# A real-time pattern recognition module via Matlab-Arduino interface

Oscar J. Suarez, D.Sc.[1,2], Nicolás Hernández Díaz, Eng.[2], and Aldo Pardo Garcia, PhD.[2]
[1]Centro de Investigación y de Estudios Avanzados - CINVESTAV, Guadalajara, México, javier11213@hotmail.com
[2]Universidad de Pamplona, Pamplona, Colombia, nicolashernandezdiaz@gmail.com, apardo13@hotmail.com

***Abstract*– Pattern recognition is a relevant research area in artificial vision, where several methods have been proposed in the last 50 years. This paper presents a real-time pattern recognition algorithm for an addition operation through two six-sided dice using an Android camera device, an IP webcam app, a graphical user interface (GUIs) from Matlab, and Arduino technology. The methodology to develop the interface and the communication between Matlab software and Arduino technology is presented. To evaluate the performance of the proposed methodology, a real-time implementation using an Arduino Mega 2560 board and Matlab is illustrated.***

***Keywords*-- Pattern recognition, Digital image processing, Artificial vision, Matlab, Arduino, Real-time systems.**

## I. Introduction

Pattern recognition plays key roles in the processes of engineering, computing, mathematics, and economy, among others [1][2][3]. Pattern recognition problems generally denote classification or description of a set of processes or events to extract information and establish properties between these processes [3][4][5][6].

In this sense, different works have developed pattern recognition algorithms using the Matlab interface [7] [8] [9] [10]. In [7], a low computational cost method for the online acquisition of electrocardiogram signals (ECG) is illustrated, whose storage and processing are managed through a graphical user interface (GUI) from Matlab. The ECG signal is sampled at 1 kHz, scanned, and fed into a microcontroller-based embedded system to convert the ECG data into a serial bitstream in RS232 format. This serial data is transmitted to a desktop personal computer at a speed of 19.2 kbps for storage. The GUI interface is designed to perform an online analysis of the ECG data to calculate different types of characteristics in the time plane, which are displayed through an interface with the ECG signal diagram. In [8], a platform was developed in Matlab for the training and evaluation of prosthesis control algorithms called BioPatRec. This platform allows the implementation of a wide variety of specialized signal processing algorithms, such as feature selection and extraction, pattern recognition, and real-time control. BioPatRec uses the implementation of a pattern recognition algorithm such as Linear Discriminant Analysis and Multilayer Perceptron; furthermore, it has a common repository of bioelectric signals that allow the reproducibility of the experiment and high-resolution comparison using a Matlab GUI interface. Furthermore, [9] showed that traffic management has gradually transformed into a digital and intelligent system due to the rapid development of the transport industry in China. Considering this problem, an automatic license plate recognition algorithm based on image processing technology is established to develop a fast and straightforward display system using a Matlab GUI interface. Finally, in [10], the results of the simulation of the biometric image processing algorithm using the UPOL database for an iris recognition system are presented. These results have an average processing speed of 4 seconds, including segmentation, feature extraction, feature selection, dimension reduction, and classification time. The developed system has a graphical user interface developed in Matlab.

On the other hand, Arduino is a microcontroller family and a software creation environment that allows developing programs to interact with the physical world. Due to its agile development capabilities and facility for quick implementation of ideas, many applications have been presented, such as those found in [11][12][13]. In [11], it was demonstrated that with adaptable Simulink models and a wide number of libraries for the Arduino IDE, the system allows electromyographic (EMG) processing as well as basic classification for actuating both basic hand models and advanced hand prostheses. In [12], the study describes the development of a robust robotic beer pourer using low-cost sensors, Arduino boards, Lego building blocks, and servo motors for the RoboBEER prototype, which evaluates foamability, bubble size, alcohol content, temperature, carbon dioxide release, and beer color. All data are evaluated using multivariate data analysis through a customized code written in Matlab. Finally, in [13], an intelligent algorithm is developed, based on fuzzy logic, to track the maximum power point (MPP) of a photovoltaic (PV) panel using Simulink Support Package for Arduino Hardware in MATLAB/Simulink.

The novelty of the present paper consists of the development of an artificial vision module that integrates a graphical user interface (GUI) from Matlab and Arduino technology (Arduino Mega 2560 board). Note that the script is developed in Matlab (.m file). This development will be used for supervision and control of gambling games, such that, through a controlled environment, the player cannot manipulate the elements involved in the game.

Digital Object Identifier (DOI):
http://dx.doi.org/10.18687/LACCEI2020.1.1.646
ISBN: 978-958-52071-4-1 ISSN: 2414-6390

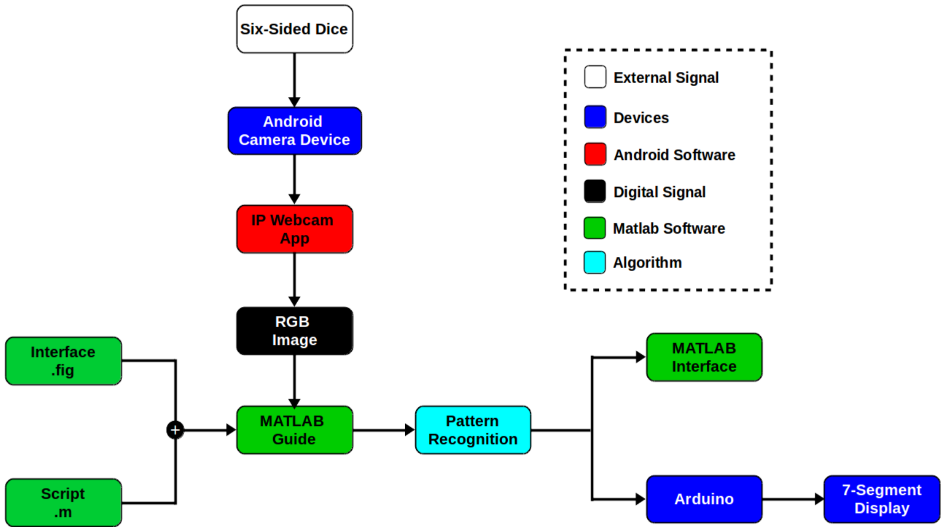


Figure 1. General scheme of pattern recognition module.

## II. General Scheme of the Pattern Recognition Module

This section presents the general scheme used in the development of pattern recognition module via Matlab-Arduino interface.

According to Figure 1, a six-sided dice is an external signal input. This signal will be obtained by a camera using an android device with IP Webcam App, which captures the RGB image to be sent to Matlab (composed of two parts: GUI interface .fig and the script .m). Afterward, a pattern recognition technique to obtain the original value given by a six-sided dice is implemented. Finally, two graphical ways to observe the real value are presented; the first is a graphical user interface, and the second is a 7-segment display.

### *A. Six-sided dice*

For a long time, the use of dice has been allowed users to experience the emotion of chance. Dice are small and throwable objects with uniquely marked sides that can rest in multiple positions. They are used to generate random numbers and are commonly used in board games [14]. Two six-sided dice are used in this paper, as shown in Figure 2.

Figure 2. Six-sided dice.

### *B. Image acquisition*

To capture an image of the environment where the dice are located, it is necessary to implement a digital camera with specific characteristics that allow information to be continuously transferred to another device for processing.

### *C. Hardware and software recommendations*

About the camera,

- Minimum resolution of 5MP.
- Able to autofocus.
- IP communication (wired or wireless).

During the development of this application, the recommended type of communication is not entirely mandatory, since classic webcams can be applied while installing the corresponding libraries in Matlab [15]. In this paper, IP communication is used for its versatility, compatibility with different types of devices, and flexibility.

IP cameras are versatile devices that allow communication to other devices connected through a network. However, they are generally expensive and difficult to access, in addition to presenting some undesirable physical characteristics (such as their size or range of mobility) that can hinder their operation in uncontrolled environments. In this paper, a camera integrated into a smartphone cellphone with an Android operating system is selected to minimize the aforementioned problems.

IP Webcam app is selected to link the phone with Matlab, and it can be found in Play Store, as shown in Figure 3. It is possible to establish communication with other devices directly by simply starting the server (see Figure 3 Red Arrow) and then taking note of the IP address (see Figure 3 Red Circle).

This configuration allows excellent versatility in this application, since cell phones are probably the most common and accessible electronic devices today.

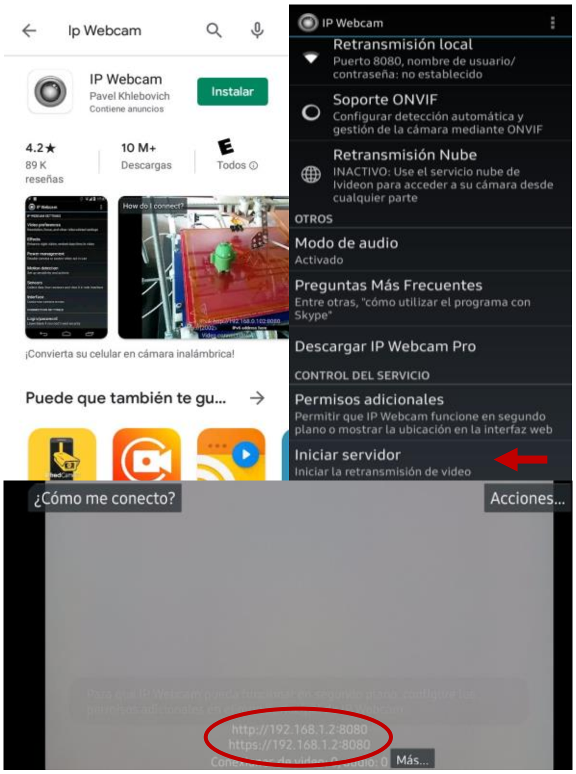

Figure 3.IP Webcam app interface.

To verify that communication between Matlab and the IP Webcam app is successful, the Matlab function "imread" is used together with a "host url" address (this corresponds to the previously taken IP address) as can be seen in Figure 4.

```
while true
    RGB1=imread('hhtp://your_ip:your_port/photo.jpg');
    imshow(RGB1)
end
```

Figure 4. Script to verify communication between Matlab and IP Webcam app.

If the process is correct, the previous link will display a Matlab figure showing with the image acquired by the Android device in real time. Note that this process can present a certain amount of delay depending on the characteristics of the device carrying out the processing, as well as the quality and speed of the network used for the connection.

### D. *Graphical user interfaces (GUI's)*

The graphical user interface from Matlab, popularly known as GUI from Matlab, is a visual programming environment for making and running programs that require continuous data entry. It provides tools to design user interfaces for custom Apps and presents different menu options such as Blank GUI (Default), GUI with unit controls, GUI with Axes and Menu, and Model Question Dialog [3] [16] [17].

For the pattern recognition module designed, a Blank GUI is selected; then, the objects that make up the graphic interface are assigned a location for their configuration. Once this process is done, the design is saved with the name "Contar" as shown in Figure 5. GUI from Matlab has two files linked to each other with the same name, but with different formats. The first is a ".fig" file extension containing information regarding the interface design, while the second is a ".m" file extension containing different types of commands that assign tasks to be performed depending on the user's interactions with the application.

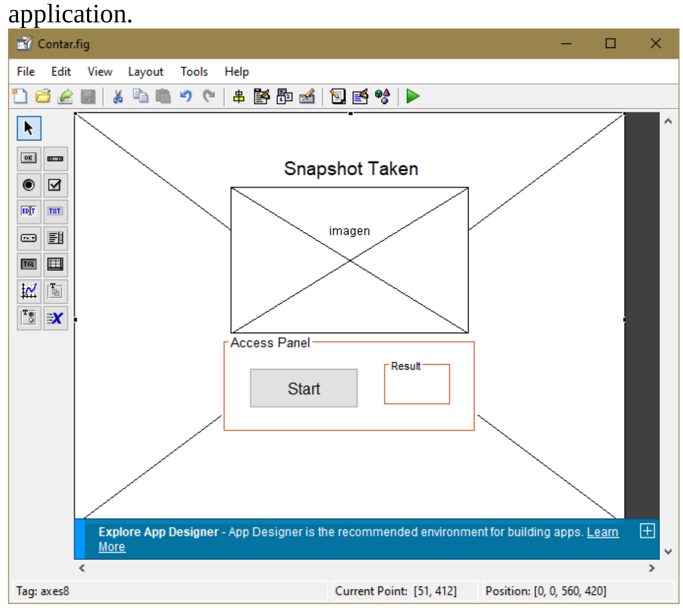

Figure 5. Final design of the graphical user interface in Matlab.

### E. *Arduino Technology*

Arduino is a general-purpose microcontroller that continually implements new improvements and utilities for hardware and software levels thanks to an extensive community of developers. Arduino has analog inputs, digital inputs, digital outputs, an analog to digital converter, pulse width modulation (PWM) outputs, and voltage supply pins. Arduino Mega 2560 is selected for its excellent features and performance [18].

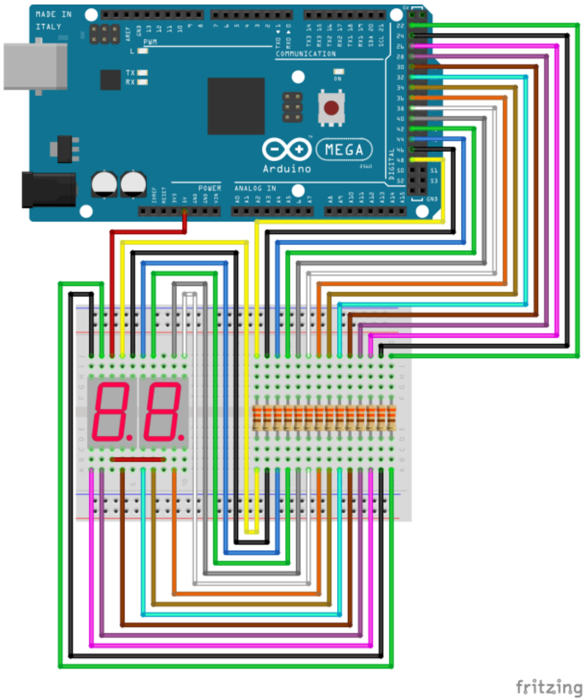


Figure 6. Connection scheme for Arduino Mega 2560 board and 7-segment displays.

To perform supervision and monitoring of the numerical value obtained after throwing two six-sided dice, two 7-segment displays type common anode with reference 5161BG are used. Figure 6 illustrates the connection scheme diagram of the Arduino Mega 2560 board and the 7-segment displays implemented. A number in the value range of 2 to 12 is displayed, representing the minimum and maximum sum that can occur when using two six-sided dice.

To verify that communication between Matlab and Arduino Mega 2560 board is successful, the script illustrated in Figure 7 is used.

```
a=arduino('COM3');% Review COM assigned to the PC to
use.
configurePin(a,'D22','DigitalOutput');% Digital Pin 22 is
defined as data output.
while true
    writeDigitalPin(a,'D22',0);% Digital Pin 22 is disabled
    LED On.
    pause(0.5)% Digital Pin 22 is kept deactivated by
    500ms.
    writeDigitalPin(a,'D22',1);% Digital Pin 22 is enabled
    LED Off.
    pause(0.5)% Digital Pin 22 is kept activated by 500ms.
end
```

Figure 7. Script to verify the communication between Matlab and Arduino Mega 2560 board.

## III. PATTERN RECOGNITION MODULE

This section presents the main contribution of this paper. The methodology used for pattern recognition on the dice and its connection to Matlab-Arduino is explained. The pseudocode to create the pattern recognition module is presented in Figure 8.

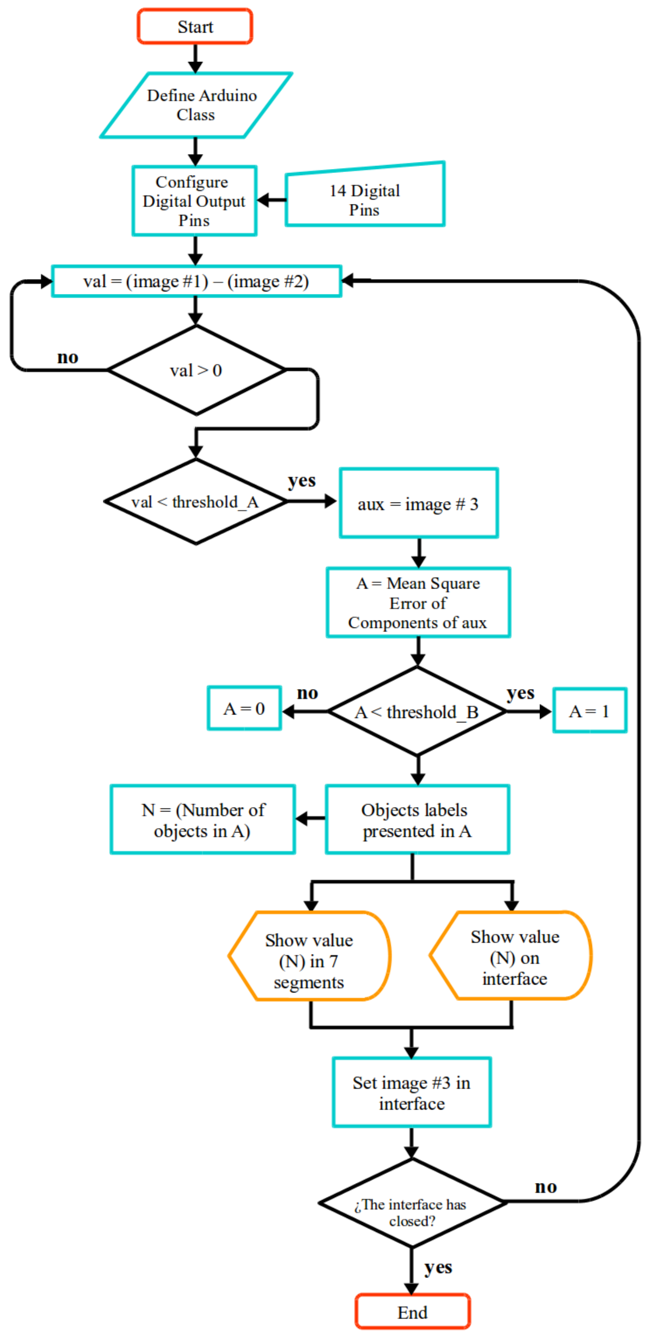


Figure 8. Pseudocode to create the pattern recognition module.

This scheme proposes a solution to the objective of identifying the numerical value obtained after throwing two six-sided dice. If the process is segregated, it first determines if the dice have stopped moving (a condition that allows the system to determine whether it is necessary to analyze the obtained image). If the condition is true, the image is processed decomposing into RGB components, setting the conversion thresholds to black and white (darkening everything that is not of interest), filtering (eliminating objects below the desired pixel limit), amplifying the desired information, and labeling it so that the algorithm can count the marks on the surface of the six-sided dice. Finally, the result obtained is printed in the Matlab graphical environment (interface illustrated in Figure 5) and in the 7-segment displays (configuration proposed in Figure 6). The previous algorithm (Figure 8) keeps the current value on the displays and Matlab interface until the dice are shaken again for another value.

In this sense, it is necessary to implement the code in two parts to execute the Matlab GUI interface. The first part is shown in Figure 9. This function is similar to the void setup () function of the Arduino IDE and is a generalized function; that is, it can be found in every GUI environment. The code only runs once and saves the initial configuration of system variables to display properties at a graphical level.

```
function Contar_OpeningFcn(hObject, eventdata, handles,
varargin)
global a
ima=imread('fondo.jpg');
axes(handles.axes8);
image(ima);
axis off;
a = arduino('COM3');
configurePin(a,'D22','DigitalOutput')% Digital Pin 22 is
defined as data output.
configurePin(a,'D24','DigitalOutput')
configurePin(a,'D26','DigitalOutput')
configurePin(a,'D28','DigitalOutput')
configurePin(a,'D30','DigitalOutput')
configurePin(a,'D32','DigitalOutput')
configurePin(a,'D34','DigitalOutput')
configurePin(a,'D36','DigitalOutput')
configurePin(a,'D38','DigitalOutput')
configurePin(a,'D40','DigitalOutput')
configurePin(a,'D42','DigitalOutput')
configurePin(a,'D44','DigitalOutput')
configurePin(a,'D46','DigitalOutput')
configurePin(a,'D48','DigitalOutput')
display_A(a,1,1,1,1,1,1,1,1,1,1,1,1,1,1);% The 7 segments are
set off.
handles.output = hObject; % Choose default command line
output for function Contar_OpeningFcn
guidata(hObject, handles); % Update handles structure
```

Figure 9. Function *Contar_OpeningFcn* of GUI interface.

The second function is shown in Figure 10, this function is similar to the void loop () function of the Arduino IDE and is particular for this application. The code has to be executed many times as indicated in the loops that describe it and its purpose is to return results according to established criteria both graphically and data transfer.

```
function Iniciar_Callback(hObject, eventdata, handles)
global a numero
RGB1=imread('http://192.168.137.195:8080/photo.jpg');%Req
  uest and acquisition of a first network image.
pause(0.5);% 500ms delay as a sample anti-slip.
RGB2=imread('http://192.168.137.195:8080/photo.jpg');%Req
  uest and acquisition of a second network image.
RGB3 = RGB2-RGB1;% The similarity between the images is
  quantified.
val = sum(sum(RGB3(:,:,1)));% The R component of  the
  second RGB image is taken and sum column by column of
  the matrix that composes forming a vector, and finally sum
  this vector by converting the component into a scalar.
while(val>0) % The loop is entered as long as the escalar is
greater than 0.
    if val<10000000% If the magnitude between the first and
second network image exceeds the threshold, dices are
considered to be in motion. If the above threshold is not
exceeded, dice are considered stationary.
RGB5=imread('http://192.168.137.195:8080/photo.jpg');% A
third image is taken through the network.
RGB_d=RGB5;% An auxiliar variable is used to operate. The third
networked image is broken down into its RGB components.
R=double(RGB_d(:,:,1));% Red.
G=double(RGB_d(:,:,2));%Green.
B=double(RGB_d(:,:,3));% Blue.
A=sqrt((R-37.41).^2+(G-37.4).^2+(B-37.4).^2);
A(A<40)=1;% White for pixel magnitudes less than 40
intensity value.
A(A>40)=0;% Black for pixel magnitudes less than 40
intensity value.
A = bwareaopen(A,40);% Remove pixels with than 40
intensity value.
A = imfill(A,'holes');% Fills delimited objects.
se = strel('disk',2);% 2px radio disc type structuring element.
A = imdilate(A,se);% Image dilation.
[~, numero]=bwlabel(A);% Tags regions and returns the
number of connected objects.
toArduino(a,numero);
set(handles.resultado,'String',numero);
axes(handles.imagen);
image(RGB5);
axis off;
        end
    break
  end
```

Figure10. GUI interface Function.

As can be seen in Figure 11 and Figure 12 *toArduino* function and *display_A* function are used respectively. In this function object (a) entering the number value identified after throwing the dices.

```
function toArduino(a,numero)
   if numero>=2 && numero<=12
      switch(numero)
        case 2
      display_A (a,0,1,0,0,0,0,0,0,0,1,0,0,1,0);
        case 3
      display_A (a,0,1,0,0,0,0,0,1,0,0,0,0,1,0);
        case 4
      display_A (a,0,1,0,0,0,0,0,1,1,0,0,1,0,0);
        case 5
      display_A (a,0,1,0,0,0,0,0,1,0,0,1,0,0,0);
        case 6
      display_A (a,0,1,0,0,0,0,0,0,0,0,1,0,0,0);
        case 7
      display_A (a,0,1,0,0,0,0,0,1,1,0,0,0,1,1);
        case 8
      display_A (a,0,1,0,0,0,0,0,0,0,0,0,0,0,0);
        case 9
      display_A (a,0,1,0,0,0,0,0,1,0,0,0,0,0,0);
        case 10
      display_A (a,1,1,1,1,0,0,1,0,0,0,0,0,0,1);
        case 11
      display_A (a,1,1,1,1,0,0,1,1,1,0,0,1,1,1);
        case 12
      display_A (a,1,1,1,1,0,0,1,0,0,1,0,0,1,0);
        otherwise
      display_A (a,0,1,0,1,0,0,0,0,0,0,0,0,0,1);
      end
   end
end
```

Figure 11. Code of function *toArduino*.

```
function display_A(a,b,c,d,e,f,g,h,i,j,k,l,m,n,o)
writeDigitalPin(a,'D22',b); %WriteDigitalPin(objeto,#Pin,1 o 0).
writeDigitalPin(a,'D24',c);
writeDigitalPin(a,'D26',d);
writeDigitalPin(a,'D28',e);
writeDigitalPin(a,'D30',f);
writeDigitalPin(a,'D46',g);
writeDigitalPin(a,'D48',h);
writeDigitalPin(a,'D32',i);
writeDigitalPin(a,'D34',j);
writeDigitalPin(a,'D36',k);
writeDigitalPin(a,'D38',l);
writeDigitalPin(a,'D40',m);
writeDigitalPin(a,'D42',n);
writeDigitalPin(a,'D44',o);
end
```

Figure 12 Code of function *display_A*.

## IV. Simulation and Real-Time Results

The objective of this system is to acquire an image, process the information through the proposed algorithms, and illustrate the addition, corresponding to the top dice faces in a graphical user interface from Matlab and the 7-segment displays connected to Arduino. For each case, the dice are shaken using a controlled environment, as in Figure 13.


Figure 13. Controlled environment for two six-sided dice.

For this paper, 33 samples are evaluated. As the minimum values is 2 and the maximum value is 12, three times peer option is considered in the real-time experimental validation. Table 1 presents system performance through testing samples, where it can be seen that the proposed methodology is 91% effective.

Table 1. System performance through testing samples.

| Measurements | Values |
|---|---|
| Samples Number | 33 |
| Correct Detection | 30 |
| Incorrect Detection | 3 |
| Performance | 91% |

Figure 14 demonstrates how useful the proposed algorithm is in determining the result obtained after shaking the dice. Note that there is a correct correlation between the image observed by the camera in Figure 13 and the addition presented by 7-segments displays.

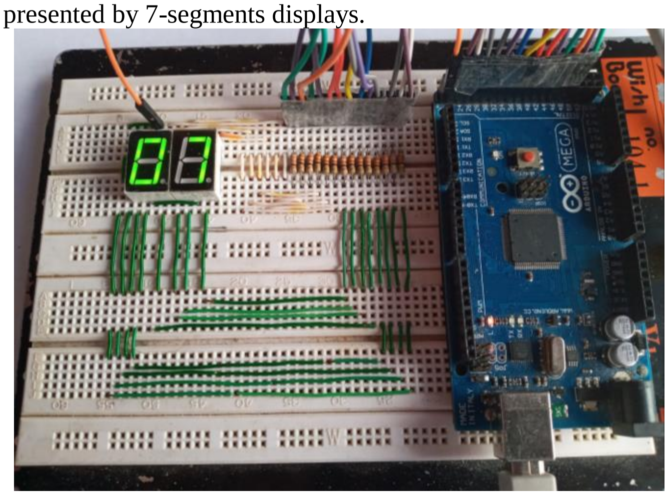

Figure 14. Full system prototype displaying the results for the first proposed scenario.

The GUI interface developed using Matlab is presented in Figure 15, showing an excellent response between the proposed simulation scenarios; the GUI interface and the two six-sided dice show the same answer, demonstrating that the proposed configuration can accurately perform the assigned task.

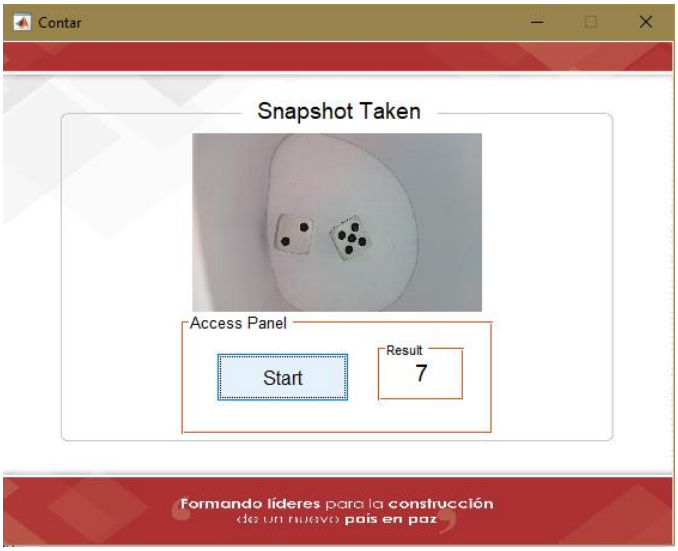


Figure 15. Pattern Recognition module for the first proposed scenario.

To evaluate a new possible scenario, dice are shaken for a new value. Figure 16 shows a real-time implementation and Figure 17 illustrates a pattern recognition module via Matlab-Arduino interface. The same value is visualized for both cases.

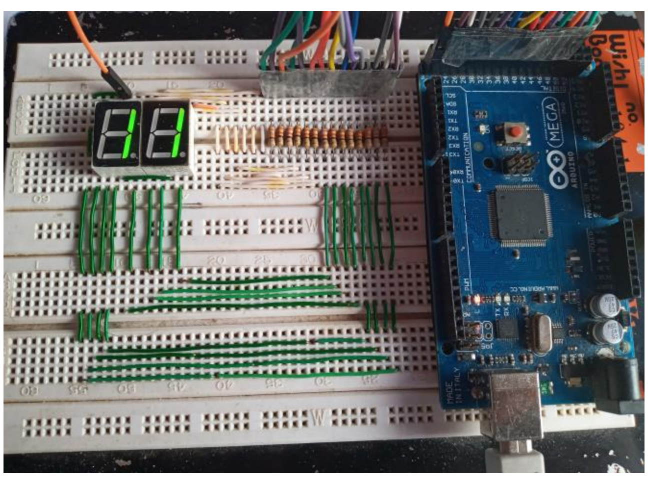

Figure 16. Full system prototype displaying the results for the second proposed scenario.

## V. CONCLUSIONS

In this work, a real-time pattern recognition module using the Matlab interface and an Arduino board is developed. The results illustrate the effectiveness of the proposed system for calculating the addition of the top faces of two dice through a controlled environment.

During the experimental results, it was observed that lighting changes and other effects related to the environment could negatively influence the results. For 33 samples obtained, there were 30 correct and 3 incorrect detections, from which it can be concluded that the proposed methodology is 91% effective. As a future work, the implementation of noise filtering and elimination algorithms will be employed to guarantee greater system robustness.


Figure 17. Pattern Recognition module for the second proposed scenario.

## ACKNOWLEDGMENT

This work is supported by Universidad de Pamplona, Pamplona, Colombia.

## REFERENCES

[1] S. Remedios, D. Armstrong, R. Graham, and L. Fischer. "Exploring the application of pattern recognition and machine learning for identifying movement phenotypes during deep squat and hurdle step movements". Frontiers in Bioengineering and Biotechnology, 8(364), pp. 1-15, 2020.

[2] S. Seo, S. Jo, S. Kim, J. Shim, S. Oh, J. Kim, and D. Kuzum. "Artificial optic-neural synapse for colored and color-mixed pattern recognition". Nature Communications, 9(1), pp. 1-8, 2018.

[3] E. Santiago, and J. Allende. "Diseño de un sistema multiagentes hibrido basado en aprendizaje profundo para la detección y contención de ciberataques". Revista Colombiana de Tecnologías Avanzadas, ISSN: 1692-7257, 2(28), pp. 115-123, 2016.

[4] K. Fu, "Applications of pattern recognition". CRC press. 2019.

[5] J. De Sa, "Pattern recognition: concepts, methods and applications". Springer Science & Business Media. 2012.

[6] L. Márquez, Y. Abdo, and F. Ángulo. "Prototipo de control de acceso a aulas y registro automático de asistencia". Revista Colombiana de Tecnologías Avanzadas, ISSN: 1692-7257, 2(26), pp. 41-47, 2015.

[7] R. Gupta, J. Bera, and M. Mitra, “Development of an embedded system and MATLAB-based GUI for online acquisition and analysis of ECG signal”. Measurement, 43(9), pp. 1119-1126, 2010.
[8] M. Ortiz-Catalan, R. Brånemark, and B. Håkansson, “BioPatRec: A modular research platform for the control of artificial limbs based on pattern recognition algorithms”. Source Code for Biology and Medicine, 8(1), 11, pp. 1-18 2013.
[9] W. Xie, and Y. Wu, “License Plate Automatic Recognition System Based on MATLAB-GUI”. The Open Automation and Control Systems Journal, 6(1), pp. 497-502, 2014.
[10] N. Jagadeesh, and C. Patil, “Iris recognition system development using MATLAB”. International Conference on Computing Methodologies and Communication IEEE, pp. 348-353, 2017.
[11] A. Attenberger and K. Buchenrieder. “An arduino-simulink-control system for modern hand prostheses”. International Conference on Artificial Intelligence and Soft Computing. Springer, pp. 433-444, Cham, 2014.
[12] C. Viejo, S. Fuentes, G. Li, R. Collmann, B. Condé, and D. Torrico. “Development of a robotic pourer constructed with ubiquitous materials, open hardware and sensors to assess beer foam quality using computer vision and pattern recognition algorithms: RoboBEER”. Food Research International, 89, pp. 504-513, 2016.
[13] M. Fannakh, M. Elhafyani, and S. Zouggar. “Hardware implementation of the fuzzy logic MPPT in an Arduino card using a Simulink support package for PV application”. IET Renewable Power Generation, 13(3), pp. 510-518, 2018.
[14] F. Stahl, “Six-sided game dice with playing card indicia”, U.S. Patent No. 5,125,660. Washington, DC: U.S. Patent and Trademark Office. 1992.
[15] J. Giraldo, M. Ruiz, C. Rosero, and L. Zapata, “Formación en competencias específicas para la industria del software colombiano. Experiencias del uso del aprendizaje basado en proyectos”. Revista Colombiana de Tecnologías Avanzadas, ISSN, 1692-7257, 1(27), pp. 7-13, 2016.
[16] B. Hunt, R. Lipsman, and J. Rosenberg, “A guide to MATLAB: for beginners and experienced users”. Cambridge University Press. 2014.
[17] D. Higham and N. Higham, “MATLAB guide”. Society for Industrial and Applied Mathematics. 2016.
[18] M. Margolis, B. Jepson and N. Weldin, “Arduino cookbook: recipes to begin, expand, and enhance your projects”. O'Reilly Media. 2020.