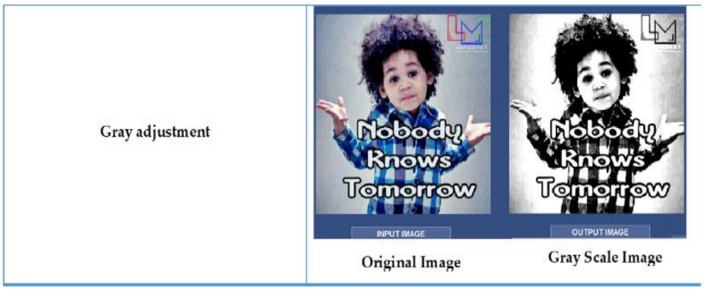


Figure 5. Effect of the grayscale image processing

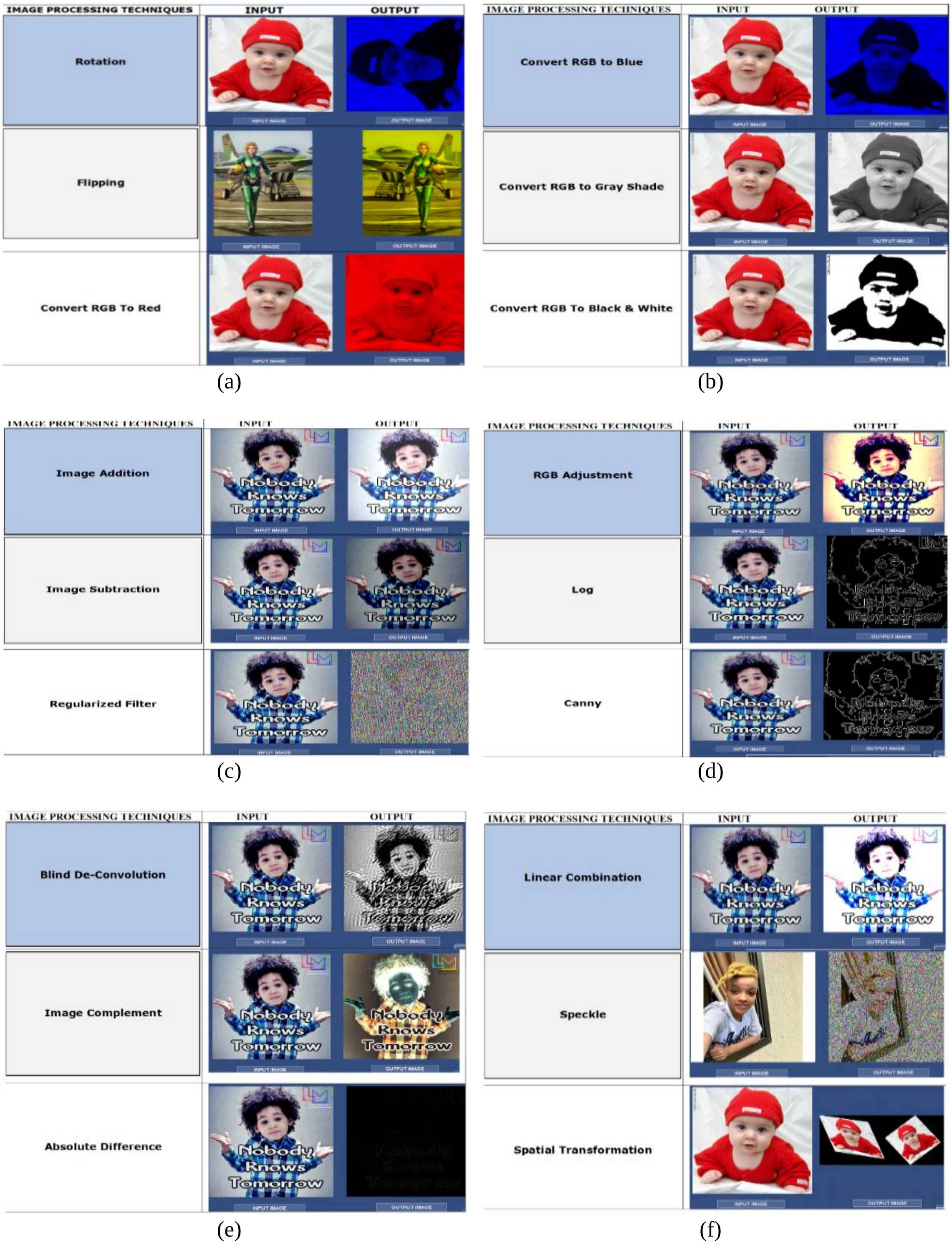
4.3. System implementation

The image processing software was created using the graphical user interface (GUI) present in MATLAB. Many programming command codes were used to create different push buttons to perform specific image processing operations and those push buttons were sub-sectioned into panels. The functions of each panel and push button are discussed as shown in:

- Axes: the GUI is used to create two axes and they are labelled as axes1 and axes2. The axes1 is meant to input image that will undergo processing by the user and the axes2 is for the output image.
- Input image: the push button reads image from the system and displays it in the axes1 in the image processing environment.
- Reset: it is used to restore pre-processed image on axes2 to the original image on axes1
- Exit: the exit button is used to close the image processing software window
- Rotation and flipping panel: This panel consists of pushbuttons for image rotation and flipping purposes in anti-clock wise, clockwise and 180 rotation button and horizontal, vertical and flip (horizontal + vertical) button.
- Color conversion panel: this colour conversion panel shows separate views the red, green and blue, and gray shade of the original image.
- Image arithmetic panel: This panel consists of sub pushbuttons which are addition, subtraction, complement of image, linear combination, and absolute difference. Image can only be converted into matrixes in order to perform some mathematical manipulation on them. In this project, the addition pushbutton was programmed to add the original image with a constant, the subtraction pushbutton was meant to subtract from the original image, while constant can multiply on the image by the multiplication pushbutton. Also, the complement button was coded to give the complement of the original image; the linear combination button was programmed to add the image with a factor of the original image and absolute difference button would reveal the difference between the original image and the gray scale of the original image.
- Deblurring panel: this panel contains the three different types of deblurring method pushbuttons which are blind deconvolution, Lucy-Richardson method, and Regularized method pushbuttons.
- Noise addition panel: It contains three types of noise pushbuttons namely speckle noise, salt and pepper noise, and Gaussian noise.
- Filtering panel: this panel contains three types of filtering method pushbuttons namely average filter, guide filer and mean filter. And each filtering pushbutton was coded to remove noise.
- Egde detection and contrast adjustment panel: Edge detection technique is applicable only to binary images, but the original image has to be converted to a binary image before it can undergo edge detection techniques. This panel consists of four different types of pushbuttons namely Sobel, Roberts, Log, and Prewitt. Also, RGB and Gray shade image adjustment pushbuttons are available in this panel for contrasting of the original image.
- Histogram: this button helps to show histogram of the original image. It shows the frequency of pixels intensity values. The x- & y- axes show the intensities of the gray level & their corresponding frequencies respectively.
- Crop image: this is used to select any particular portion of the original image that the user wants to crop.
- Blur edge: this pushbutton is for blurring the edges of the original image.
- Sharpness: it make the original image to have a clearer view (sharpness).
- Measure distance: this button was coded to perform measurement in pixels on the original image but the operation takes place on 2 axes.
- Geometric transformation: it helps to transform the original image.

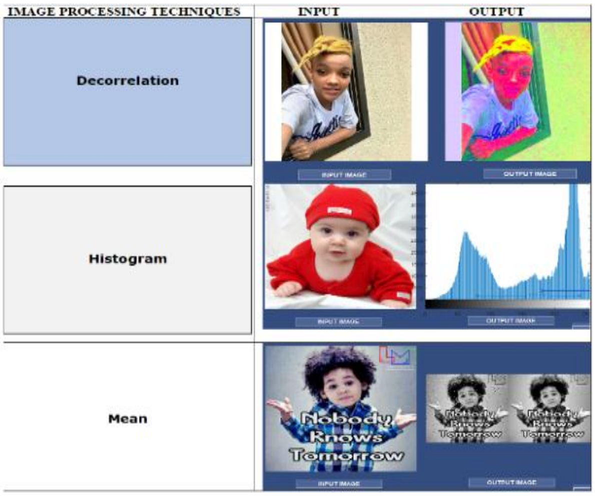
5. SIMULATION RESULTS

This section presents the performances of a MATLAB-based image processing technique using the various algorithms inherent in MATLAB. The results are the output of the processed images using the software developed. The results are from all the operations performed on the images using the image processing toolbox in MATLAB, while employing all the algorithms and necessary MATLAB command codes. Figure 6 shows the results of the various image processing operations using MATLAB.



(a) (b) (c) (d) (e) (f)

Figure 6. Results of image processing operations using MATLAB-based analytics (a)-(f)



(g)

Figure 6. Results of image processing operations using MATLAB-based analytics (*continue*)

6. CONCLUSION

The present study discussed a novel image processing technique using MATLAB-based analytics. It also presented an empirical-based model that uses image processing applications with features derived from DCT coefficients. The system was evaluated in MATLAB. We developed algorithms for color detection in images and provided updated MATLAB programming codes for state-of-the-art image processing operations. With different algorithms running on MATLAB toolbox, we provided extensive simulations of several images to evaluate the performance of the proposed approach. The results present the proposed technique as a veritable state-of-the-art approach for digital image processing application. This study aims at providing readers with different image processing applications running on MATLAB platform and to provide researchers with wider knowledge of MATLAB-based image processing techniques that can be exploited for several application specifics. With MATLAB, the various steps employed for image processing can be properly documented and replicated. And MATLAB-based image processing algorithms are more advanced compared to other state-of-the-art image processing applications.

REFERENCES

[1] S. K. Dewangan, "Importance & Applications of Digital Image Processing," *International Journal of Computer Science & Engineering Technology (IJCSET)*, vol. 7, no. 7, pp. 316-320, 2016.

[2] D. Zhang, F. Wang, R. Burgos, and D. Boroyevich, "Common mode circulating current control of interleaved three-phase two-level voltage-source converters with discontinuous space-vector modulation," *2009 IEEE Energy Conversion Congress and Exposition*, 2009, pp. 2801-2807, doi: 10.1109/ECCE.2009.5316313.

[3] R. S. Kavita, R. Bala, and S. Siwach, "Review paper on overview of image processing and image segmentation, " *International journal of Research in Computer applications and Robotics*, vol. 1, no. 7, 2013.

[4] H. Zhang, Z. Zhang, and Z. Pei, "Design and Implementation of Image Processing System Based on MATLAB," in *International Conference on Logistics Engineering, Management and Computer Science (LEMCS 2015)*, 2015, pp. 1356-1359, doi: 10.2991/lemcs-15.2015.270.

[5] R. Buksh, S. Routh, P. Mitra, S. Banik, A. Mallik, and S. Gupta, "MATLAB based Image Editing and Color Detection," *International Journal of Scientific and Research* Publications, vol. 4, no. 1, pp. 1-6, 2014.

[6] J. Nagi, S. K. Ahmed, and F. Nagi, "A MATLAB based face recognition system using image processing and neural networks," in *4th International Colloquium on Signal Processing and its Applications*, vol. 2, pp. 83-8, March 2008.

[7] T. S. Mohammed and A. S. Ibrahim, "Simulation and Analysis for Activities in Image Recognition Using MATLAB," *Computer science and information technology*, vol. 3, no. 1, pp. 22-30, 2015, doi: 10.13189/csit.2015.030104.

[8] I. Ahmad and K. Pothuganti, "Design & implementation of real time autonomous car by using image processing & IoT," *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2020, pp. 107-113, doi: 10.1109/ICSSIT48917.2020.9214125.

[9] L. Yang, "Medical Application of Digital Image Processing Based on MATLAB," *International Journal of Innovative Science, Engineering & Technology*, vol. 3, no. 3, pp. 148-155, 2016.

[10] E. Leavline, A. Antony, and G. Singh, "On Teaching Digital Image Processing with MATLAB," *American Journal of signal processing*, vol. 4, no. 1, pp. 7-15, 2014, doi: 10.5923/j.ajsp.20140401.02.

[11] M. S. Alkoffash, M. J. Bawaneh, H. Muaidi, S. Alqrainy, and M. Alzghool, "Survey of Digital Image Processing Techniques in Character Recognition," *International Journal of Computer Science and Network Security*, vol. 14, no. 3, pp. 65-71, 2014.

[12] P. S. Duth and M. M. Deepa, "Color Detection in RGB-modeled Images using MATLAB," *International Journal of Engineering & Technology*, vol. 7, no. 7, pp. 29-33, May 2018, doi: 10.14419/ijet.v7i2.31.13391.

[13] V. Goel, S. Singhal, T. Jain, and S. Kole, "Specific color detection in images using RGB modelling in MATLAB, " *International Journal of Computer Applications*, vol. 161, no. 8, pp. 38-42, March 2017, doi: 10.5120/ijca2017913254.

[14] S. Kaur and M. Sandhu, "Analysis of Various Image Segmentation Techniques Using MATLAB," *International Journal of Advanced Research in Computer Science and Software Engineering Research Paper*, vol. 6, no. 4, pp. 90-94, 2016.

[15] D. Sahu and C. Dewangan, "Identification and Classification of Mango Fruits Using Image Processing," *IJSRCSEIT: International Journal of Scientific Research in Computer Science, Engineering and Information* Technology, vol. 2, no. 2, pp. 203-210, 2016.

[16] A. Mishra, P. Asthana, and P. Khanna, "The Quality Identification of Fruits in Image Processing using MATLAB," *Department International Journal of Research in Engineering and Technology*, vol. 3, no. 10, pp. 92-95, 2014, doi: 10.15623/ijret.2014.0322019.

[17] G. Sharma, "Performance analysis of image processing algorithms using matlab for biomedical applications," *International Journal of Engineering and Manufacturing*, vol. 7, no. 3, pp. 8-19, May 2017, doi: 10.5815/ijem.2017.03.02.

[18] A. Abdullah, W. Palash, A. Rahman, K. Islam, and A. Alim, "Digital image processing analysis using Matlab, " *American Journal of Engineering Research (AJER)*, vol. 5, no. 12, pp. 2320-0936, 2016.

[19] A. Saxena, A. Samal, and D. Ray, "Vehicle Detection by Image Processing using MATLAB: A Colour based Approach," *International Journal of Industrial Electronics and Electrical Engineering*, vol. 6, no. 3, pp. 65-67, 2018.

[20] K. Senthilkumar, V. Ellappan, and A. R. Arun, "Traffic analysis and control using image processing," in *IOP conference series: materials science and engineering*, vol. 263, no. 4, p. 042047, 2017, doi: 10.1088/1757-899X/263/4/042047.

[21] A. Frank, Y. S. K. Al Aamri, and A. Zayegh, "IoT based Smart Traffic density Control using Image Processing," *2019 4th MEC International Conference on Big Data and Smart City (ICBDSC)*, 2019, pp. 1-4, doi: 10.1109/ICBDSC.2019.8645568.

[22] A. Utikar and V. Yerande, "Relative Study of Image Processing Filter with MATLAB, SCILAB, RASPBERRY Pi," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 4, no. 11, pp. 20181-20192, 2016.

[23] A. Devaraj, K. Rathan, S. Jaahnavi, and K. Indira, "Identification of Plant Disease using Image Processing Technique," *2019 International Conference on Communication and Signal Processing (ICCSP)*, 2019, pp. 0749-0753, doi: 10.1109/ICCSP.2019.8698056.

[24] M. K. Singh and S. Chetia, "Detection and classification of plant leaf diseases in image processing using MATLAB, " *International journal of life sciences Research*, vol. no. 4, pp. 120-124, 2017.

[25] G. Kshirsagar and A. N. Thakre, "Plant disease detection in image processing using MATLAB," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 6, no. 4, pp. 113-116, 2018.

[26] V. Roy, G. Mishra, R. Mannadiar, and S. Patil, "Fake Currency Detection Using Image Processing," *International Journal of Computer Science and Mobile Computing*, vol. 8, no. 4, 2019.

[27] A. S and D. M. Sasikumar, "Fake Currency Detection," 2019 International Conference on Recent Advances in *Energy-efficient Computing and Communication (ICRAECC)*, 2019, pp. 1-4, doi: 10.1109/ICRAECC43874.2019.8994968.

[28] A. Tukur, "Fingerprint recognition and matching using Matlab," *The International Journal of Engineering and Science (IJES)*, vol. 4, no. 12, pp. 01-06, 2015.

[29] R. Koshy and V. V. Nair, "Black Box for Accident Analysis Using MATLAB-Image Processing," *International Journal of Computer Applications Technology and Research*, vol. 6, no. 5, pp. 229-233, 2017, doi: 10.7753/IJCATR0605.1003.

[30] M. I. Ashqer and M. Bikdash, "Parking Lot Space Detection Based on Image Processing," *2019 SoutheastCon*, 2019, pp. 1-6, doi: 10.1109/SoutheastCon42311.2019.9020584.

[31] S. H. I. Putera and Z. Ibrahim, "Printed circuit board defect detection using mathematical morphology and MATLAB image processing tools," *2010 2nd International Conference on Education Technology and Computer*, 2010, pp. V5-359-V5-363, doi: 10.1109/ICETC.2010.5530052.

[32] D. Goswami, "Edge Detection Technology using Image processing in MATLAB," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 3, no. 5, pp. 3466-3471, 2015.

[33] G. A. D. Morais, E. Fujiwara, M. C. P. Soares, and G. Lucimara, "Development of an image processing MATLAB algorithm for cell growth analysis," *Revista dos Trabalhos de Iniciação Científica da UNICAMP*, no. 26, 2018, doi: 10.20396/revpibic262018557.

[34] H. Kakui, M. Yamazaki, N. B. Hamaya, and K. K. Shimizu, "Pollen grain counting using a cell counter," in *Pollen and Pollen Tube Biology*, Humana, New York, NY, vol. 2160, pp. 1-11, 2020, doi: 10.1007/978-1-0716-0672-8_1.

[35] Y. C. Samarawickrama and C. D. Wickramasinghe, "Matlab based automated surface defect detection system for ceremic tiles using image processing," *2017 6th National Conference on Technology and Management (NCTM)*, 2017, pp. 34-39, doi: 10.1109/NCTM.2017.7872824.

[36] A. Bansal, R. Bajpai, and J. P. Saini, "Simulation of Image Enhancement Techniques Using Matlab," *First Asia International Conference on Modelling & Simulation (AMS'07)*, 2007, pp. 296-301, doi: 10.1109/AMS.2007.92.

[37] G. A. Kumar and A. Kusagur, "Evaluation of image denoising techniques a performance perspective," *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, 2016, pp. 1836-1839, doi: 10.1109/SCOPES.2016.7955762.

[38] P. Srujana, J. Priyanka, V. Y. S. S. S. Patnaikuni, and N. Vejendla, "Edge Detection with different Parameters in Digital Image Processing using GUI," *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, 2021, pp. 795-802, doi: 10.1109/ICCMC51019.2021.9418327.

[39] T. Sahoo and S. Pine, "Design and simulation of various edge detection techniques using Matlab Simulink," *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, 2016, pp. 1224-1228, doi: 10.1109/SCOPES.2016.7955636.

[40] M. W. Khan, "A survey: Image segmentation techniques," *International Journal of Future Computer and Communication*, vol. 3, no. 2, p. 89, 2014.

[41] A. Abdulrahman and S. Varol, "A Review of Image Segmentation Using MATLAB Environment," *2020 8th International Symposium on Digital Forensics and Security (ISDFS)*, 2020, pp. 1-5, doi: 10.1109/ISDFS49300.2020.9116191.

[42] T. Khan and A. H. Pathan, "Hand Gesture Recognition based on Digital Image Processing using MATLAB," *International Journal of Scientific & Engineering Research*, vol. 6, no. 9, pp. 338-346, 2015.