SEj
Vol. 3, N° 1, 2019

Journal of Sciences and Engineering

A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

Movement´s detection of the view to generate orders

[Deteccion de movimientos de la vista para generar ordenes]

Huarote Zegarra Raul Eduardo[a] , Vega Lujan Yensi[b], Romero Valencia Monica Patricia[a]
[a]Universidad Nacional Federico Villarreal – Perú
[b]Universidad Nacional de Trujillo - Perú

raulhuarote@yahoo.es

Received: 28 November 2019; Accepted: 16 December 2019; Published: 30 December 2019

Resumen

El presente artículo de investigación se basa en el uso de la visión computacional como herramienta para detectar las órdenes generadas a partir de movimientos de la vista, para ello también hay que discriminar cuando realiza una orden y cuando no, donde se aprovecha también el procesamiento digital de imágenes. En un computador de regular característica se logró implementar en un software el esquema creado para lograr generar ordenes visuales en el lenguaje Python 3.8.1, tomando como objetos de prueba las 100 ordenes generadas con la vista y tomando estos datos para las pruebas respectivas. Se ha logrado como resultado de manera satisfactoria la generación de ordenes visuales. La prueba realizada dio como resultado la sensibilidad de 0.96, especificidad de 0.98, verosimilidad positiva de 48 y verosimilidad negativa de 0.04. Tener en cuenta que se realizó el testeo de 100 intentos divididos en 10 intentos por cada persona (5 izquierda y 5 derecha)

Palabras clave: Visión computacional, orden visual, movimiento visual

Abstract

This research article is based on the use of computational vision as a tool to detect orders generated from movements of sight, for this we must also discriminate when an order is made and when not, where processing is also used Digital images In a regular feature computer, the scheme created to generate visual orders in the Python 3.8.1 language will be implemented in software, taking as a test object the 100 orders generated with the view and taking this data for the respective tests. The generation of visual orders has been successfully achieved. The test carried out resulted in the sensitivity of 0.96, specificity of 0.98, positive likelihood of 48 and negative likelihood of 0.04. Take into account that the test of 100 attempts divided into 10 attempts by each person (5 left and 5 right) was performed.

Keywords: Artificial neural networks, BackPropagation, prediction

Please cite this article as: Huarote R, Vega Y, Romero M.; Movement´s detection of the view to generate orders, Journal of Sciences and Engineering, vol. 3, no 2, 2019, pp. 13-24. https://doi.org/10.32829/sej.v3i2.113

SEj
Vol. 3, N° 1, 2019
Journal of Sciences and Engineering
A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

1. Introduction

Investigaciones demuestran que las órdenes se generan a partir de movimientos de la vista, considerando que no necesariamente un movimiento de la vista puede ser una orden, por tanto, este problema desprende otro problema como es identificar realmente una orden visual de las que no. Para generar una orden necesariamente se debe analizar el movimiento del iris, por tanto, se debe analizar que funciones y filtros son necesarios para identificar la zona de interés, tales como recorte, suavizamiento, binarización, erosión, dilatación, gradiante de Hough, entre otros. Partiendo desde la captura de la escena a partir de una cámara, hasta la generación de orden visual. Comprobando los resultados estadísticos por la matriz de confusión, expresado por Comber et al. (2017) donde es un estándar para informar sobre la exactitud de datos, tal como se muestra en la Figura 1 donde TP serán los verdaderos positivos, FN los falsos negativos, FP los falsos positivos y TN los verdaderos negativos. Logrando obtener la matriz de sensibilidad, especificidad, razón de verosimilitud positiva y razón de verosimilitud negativa a partir de la ecuación (1), (2), (3) y (4) respectivamente (Fernández & Díaz, 2003)

		Clasificator	
		+	-
True Values	+	TP	FN
	-	FP	TN

Figure 1. Confusion Matrix

$$Sensitibity = \frac{TP}{TP + FN} \quad \text{.....(1)}$$

$$Specificity = \frac{TN}{TN + FP} \quad \text{.....(2)}$$

$$Rv+= \frac{Sensitibity}{1 - Specificity} \quad \text{.....(3)}$$

$$Rv-= \frac{1 - Sensitibity}{Specificity} \quad \text{.....(4)}$$

Given that we are constantly on the move and that objects also move in our visual field, although we are still our eyes and head constantly move to one that is imperceptible to the human eye.

According to Portillo (2012) there are different ways of how to identify the movement according to a sequence of scenes, for example in equation (5) it shows as matrix C that represents the subtraction of two gray-tone matrices of a time t and ti +1, and that this result brings small values that may not necessarily be considered as movements, so the threshold is used, which should be

SEj
Vol. 3, N° 1, 2019
Journal of Sciences and Engineering
A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

in minimum values (<5), having a binary output, where 1 represents a movement and 0 does not. There is also another method that is the movement by image history, where a sequence of n images is captured and a weighted sum evaluation is performed.

$$C = M_{t+1} - M_t \qquad \text{.....(5)}$$

Where:

$$S_{i,j} = \begin{cases} 1 & C_{i,j} > threshold \\ 0 & Opposite\ case \end{cases}$$

Since Louis Emile Javal in France in 1879 observed that to perform the reading involves a series of short stops and quick exits, generating a series of concerns from the year 1900 from this research. Edmund Huey created the first object (special lens) to track the view, later Guy Thomas Buswell created the first non-invasive lens that is nothing more than a camera to record the visual sequence of a specific text, where he enters to carve the Digital image processing for eye tracking.

Being one of the outstanding researchers als Yarbus (1967) taking textually where it says “All the records show conclusively that the character of the eye movement is completely independent or practically depending on the material of the image and how it was done, provided that be it flat or almost flat”, therefore the later works of the movement of the view have a hue of independence in movements. Another opinion of (Mees, Jacobsen, & Göpferich, 2008) makes the follow-up analysis of the view in order to understand and improve the comprehension of text, as well Schilperoord (1996) mentions that “the pauses in the production of text are reflections behavioral of the cognitive processes involved in the exchange of state of attention”.

Currently, vision monitoring has different applications in research, such as Wu et al. (2019) in his research, he identifies vision monitoring according to culture in color preference, taking as an example the Asian culture. In other investigations we find Mehrubeoglu & Nguyen (2018) where he manages to authenticate keys based on the look, specifically in the eye tracking. Also in the patented application of Fayez & Small (2019) where they apply sight tracking to be able to perform game movement.

For the implementation of the source code of the present investigation, it was carried out in the Python 3.8.1 language and the OpenCV 4.2 graphic libraries were used. Given that the language has libraries that are added to Python, but the concern of knowing the operation of the filters and see how tests can be performed, was more motivated to do it manually.

2. Materials and Methods

2.1. Artificial intelligence

According to Huarote (2018) “the way to emulate as much as possible each of the human's own abilities, organizing information, hardware and software technologies for its implementation”. On the other hand, according to Omil (2019) he mentions that artificial intelligence can be defined as “as the ability and capacity of a computer, computer network or network of computer-controlled robots to perform the tasks commonly associated with intelligent human beings”. According to Saavedra (2016), he mentions that “the intelligence function as an element of public policy at the national and strategic level is undergoing important changes within today's global and interdependent society”. According to Nilsson (1998), an intelligent system is a computer program that combines characteristics and behaviors similar to that of human or animal intelligence.

2.2. Computational vision

According to Enrique (2007) “it is the study of these processes, to understand them and build machines with similar capacities”. Therefore, computational vision is responsible for finding information from a scene.

SEj
Vol. 3, N° 1, 2019

Journal of Sciences and Engineering

A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

2.3. Processing and digital analysis of images

According to Morales (2011) “artificial vision or computer vision systems, terminology currently very common use, try to encompass a set of procedures related to the processing and digital analysis of images, which cover a host of mathematical, physical, computational and engineering techniques and tools with applications in numerous fields of modern life”. Also for Gutiérrez (2003) digital image processing is the human perception “the human being uses digital images, either to save them or to modify them, such as remote sensing, medical or photographic images, etc.”. Therefore, it considers for digital images that “They come from spatial and intensity sampling of the optical image. They are formed by an array of elements (pixels)”. This research uses theory, algorithm development, filters and programs to find information.

2.4. Eye follow-up

According to Müller et al. (2016) electrooculography is a technique that records eye movements by analyzing the variations of potentials of the cornea and the retina. For this, the variation of the signal by means of the electrodes makes the movement and direction of the pupil's movement recognized (See Figure 2).

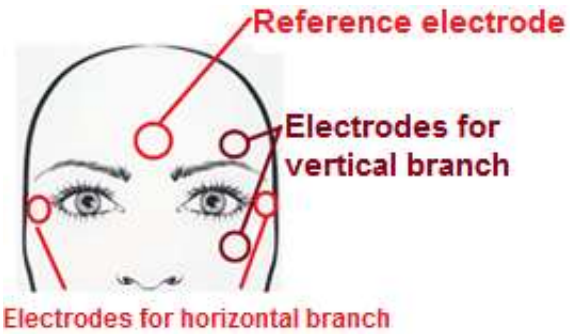

Figure 2. Position of the electrodes for Electrooculography

Another method is the one proposed in this investigation, when the movement of the iris is captured by a camera attached to a computer to calculate the position based on a reference point, which in this case is the left corner, additionally you have to compare the areas white tone (sclera) on the left (A1) with respect to the right (A2), to know where you are looking, obviously this process is done by digital image processing, as shown in Figure 3.

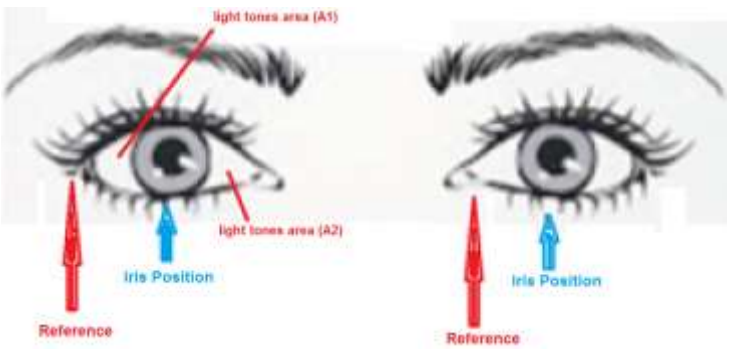

Figure 3: Position of the iris in reference to the corner of the eye

SEj
Vol. 3, N° 1, 2019
Journal of Sciences and Engineering
A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

2.5 Order Generator

To (BienSalud, 2015) generate orders is the proper functioning of the human organism and that not only depends solely on the orders sent by the brain through the nervous system. For the order to be given effectively, it must necessarily be carried out by means of a constant flow of information both transmitted and received by the organic system performing its functions in an appropriate manner.

To generate the orders part of the brain to the rest of the body, which is a function of the set of necessary actions with the intention of giving a message to the organ capable of carrying it out, through the channel that is the nervous system. Having as a result the generation of visual orders. For (Querelle, 2018) he has an opinion regarding the generation of orders “it is the movement has to do with the sensation of rapid displacement, such as seeing a motorcycle or a car at high speed, but it is caused by an invisible effect, which acts over bodies, called strength”. Therefore, generating an order is a physical event linked to the neuronal reaction and transmitted by a system that acts as a channel to carry out that order.

3. Results

Proposal of the model to detect movements of the view to generate orders

The model to be implemented is given by a sequence of steps, to find generate the order, as shown in Figure 4, starting from the acquisition of the scene by a camera, followed, it is necessary to resize to 800x600 to have a standard dimension to process the image, followed by converting the resized scene that is in RGB to gray, followed by the haarcascade with respect to the face (with the haarcascade_frontalface_default.xml) to reduce the scene to process, followed by continuing to reduce the scene by finding the eyes (with the haarcascade_lefteye_2splits.xml), before locating the eye scene it is necessary to improve the image using the medium filter (medianBlur), from this step the Hough transform is used, ideal to find a circle within the scene that in this case is the iris, once the iris is located it is necessary to know the amount of white tones found on its left and right, with this information can be generated a histogram of white tones, from the histogram with the statistical method of the correlation coefficient a line can be generated, once you have the line you can find the slope, once the slope has to be taken count the positive or negative values, for this two thresholds are considered to be able to consider that an order is being given, these threshold values have been obtained by experiment, being U1 = -0.1 and U2 = 0.2, so if it is less than U1 is considered an order to the left and if it is greater than U2 it is considered to be an order to the right, as the amount of left and right values is constant along the path of each frame, it is necessary to perform an evaluation of orders, creating a variance (Cartusia, 1998) as seen in equation (6), so if you have a variance of 0 from 15 orders, this indicates that you are really going to give an order (left or right) as the case may be) and also discriminate against a simple look (or that is not an order generated by sight).

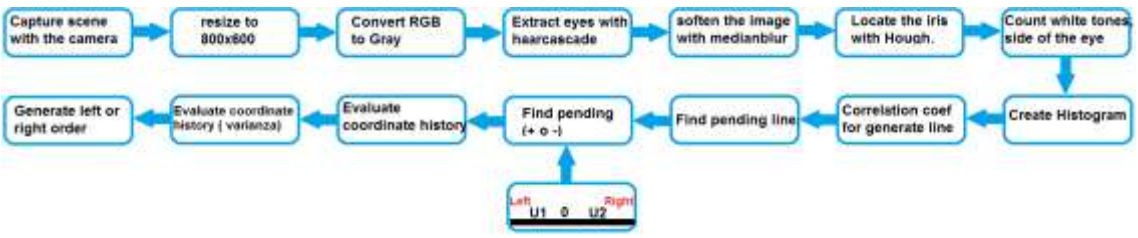

Figure 4: Sequence of steps to generate order from view

SEj
Vol. 3, N° 1, 2019

Journal of Sciences and Engineering

A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

$$S_X^2 = \frac{\sum_{i=1}^{N}(X_i - \overline{x})^2}{N-1} \qquad \text{.....(6)}$$

Algorithm to generate the order

```
MIENTRAS Fin de escena
        RGB = capturar_escena
        RGB = redimensionar(RGB,800,600)
        Gray = convertir_a_gris(RGB)
        Faces = Buscar_rostro_harracascade(Gray)
        DESDE f HASTA Faces HACER
        Ojos= Buscar_ojos_harracascade(Gray)
                DESDE o HASTA Ojos HACER
        Escena = mejorar_escena_blured(o)
        EncontrarIris(x,y) = hough(escena)
        Matriz=Encontrar_area_blanco_izquierda_derecha(x,y)
        Histograma = Crear_histograma(Matriz)
        Linea = Crear_linea(Hiatograma)
        M = encontrar_pendiente(Linea)
        Si m<-0.1
        Agregarlista('i')
        Si m>0.2
        Agregarlista('d')
        Si tam_lista>20
        Si varianza(lista)=0
        Orden = lista[0]
        Borrar_lista
        FIN DESDE
FIN MIENTRAS
```

Analyzing the algorithm based on computational vision to generate visual order, in the first line it indicates to travel the eyes found in the scene, in the second line it is reduced from the scene of the eyes to the position specifically the eye (not the eyebrows), the same for the third line, but in RGB, in the fourth line the Hough transform is applied, which is nothing more than a circle in the circle of any part found in that scene, bone finding the iris, in the fifth line it evaluates if there is at least one circle, if it exists in the sixth line, it rounds the values of the circle, in the seventh it goes through how many circles there are, in the eighth line it evaluates whether the radius of that circle is greater than 10, if so in the ninth line it captures the height and width, so that in line 10 I evaluated the consistency of the values of width, height, position, etc. and on line 11 highlight the iris found. This is visualized in figure 5.

SEj
Vol. 3, N° 1, 2019
Journal of Sciences and Engineering
A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

```
for (ex,ey,ew,eh)in eyes:
    imgGrayencirculado = roi_gray[ey+int(eh/4):ey+int(3*eh/4)+20, ex:ex+ew]
    imgColorencirculado = roi_color[ey+int(eh/4):ey+int(3*eh/4)+20, ex:ex+ew]
    circles=cv2.HoughCircles(imgGrayencirculado,cv2.HOUGH_GRADIENT,1,20,param1=50,param2=30,minRadius=7,maxRadius=60)
    if circles is not None:
        circles = np.uint16(np.around(circles))
        for i in circles[0,:]:
            if i[2]>10:
                hh,ww = imgGrayencirculado.shape
                if (imgGrayencirculado.size>0 and len(imgGrayencirculado)>0 and hh>0 and ww>2*i[2] and hh>2*i[2] and i[1]-i[2])>0 and (i[0]-i[2])>0 and hh>2*i[2] and ww>2*i[2]:
                    cv2.circle(imgColorencirculado,(i[0],i[1]),i[2],(0,0,255),1)
```

Figure 5. Source code to find the iris

Tests were given giving encouraging results, the evaluation of the orders was carried out with 10 people, trying to give orders both to the left and to the right, the data obtained are displayed in table 1.

Table 1: Data collection for the evaluation of results

Person	Intent	Total intents	TP	FN	TN	FP
1	Left	5	2	0	3	0
	Right	5	2	0	3	0
2	Left	5	3	0	2	0
	Right	5	2	0	3	0
3	Left	5	2	1	2	0
	Right	5	2	0	3	0
4	Left	5	2	0	3	0
	Right	5	3	0	2	0
5	Left	5	2	0	3	0
	Right	5	2	0	2	1
6	Left	5	2	0	3	0
	Right	5	3	0	2	0
7	Left	5	2	0	3	0
	Right	5	2	0	3	0
8	Left	5	2	0	3	0
	Right	5	2	1	2	0
9	Left	5	3	0	2	0
	Right	5	2	0	3	0
10	Left	5	2	0	3	0
	Right	5	2	0	3	0
			44/100	2/100	53/100	1/100

Applying equation (1), the sensitivity of

$$Sensitibity = \frac{TP}{TP + FN} = \frac{\frac{44}{100}}{\frac{44}{100} + \frac{2}{100}} = 0.96$$

SEj
Vol. 3, N° 1, 2019
Journal of Sciences and Engineering
A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

Applying equation (2) the specificity of

$$Specificity = \frac{TN}{TN + FP} = \frac{\frac{53}{100}}{\frac{53}{100} + \frac{1}{100}} = 0.98$$

Applying equation (3), the positive likelihood ratio of

$$Rv+= \frac{Sensitibity}{1 - Specificity} = \frac{0.96}{1 - 0.98} = 48$$

Applying equation (4), the negative likelihood ratio of

$$Rv-= \frac{1 - Sensitibity}{Specificity} = \frac{1 - 0.96}{0.98} = 0.04$$

Next, it shows in Figure 6 the results generated by applying the source code made to a person of the 100.

SEj
Vol. 3, N° 1, 2019

Journal of Sciences and Engineering

A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

```
>>>
======== RESTART: I:/SraArt.py ========

Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7290785674704933
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7290785674704933
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7290785674704933
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Derecha Prom_m =  1.0  M =  0.0
Centro Prom_m =  1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
Izquierda Prom_m =  -1.0  M =  0.0
Centro Prom_m =  -1.0  M =  0.7063197941229304
>>>
```

Figure 6. Python result with OpenCv to generate order from view

Figure 7 shows the results of the evaluation of the 10 people, and a total of 100 attempts (10 attempts each), showing in blue and lead the correct values when performing the tests, indicating that there are more results correct, both for the orders generated and the orders not generated, in a small proportion the values are shown in which it represents an error in the estimate and the evidence (represented in yellow and orange). Therefore the results shown are encouraging. These data are being obtained from table 1 which are of the 20 tests performed by people.

SEj
Vol. 3, N° 1, 2019
Journal of Sciences and Engineering
A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

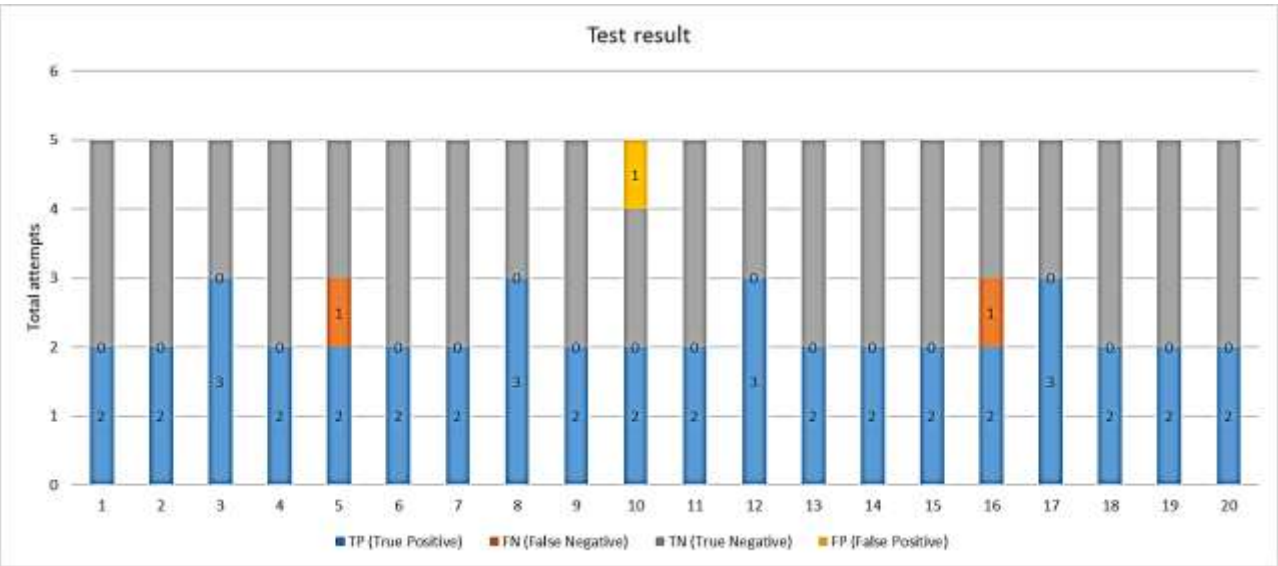

Figure 7. Results after the evaluation of results in Table 1.

Source code

To carry out the process of generating order from the view it has been necessary to perform the coding in a language that allows working with OpenCv functions and filters, this to separate each of the functions of digital image processing as shown in the Figure 8, taking into account that functions have been created to be able to have the calculation accurately, showing the complete source code of the tests performed to obtain the results of Table 1.

SEj
Vol. 3, N° 1, 2019

Journal of Sciences and Engineering

A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

```
from PIL import Image
import numpy as np
import cv2
from playsound import playsound
def pendiente(y):
    x = np.arange(len(y))
    return np.corrcoef(x, y)[0,1]
def varianza(y):
    return np.std(y)**0.5

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_lefteye_2splits.xml')
cap = cv2.VideoCapture('prueba2.mp4')
historiaX = []
historiaMLineas = []
estadoAnterior ="Centro"
estadoActual = estadoAnterior
while (cap.isOpened()):
    ret, img = cap.read()
    img = cv2.resize(img, None, fx=0.25, fy=0.25, interpolation=cv2.INTER_AREA)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3,5,minSize=(30, 30))
    for (x,y,w,h) in faces:
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = img[y:y+h, x:x+w]
        eyes = eye_cascade.detectMultiScale(roi_gray)
        for (ex,ey,ew,eh)in eyes:
            imgGrayenCirculado = roi_gray[ey+int(eh/9):ey+int(3*eh/4)+20, ex:ex+ew]
            imgColorenCirculado = roi_color[ey+int(eh/4):ey+int(3*eh/4)+20, ex:ex+ew]
            kernel = np.ones((5,5),np.uint8)
            circles=cv2.HoughCircles(imgGrayenCirculado,cv2.HOUGH_GRADIENT,1,20,param1=50,param2=30,minRadius=7,maxRadius=50)
            if circles is not None:
                circles = np.uint16(np.around(circles))
                for i in circles[0,:]:
                    if i[2]>10:
                        hh,ww = imgGrayenCirculado.shape
                        if (imgGrayenCirculado.size>0 and len(imgGrayenCirculado)>0 and hh>0 and ww>2*i[2] and hh>2*i[2] and i[1]-i[2]>0 and (i[0]-i[2])>0
                        and hh>2*i[2] and ww>2*i[2]:
                            cv2.circle(imgColorenCirculado,(i[0],i[1]),i[2],(0,0,255),1)

                            histo = imgGrayenCirculado[(i[1]-i[2]):(i[1]+i[2]), (i[0]-i[2]):(i[0]+i[2])]

                            if len(histo)>20 and i[1]-i[2]<i[1]+i[2] and i[0]-i[2]<i[0]+i[2]:
                                histo = cv2.erode(histo,kernel,iterations = 6)
                                histo = cv2.dilate(histo,kernel,iterations = 6)

                                ret,histo = cv2.threshold(histo,50,255,cv2.THRESH_BINARY_INV)
                                histo2 = histo.copy()
                                suma = np.sum(histo2//255,axis=1)
                                while suma[0]==0:
                                    histo2 = np.delete(histo2,(0),axis=0)
                                    suma = np.sum(histo2//255,axis=1)
                                histo2 = np.flip(histo2,axis=0)
                                suma = np.sum(histo2//255,axis=1)
                                while suma[0]==0:
                                    histo2 = np.delete(histo2,(0),axis=0)
                                    suma = np.sum(histo2//255,axis=1)
                                histo2 = np.flip(histo2,axis=0)

                                suma = np.sum(histo2//255,axis=0)
                                maximo = np.max(suma)
                                for i in range(len(suma)):
                                    cv2.line(img, (2*i+10, 50-(maximo - suma[i])), (2*i+10, 50), (0,0,255))
                                    cv2.line(img, (2*i+11, 50-(maximo - suma[i])), (2*i+11, 50), (0,0,255))

                                m = pendiente(suma)
                                historiaMLineas.append(m)
                                if len(historiaMLineas)>15:
                                    historiaMLineas.pop(0)
                                    mp = np.median(historiaMLineas)
                                    cv2.line(img, (10, 100-int(35*mp)), (70, 100-int(35*mp)+100), (0,255,0))
                                    cv2.putText(img,"m="+str(round(mp,4)),(10,120), cv2.FONT_HERSHEY_SIMPLEX, 0.4,(0,255,0),1,cv2.LINE_AA)

                                v = varianza(suma)
                                umbralDerecha = 0.2
                                umbralIzquierda = -0.1
                                if m>umbralDerecha:
                                    historiaM.append(1)
                                else:
                                    if m<umbralIzquierda:
                                        historiaM.append(-1)
                                    else:
                                        historiaM.append(m)

                                M = 0
                                if len(historiaM)>15:
                                    historiaM.pop(0)

                                    M = varianza(historiaM)
                                    if M==0 and np.median(historiaM)>0:
                                        estadoActual ="Derecha"

                                    if M==0 and np.median(historiaM)<0:
                                        estadoActual ="Izquierda"

                                    if M>0.4:
                                        estadoActual ="Centro"

                                    if (estadoActual!=estadoAnterior):
                                        print(estadoActual,"Pron_m = ",np.median(historiaM)," M = ",M)
                                        estadoAnterior = estadoActual
                                        if estadoActual=="Izquierda":
                                            pts = np.array([[10,200],[20,180],[30,190],[50,190],[90,210],[20,210],[20,220]], np.int32)
                                            pts = pts.reshape((-1,1,2))
                                            cv2.fillPoly(img, np.int_([pts]), (0, 255, 0))
                                            playsound('izquierda.mp3')
                                        if estadoActual=="Derecha":
                                            pts = np.array([[10,190],[40,190],[40,200],[40,220],[40,210],[10,210]], np.int32)
                                            pts = pts.reshape((-1,1,2))
                                            cv2.fillPoly(img, np.int_([pts]), (0, 255, 0))
                                            playsound('derecha.mp3')
    cv2.imshow('img',img)
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break
cap.release()
cv2.destroyAllWindows()
```

Figure 8. Source code to perform the visual order generation process

SEj
Vol. 3, N° 1, 2019

Journal of Sciences and Engineering

A publication of
CINCADER
Centre of Research and Training for Regional Development
Online at www.journals.cincader.org

ISSN 2523-9503
DOI: https://doi.org/10.32829/sej.v3i2.113

4. Conclusions

It is demonstrated that the generation of visual orders can be performed using digital image processing as a tool, with the necessary filters provided by Opencv.
The results were encouraging, as it resulted in a sensitivity of 0.96, which represents 96% effectiveness, a specificity of 0.98, also found the positive likelihood ratio of 48 and a negative likelihood ratio of 0.04, the latter having a value close to 0.

References

BienSalud, B. 2015. Mensajes del cerebro al resto del cuerpo. http://www.revistabiendesalud.com/general/mensajes-del-cerebro-al-resto-del-cuerpo/

Cartusia, L. 1998. Bioestadística — Métodos y Aplicaciones. Malaga: U.D. Bioestadística - Facultad de Medicina.

Comber, A., Fisher, P., Brunsdon, C., & Khmag, A. 2017. Spatial analysis of remote sensing image classification accuracy. Remote Sensing of Environment, 237-246.

Enrique, S. L. 2007. Vision Computacional. Neuherberg, Alemania: Monterrey.

Fayez, F., & Small, D. 2019. INTERACTING WITH GAME ELEMENTS USING EYE MOVEMENT TRACKING. US Patent & Trademark Office.

Fernández, P. S., & Díaz, P. S. 2003. Pruebas diagnósticas. Coruña (España): Unidad de Epidemiología Clínica y Bioestadística.

Gutiérrez, E. A. 2003. Procesamiento digital de imagen. España: Universidad de León.

Huarote Zegarra, R. E. 2018. Optimizacion de ruta corta usando algoritmo genetico generacional. Lima-Perú: Ciencia & Desarrollo.

Jenkins, B. A. 1970. Time Series Analysis Forecasting, and Contro. San Francisco: Soc. Ind. Appl. Math.

Mees, I. M., Jacobsen, A. L., & Göpferich, S. 2008. Looking at Eyes: Eye-tracking Studies of Reading and Translation Processing. Frederiksberg: eBook Collection (EBSCOhost).

Mehrubeoglu, M., & Nguyen, V. 2018. Real-time eye tracking for password authentication. IEEE.

Morales, R. R. 2011. Procesamiento y analisis digital de imagenes. Mexico: RA-MA.

Müller, J., Wendt, D., Kollmeier, B., & Brand, T. 2016. Comparing eye tracking with electrooculography for measuring individual sentence comprehension duration. PLoS ONE 11.

Nilsson. 1998. Artificial Intelligence: A New Synthesis. EEUU: Morgan Kaufmann Publishers, Inc.

Omil, J. C. 2019. Inteligencia artificial ¿Dr. Jekyll o Mr. Hyde? Mercados y Negocios, 7.

Portillo, J. P. 2012. Detección de movimiento de objetos mediante secuencias de vídeo. Instituto Politécnico Nacional de México.

Querelle. 2018. Profesor en linea. Obtenido de http://www.profesorenlinea.cl/

Saavedra, B. 2016. Inteligencia Estratégica en un mundo globalizado en Latinoamérica: Retos y desafíos en el siglo XXI. Washington: Revista Policía y Seguridad Pública.

Schilperoord, J. 1996. It's about Time: Temporal Aspects of Cognitive Processes in Text Production. Amsterdam: Rodopi.

Wu, B., Nishimura, S., Jin, Q., & Zhu, Y. 2019. Culture-Based Color Influence Paths Analysis by Using Eye-Tracking Devices. IEEE.

Yarbus, A. L. 1967. Eye Movements and Vision. Plenum Press.