# RECONOCIMIENTO FACIAL DE VIDEO EN STREAMING UTILIZANDO PYTHON 3 Y OPEN CV

Como aplicar la librería OPEN CV para el reconocimiento de rostros

Jeison Robles Arias

roblesjeison@gmail.com

***Resumen***

Este trabajo explora las funcionalidades de la librería de código abierto OPEN CV en el análisis y procesamiento de imágenes secuenciales para la consignación de patrones e identificación de rostros con varianza en condiciones de luz, modificaciones en las expresiones faciales y siempre en capturas frontales con un gran rango de gestos y ángulos de captura. El trabajo se desarrolla en Python 3 y utilizando Spyder 4 de anaconda.

*Palabras Clave:* Open CV, Reconocimiento Facial.

***Abstract***

This work explores the functionalities of the open source library OPEN CV in the analysis and processing of sequential images for the consignment of patterns and identification of faces with variance in light conditions, modifications in facial expressions and always in frontal shots with a large range of gestures and capture angles. The work is carried out in Python 3 and using Spyder 4 from anaconda.

*Keywords:* Open CV, facial recognition.

## I. INTRODUCCIÓN

El reconocimiento facial en la actualidad se ha vuelto una herramienta indispensable en un mundo cada vez mas movido, globalizado y tecnológico y en el cual la detección ágil y precisa de la identidad de usuarios es cada vez mas necesaria. Transacciones bancarias, mitigaciones de fraudes electrónicos, suplantación de identidad, pagos electrónicos, ignición de automotores, validaciones aeroportuarias, son solo algunas de las aplicaciones en las que el reconocimiento facial ha incursionado. A lo largo de los últimos anos y con el crecimiento de la capacidad de procesamiento de imágenes cada vez mas detalladas gracias al poder computacional, se ha abierto una basta carta de posibilidades.

## II. METODOLOGÍA

Para este trabajo se explora la diferencia entre la identificación facial y el reconocimiento facial utilizando la herramienta: Local Binary Patterns Histograms de CV2.

Se comienza con un trabajo modular de extracción de identidad bibliometría en un simple ensayo de lectura y extracción de rostros utilizando un tope de 500 archivos de comparación, los cuales son en el momento escalados debido al peso del procesamiento de imágenes y las “n” dimensiones que las componen, se selecciona una matriz escalada a 150 por 150 pixeles, con lo que se pierde calidad pero se mejora el rendimiento del futuro modelo. Este problema es fácilmente solucionable al utiliza herramientas que corran sobre GPUs como Google Colaboratory.

Por conceptos de orden, el proyecto es almacenado en una carpeta especifica y se utiliza una carpeta “Data” que contendrá toda la información de individual de cada sujeto. Para este ejemplo se muestra el panorama de la carpeta DATA:

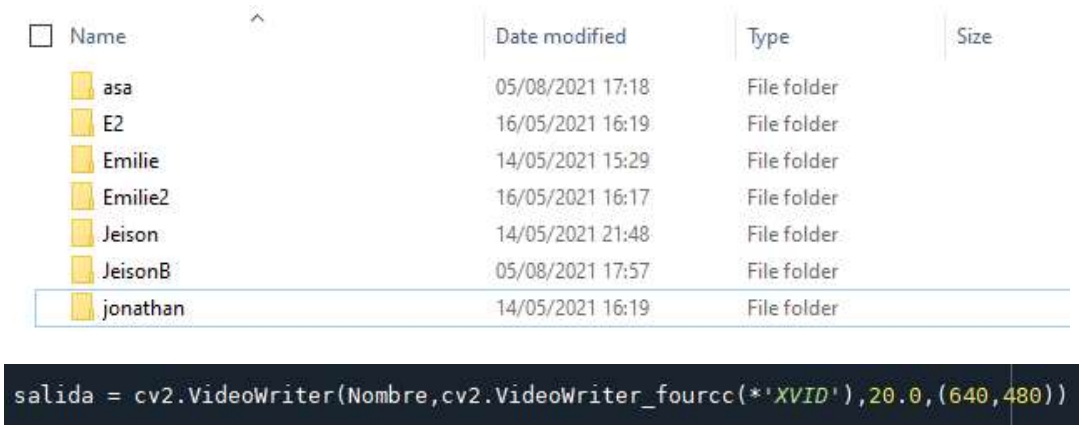


*Figura 1. Rutas creadas mediante OS Python 3 y VideoWriter() de CV2.*

El nombre de la carpeta es fácilmente editable mediante el uso del modulo Os de Python, el cual proporciona una forma portátil de utilizar la funcionalidad dependiente del sistema operativo en creación de carpetas y rutas de datos que posteriormente serán accesadas por los modelos en su procedimiento de entrenamiento.

Ademas se explora paso a paso una base de Python para la extracción y lectura de videos, los cuales pueden ser grabados o vía streaming utilizando un periférico (cámara de video).

Durante la etapa de extracción de videos o lo que en el mundo de la biometria se conoce como empadronamiento de un sujeto, se realiza la identificación de rostros mediante modelos de histogramas binarios y los mismos se almacenan en una ruta especifica, la cual debe ser pura, refiriendo a que no debe contener información basura de otros rostros o ser ejecutado en un ambiente de luces muy distinto al ambiente de identificación una vez se encuentre en producción.

Se expone el diagrama de bloques utilizado para resolver este trabajo, siendo la explicación anterior, la correspondiente al primer bloque.

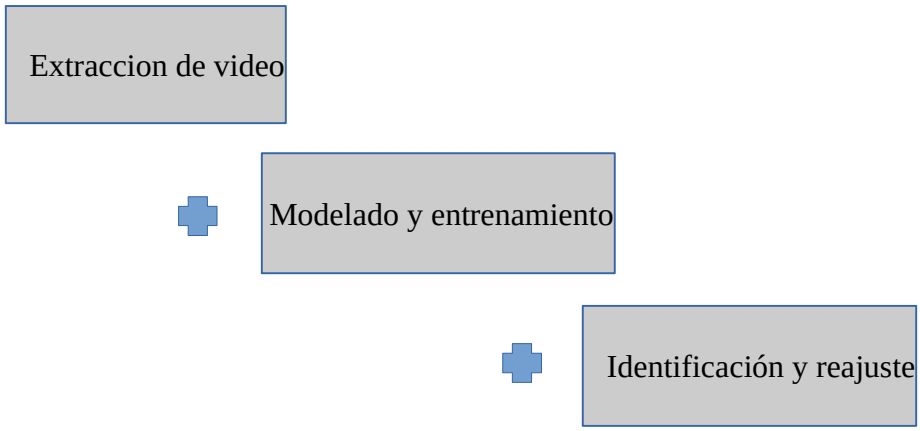


*Figura 2. Diagrama de islas de alto nivel de macro procesos.*

Las librerías utilizadas durante el desarrollo de este proyecto son las siguientes:

```
import cv2
import os
import imutils
```

*Figura 3. Librerías utilizadas en Python 3.*

Todas siendo previamente instaladas y comprobadas en el ambiente “Spyder”

Extractor de video de entrada vía Streaming

```
def extraccionDeRostros(personName):
    if not os.path.exists('Rostros encontrados'):
        print('Carpeta creada: Rostros encontrados')
        os.makedirs('Rostros encontrados')


    dataPath = 'C:\Jeison\Python\RECONOCIMIENTO FACIAL\Data'#Cambia a la ruta donde hayas almacenado Data
    personPath = dataPath + '/' + personName

    if not os.path.exists(personPath):
        print('Carpeta creada: ',personPath)
        os.makedirs(personPath)


   cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)

    #cap = cv2.VideoCapture('ATM1.mp4')

    faceClassif = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    count = 0

    while True:

        ret, frame = cap.read()

        if ret == False:

            print ('me sali en count :' ,count)
            break

        frame =  imutils.resize(frame, width=640)
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        auxFrame = frame.copy()
        faces = faceClassif.detectMultiScale(gray,1.3,5)

        for (x,y,w,h) in faces:
            print(count)

            cv2.rectangle(frame, (x,y),(x+w,y+h),(0,255,0),2)
            rostro = auxFrame[y:y+h,x:x+w]
            rostro = cv2.resize(rostro,(150,150),interpolation=cv2.INTER_CUBIC)

            #Escribo en carpeta el cuadro sacado
            cv2.imwrite(personPath + '/rotro_{}.jpg'.format(count),rostro)

            count = count + 1

        #abre el framework view
        cv2.imshow('frame',frame)



        k =  cv2.waitKey(1)

        if k == 27 or count >= 500:

            break

    cap.release()
    cv2.destroyAllWindows()


extraccionDeRostros('JeisonB')
```

*Figura 4. Codigo inicial de almacenamiento de video y reconocimiento facial*

El resultado de este código se puede verificar directamente en la ruta especificada y la carpeta **“.../DATA/”+Sujeto**

Las matrices de pixeles se pueden visualizar en dicha ruta, se presentan algunos ejemplos del producto:

Tome en cuenta que se realizaron los empadronamientos en condiciones, emociones, edades y géneros distintos

*Figura 5. Matrices Fotogramas de 150 px por 150 px*

Cada una de estas matrices sera estudiada pixel por pixel mediante el método de histograma de patrones binarios de lo cual se presenta un ejemplo de su funcionamiento. Este trabajo es muy similar en lo que se puede estudiar en *redes neuronales convolucionales* para el procesamiento de de imágenes y sus "n" dimensiones de análisis.

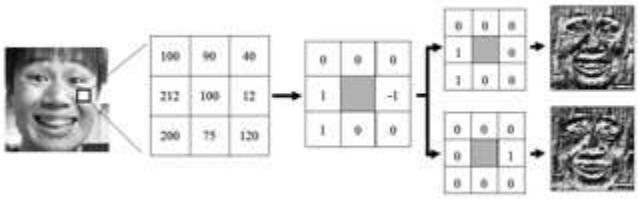


*Figura 6. Ejemplo básico de funcionamiento de computer vision  por patrones binarios*

**Entrenamiento del modelo:**

El modelo es entrenado mediante el siguiente código, el cual realiza tantas iteraciones como fotogramas encuentre en el espacio matricial de las rutas almacenadas:

```
def entrenador():

    dataPath = 'C:\Jeison\Python\RECONOCIMIENTO FACIAL\Data'#Cambia a la ruta donde hayas almacenado Data
    peopleList = os.listdir(dataPath)


    print('Lista de personas: ', peopleList)

    labels = []
    facesData = []
    label = 0
    for nameDir in peopleList:
        personPath = dataPath + '/' + nameDir
        print('Leyendo las imágenes')
        for fileName in os.listdir(personPath):
            print('Rostros: ', nameDir + '/' + fileName)
            labels.append(label)
            facesData.append(cv2.imread(personPath+'/'+fileName,0))
            # image = cv2.imread(personPath+'/'+fileName,0)
            # cv2.imshow('image',image)
            #cv2.waitKey(10)
        label = label + 1



    face_recognizer = cv2.face.LBPHFaceRecognizer_create()
    print("Entrenando...")
    face_recognizer.train(facesData, np.array(labels))
    face_recognizer.write('modeloLBPHFace.xml')
    print("Modelo almacenado...")
```

*Figura 7. Codigo de entrenamiento de modelo con image show comentado para velocidad de calculo computacional*

Al ejecutarse el mismo recorre todo el arreglo de fotogramas previamente capturados y los somete a un estudio de patrones mediante el uso del metodo: **Local Binary Patterns Histograms de CV2**

Por conveniencia y fines de ejemplo se aplica un "image show" para desplegar el fotograma bajo estudio, este paso se podria evitar para optimizar el rendimiento de calculos y limitar el procesamiento al analisis de patrones binarios.

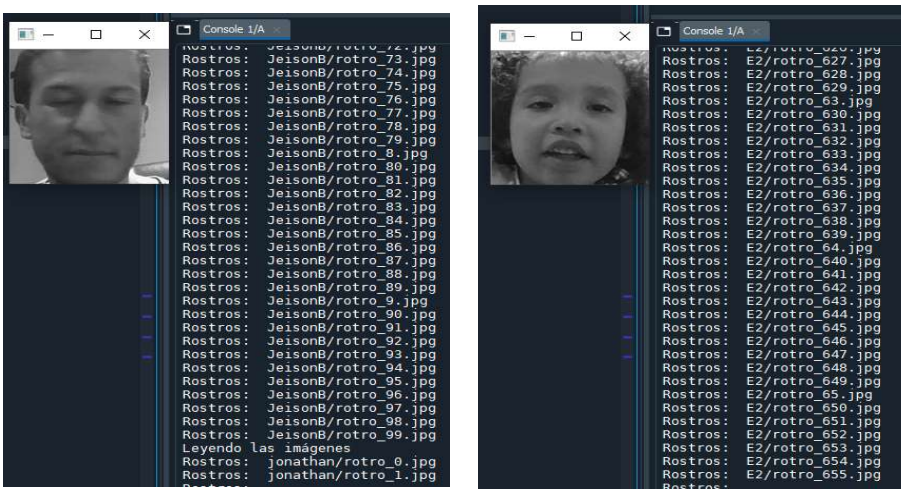


*Figura 8. Extracción de entrenamiento de los patrones binarios para dos sujetos distintos. Entrenamiento sobre 1500 fotogramas.*

Como se observa en las capturas superiores, se tiene un indicador para comprender el proceso y el punto por el que se lleva el entrenamiento. Se trabaja de esta forma como una buena practica para validar la simetría de los datos y garantizar que el calculo se realice de forma homogéneo.

**Reconocimiento Facial:**

La etapa final del proceso cuenta con una interfaz de video que refleja la visión del computador y que básicamente se utiliza como una guía visual al usuario para representar la identificación de su nombre. Es claro que para ambientes de producción, esta interfaz de salida puede evitarse y simplemente utilizar el dato promedio de reconocimiento como una condición en toma de decisiones.

Decisiones como apertura de puertas, desbloqueo de información, permitir el "Log In" en diferentes aplicaciones, entre muchos otros. Otra razón para evitar esta interfaz es debido al costo incremental que se tendría en producción en masa, las pantallas pueden utilizarse unicamente en las estaciones de empadronamiento, las cuales tienden a ser menor cantidad.
Continuando con e ejemplo se muestran capturas del modulo de reconocimiento facial, el cual esta diseñado para colocar un recuadro, que puede ser pintado de colores verde o rojo, de acuerdo a si se reconoce el rostro o no respectivamente. En caso de un reconocimiento positivo, se lee de la base de datos de empadronamiento este dato y despliega su información.
Es claro que a nivel de industria, los datos de escritura y consulta pueden ser muchos mas, proporcionando información expedita y certera a la hora del reconocimiento.
Alguna información usual podría ser:

- Departamento
- Puesto

- Nivel de autorización
- Horarios
- Alertamientos

Como puede notarse, al solucionar el problema de empadronamiento y reconocimiento, puede tenerse la información deseada y acceder a la misma mediante los datos biometricos.

Se presentan ejemplos de capturas y su respectivo reconocimiento:

Rostros empadronados (Umbral de aceptación definido en 60)

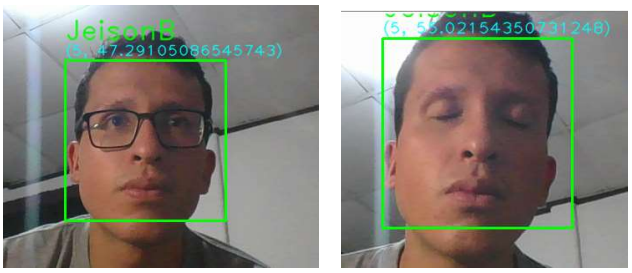


*Figura 9. Reconocimiento positivo de sujeto adulto en distintas tomas*

Rostro no reconocido:

Este ejemplo fue tomado previamente al entrenamiento con fines de mostrar la construcción ante una identificación negativa: Se supera la banda de aceptación de 60.


*Figura 10. Reconocimiento negativo de sujeto femenino sin entrenamiento con medida 66.*

Luego del entrenamiento:


*Figura 11. Reconocimiento positivo de sujeto femenino post entrenamiento al suministrar 1500 epochs y 1500 batch para mayor estabilidad.*

# III. RESULTADOS Y DISCUSIÓN

Los resultados obtenidos en general son moldeables de acuerdo a las necesidades del negocio en el que se desee operar, por ejemplo, para aplicaciones de prevención de fraudes y de seguridad se debe aumentar la cantidad de fotogramas en la ruta de estudio con la finalidad de aumentar la fiabilidad del modelo, proporcionando miles de micro variaciones en los histogramas creados. Por esta razón siempre debe tenerse claro el alcance y equilibrar el costo beneficio del algoritmo de aprendizaje de acuerdo a la necesidad.

En general para un proceso de empadronamiento con 1500 fotogramas se requieren entre 3 y 5 minutos para completar el proceso, esto es variable de acuerdo a la necesidad.

El reconocimiento facial, lo que llamamos consulta, se logra entre los 0.2s y 1s (Retraso aceptable de identificación)

El umbral de validación se puede modificar de acuerdo a las necesidades.

Siempre puede incursionarse con algoritmos de un espectro mas alto tales como:

EigenFaces: EigenFaces en OpenCV, cv2.face.EigenFaceRecognizer_create()

FisherFaces: FisherFaces en OpenCV, cv2.face.FisherFaceRecognizer_create()

Para este proyecto fue utilizada una funcion de grabacion y almacenamiento del video, con finalidades de mantener un orden a lo largo del proyecto, sin embargo, siempre y cuando las capacidades del equipo de computo o server lo permita, puede unificarse la extraccion de fotogramas con el almacenamiento de los mismos, de esta forma se logra evitar el procedimiento de escritura de los video, lo que por general tienen a pesar unos 20 – 30 MB.

Emilie Robles Villanueva

| | |
|---|---|
| Location: | C:\Jeison\Python\RECONOCIMIENTO FACIAL |
| Size: | 24,6 MB (25.803.484 bytes) |
| Size on disk: | 24,6 MB (25.804.800 bytes) |

*Figura 12. Video y espacio consumido*

Claramente almacenar el video incrementaría mucho los requerimientos debido a su peso y el volumen de sujetos.

Los 1500 fotogramas almacenados tienen un equivalente de 12 MB, sin embargo al generar el modelo de datos, estos se pueden archivar o comprimir en otros sitios ajenos al proyecto.

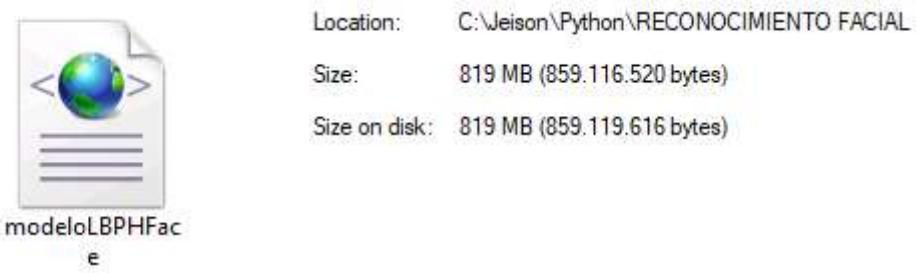


*Figura 13. Modelo almacenado y espacio consumido*

## IV. CONCLUSIONES

En general se puede concluir que computer visión tiene un efecto importante en la tecnología y soluciones ingenieriles actuales, para este caso es importante contar con librerías potentes dedicadas al diseño y producción de algoritmos específicos de visión por computador.

Se puede ademas concluir que el método de calculo de histograma por patrones binarios es de un uso amplio y efectivo en la determinación de rostros, este método es importante utilizarlo durante las dos etapas relacionadas a la visión, como lo son el reconocimiento de rostros y la identificación de la identidad de dicho rostro.

## V. REFERENCIAS

O.C.V. (s. f.). *OPENCV*. https://opencv.org/. Recuperado 5 de agosto de 2021, de https://opencv.org/

*¿Cómo calcular los histogramas de patrones binarios locales con OpenCV?* (s. f.). https://www.it-swarm-es.com/es/opencv/como-calcular-los-histogramas-de-patrones-binarios-locales-con-opencv/1070480484/. Recuperado 5 de agosto de 2021, de https://www.it-swarm-es.com/es/opencv/como-calcular-los-histogramas-de-patrones-binarios-locales-con-opencv/1070480484/