

## **Machine Intelligent System Algorithm to Recognize Different Shaped Color Targets**

**Duaa A. Taban<sup>1</sup>, Ali Al-Zuky<sup>2</sup>, Anwar H. Al-Saleh<sup>3</sup>, Haidar J. Mohamad<sup>4</sup>**

**<sup>1,2,4</sup> Department of physics, College of Science, Mustansiriyah University**

**<sup>3</sup> Department of Computer Science, College of Science, Mustansiriyah University**

[rawanali25@yahoo.com](mailto:rawanali25@yahoo.com), [prof.alialzuky@uomustansiriyah.edu.iq](mailto:prof.alialzuky@uomustansiriyah.edu.iq),  
[anwar.h.m@uomustansiriyah.edu.iq](mailto:anwar.h.m@uomustansiriyah.edu.iq), [haidar.mohamad@uomustansiriyah.edu.iq](mailto:haidar.mohamad@uomustansiriyah.edu.iq)

### **Abstract**

Geometrical shape plays a vital role in computer vision applications. In this study, a new suggested method used to detect geometric shapes (square and circle) of different colors (red, green, and blue), and different arrangements within the same input image. The introduced algorithm classified 256 tested input images at the same time and feedback the recognized index number to each individual image. These tested images have different details for the same color target by means occurrences like different in orientation, position, and partially appeared shape. A new idea of indexing number used to describe shape and color of input target images for all possibility of occurrences. There were 42 possible arrangements of image cases because of the color and shape targets that used in this study. The introduced algorithm detects shape, and color with an accuracy of 100%.

### **الخلاصة**

الأشكال الهندسية تلعب دوراً مهماً في تطبيقات الرؤية الحاسوبية. لذلك في هذه الدراسة تم إقتراح نظرية جديدة لكشف نوعين من الأشكال الهندسية (دائرة ومربع) حيث كل شكل له ثلاثة ألوان (أحمر و أخضر و أزرق). هذه الأشكال الهندسية تم كشفها بترتيبات مختلفة داخل مجال الصورة حيث يوجد ٤٢ ترتيب مختلف لهذه الأشكال. تقوم الخوارزمية المقترحة بتصنيف ٢٥٦ صورة على اساس تفاصيل الأشكال الهندسية الموجودة وتعطي كل صورة دليل معين. حيث أن كل صورة من هذه الصور تختلف باختلاف شكل ولون وترتيب الأشكال الهندسية المستعملة وكذلك باختلاف موقع الشكل داخل الصورة. تقوم الخوارزمية المقترحة بتصنيف الأشكال الهندسية وتحديد نوعها ولونها بنسبة كشف ١٠٠%.

**Keywords:** computer vision, geometric shape recognition, target indexing

## **1. Introduction**

In daily life, different targets with different shapes observe in a public places or workplace with human eyes. The human brain can easily classify and distinguish these targets, but the computer does not have this ability. Therefore, computers need to be programmed to classify or recognize different targets, and this is not a simple task. However, computer vision techniques make this task achievable [1]. Thus, computer vision can be developed to detect shape targets and analyze them as an advanced stage [2]. Shape detection method depends on analyzing color and texture features. There are a number of visual information aspects and among them target detection which certainly has a wide range of applications in robotics, fingerprint analysis, handwriting mapping, face recognition, remote sensors and monitoring system for agriculture fields [3]. As long as the machine programmed correctly, a direct interaction between human being and machine is not necessary [4] [5]. Hough transform technique is commonly used to analyze the image for circle detection and discriminate it from another shape such as square, polygon, triangle, etc. [6]. The position of a target within the input image can be found by specifying the center of the target area in the image plane. However, the center of the area is a point and relatively insensitive to image noise [7]. Target color is one of the most important characteristics in image analysis. The color content of the target is described using RGB color bands histograms [8, 9].

Many researchers have been conducted to study shape detection in different ways like using Extent parameter to recognize shapes of different color. Where the recognition is sensitive to the existence of shadow and give wrong results for high shadow environment [10]. shape detection based on the location of the edges to calculate the area of the target, but it is applicable only on grey images [11]. Different interface used namely Raspberry Pi and Odroid c2 to detect target edges. Where the introduced edge detection methods were commonly used and the output results were well-known [12]. A shape factor code in MATLAB software used to recognize more than one target in the image [13]. Python and MATLAB software used to detected contours, shapes and colors of various geometrical figures by using Open Source Computer Vision Library (Open CV) [14].

In this study, the suggested method to detect more than one target within the input image depends on introducing a new algorithm. This algorithm coded the input target images using a new method based on an index number. The tested captured images are different in geometrical shapes (square and circle), colors (red, green, and blue), and arrangements (one or two objects). The introduced algorithm describes the input image case (input target shape, and color) using six- digits binary number as an index table. The input images have same features like similar in target shape but differ in a position within the image plane, the orientation of the target, and space between targets (in case of two targets). The occurrences for each 256 input image solved using the introduced algorithm to feedback 42 image cases, where the possible cases for this study are 42. As a result, the output detection of input targets using the designed algorithm gives high accuracy and easy in implementation.

## **2. Suggested Imaging System**

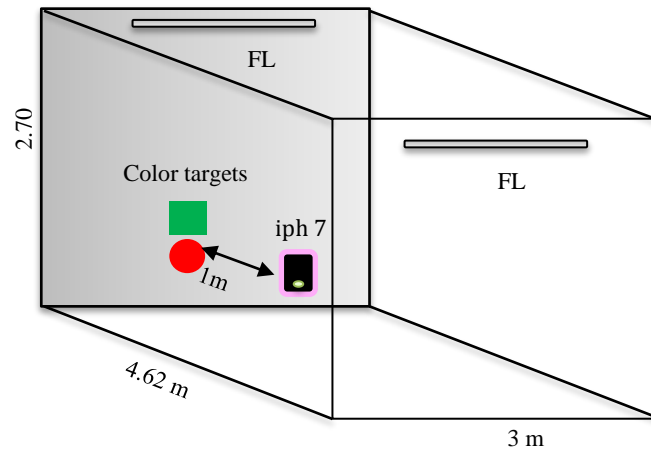
In this study, starting from acquiring an image, preprocessing, analysis the objects, and to the final step of detect objects, a new algorithm is suggested to perform this approach. A new idea used to solve multi-object detection depending on the new indexing method. Each input image indexed by giving a unique number depending on a number describes the target shape and color. The indices of target details can be used to make a decision system, later on, to control ON/OFF hardware devices that interfaced with a computer.

The tools and software used in this work are:

- 12-mega-pixel wide-angle iPhone 7s plus camera, with a ( $f/1.8$ ) aperture for wide-range lens and ( $f/2.8$ ) aperture for a telephoto lens.
- Two types of geometric shape (square and circle) made of cardboard with three colors (red, green, and blue) double for each shape and color (i.e. 12 geometric shape targets).
- MATLAB R2017a software running on Windows 10 laptop.

The tools and software applied to design a smart system to detect a multi-target within an input image. Figure (1) shows the sketch of the suggested imaging system, where the target colored shapes (circle or square) hold by hand in front of off-white wall color in order to increase the contrast of output image results. The room dimension is 4.62m length, 3m width, and 2.70 m height with appropriate lighting condition. The distance between the iPhone camera (iph7) and targets is 1m. Lighting condition parameter is important by means of detect targets in the suggested imaging system. This parameter affects image lighting quality

and the color of the captured image. Therefore, the lighting condition is satisfied by using two normal fluorescent lamps (FL) and environment luminance was 109 Lux measured by a Luxmeter device. These parameters and conditions are fixed for all cases. The acquired images export from the iPhone to the PC in order to apply the suggested algorithm.





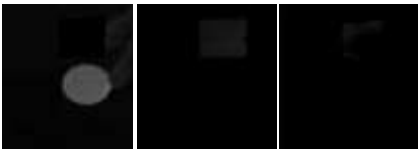

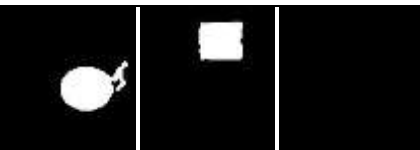
**Figure (1) Environment sketch of the suggested imaging system.**

### **3. Suggested Algorithm to detect Shape and Color Target**

In the first step, it is necessary to convert RGB input image bands into a grey image to extract the red, green and blue of grey color bands by using *imsubtract* Matlab function. Then the median filter applied to remove noise from each grey band. The enhanced three bands converted into binary to obtain the results of detection color targets. Merge three binary bands, then apply *regionprops* and *imfindcircles* Matlab functions to obtain three bands merged image as in step 5 shown in Figure (2) of the algorithm. The benefit of applying *regionprops* is to get number and center coordinates of objects. While *imfindcircles* were used to calculate the number of diameters and center coordinates of the detected circle. The last step is detecting the shape of the targets by the number of objects and the number of diameters from the previous step. The position of target determined by using a center coordinates of detected object and circle. The introduced algorithm classified 256 input images (images may include one or two targets) into 42 different cases. There are six cases for one target and 36 cases for two targets. The output code from the introduced algorithm solved the similarity of the input images. The similarity means the same input image for one or two shapes but different in the orientation, position in the image plane, or space between shapes (for two target case). Figure 2 (a and b) shows some of the input images cases for one and two targets, respectively. The steps of the algorithm and the output images are shown in Figure (3).



**Figure (2) (color online) Input image set (a) one target set (b) two target set.**

<p><b>Input:</b> color target image (I) as shown below</p> 	
<p><b>Output:</b> (SD) which describes (Shape, Color, and Number of targets), and output index code (id code) for each input image.</p>	
Algorithm steps	Output
<p><b>Step 1:</b> convert (I) it into a grey image (Igray).</p>	
<p><b>Step 2:</b> extract out red Ir, green Ig, and blue Ib targets from the color image (I) as follow:  <math>I_r = I_R - I_{gray}</math>; <math>I_g = I_G - I_{gray}</math>; <math>I_b = I_B - I_{gray}</math>                  Where: IR, IG, IB are red, green, and blue bands.</p>	
<p><b>Step 3:</b> applying a median filter mask on images Ir, Ig, and Ib.</p>	
<p><b>Step 4:</b> convert images Ir, Ig, Ib into binary using thresholds value (thr1, thr2, and thr3) respectively</p>	
<p><b>Step 5:</b> merge images matrices Ir, Ig, and Ib as follow:  <math>I_a = \left[ \frac{I_r}{255} + \frac{2 * I_g}{255} + \frac{3 * I_b}{255} \right]</math></p>	
<p><b>Step 6:</b> determine the properties of the image (Ia) by using Matlab function (<i>reigonprops</i>) and determine the number of targets (nt) in the image.                  i1 is x-axis coordinate of the first target center. j1 is y-axis coordinate of the first target center.                  i2 is x-axis coordinate of the second target center. j1 is y-axis coordinate of the second target center.</p>	
<p><b>Step 7:</b> detect the number of circles in (Ia) using Matlab statement  <math>[C \ D] = \text{imfindcircles}(I_a)</math></p>	

<p>Where C represents x,y center of the circle C (1) is x-axis coordinate of circle center, C (2) is y-axis coordinate of circle center. D represents the diameter of the determined circle.</p> <p>The number of circles computed by using Matlab statement  <code>nc=length (D)</code>                  where nc is the number of circles in the image.</p>
<p><b>Step 8:</b> for two color targets in the image plane (sh1 for the first shape, sh2 for the second shape) then, the following condition is used:</p> <ol style="list-style-type: none"> <li>1. For two squares in image:  <code>nc=0</code>  <code>sh1=0; sh2=0</code></li> <li>2. For two circles in image:  <code>nc=2</code>  <code>sh1=1; sh2=1</code></li> <li>3. For one circle and one square in image:                      if <math>  (C (2)-j1)   &lt;   (C (2)-j2)  </math>  <code>sh1=1; sh2=0</code>                      else  <code>sh1=0; sh2=1</code>                      End</li> <li>4. For one target in image:                      One circle <code>nc=1, sh=1</code>                      One square <code>nc=0, sh=0</code></li> </ol>
<p><b>Step 10:</b>                  Output <i>id</i></p>

**Figure (3) Shape and color target detection algorithm.**

#### **4. New Indexing Technique**

A new suggested idea to describe the color, and shape of targets depending on the indexing method. The suggesting indexing method consists of six-digits of binary and decimal code. The shape represented by 1-digit binary number i.e. 1 or 0 for circle and square, respectively. The color represented by two-digit binary numbers 00, 01, and 10 to represent the red, green, and blue color of the input target shape, respectively. Table (1) shows the suggested binary code number for one and two targets for the input image plane. For one target representation, there are three digits fixed to 1 value which are *bn2*, *bn5*, and *bn6* i.e.  $bn2=bn5=bn6=1$  this idea is used to avoid similarity code with two targets code. First digit *bn1* represents the target shape which 0 for square, and 1 for the circle. Third digit (*bn3*) and fourth digit (*bn4*) represent the first and second bit of target color, respectively, so the input image with one target has six bits to represent it. For instance, if the circle is green in the table (1) it can be represented by  $bn=1\ 1\ 0\ 1\ 1\ 1$ . The digits  $bn2=bn5=bn6=1$  are fixed

number, the circle represented by  $bn1=1$ , binary code of green color is  $bn=0$  1, so,  $bn3=0$ , and  $bn4=1$ .

Two targets in the input image plane have six digits to represent it. The first digit ( $bn1$ ) and second digit ( $bn2$ ) represent upper and lower target shape respectively. Third ( $bn3$ ) and fourth digits ( $bn4$ ) represent the first and the second bit of upper target color, respectively. Fifth ( $bn5$ ) and sixth digits ( $bn6$ ) represent the first and the second bit of lower target color, respectively. For instances, if the two targets representation in the table (1) is a blue circle for the upper shape and a red square for the lower shape, the binary code will be  $bn=1$  0 1 0 0 0. The first and second digits  $bn1=1$  and  $bn2=0$  represent circle and square which is upper and lower target shape, respectively. The binary code for blue color  $bn=10$  which is the color of the upper target, so  $bn3=1$  and  $bn4=0$ . The color of the red lower target is  $bn=0$  0, so  $bn5=0$ , and  $bn6=0$ .

**Table (1) calculation of binary number  $bn$  to represent one and two targets in the input image plane.**

Output binary code number for one target in the image plane						
Target representation	target shape	Fixed number	1 <sup>st</sup> bit of target color	2 <sup>nd</sup> bit of target color	Fixed number	Fixed number
or	$bn1=0$ or 1	$bn2=1$	$bn3=0$ or 1	$bn4=0$ or 1	$bn5=1$	$bn6=1$
Output binary code number for two targets in the image plane						
Target representation	upper target shape	lower target shape	1 <sup>st</sup> bit of upper target color	2 <sup>nd</sup> bit of upper target color	1 <sup>st</sup> bit of lower target color	2 <sup>nd</sup> bit of lower target color
↔ ↔	$bn1=0$ or 1	$bn2=0$ or 1	$bn3=0$ or 1	$bn4=0$ or 1	$bn5=0$ or 1	$bn6=0$ or 1







The introduced output code number  $C$  for one and two targets are shown in the table (2). For one target,  $C$  has six decimal codes [ $C1$ ,  $C2$ ,  $C3$ ,  $C4$ ,  $C5$ , and  $C6$ ], the  $C1$  represents target shape,  $C3$  and  $C4$  represent target color. While  $C2$ ,  $C5$ , and  $C6$  have fixed one-digit number.  $C1$ ,  $C3$ , and  $C4$  have two cases numbers for each of them (i.e.  $C1=0$  or 32,  $C3=0$  or 8, and  $C4=0$  or 4). The value of  $C1$ ,  $C3$ , and  $C4$  is determined to be depend on values of  $bn$ , for example,  $C1=0$  if  $bn=0$  and  $C1=32$  if  $bn=1$ , and the same procedure for  $C3$ , and  $C4$ . In table (2) for example, if the shape is



a green circle the  $bn$  equal to [1 1 0 1 1 1], therefore  $C = [32, 16, 0, 4, 2, 1]$ ,  $C1=32, C2=16, C3=0, C4=4, C5=2,$  and  $C6=1$ .

The code numbers  $C$  for two targets the  $C1$  and  $C2$  representing the upper and lower target position, respectively.  $C3$  and  $C4$  representing the upper target color while  $C5$  and  $C6$  represent the lower target color. Each digit of  $C$  has two cases numbers (i.e.  $C1= 0$  or  $32, C2= 0$  or  $16, C3= 0$  or  $8, C4=0$  or  $4, C5= 0$  or  $2,$  and  $C6= 0$  or  $1$ ), these values related to  $bn$  value from table (1). For example, the value of  $C^1=0$  if  $bn=0$  and  $C^1=32$  if  $bn=1$ . For example, if the upper shape is a blue circle and lower shape is a red square in the table (2) then the  $bn$  equal to [1 0 1 0 0 0]. Therefore,  $C$  for this case  $C = [32, 0, 8, 0, 0, 0]$  for upper shape  $C1=32,$  for lower shape  $C2=0. C3=8$  and  $C4=0$  which represent upper target color, while the color of the lower target represented by  $C5=0$  and  $C6=0$ .

**Table (2) calculation of decimal code number to represent one and two targets in the input image plane.**

Output code number for one target in the image plane						
Target representation	C of the target shape	Fixed number	1 <sup>st</sup> bit of the target color	2 <sup>nd</sup> bit of the target color	Fixed number	Fixed number
 or 	$C1= 0$ or $32$	$C2= 16$	$C3= 0$ or $8$	$C4= 0$ or $4$	$C5=2$	$C6=1$
Output code number for two targets in the image plane						
Target representation	C of the upper target shape	C of the lower target shape	1 <sup>st</sup> bit of the upper target color	2 <sup>nd</sup> bit of the upper target color	1 <sup>st</sup> bit of the lower target color	2 <sup>nd</sup> bit of the lower target color
 ⇌   	$C1= 0$ or $32$	$C2= 0$ or $16$	$C3= 0$ or $8$	$C4=0$ or $4$	$C5= 0$ or $2$	$C6= 0$ or $1$















































































The output index code  $id$  can be extracted from the table (2) which is referred to the input image case. So, for each input image, there is a specific  $id$  number. The  $id$  code for one and two targets can get using the following equation:

$$id = \sum_{i=1}^6 Ci \quad (1)$$



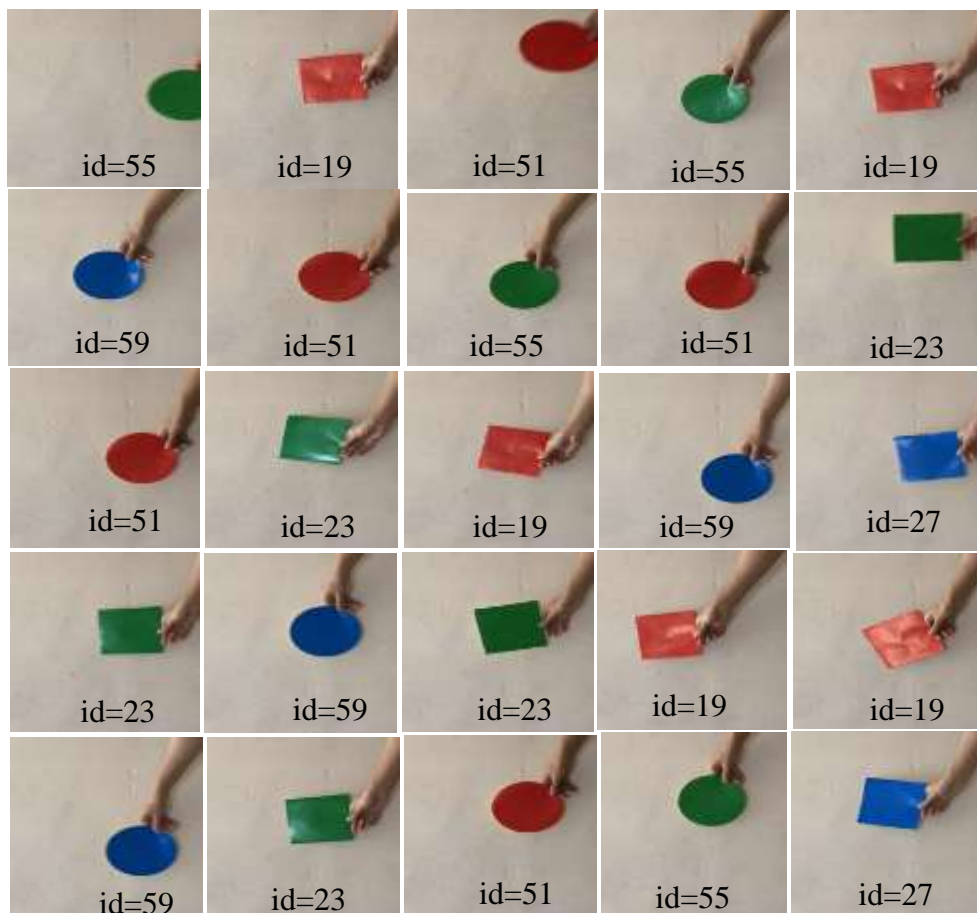
In this study the suggest algorithm described the shape and color of input image case using *bn*. Table (1) illustrate how *bn* is used for shape description of input images, which is binary code consists of six bits to describe one, and two targets in the image plane. For 256 input images, there are 42 output images named as its output index (*id*) that acquire from equation (1). Table (3) shows a diagram of 42 output image cases (six images of one target and 36 images of two targets) corresponding to their output indices *id*.

**Table (3) (color online) Output indices.**

series	case	occ	id	series	case	occ	id	series	case	occ	id
1		5	0	15		5	21	29		7	41
											
2		9	1	16		7	22	30		5	42
											
3		8	2	17		5	23	31		5	48
											
4		8	4	18		7	24	32		6	49
											
5		8	5	19		6	25	33		5	50
											
6		7	6	20		5	26	34		5	51
											
7		8	8	21		4	27	35		5	52
											
8		8	9	22		5	32	36		4	53
											
9		7	10	23		8	33	37		4	54
											
10		5	16	24		8	34	38		4	55
											
11		8	17	25		9	36	39		5	56
											
12		8	18	26		5	37	40		6	57
											
13		5	19	27		6	38	41		4	58
											
14		7	20	28		6	40	42		4	59
											

**5. Results**

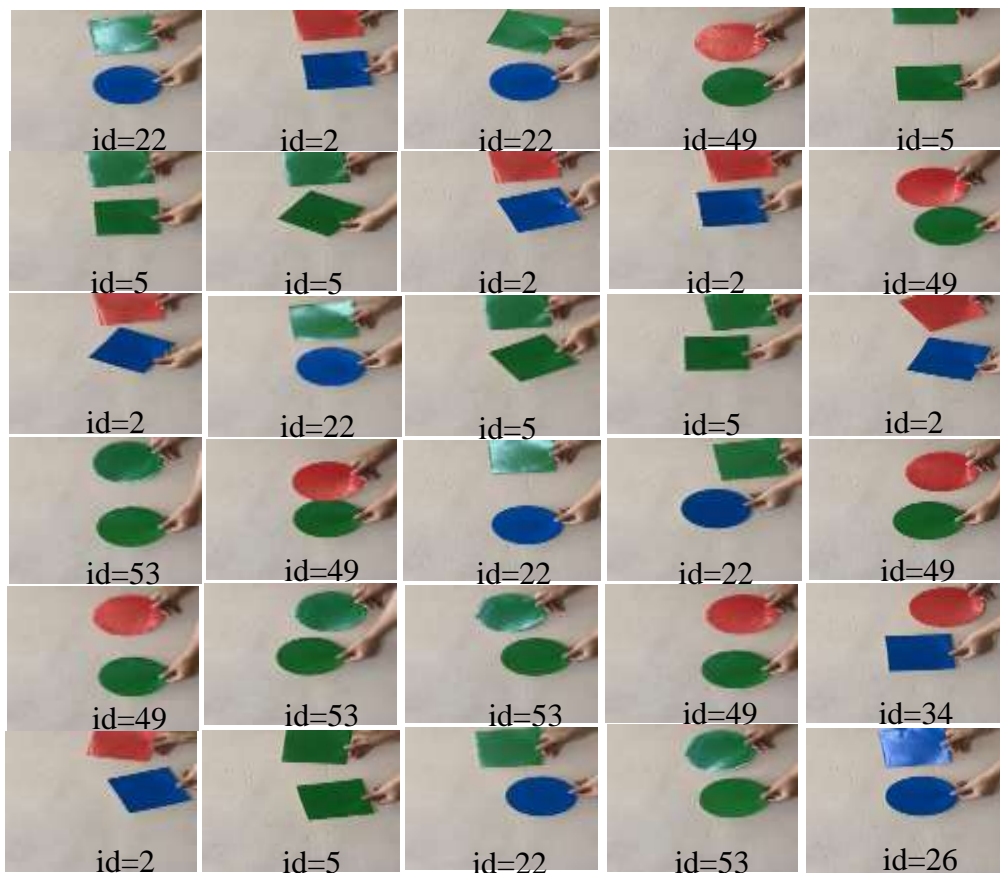
The main idea consists of distinguishing objects within an input image to be used in different applications. Therefore, a new idea of programming employed within the suggested designed algorithm to index every input object. The studied objects were circle and square with different colors, and this study can be extended to include more shapes and colors. These shapes are chosen because of its frequently used in computer vision applications within manufacturing use. According to using two shapes and three colors, the possibility maximum cases are 42 case only. The studied input images were 256 images because of the occurrences of each one or two targets. Therefore, the introduced algorithm tested the 256 randomly input images as input and feedback 42 image cases named as its output indices (*id*). Figure (4) shows output indices for some of randomly input images of one target in an image plane with its recognition *id* number.



**Figure (4) (color online) The tested image sets and output indices for one target image cases.**

Figure (4) shows some of the occurrences of one shape in the image plane. The output *id* is the results of the introduced algorithm which showed on each image. These images were tested all together and stored with a name consist of original image name and *id* (image name\_id) only in programming use.

The difficulty increased using two objects within the input images. The possibility of having more than one image in a single case is increased, which named as occurrences in the table (3). Figure (5) shows some of these occurrences and classified correctly as its output *id*. The strategy of the introduced algorithm for testing two objects is based on detecting the circle shape. Therefore, the output *id* resulted by checking the number of the objects first then check each object, whether it is a circle or square. Figure (5) shows output indices for the random input images of different cases. For clarity, it is difficult to show all case occurrences because it is out of range of this study. The same strategy of the stored image used in one target case is used to name the output image. All possibilities are considered by means of showing some part of the object, as shown in Figure (5), to test the performance of the suggested algorithm.



**Figure (5) (color online) The tested input image and its output indices for two target image cases**

### **. Conclusion and Discussions**

In this study, a suggested algorithm is used to detect target shape and color by classifying 256 images (with one, and two targets in the image plane) consist of 42 sets. Each set has a specific number of images similar in target shape and color but differs in orientation, target position in the image plane, and spaces between targets. This algorithm has advantages of eliminating noise or clutter from images by separating colored bands and extracting out the color target red, green, and blue. The introduced algorithm detects different orientation for the square shape efficiently in both one and two objects in the input image. In the case of two targets, the output *id* does not affect by the spaces between targets and even if the two targets are attached. The objects in some cases partially appear, however, it is still recognized. The effect of luminosity on the input images has disadvantages on recognizing shapes, however, the steps of the proposed algorithm can eliminate this effect and determined output *id* efficiently. As a result, the designed algorithm detects shape and color with an accuracy of 100%.

### **References**

- [1] S. Gomez, Shape recognition using machine learning, in: Computing Congress (CCC), 2011 6th Colombian, IEEE, 2011, pp. 1-3.
- [2] A.G. Raghav Puri, Manas Sikri Contour, Shape, And Color Detection Using Open Cv – Python International Journal of Advances in Electronics and Computer Scienc, 5 (2018).
- [3] D. Pandey, P. Singh, A Review of shape Recognition techniques, Int. J. Emerg. Res. Manag. Technol, 3 (2014) 40-43.
- [4] M.F. Zakaria, H.S. Choon, S.A. Suandi, Object shape recognition in image for machine vision application, International Journal of Computer Theory and Engineering, 4 (2012) 76.
- [5] K. Chattopadhyay, J. Basu, A. Konar, An efficient circle detection scheme in digital images using ant system algorithm, arXiv preprint arXiv:1106.0962, (2011).
- [6] D. Vernon, Machine Vision, Automatic Visual Inspection and Robot Vision, in, Printed in Great Britain by Cambridge Press, 1991.
- [7] R. Jain, R. Kasturi, B.G. Schunck, Machine vision, McGraw-Hill New York, 1995.
- [8] V. Goel, S. Singhal, T. Jain, S. Kole, Specific Color Detection in Images using RGB Modelling in MATLAB, International Journal of Computer Applications, 161 (2017).
- [9] H.H. Abbas, A proposed Algorithm for Interactive Geometric Shapes Recognition, Eng. & Tech. Journal vol. 32 Part (A), No.5, (2014).
- [10] S. Rege, R. Memane, M. Phatak, P. Agarwal, 2D geometric shape and color recognition using digital image processing, International journal of advanced research in electrical, electronics and instrumentation engineering, 2 (2013) 2479-2487.
- [11] V. Kumar, S. Pandey, A. Pal, S. Sharma, Edge Detection Based Shape Identification, arXiv preprint arXiv:1604.02030, (2016).
- [12] S.V. Sivasankari S, Performance Analysis of Shape Recognition Algorithms on Embedded Boards RASPBERRY PI and ODROID C2, International Journal of Pharmaceutical Sciences Review and Research, 43 (2017) 103-106.
- [13] H.M. Zangana, A New Algorithm for Shape Detection, IOSR Journal of Computer Engineering (IOSR-JCE), 19 (2017) 71-76.
- [14] R. Puri, A. Gupta, Contour, Shape & Color Detection using OpenCV-Python, 2018.