Proceedings of Science and Mathematics
Faculty of Science, Universiti Teknologi Malaysia

Vol. 5, 2021, page 140-150

Applications of Linear Algebra in Digital Image Processing

[1]Nurul Amira Suhaimi and [2]Fong Wan Heng

[1,2]Department of Mathematical Sciences
Faculty of Science, Universiti Teknologi Malaysia,
81310 Johor Bahru, Johor, Malaysia.

e-mail: [1]nurulamirasuhaimi@graduate.utm.my, [2]fwh@utm.my

Abstract Concepts in linear algebra play a vital role in the applications of Digital Image Processing in reaching the goal of computer visualizations. The purpose of this study is to apply some basic linear algebra concepts as the foundation of image processing, focusing on matrix operations and geometric transformations. This research presents some examples on the concept of matrix operations, geometric transformations and filter on images which are used to develop the algorithm of matrices in image processing as a representation of the images, which are later computed using MATLAB. The results shows that it follows some of the properties of matrix operations.

Keywords linear algebra; digital image processing; MATLAB; matrix operation; geometric transformation; image manipulation.

1 Introduction

Linear algebra plays a vital role in computer visualization [1]. It was stated in [2] that one of a ways to improve the capability of image processing is by applying the tools of linear algebra such as matrix, least square, single value decomposition, homogeneous coordinates, eigenfaces, and many more [1]. Thus, by using linear algebra computations, we can extract valuable information on images to be manipulated [1]. The purpose of this project is to apply some basic linear algebra concepts as the foundation of image processing, focusing on matrix operations and geometric transformations.

Matrix operations can be used to eliminate the background of images or identify parts of images, enhance the brightness of images and add filters on images; whereas geometric transformations can align different images together. Since there is a relation between matrices and digital images [2], some basic linear algebra concepts can be explored from the applications of Digital Image Processing (DIP).

2 Literature Review

In this chapter, some applications of linear algebra, historical background of image processing, and concepts of Digital Image Processing are discussed.

2.1 Some Applications of Linear Algebra

In this section, some applications of linear algebra from previous researches are discussed.

Linear algebra for computer vision was studied by Garg and Kaundal [1]. They stated that some basic concepts in linear algebra like matrix, least square, single value decomposition and many more were helpful in computer vision. Moreover, some of the applications of linear algebra in computer vision are camera modelling, computer gaming, image processing, convolution, pose estimation and structure-from-motion.

According to Halim [3], linear algebra was applicable on a variety of fields such as physics, engineering and computer science since machine learning is widely used in these fields which needed the algorithms, data analysis and categorizations. In his paper, methods where machine learning algorithms use mathematics, especially matrix and vector, are described.

2.2 Historical Background of Image Processing

Digital Image Processing refers to the processing of a digital image by means of a digital computer, and the study of algorithms for its transformations [4]. We can utilize a number of mathematical techniques on Digital Image Processing since the data of a digital image is in the matrix form. Moreover, Digital Image Processing encompasses a wide and varied field of application, such as area of image operation and compression, computer vision, and image analysis which are also called as image understanding [4]. Computational methods in linear algebra have become an important aspect of the subject since Digital Image Processing algorithms can be written in terms of matrix equations [4].

2.3 Some Concepts of Digital Image Processing

In this section, some concepts of Digital Image Processing from previous researches are discussed.

Matrix operations, geometric transformations, edge detection and many other mathematical operations can be used to manipulate images by applying those operations to the matrix that is associated to each image as stated by Joy [5]. In the paper, the demonstrations of a binary, greyscale and colour image were explained. Not only that, matrix operation, geometric transformation, edge detection and some image processing techniques used on images were also described in the paper.

Image filter has been extensively used in a lot of editing software and even in applications that use camera to capture or record images. In [6], Torres explained the retrieval techniques and algorithms used on image filter. In order to achieve the filter retrieval, he analysed the properties of the image. As a result, he revealed that taking the nearest images to the original from the database will benefit more in filter retrieval.

3 Matrices and Digital Images

3.1 The Relation between Matrices and Digital Images

The images that can be seen on television, mobile phone, and also laptop are some of the instances of digital images in our daily lives. Discrete images that contain a sample of two-dimensional function which is $f(x, y)$ on a regular grid can be called as a digital image [5]. In the function, every single pair of coordinates $(x, y) \epsilon \mathbb{Z}^2$ presented by one matrix element is known as a pixel [5]. Thus, we can say that there is a relation between matrices and digital images since the digital image is presented by a pixel matrix [2]. The digitized image can be represented as Equation (1):

$$f(x,y) = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1n} \\ f_{21} & f_{22} & \cdots & f_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ f_{m1} & f_{m2} & \cdots & f_{mn} \end{bmatrix}, \tag{1}$$

where *m* is the number of rows and *n* is the number of columns. The function f_{ij} where $i, j \in \mathbb{Z}$, corresponds to the coordinate (i, j).

3.2 Types of Digital Images

There are four types of digital image categories in image processing which are greyscale image, binary image, colour image and indexed image.

Most people thought of greyscale image as an image that only contains black and white colour. However, a greyscale image in fact contains various shades of grey [7] where the brightness of the corresponding pixels are determined by every single element of the matrix [8]. In addition, a greyscale image can also be called as a monochrome image. The elements of the matrix consist of integer numbers in the range starting from 0 which represents the black colour as the darkest pixel, until 255 which represents the white colour as the brightest pixel. Furthermore, digital images can be called as binary images or Boolean images where each pixel only uses two colours; either 0 that represents black colour, or 1 that represents white colour [8].

Moreover, colour images are used using true colour, or we called it as RGB colour model which contains three main colour components: R for red, G for green, and B for blue. A colour can be stored quantitatively by assigning a value between 0 and 255. We know that any colour can be described by a 3-vector. An image is made up of *m* x *n* pixels, and each pixel has its own colour and vector. Therefore, the image can actually be stored in an $m \times n \times 3$ matrix, or an $m \times n$ matrix where each entry contains a 3-vector. The colour image actually contains three functions that are pasted together which determine the intensity of the pixel. We call this function as a 'vector-valued' function as stated in Equation (2):

$$f(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix}, \tag{2}$$

where the function $f(x, y)$ is made up of three different vector-valued function: $r(x, y)$ for red value function, $g(x, y)$ for green value function and $b(x, y)$ for blue value function.

Additionally, an indexed image is defined as an image that consists of a matrix with its own colour map which can be called as a colour palette [9]. This colour map is actually a list of all colours used in the image itself. The differences between RGB image and indexed image is that, for indexed image, each pixel does not give its colour but has its own value called index to the colour in the map.

3.3 Concepts of Image Processing

In [10], Caridade stated that there is a relation between linear algebra and image processing. In this subsection, matrix operations and geometric transformations in image processing are discussed.

3.3.1 Matrix Operations

In linear algebra, matrix operations is the first concept that students get exposed to [5]. The matrices contain each element of the pixel value with the same dimension of rows and columns, (m, n). Tanimoto [11] stated that addition of the image matrices can enlighten an image, while subtraction of it can be used to darken an image; multiplication can boost the contrast of the image, while division lowers the contrast and darkens it at the same time. Firstly, let there be two matrices for images A and B which are represented as the following matrices in Equation (3) and Equation (4) respectively:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \tag{3}$$

$$B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{bmatrix}. \tag{4}$$

Definition 1 [5] : Matrix addition is defined as the combination of two matrices A and $B,$ that represents an image each with the same dimension of rows *m* and columns *n*. The matrix obtained is shown in Equation (5) below:

$$A + B = \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} & \cdots & a_{1n} + b_{1n} \\ a_{21} + b_{21} & a_{22} + b_{22} & \cdots & a_{2n} + b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} + b_{m1} & a_{m2} + b_{m2} & \cdots & a_{mn} + b_{mn} \end{bmatrix}. \tag{5}$$

Below are the properties of matrix addition that are stated without proof [12]:

(a) Commutative law: $A + B = B + A,$

(b) Associative law: $A + (B + C) = (A + B) + C.$

Definition 2 [5] : Matrix subtraction is defined as the deduction of two matrices A and $B,$ that represents an image each with the same dimension of rows *m* and columns *n*. The matrix obtained is shown in Equation (6) below:

$$A - B = \begin{bmatrix} a_{11} - b_{11} & a_{12} - b_{12} & \cdots & a_{1n} - b_{1n} \\ a_{21} - b_{21} & a_{22} - b_{22} & \cdots & a_{2n} - b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} - b_{m1} & a_{m2} - b_{m2} & \cdots & a_{mn} - b_{mn} \end{bmatrix}. \tag{6}$$

Definition 3 [5] : Matrix multiplication is the product of matrix *A* and *B*, where *A* represents an $m \times p$ matrix and *B* represents a $p \times n$ matrix. The matrix obtained is shown in Equation (7) below:

$$A \times B = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix}, \tag{7}$$

where $c_{ij} = a_{ij}b_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{ip}b_{pj} = \sum_{k=1}^{p} a_{ik}b_{kj}$ when $i = 1, 2, 3, \dots, m$ and $j = 1, 2, 3, \dots, n.$

Below are some of the properties of matrix multiplication that are stated without proof [12]:

(a) Associative law: $A(BC) = (AB)C,$

(b) Distributive law: $A(B + C) = AB + AC.$

3.3.2 Geometric Transformations

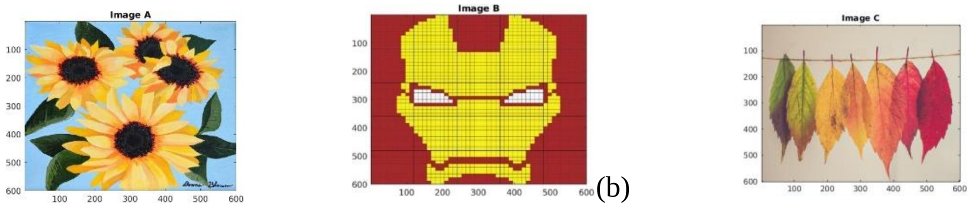
Geometric transformations of matrices can be applied to make some movements such as translation, rotation and scaling of an image [5].

To begin with, let there be an arbitrary point $p = (x_1, y_1)$ as a pixel that we want to scale in order to perform the transformation. Based on [5], there are three steps on scaling an image to a certain arbitrary point. Firstly, by applying a translation $T = (-x_1, -y_1)$ to all the pixel values, we can move the pixel $p = (x_1, y_1)$ to the origin of the coordinate system. After that, by applying a scaling $S = (s_x, s_y)$ to all the pixel values, we can change the dimensions of the image. Lastly, a back translation $T = (x_1, y_1)$ is applied to all the pixel values in order to move the pixel $p = (x_1, y_1)$ into its original position. This transformation can be called as a composite transformation, T_c and the matrix of the transformation obtained is shown in Equation (8) below:

$$\begin{aligned} T_c &= T(x_1, y_1) \times S(s_x, s_y) \times T(-x_1, -y_1) \\ &= \begin{bmatrix} 1 & 0 & x_1 \\ 0 & 1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -y_1 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} s_x & 0 & x_1(1 - s_x) \\ 0 & s_y & y_1(1 - s_y) \\ 0 & 0 & 1 \end{bmatrix}. \end{aligned} \tag{8}$$

4 Results and Discussion

In this section, some concepts of matrix operations and geometric transformations are used to develop the algorithm of matrices in image processing as a representation of the images, and to compute digital images in MATLAB programming. The equations and properties listed before are used for this purpose. For example, three images, *A*, *B*, and *C* in Figure 1 are used to perform the operations in order to show how images are manipulated using matrix operations and geometric transformations.



(a) Image *A* (b) Image *B* (c) Image *C*

Figure 1 : Images used to perform the operations: (a) Image *A* (b) Image *B* (c) Image *C*.

4.1 Matrix Operations

In this subsection, three types of matrix operations are used to compute the digital images in MATLAB programming which are matrix addition, matrix subtraction and matrix multiplication.

The first operation is the matrix addition. For the commutative law, $A + B = B + A$, let image *D* be an addition of matrix from image *A* to image *B*, and image *E* be an addition of matrix from image *B* to image *A*. Based on the properties of matrix addition stated before, the commutative law can be applied using the algorithm: `D=image(A+B)` and `E=image(B+A)`. Based on Figure 2, the commutative law is satisfied since the outputs of the images for both $A + B$ and $B + A$, that are images D and E respectively, are equal when the matrix addition is performed.

Image A+B
Image B+A
100
200
300
400
500
600

(a) Image $A + B$ (b) Image $B + A$

Figure 2 : Commutative law image output: (a) Image $A + B$ (b) Image $B + A$.

For the associative law, $A + (B + C) = (A + B) + C$, let image *F* be an addition of matrix from image *A* to images *B* and *C*, while image *G* an addition of matrix from images *A* and *B* to image *C*. Based on the properties of matrix addition stated before, the associative law can be applied using the algorithm: `F=image(A+(B+C))` and `G=image((A+B)+C)`. Based on Figure 3, the associative law is satisfied since the outputs of the images for both $A + (B + C)$ and $(A + B) + C$, that are images *F* and *G* respectively, are equal when the matrix addition is performed.

Image A+(B+C)
Image (A+B)+C
100
200
300
400
500
600

(a) Image $A + (B + C)$ (b) Image $(A + B) + C$

Figure 3 : Associative law image output: (a) Image $A + (B + C)$ (b) Image $(A + B) + C$.

The second operation is the matrix subtraction. Let image *H* be a deduction of matrix of image *B* from image *A*, while image *J* a deduction of matrix of image *A* from image *B*. Based on the properties of matrix subtraction, $A - B \neq B - A$ can be shown using the algorithm: `H=image(A-B)`and `J=image(B-A).` Based on Figure 4, the properties of matrix subtraction is satisfied since the outputs of the images for both $A - B$ and $B - A$, that are images H and J respectively, are not equal when the operation is performed.

Image A-B
Image B-A
100
200
300
400
500
600

(a) Image $A - B$ (b) Image $B - A$

Figure 4 : Matrix subtraction image output: (a) Image $A - B$ (b) Image $B - A$.

The third operation is the matrix multiplication. The same three images *A*, *B*, and *C* in Figure 4.1 are used to perform the matrix multiplication operations. For the associative law $A(BC) = (AB)C$, let image *M* be a multiplication of matrix from image *A* to image *BC* such that $A(BC)$, while image *N* a multiplication of matrix from image *AB* to image *C* such that $(AB)C$.

Based on the properties of matrix multiplication stated before, the associative law can be applied using `immultiply` function. Based on Figure 5, the associative law is satisfied since the outputs of the images for both $A(BC)$ and $(AB)C$, that are images M and N respectively, are equal when the matrix multiplication is performed.

Image Ax(BxC)
Image (AxB)xC
100
200
300
400
500
600

(a) Image $A(BC)$ (b) Image $(AB)C$

Figure 5 : Associative law image output: (a) Image *A*(*BC*) (b) Image (*AB*)*C*.

For the distributive law, $A(B + C) = AB + AC$, let image P be a linear combination of multiplication of matrix from image *A* with image (*B*+*C*) such that *A*(*B*+*C*), while image Q a linear combination of an addition of matrix from image *AB* to image *AC* such that *AB*+*AC*. Based on the properties of matrix multiplication stated before, the distributive law can be applied using `immultiply` function. Based on Figure 6, the distributive law is satisfied since the outputs of the images for both *A*(*B*+*C*) and *AB*+*AC*, that are images *P* and *Q* respectively, are equal when the matrix multiplication is performed.

Image A(B+C)
Image AB+AC
100
200
300
400
500
600

(a) Image $A(B + C)$ (b) Image $AB + AC$

Figure 6 : Distributive law image output: (a) Image *A*(*B*+*C*) (b) Image *AB*+*AC*.

4.2 Geometric Transformations

In this section, three types of geometric operations are used to compute the digital images in MATLAB programming which are translation of an image, rotation of an image, and scaling of an image.

The first type of geometric transformation is the translation of an image using `imtranslate`(*X*, [*p*, *q*]), where *X* is the image to be translated, *p* is the number of rows for the image to be shifted, and *q* is the number of columns for the image to be shifted. The specified values of *p* and *q* to be shifted need to be positive for the image to appear within the original image boundaries. If the specified values of *p* and *q* are negative or more than the range of the image boundaries, the image will be cropped out from the boundaries. Based on Figure 7, the image are shifted in both directions when the values of [*p*, *q*] are specified as [100, 100], [−100, −100] and [100, −100].

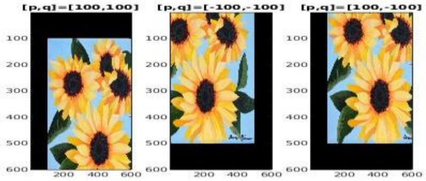


Figure 7: The output of image *A* when [*p*, *q*] = [100, 100], [−100, −100] and [100, −100].

The second type of geometric transformation is the rotation of an image. In order to rotate an image, the function `imrotate` is used by specifying the desired rotation angle θ in degrees. The value specified for the rotation angle θ determines either the rotation is clockwise or counter clockwise. If the specified angle θ is positive, the image rotates in counter clockwise direction. On the other hand, if the specified angle θ is negative, the image rotates in clockwise direction. Figure 8 shows the output of image *A* when $\theta = 50°$, 100°, −50° and −100°.

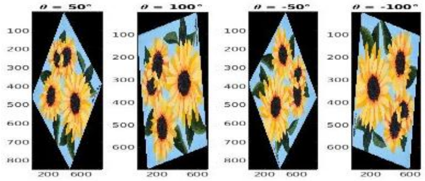


Figure 8 : The output of image *A* when $\theta = 50°$, 100°, −50° and −100°.

The third type of geometric transformation is the scaling of an image. In order to shrink or magnify an image, the function `imresize` is used to resize the image. There are two ways to use this function which are `imresize`(*X*, *k*) and `imresize`(*X*, [*m*, *n*]), where *X* is the image to be resized, *k* is the constant, *m* is the scale for the number of rows for the image, and *n* is the scale for the numbers of columns for the image. Figure 9 shows the output of image *A* using `imresize`*(X, k)* and Figure 10 shows the output of image *A* using `imresize`(*X*, [*m*, *n*]).

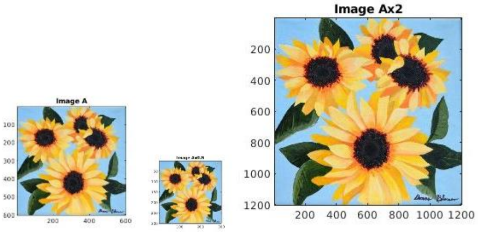


Figure 9 : The output of image *A* using `imresize`*(X, k)* when *k* = 1, 0.5, and 2.

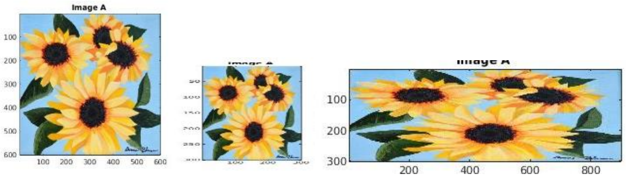


Figure 10 : The output of image *A* using `imresize`(*X*, [*m*, *n*]) when [*m*, *n*] = [600, 600], [300, 300], and [300, 900].

4.3 Filter of Images

In this subsection, four types of filter images are presented which are brightness filter of an image, RGB filter of an image, greyscale filter of an image, and sepia filter of an image.

The brightness of an image can be applied using the function `immultiply`(X, k), where X is the image to be manipulated and k is the scalar value. Based on Figure 11, we can see that when image A is multiplied with all three scalars of k = 0.5, 1.5 and 2.5, the brightness of the image varies. The larger the value of k, the brighter the image is.

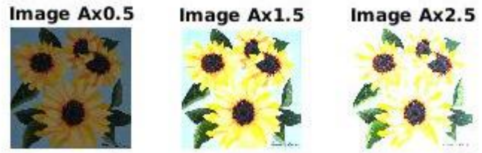


Figure 11 : The output of image A when k = 0.5, 1.5 and 2.5

RGB filter of an image is used to create a new image by separating the red, green and blue components from the actual image since an image is a combination of the three colours function. By letting a certain function equal to zero, a filter of a specific colour out of the three colours can be created. The output for RGB filter of image A is shown in Figure 12.

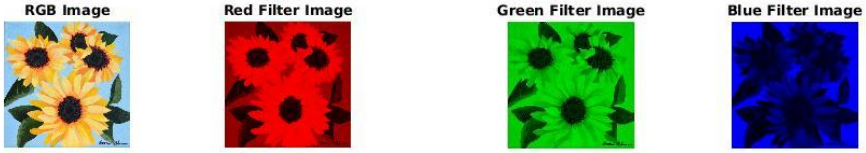


Figure 12 : RGB filter of image A.

The greyscale filter of an image helps to create a new image by converting the RGB image or colour map it into greyscale using the `rgb2gray` function. This function helps to keep the luminance while removing the saturation and hue information from the image. The output of the greyscale filter for image A is shown in Figure 13.

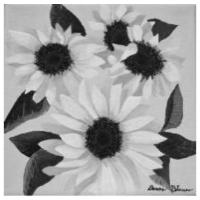


Figure 13 : Greyscale filter on image A.

Sepia filter of an image helps to create a new brownish vintage colour image based on specified values of the red component, blue component and green component. Based on Microsoft, the sepia filter is given as a function of $rf(x,y) = \begin{bmatrix} rr(x,y) \\ rg(x,y) \\ rb(x,y) \end{bmatrix} = \begin{bmatrix} 0.393r + 0.769g + 0.189b \\ 0.349r + 0.686g + 0.168b \\ 0.272r + 0.534g + 0.131b \end{bmatrix}$, where the function $rf(x,y)$ is made up of three different vector-valued function: $rr(x,y)$ for red value function, $rg(x,y)$ for green value function and $rb(x,y)$ for blue value function. The output of the sepia filter on image A is shown in Figure 14.

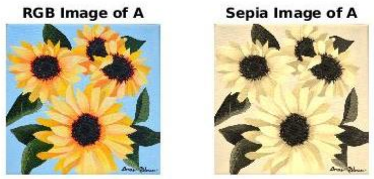


Figure 14 : The output for RGB image and sepia filter on image *A*.

5 Conclusion

The relation between matrices and digital images are presented alongside with four types of digital images which are the greyscale images, binary images, colour images and indexed images. The concept of manipulating images are studied by applying mathematical operations to the matrices that are bounded in each image. This research also shows the result of some examples on the concept of matrix operations for matrix addition, matrix subtraction and matrix multiplication, geometric transformations for translation of image, rotation of image and scaling of image, and lastly, filters on images such as brightness filter of an image, RGB filter of an image, greyscale filter of an image and sepia filter of an image, which are used to develop the algorithm of matrices in image processing as a representation of the images, and are later computed using MATLAB programming.

References

[1] Garg, A., and Kaundal, K. A Study of Linear Algebra for Computer Vision. *International Journal of Innovative Research in Computer and Communication Engineering*. 2017. 3: 4169-4176.

[2] Vuckovic, V. Image and Its Matrix, Matrix and Its Image. *Faculty of Mathematics, Belgrade, Review NCD.* 2008. 12: 17-31.

[3] Halim, S. Application of Linear Algebra in Machine Learning. *Faculty of Computer Science, Balkh University, Balkh, Afghanistan, IRJET.* 2020. 7(2): 3188-3197.

[4] Cao, L. Singular Value Decomposition Applied to Digital Image Processing. *Division of Computing Studies, Arizona State University Polytechnic Campus, Mesa, Arizona State University Polytechnic Campus*. 2006. 1-15.

[5] Joy, D. Image Processing through Linear Algebra. *Department of Mathematics, College of Engineering Trivandrum, Infokara Research.* 2020. 9(1): 742-753.

[6] Torres, P., Colton, S., and Ruger, S. Experiments in Example-based Image Filter Retrieval. *Department of Computing, Imperial College, London, UK and Knowledge Media Institute, the Open University, UK.* 2012.

[7] Bakri, N., Ibrahim, R., Awang, T. S., and Zin, Z. M. Linking Mathematics and Image Processing Through Common Terminologies. *Procedia - Social and Behavioural Sciences.* 2013. 102: 454-463.

[8] Pesco, D. U. and Bortolossi, H. J. *Matrices and Digital Images*. Institute of Mathematics and Statistics, Fluminense Federal University. 2012.

[9] Tyagi, V. *Understanding Digital Image Processing.* Department of Computer Science and Engineering, Jaypee University of Engineering and Technology, India: CRC Press. 2020.

[10] Caridade, C.M. R. Applications (Ideas) in Linear Algebra with Digital Image Processing. Can We Do, Teach, Motivate and Evaluate? *Journal of Information Systems Engineering and Management,* 2019. 4(4): em0103.

[11] Tanimoto, S. L. Exploring Mathematics with Image Processing. *IFIP World Conferenceon Computers in Education VI,* USA: Springer Science and Business Media Dordrecht. 1995. 806-814.

[12] Dianat, S. A. and Saber, E. S. *Advanced Linear Algebra for Engineers With MATLAB.* New York, USA: CRC Press. 2017. 1-40.