Comparison between Open CV and MATLAB Performance in Real Time Applications

Ammar Sameer Anaz ammar3303@gmail.com
Diyaa Mehadi Faris d.mfaris@yahoo.com
Department of Computer Engineering
Technical College / Mosul

Abstract:

The most important operation in the image processing field is detect the objects inside an image or video, which it is used in many applications, especially in real time systems. Real time image processing in modern systems demands fast technologies, and there are variety methods to achieve this goal. This paper compare between the performance of (MATLAB) and (Open CV) to detect circular shapes that have specific color (blue) from live video in a real time. The input is a live video with resolution (640*480) pixels, taken by a web camera, and then processed by a sequence of image processing operations. The results refer to a program written in (Open CV) speed up the system up to 174% more than the same program written in (MATLAB) to do the same job.

Keywords: Image processing, Open CV, MATLAB

مقارنة بين أداء ألـ (Open CV) وألـ (MATLAB) في تطبيقات الزمن الحقيقي

عمار سمير عناز **ضياء مهدي فارس**
قسم هندسة الحاسوب
الكلية التقنية / الموصل

الخلاصة:

أحد أهم عمليات المعالجة الصورية هو الكشف عن الأشكال داخل صورة أو فيديو، والتي تستخدم في كثير من التطبيقات وخصوصا في معالجات الزمن الحقيقي. النظم الحديثة من معالجات الزمن الحقيقي تتطلب تكنلوجيا عالية السرعة، وكما هو معلوم هنالك طرق متنوعة لتحقيق هذا الهدف. هذا البحث يقارن بين أداء ألـ (MATLAB) وألـ (Open CV) لإيجاد الاشكال الدائرية التي لها لون معين (أزرق) داخل فديو مباشر في الزمن حقيقي، الادخال هو فيديو مباشر ملون بدقة (640*480) نقطة، مأخوذ من كامرة، ثم معالجته خلال سلسلة متتابعة من عمليات المعالجة الصورية. لقد أشارت النتائج إلى أن البرنامج المكتوب بالـ (Open CV) قد زاد من سرعة النظام بنسبة (174%) مقارنة بنفس البرنامج المكتوب بالـ (MATLAB) لأداء ذات العمل.

Received: 13 – 5 - 2014 **Accepted: 15 – 8 - 2015**

1. INTRODUCTION

Nowadays, images and videos are all over the world, and the demand for high-speed and convenient real time processing for these images and videos is increased, especially after the development in digital communications and digital images devices. An efficient real time system is needed to extract more information from images and videos.

Matuska *et al*. in 2012 used images with various resolution, to calculate CPU Time consumption for image processing algorithm in MATLAB and Open CV. The results of their work show that Open CV is faster than MATLAB in some algorithm from 4 to 30 times and in some case up to 100 times [1].

Sharmila *et al*. in 2015 propose image conversion function, and test it on various images for data conversion and contrast stretching by using Open CV and MATLAB. The final calculation reveals the MATLAB code consumes more time for the same conversion. Moreover, the total time taken for the conversion in Open CV falls less than three seconds [2].

In this paper, the input frame from the camera is segmented to pick up the target color (blue), after that smooth the resultant image to reduce noise, and at the end circles detect. The rest of the paper is structured as follows: Section 2 discusses image processing steps, starting by frame reading, segmentation, smoothing, and ended by circular Hough transforms. Section 3 introduces Steps of work, in Section 4 the results and calculation of the comparison between MATLAB and OpenCV approaches. The conclusions are introduced in Section 5. Section 6 indicates References. At the time of writing, the latest release is Open CV v.2.4.7 and MATLAB R2013a (8.1.0.604) where available and used in this paper.

2. IMAGE PROCESSING

The block diagram in Figure (1) describes the sequence of image processing system, which is used to process live video from web camera. First, the system read a frame form input device (web camera). Then separate a specific color by using histogram segmentation and multi-level thresholding. After that, smooth the resultant frame by Gaussian filter. Finally, apply a circular Hough transform function to detect circular object inside the frame.

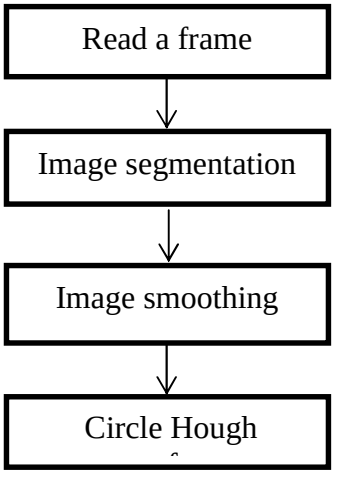


Figure (1): The block diagram of image processing system

2.1. Read a Frame

The first step in any system that interact with real time video processing is reading the frame from an input device, the decrease in reading time leads to increase system efficiency. This paper is started by calculating the time conception in reading the frame from input device. Then made a comparison to find out, which is the best programing language (MATLAB or Open CV) to do that. The camera specifications that used as input device are listed in table (1).

Table (1): The camera specifications

Frame rate	30 fps (frame per second)
Video format	Mp4
Resolution	640*480 pixel

2.2. Image Segmentation

Image segmentation is the separation of an image into categories or regions, which belong to different objects or parts of items. Every pixel in an image is classified to one of a number of these categories, a segmentation should meet the following requirements:

1. Pixels in the same category have analogous grayscale of multivariate values and form a connected region.
2. Adjacent pixels, which have different categories, have different values.

Segmentation often is the significant step in image analysis, there is something should be taken in consideration, each pixel as a unit of observation to working with objects (or parts of objects) in the image, composed of numerous pixels. If segmentation is done fine then all other stages in image analysis are made simpler [5].

2.2.1. Color image segmentation by multi-level thresholding

Global histogram of a digital image is a popular tool for real- time image processing due to its simplicity in implementation. It serves as an important basis of statistical approaches in image processing by producing the global description of the image's information. In color image segmentation, color of a pixel is given as three values corresponding to the three component images red, green and blue (RGB). Component histograms provide additional information about the intensity distribution within the individual color channels. When computing component histograms, each color channel is considered a separate intensity image and each histogram is computed independently of the other channels [6].

Thresholding is the simplest and most generally used way for segmentation, specified a threshold value (T), or two values (T_1 and T_2) for multi-level thresholding. The pixel located at pattern position (i,j), thresholding creates a binary image b(i,j) from an intensity image I(i,j), The multi-level thresholding is described in the following equation [3] [7].

$$b(i,j) = \begin{cases} 1 & if \ \ T_2 \geq I(i,j) \geq T_1 \\ 0 & otherwise \end{cases} \qquad \ldots\ldots\ldots\ldots\ldots (1)$$

In this paper, histogram segmentation by multi-level threshold is applied directly to each component of a color space, and then the results is combined to obtain a final segmentation result. The threshold values of each color space, which is used in this paper, are shown in table (2).

Table (2): The thresholds values

	T_1	T_2
Red	0	70
Green	70	170
Blue	165	255

2.3. Image Smoothing

Smoothing, also called blurring, is a simple and normally used image processing operation. There are many reasons for smoothing, but it is typically done to reduce noise or camera artifacts. Smoothing is also important to reduce the resolution of an image in a principled way. In practice, some different types of noise usually companied in image. Noise is a pixel that vary from its actual value in an image. Noise happen in digital image, while transmit image on computer networks, or pixel value in image obtained by image input devices that does not express the actual intensity from real scene produces noise [8].

Therefore, a respectable image smoothing algorithm should be able to deal with different types of noise. However, blur and offsets of the edges regularly caused by image smoothing.

In image analysis and interpretation the edge information is much imperative. So, the precision of edge's position in image should be considered in smoothing process. [9].

2.3.1. Gaussian smoothing filter.

This type of smoothing used to 'blur' images to remove noise by a two dimensions convolution operation, as a 'point-spread' function. The need to produce a discrete approximation to the Gaussian function before convolution can performed, because of the image is stored as a collection of separate pixels. The Gaussian smoothing can be applied using standard convolution methods, the function of Gaussian smoothing showed below [8] [10]:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \text{..... (2)}$$

Where **x** and **y** are the spatial dimensions, while **σ** is the standard deviation of the Gaussian distribution**.** In this paper, the Gaussian filter is used as a smoothing filter, with values; **x** =9, **y** = 9, and **σ** = 0.1.

2.4. Circular Hough Transforms

One of the most frequently used algorithms to recognize different shapes in an image is Hough Transform. Hough Transform was invented by Paul Hough in 1962 and patented by IBM. Later then in 1972 Hough transform was modified, which is used commonly in the present day under the name Generalized Hough Transform. An extended form of General Hough Transform, used to identify circles is Circular Hough Transform. A flow-chart of circular Hough transform is shown in Figure (2). The edge detected from the Canny edge detector forms the input to extract the circle using the circular Hough transform [11].

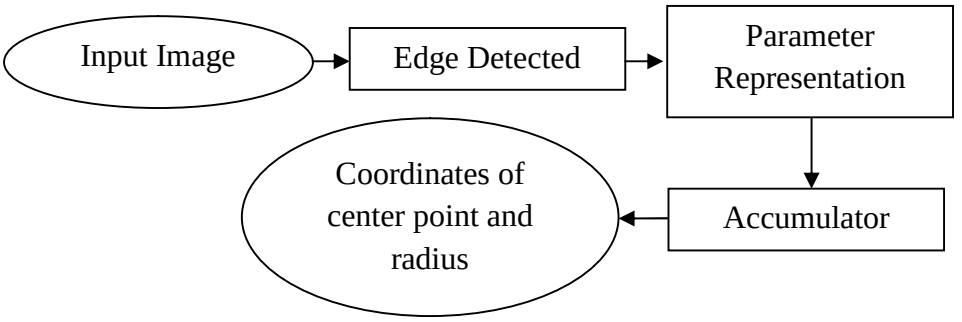


Figure (2): Flow-chart of circular Hough Transform

2.4.1. Parameter representation:

The equation of the circle is:

$$r^2 = (x-a)^2 + (y-b)^2 \quad \text{..... (3)}$$

As can be seen there are three parameters in the equation: a, b and r. where: a, and b are the center of the circle in the x & y coordinates respectively and r is the radius. The factor representation of the circle is [12]:

$$x = a + r * \cos(\theta) \quad \text{..... (4)}$$

$$y = b + r * \sin(\theta) \quad \text{..... (5)}$$

2.4.2. Accumulator:

As shown in Figure (3) the black circles represents a set of edge points within the image. Each edge point contributes a circle of radius to an output accumulator space indicated by the dotted circles. The output accumulator space has a peak where these contributed circles with desired radius overlap at the center of the original circle [13].

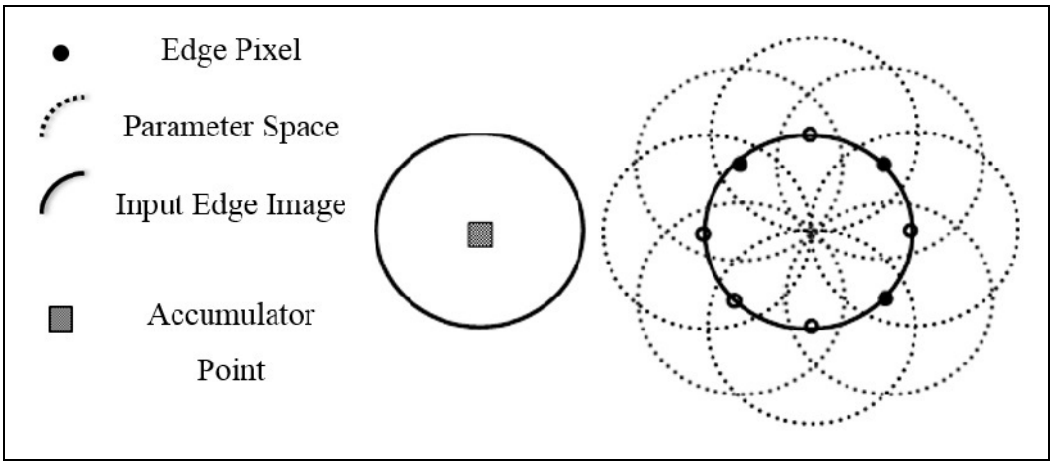


Figure (3): Illustration of circular Hough transform

3. STEPS OF WORK

The flow chart in Figure (4) describe the processing sequence and Figure (5) show the output from each step.

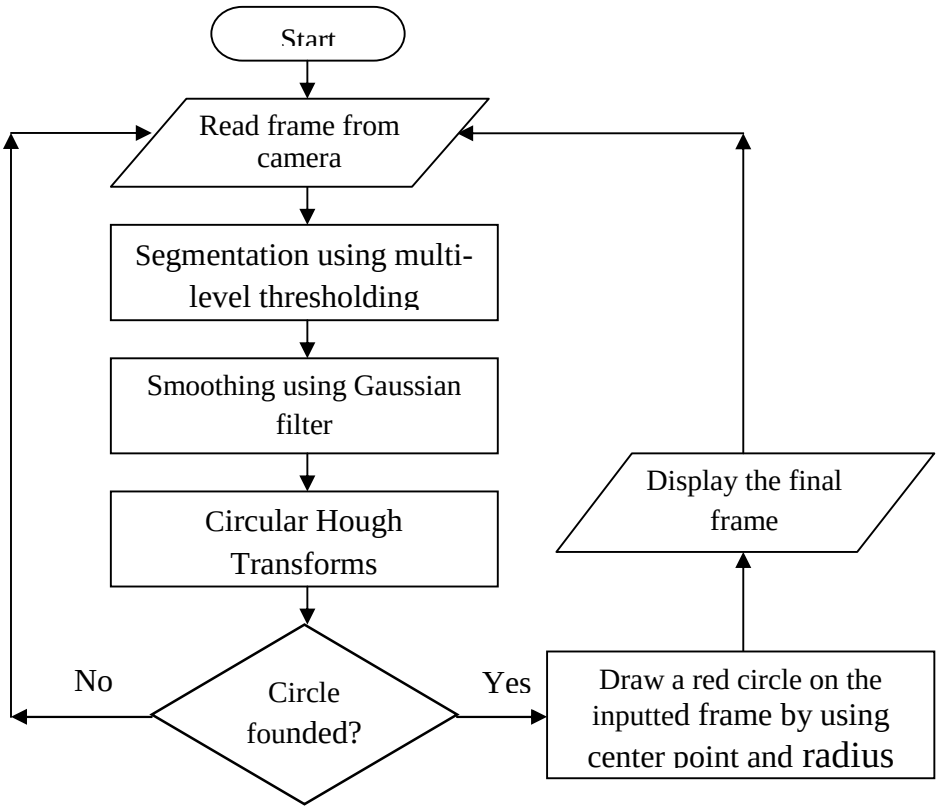
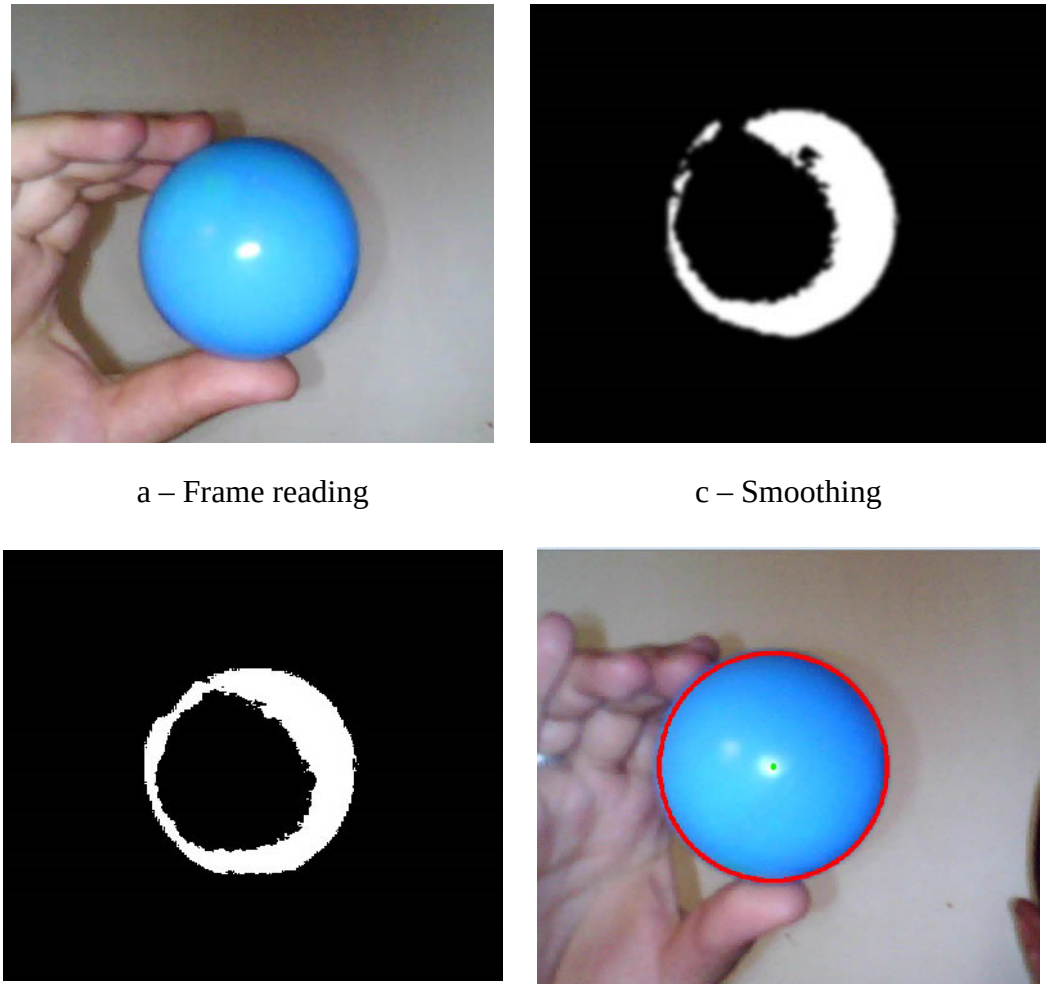


Figure (4): Flow-chart of processing steps

A ball with a blue color as shown in Figure (5 - a) is used as a target object in this paper. The position and distance of the ball are altered every time during the computing of the time for each step in random form. The *tic-toc* instructions in MATLAB were used to calculate the time of the execution in the space of seconds. While in Open CV (C++ language) *clock_t* function from *time.h* header was used to store the processor time, then the starting time was subtracted from the ending time to get the execution time, this had been done for each step of the processing.

The time taken in the space of milliseconds in C++, so the results obtained from MATLAB converted to milliseconds to achieve maximum accuracy. In every stage of the process, the execution time was calculated for hundreds of iterations then average time was found in milliseconds for each step in both programs (MATLAB and Open CV). Finally the two average times of each processing stage for MATLAB and Open CV was compared.



a – Frame reading c – Smoothing

b – Segmentation d – Circle Hough transform

Figure (5): The results of processing stages

4. RESULTS AND CALCULATION

Comparison between MATLAB and Open CV: reading the frame from input device and histogram segmentation by multi-level thresholding was applied. After that, the resultant image will be smoothed by Gaussian filter, and then processed by Hough transform filter. The average time was calculated in millisecond for each step of execution, and the results are showed in Table (3) and Figure (6).

Table (3): The results

	Frame reading	Segmentation	Smoothing	Hough transform
MATLAB	521.333	40.186	957.733	709.466
OpenCV	422.266	13.466	18.133	31.066

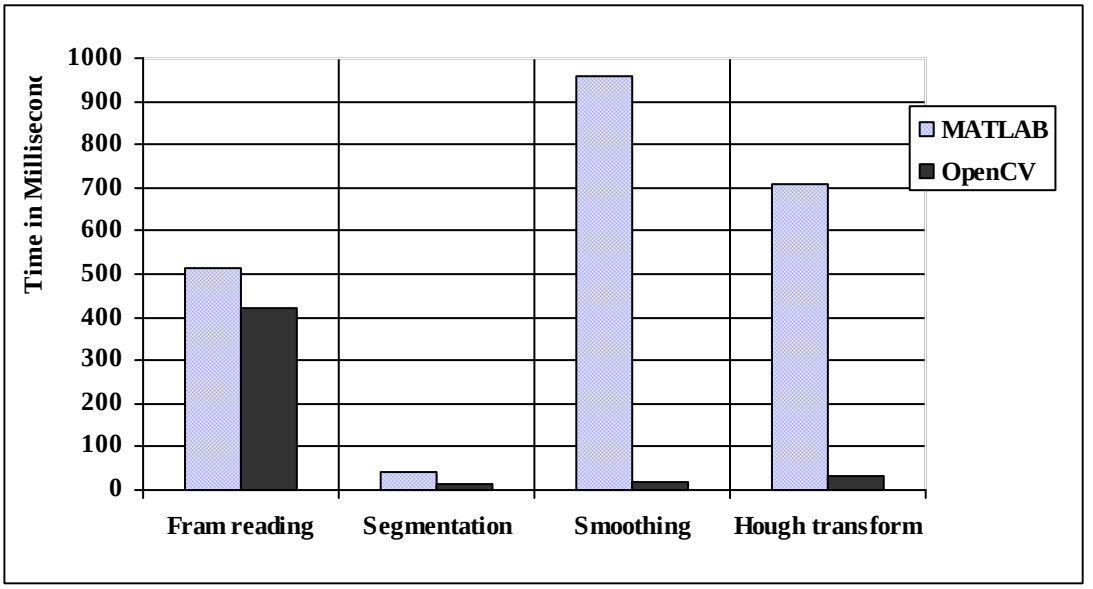


Figure (6): Illustration of results

The results show that, the Open CV is faster than the MATLAB by about:

A. 10% in reading frame from the camera.
B. 2.6% in segmentation step.
C. 94% in smoothing step.
D. 68% in Hough Transform step.

Hence, the overall speedup ratio is about 174%. The main reason of the OpenCV processing speed in image processing is that, in MATLAB it is not necessary to care about memory allocation and memory leak, but it is very important task in every processing stage in Open CV. The effect of memory allocation is showed clearly in tasks that require more memory to process like smoothing and circle Hough transform process.

In addition, MATLAB is a pretty high-level scripting language, and it is built on Java, and Java is built upon C. Therefore, when run the MATLAB program, the computer is busy trying to interpret all that MATLAB code. Then turns it into Java, and then finally executes the code. Open CV is basically a library of functions written in C. Hence it is closer to directly provide machine language code to the computer to execute. Thus more image processing done at computers processing cycles, and not more interpreting. Because of that, programs written in Open CV run much faster than similar programs written in MATLAB.

5. CONCLUSION

The results show that the code, which is written by visual C++ and Open CV library, to find a circular object with specific color from a live video, taken by a web camera, speeds up the system by about 174%, than the code, which is written by MATLAB, for the same operations.

REFERENCES

[1] Slavomir M, Robert H, and Miroslav B "The Comparison of CPU Time Consumption for Image Processing Algorithm in Matlab and Open CV", University of Zilina, IEEE, 2012, pp.75- 78.

[2] Sharmila B, Karalan N, and Nedumaran D, "Image Processing on DSP Environment Using OpenCV", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 2, February 2015, pp. 489-493.

[3] Gary B, and Adrian K, "Learning Open CV", USA, September 2008, pp. 8 -109.

[4] Issue of MathWorks, "MATLAB Programming Fundamentals", September 2013 (Release 2013b).

[5] Miyoun J, Myeongmin K, and Myungjoo K, "Variational Image Segmentation Models Involving Non-smooth Data-Fidelity Terms", Springer Science, 2014, pp. 277-308.

[6] Rafika H and Ezzedine B, "Color Image Segmentation by Multilevel Thresholding using a Two Stage Optimization Approach and Fusion", International Journal of Engineering and Innovative Technology (IJEIT), Volume 3, Issue 11, May 2014, pp. 14-20.

[7] Rohan K, Ashok K, and Sanjay B, "Review: Existing Image Segmentation Techniques", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 4, April 2014, pp.153-156.

[8] Nayan P, Abhishek S, Mayur M, and Kruti D, "A Study of Digital Image Filtering Techniques in Spatial Image Processing", IEEE, 2014, pp. 1-7.

[9] Meng L and Subhashis G, "Bayesian Multiscale Smoothing of Gaussian Noised Images", International Society for Bayesian Analysis, Volume 9, Number 3, 2014, pp. 733-758.

[10] Aditya G, Akhilesh B, and Kuntal C, "A Comprehensive Review of Image Smoothing Techniques", International Journal of Advanced Research in Computer Engineering & Technology, Volume 1, Issue 4, June 2012, pp. 315-319.

[11] Masoud N, Ronak K, and Mehdi H, "Detecting circular shapes from areal images using median filter and CHT", Global Journal of Computer Science and Technology, Volume 12, Issue 2, 2012, pp. 1-5.

[12] Dodiya B, Anu M, and Patel J, "Human Face, Eye and Iris Detection in Real-Time Using Image Processing", Journal of Engineering Research and Applications, Vol. 4, Issue 5, 2014, pp.27-31.

[13] Nitasha G, Shammi S, and Reecha S "Comparison Between Circular Hough Transform And Modified Canny Edge Detection Algorithm For Circle Detection", International Journal of Engineering Research & Technology, Vol. 1 Issue 3, May – 2012, pp. 1-5.

The work was carried out at the Technical College / Mosul