



Instituto Politécnico Nacional



Escuela Superior de Ingeniería Mecánica y Eléctrica

Ingeniería en Comunicaciones y Electrónica

PROYECTO: Cerradura automatizada mediante reconocimiento facial y transmisión en vivo por red de área local

NOMBRE

Willy Martinez Hernández

No. BOLETA

2014030808

PROF.GALICIA GALICIA ROBERTO

FECHA DE ENTREGA: 18 de junio 2021

## **Planteamiento del problema**

Con el avance creciente de la tecnología hoy en día se necesitan nuevas soluciones a ciertas problemáticas y requerimientos que plantean estos cambios sociales y las nuevas tendencias de nuestra forma de vida, facilitando el diseño de casas y hogares más humanos, más personales, polifuncionales y flexibles, con este proyecto se busca crear una solución actual y tecnológica mediante la implementación Un sistema domótico que es capaz de recoger información proveniente de un sensor o entrada, procesarla y emitir órdenes a unos actuadores o salidas. El sistema puede enviar información a la red de área local de comunicación.

## **Objetivo**

Se desarrollarán las habilidades necesarias para desarrollar e implementar un proyecto mediante IoT

## **Objetivos particulares**

Desarrollar un sistema de control para una cerradura eléctrica mediante reconocimiento facial que indique al usuario si puede entrar o no y adicionalmente se puede monitorear mediante internet.

## **Descripción del proyecto**

El sistema de bloqueo de puertas con reconocimiento facial es capaz de tomar decisiones basadas en la tecnología de reconocimiento facial. El sistema utiliza una cámara web y una Raspberry Pi. Es capaz de realizar reconocimiento facial por sí solo, como detección de rostros, extracción de características, reconocimiento de rostros utilizando bibliotecas OpenCV.

Así mediante esta herramienta y programación se logra realizar un sistema de monitorización y control automático en una cerradura eléctrica que permite o no acceder a una ubicación abriendo y cerrando.

## **Desarrollo**

### **Reconocimiento Facial [1], [2]**

Para el desarrollo de este proyecto se utilizó el programa de visión artificial llamado OpenCV que es una popular biblioteca de visión por computadora iniciada por Intel en 1999. La biblioteca multiplataforma se centra en el procesamiento de imágenes en tiempo real e incluye implementaciones sin patente de los últimos algoritmos de visión por computadora. En 2008, Willow Garage se hizo cargo del soporte y OpenCV 2.3.1 ahora viene con una interfaz de programación para C, C++, Python y Android. OpenCV se publica bajo una licencia BSD, por lo que se utiliza tanto en proyectos académicos como en productos comerciales.

OpenCV 2.4 ahora viene con la muy nueva clase FaceRecognizer para reconocimiento facial, por lo que puede comenzar a experimentar con el reconocimiento facial de inmediato.

Los algoritmos disponibles actualmente son:

- Eigenfaces (Eigenface Recognizer)
- Fisherfaces (FisherFaceRecognizer)
- Local Binary Patterns Histograms (LBPH Face Recognizer)

Para la realización de este sistema se decidió utilizar el método LBPH.

Eigenfaces y Fisherfaces adoptan un enfoque algo holístico para el reconocimiento facial. Trata sus datos como un vector en algún lugar de un espacio de imagen de alta dimensión. El enfoque de Eigenfaces maximiza la dispersión total, lo que puede generar problemas si la varianza es generada por una fuente externa, porque los componentes con una varianza máxima en todas las clases no son necesariamente útiles para la clasificación. Entonces, para preservar algo de información discriminativa, aplicamos un análisis discriminante lineal y lo optimizamos como se describe en el método Fisherfaces. El método Fisherfaces funciona muy bien para el escenario restringido que asumimos en nuestro modelo.

Ya que no se puede garantizar una configuración de luz perfecta en sus imágenes o 10 imágenes diferentes de una persona. Entonces, ¿qué pasa si solo hay una imagen para cada persona? Nuestras estimaciones de covarianza para el subespacio pueden ser terriblemente incorrectas, al igual que el reconocimiento.

Descripción algorítmica del método LBPH

Se puede dar una descripción más formal del operador LBP como:

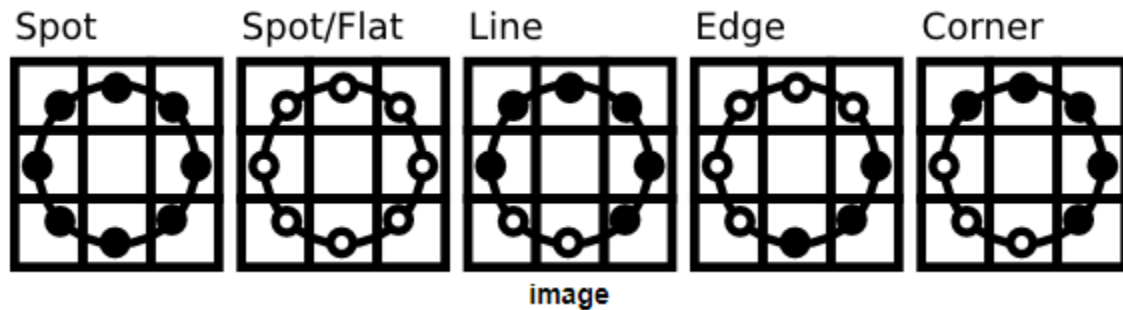
$$LBP(x_c, y_c) = \sum_{p=0}^{P-1} 2^p s(i_p - i_c)$$

, con como píxel central con intensidad  $i_c$ ; e  $i_p$  siendo la intensidad del píxel vecino. es la función de signo definida como:

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{else} \end{cases}$$

Esta descripción le permite capturar detalles muy finos en imágenes. Los autores pudieron competir con los resultados más avanzados para la clasificación de texturas. Poco después de que se publicó el operador, se observó que un vecindario fijo no codifica los detalles que difieren en escala. La idea es alinear un número

reducido de vecinos en un círculo con un radio variable, lo que permite capturar los siguientes barrios:



Para un Punto dado la posición del vecino se puede calcular mediante:

$$x_p = x_c + R \cos\left(\frac{2\pi p}{P}\right)$$

$$y_p = y_c - R \sin\left(\frac{2\pi p}{P}\right)$$

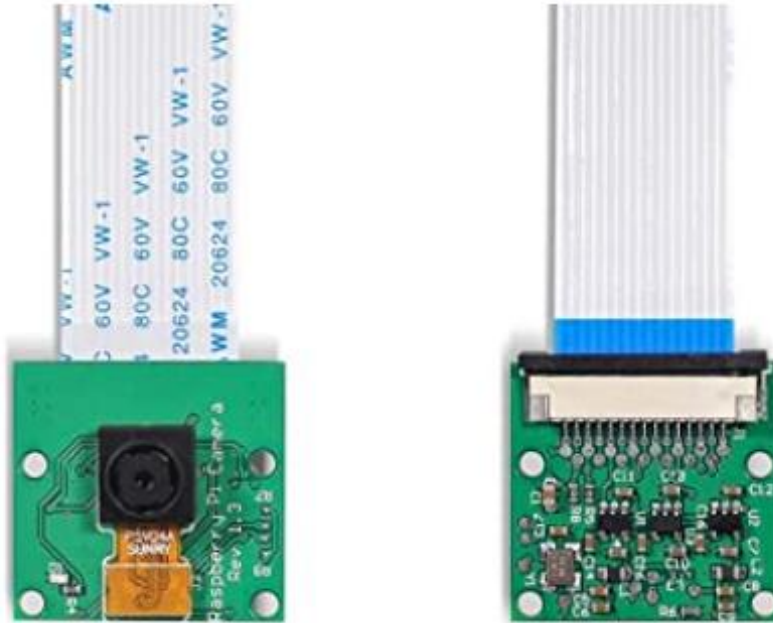
Donde R es el radio del círculo y P es el número de puntos muestrales

El operador es una extensión de los códigos LBP originales, por lo que a veces se denomina LBP extendido (también denominado LBP circular). Si una coordenada de puntos en el círculo no se corresponde con las coordenadas de la imagen, el punto se interpola. La informática tiene un montón de esquemas de interpolación inteligentes, la implementación de OpenCV hace una interpolación bilineal:

$$f(x, y) \approx \begin{bmatrix} 1-x & x \end{bmatrix} \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1-y \\ y \end{bmatrix}.$$

Entonces, lo que queda por hacer es cómo incorporar la información espacial en el modelo de reconocimiento facial. La representación propuesta por Ahonen consiste en dividir la imagen LBP en regiones locales y extraer un histograma de cada una. El vector de características espacialmente mejorado se obtiene concatenando los histogramas locales (no fusionándolos). Estos histogramas se denominan histogramas de patrones binarios locales.

**Módulo de cámara [3]**



Labists Raspberry Pi Módulo de cámara 1080P 5 M Pixels OV5647 Sensor con soporte de caja.

Características:

Este módulo de cámara Raspberry pi utiliza el sensor OV5647, capaz de capturar vídeo 1080P60, imagen fija de 5M. Carcasa transparente de alta calidad para la cámara Raspberry Pi de cualquier destrucción.

Cintas largas (15 cm) y cortas (10 cm) se adaptan a diferentes proyectos de Raspberry Pi.

La placa de la cámara se conecta a la Raspberry Pi a través de un cable plano de 15 vías. Solo hay que hacer dos conexiones: el cable plano debe estar conectado a la PCB de la cámara y a la propia Raspberry Pi. Debe colocar el cable en la forma correcta o la cámara no funcionará. En la placa de circuito impreso de la cámara, la parte trasera azul del cable debe mirar hacia afuera de la placa de circuito impreso, y en la Raspberry Pi debe mirar hacia la conexión Ethernet (o hacia dónde estaría el conector Ethernet si está usando un modelo A).

Para realizar la configuración de la cámara necesitaremos realizar una actualización antes con los siguientes comandos

```
$ sudo apt update
```

```
$ sudo apt full-upgrade
```

```
$ sudo raspi-config
```

Utilice las teclas del cursor para seleccionar y abrir las Opciones de interfaz, luego seleccione Cámara y siga las indicaciones para habilitar la cámara.

Al salir raspi-config, le pedirá que reinicie. La opción de habilitación garantizará que, al reiniciar, el firmware de la GPU correcto se ejecutará con el controlador y la sintonización de la cámara, y la división de memoria de la GPU es suficiente para permitir que la cámara adquiera suficiente memoria para funcionar correctamente.

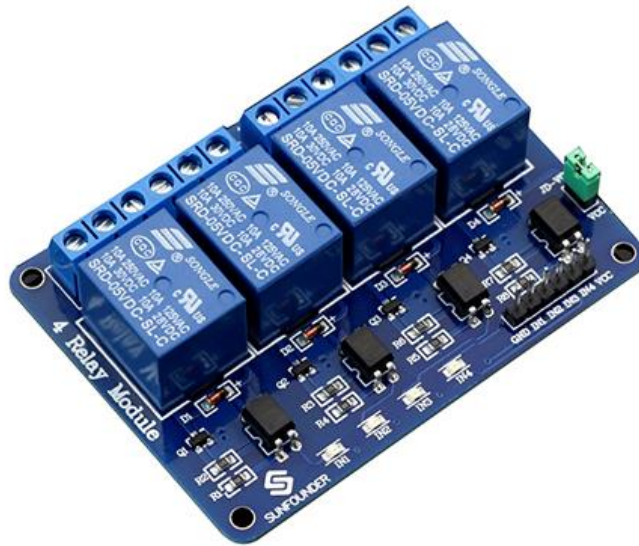
Para probar que el sistema está instalado y funcionando, pruebe el siguiente comando:

```
raspistill -v -o test.jpg
```

La pantalla debe mostrar una vista previa de cinco segundos de la cámara y luego tomar una foto, guardada en el archivo test.jpg, mientras muestra varios mensajes informativos.

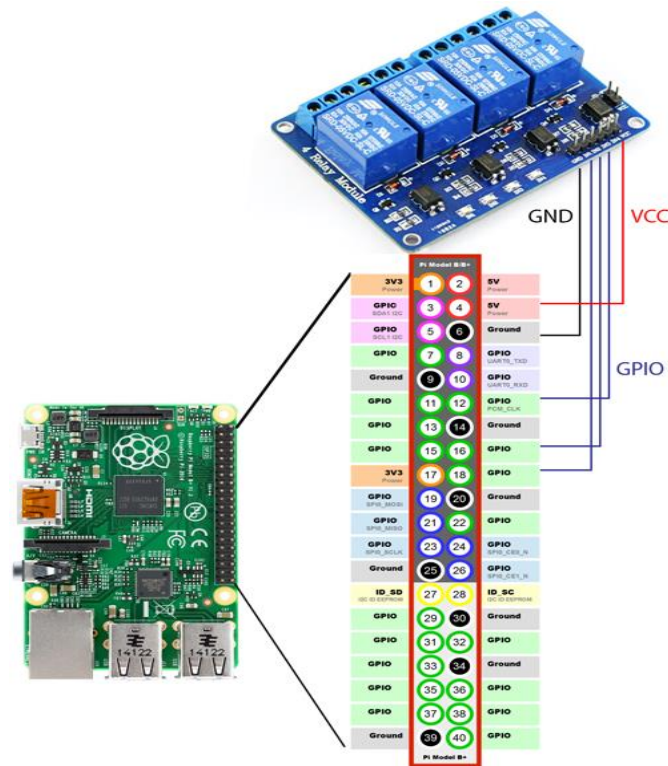
### **Método de apertura de la cerradura eléctrica**

Para la apertura de la cerradura será necesario implementar algunos relevadores, en este proyecto se utilizó un módulo de relevadores abriendo la posibilidad de controlar mas que solo la cerradura, sin embargo, para este ejemplo únicamente utilizaremos la cerradura.



### **Diagrama esquemático de conexión**

De manera general la conexión del módulo de relevadores será de la siguiente manera ¿, con los pines IN1, IN2, IN3, IN4 a los pines GPIO de la Raspberry y los pines VCC y GND a los respectivos de una fuente externa, aunque la imagen muestre una conexión a la Raspberry en estos pines, pero intentaremos preservar estos para otros fines.



Debido a que estamos usando una configuración normalmente abierta, no hay contacto entre los enchufes COM y NO a menos que active el relé. El relé se activa cuando la entrada desciende por debajo de aproximadamente 2 V. Eso significa que, si envía una señal BAJA desde el Raspberry, el relé se enciende, y si envía una señal ALTA, el relé se apaga; funciona con lógica invertida.

Dado que es un actuador mecánico en cierta medida no necesitaremos ninguna librería extra para controlarlo.

## LCD

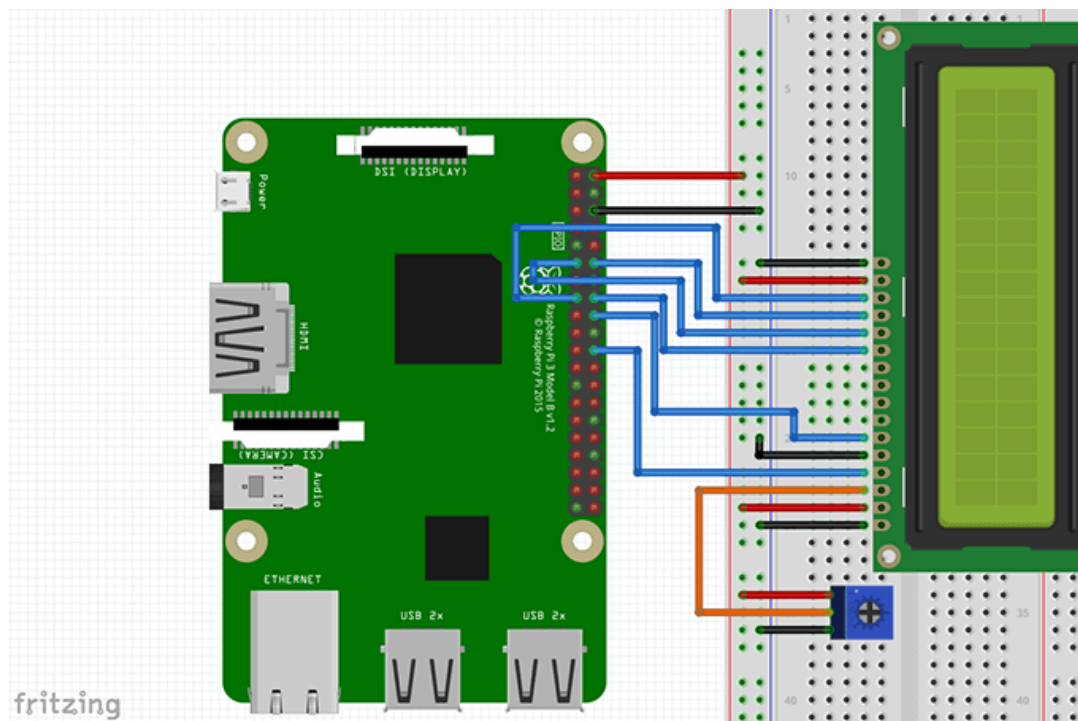
Para la utilización del display lcd 16x2 necesitaremos una biblioteca llamada pigpio ya que la propia biblioteca de la lcd está basada en la biblioteca wiringpi para C++, para la instalación de necesaria necesitaremos ingresar el comando

```
$ sudo apt-get install pigpio python-pigpio python3-pigpio.
```

```
$ sudo pip install RPLCD
```

El cableado se realizará de la siguiente manera:





Podemos notar que necesitamos un potenciómetro para controlar la intensidad del contraste de la pantalla LCD.

Para conocer cada uno de los comandos necesarios para inicializar el display y escribir en él se dejará la documentación necesaria en las fuentes de este proyecto [4].

Este LCD 16x2 dentro de este proyecto se utilizará para darle la indicación al usuario que ha sido o no reconocido mediante el texto impreso en esta ya sea “Abriendo” o “Cerrando” respectivamente para cada caso.

### **Cerradura Utilizada**

Los solenoides son básicamente electroimanes: se hacen de una gran bobina de alambre de cobre con una armadura (un lingote de metal) en el medio. Cuando se energiza la bobina, el centro de metal se mueve en la bobina. Esto hace que el solenoide sea capaz de tirar de un extremo.

Este solenoide, en particular, es agradable y fuerte, y tiene un lingote con un corte inclinado y un soporte bueno. Se trata básicamente de una cerradura electrónica, diseñada para un gabinete de base o de seguridad o de puerta. Normalmente, el bloqueo está activo por lo que no puede abrir la puerta porque el lingote del solenoide está en el camino. No utiliza ningún poder en este estado.

Cuando se aplica 9-12VDC, el lingote se contrae para que no sobresalga más y la puerta se pueda abrir. Los solenoides vienen con el lingote inclinado como se muestra arriba, pero se puede abrir con los dos tornillos de cabeza Phillips y darle



la vuelta rotándolo 90, 180 o 270 grados para que coincida con la puerta que desee usar con ella. Voltaje de Trabajo de 12VDC (pueden usar de 9 a 12 voltios DC, pero voltajes menores generan movimientos más lentos).

Diseñado para estar activado por un lapso de 1~10 segundos como máximo. Es adecuado para puerta, cajón, almacenamiento médico, vitrina, gabinete, ventana o caja de seguridad.

Especificaciones:

Tamaño: 65x38x28mm (L x W x H)

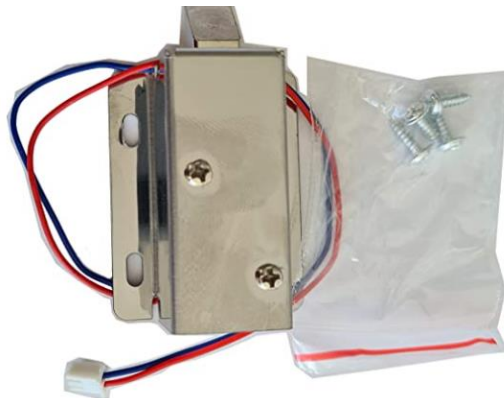
Voltaje: 12 V

Corriente: 0.8A

Potencia: 9.6 W

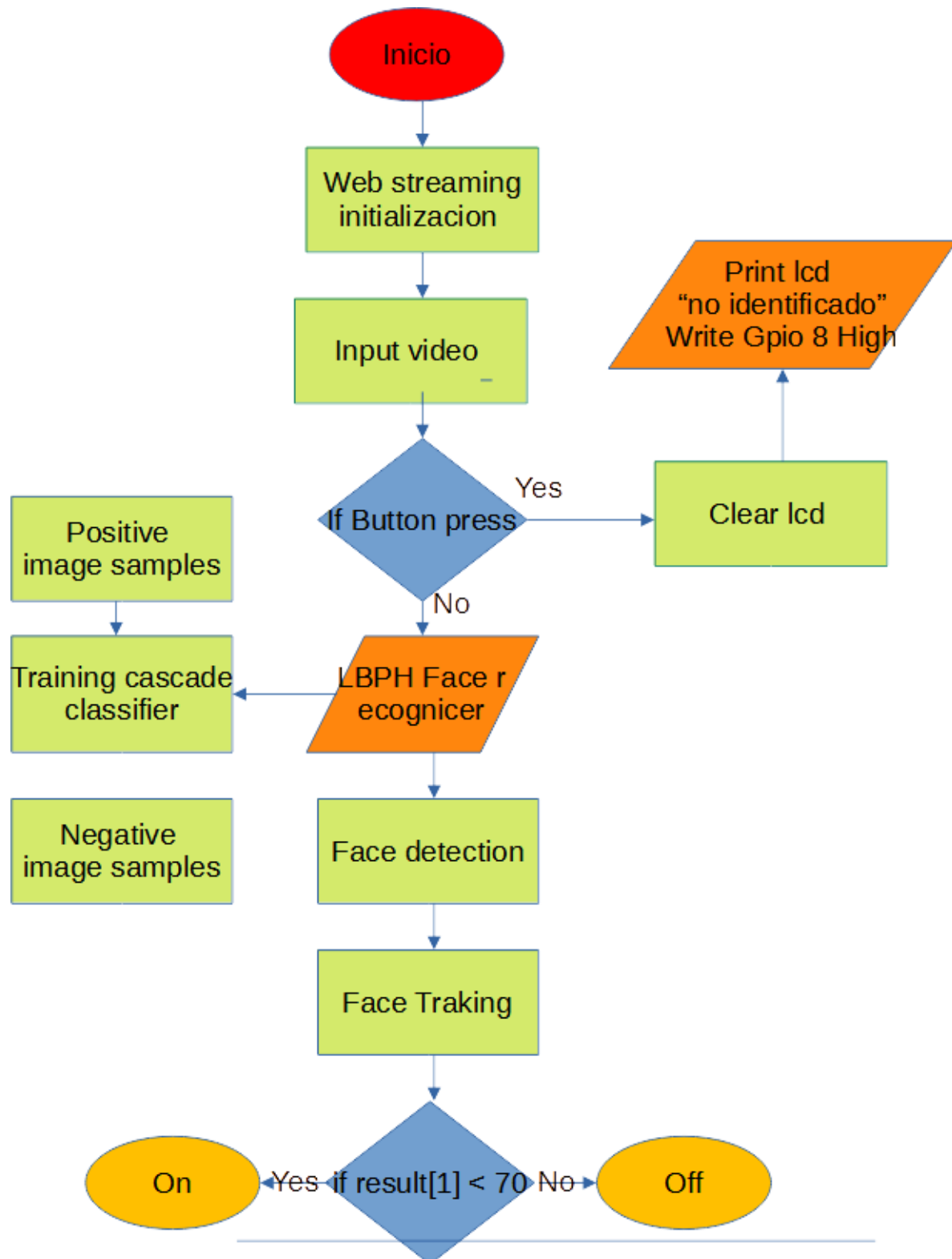
Diseñado para 1~10 segundos el tiempo de activación

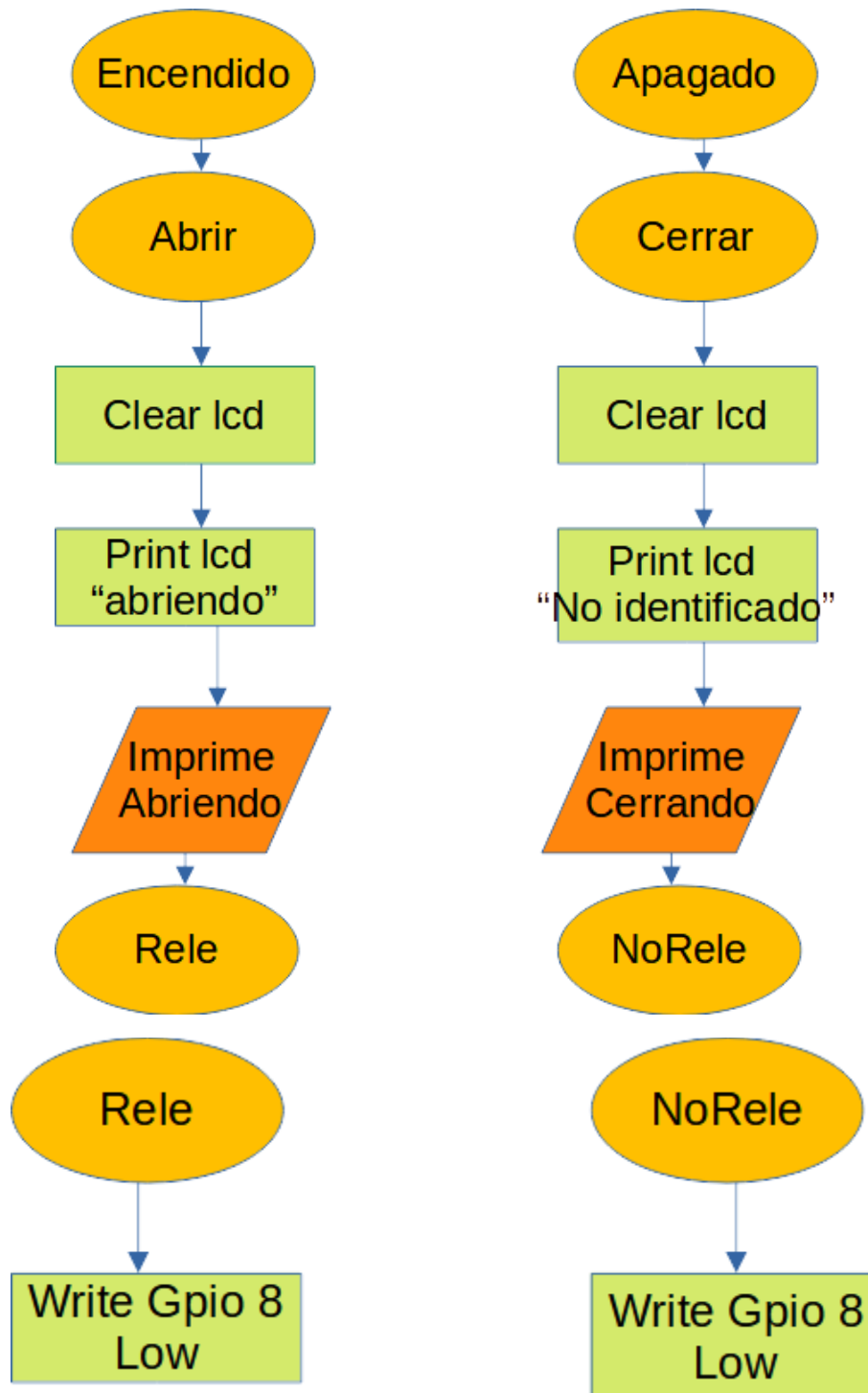
Longitud del cable: 38 cm Peso: 142g



## Implementación

En la generalidad de este proyecto no ahondaremos en cada uno de los scripts ya que requeriría un documento el estudio de cada script sin embargo lo que hare será mostrar las características del proyecto y sus principales funciones de manera general y mas adelante en el documento se explicara un poco mas a detalle como funciona.

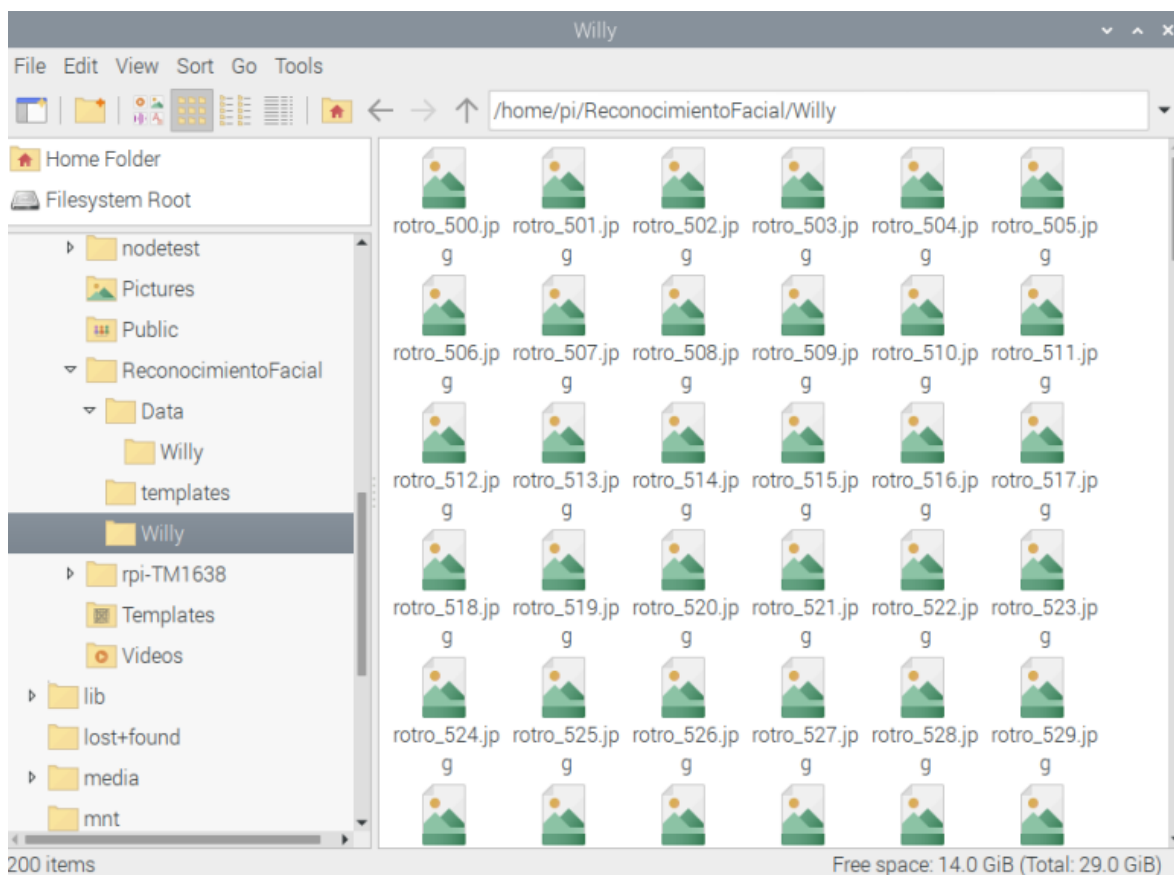
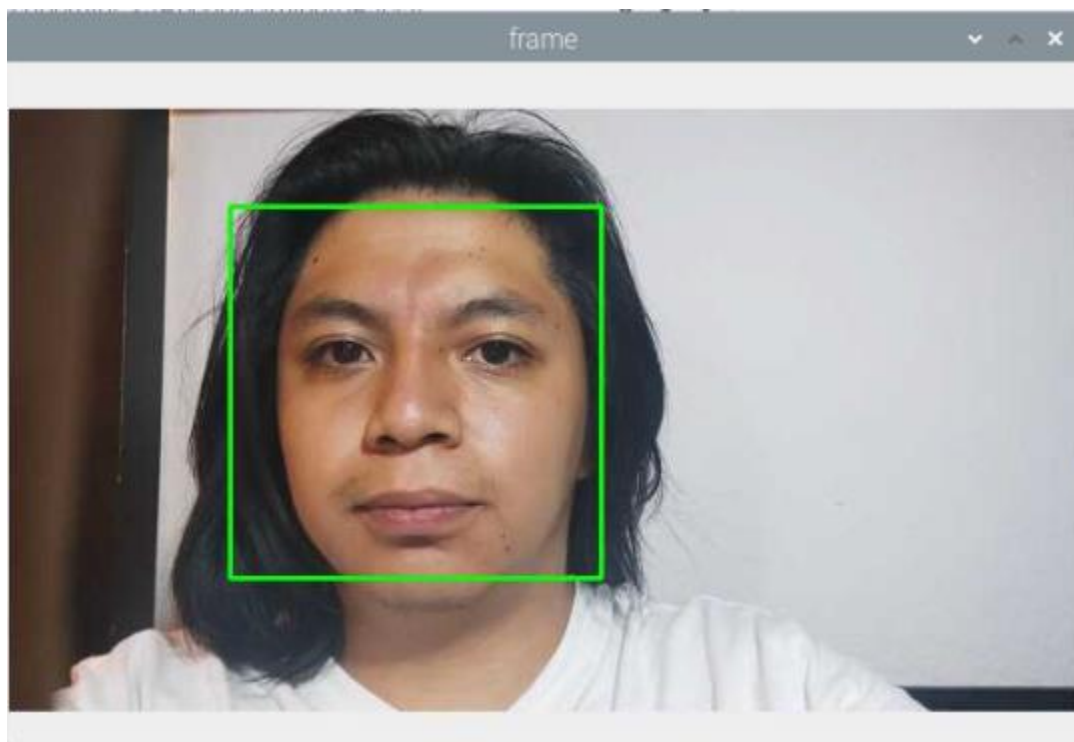




Comenzamos con el programa de reconocimiento facial, este se compone por 3 scripts distintos el primero se encarga de tomar las muestras para poder ingresar usuarios dentro del sistema.

Podemos hacerlo de dos maneras distintas, mediante un video previamente grabado o tomando los frames de la cámara directamente, en este caso se utilizo un video previamente tomado, así mismo podemos ver que después de ejecutar el

código se ha creado una carpeta con la etiqueta del usuario, así como las muestras tomadas.

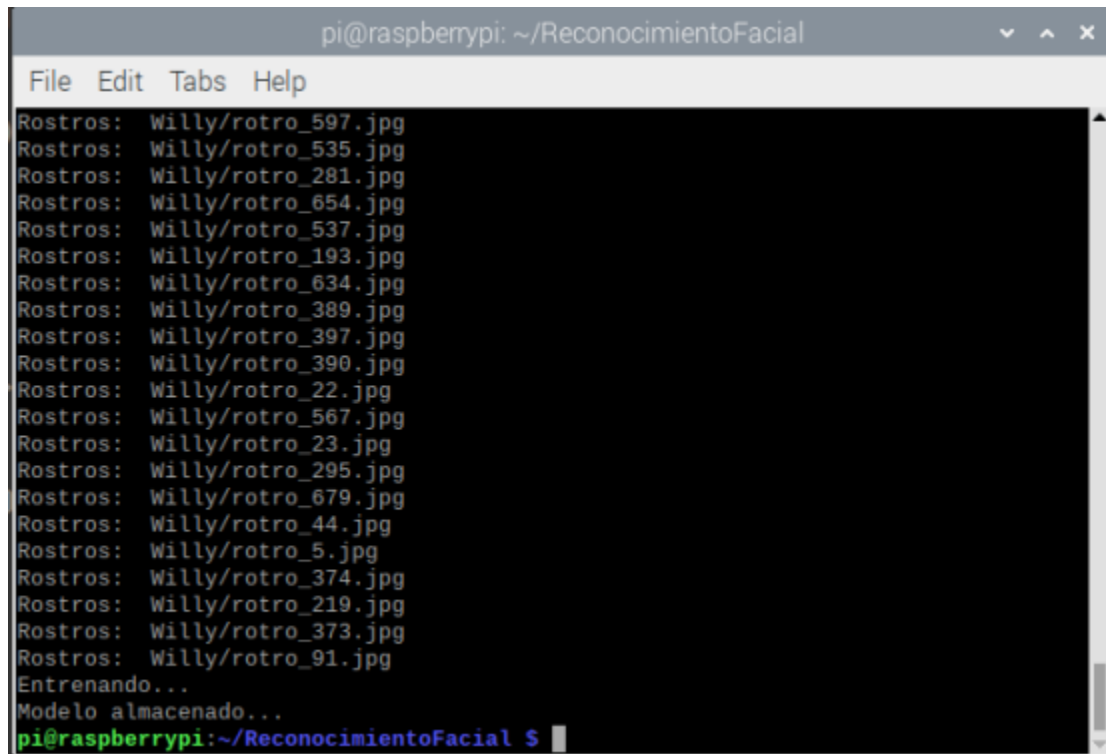


Para realizar el reconocimiento facial necesitaremos de los rostros de las personas que deseemos reconocer. Estos rostros deberán denotar variedad de expresiones como: felicidad, tristeza, aburrimiento, sorpresa, entre otros. Otro aspecto que deben tener estas imágenes es la variación de condiciones de luz, que las personas lleven lentes o que no los lleven, incluso que cierren los ojos o guiñen uno.

Es recomendable que se realice la recolección de estas imágenes en el escenario o ambiente en donde se vaya a aplicar el reconocimiento facial. Toda esta variedad de imágenes que se obtenga de los rostros contribuirá al desempeño de los algoritmos que usemos el día de hoy (y claro, según vayas experimentando podrás añadir o quitar algunas de estas condiciones que he nombrado).

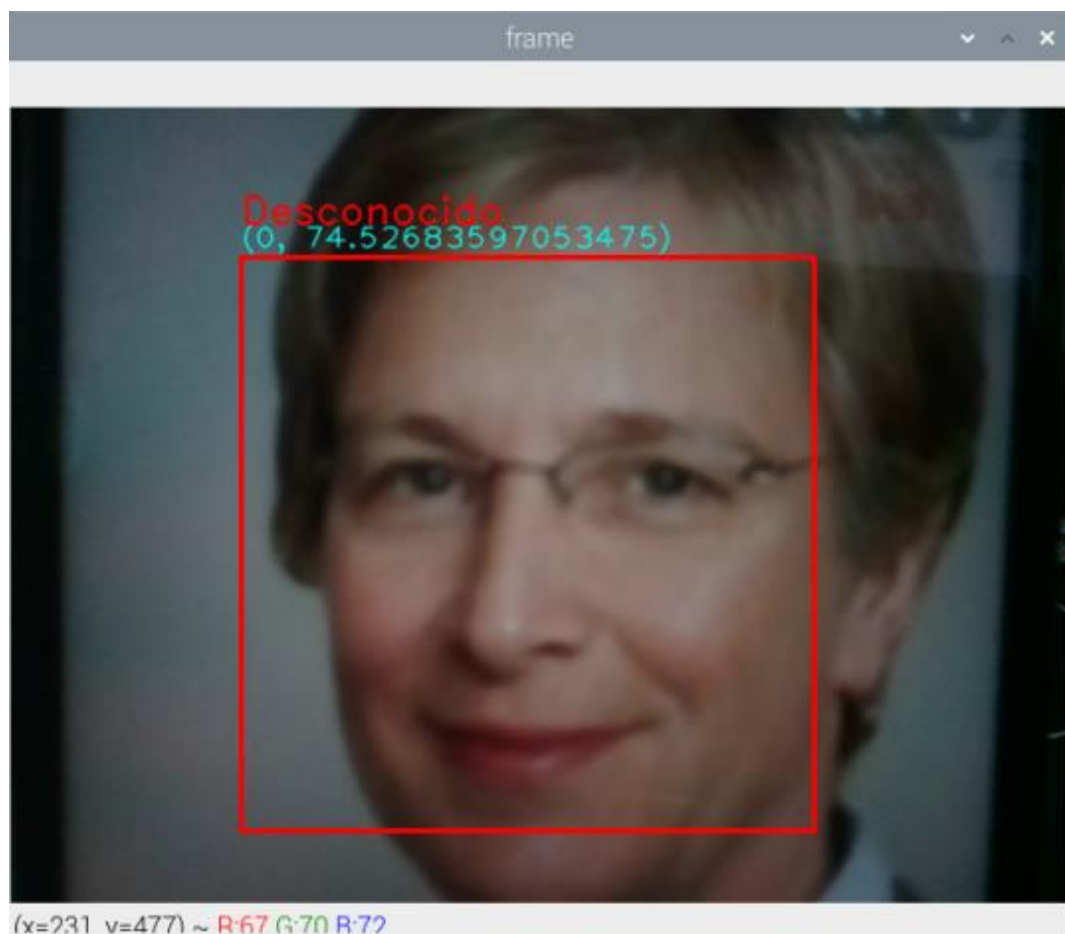
En este programa llamado se almacenarán automáticamente 300 rostros dado un video.

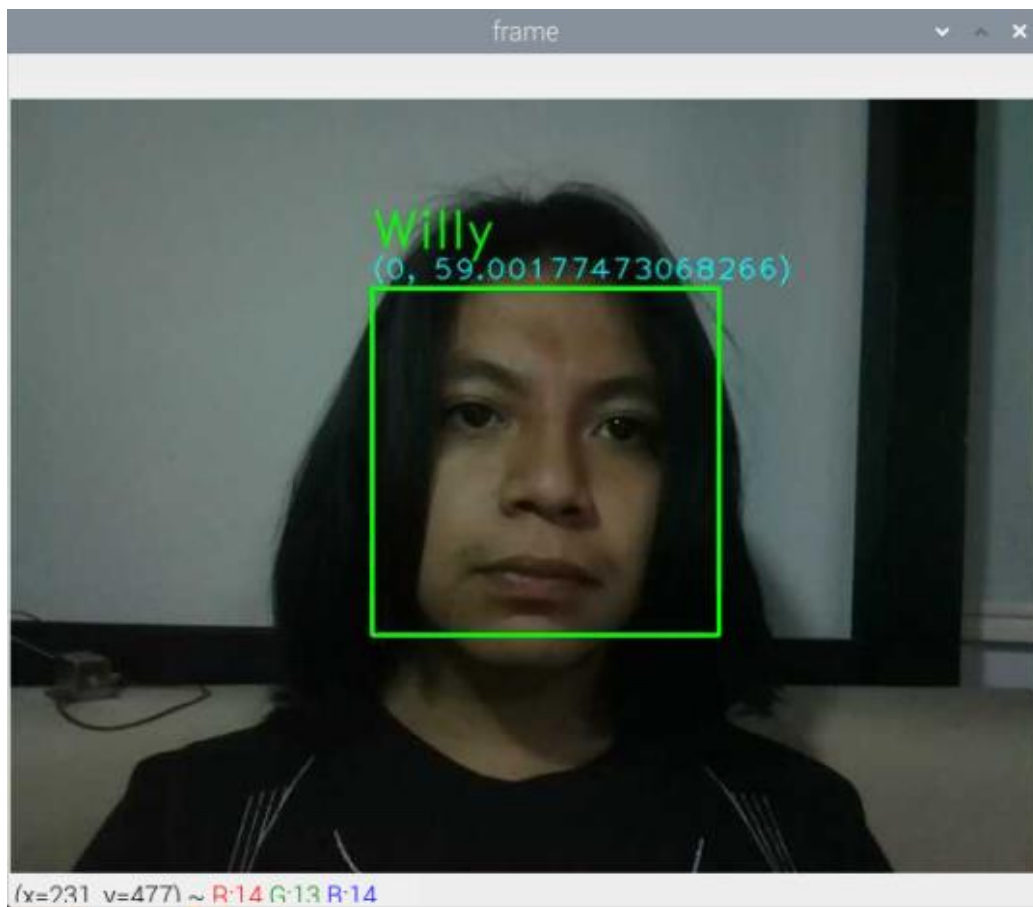
El segundo se encargará de “entrenar” al algoritmo para hacer el reconocimiento de los usuarios ingresados.



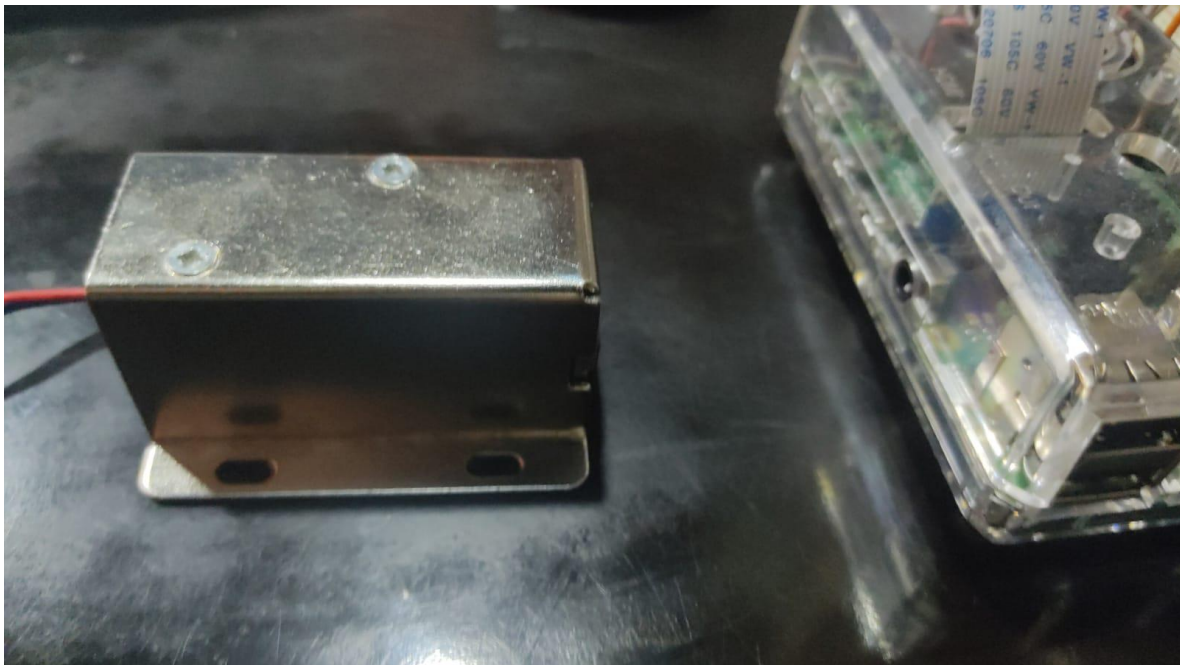
```
pi@raspberrypi: ~/ReconocimientoFacial
File Edit Tabs Help
Rostros: Willy/rotro_597.jpg
Rostros: Willy/rotro_535.jpg
Rostros: Willy/rotro_281.jpg
Rostros: Willy/rotro_654.jpg
Rostros: Willy/rotro_537.jpg
Rostros: Willy/rotro_193.jpg
Rostros: Willy/rotro_634.jpg
Rostros: Willy/rotro_389.jpg
Rostros: Willy/rotro_397.jpg
Rostros: Willy/rotro_390.jpg
Rostros: Willy/rotro_22.jpg
Rostros: Willy/rotro_567.jpg
Rostros: Willy/rotro_23.jpg
Rostros: Willy/rotro_295.jpg
Rostros: Willy/rotro_679.jpg
Rostros: Willy/rotro_44.jpg
Rostros: Willy/rotro_5.jpg
Rostros: Willy/rotro_374.jpg
Rostros: Willy/rotro_219.jpg
Rostros: Willy/rotro_373.jpg
Rostros: Willy/rotro_91.jpg
Entrenando...
Modelo almacenado...
pi@raspberrypi:~/ReconocimientoFacial $
```

Finalmente, el ultimo script se encargará de que mediante el módulo de cámara se tome el video del cual queremos reconocer al usuario así mismo cuando lo reconozca o no así mismo implementará las variables con las que se controlará que texto se imprimirá y si activa o no los relevadores.









## Web Streaming

Para la implementación del web streaming se utilizó se utilizó un framework llamado Flask para la creación del buffer que se envía mediante un socket dentro del mismo

framework y adicionalmente un diseño de página web mediante HTML en este caso muy sencillo únicamente para visualizar el streaming.

Para instalarlo se necesita el siguiente comando:

```
$ pip install Flask
```

También es necesario que tengamos nuestra aplicación dentro de una carpeta que en este caso se optó por la misma que el reconocimiento dentro de esta misma tendremos que crear otra carpeta más con el nombre “Templates” donde se almacenara nuestro HTML esto es de manera obligatoria cuando utilizamos flask.

Este script se encarga de realizar un “muestreo” frame a frame de nuestro reconocimiento facial y de la misma manera es enviado a nuestra web mediante un buffer creado para poder visualizar de esta manera los frames.

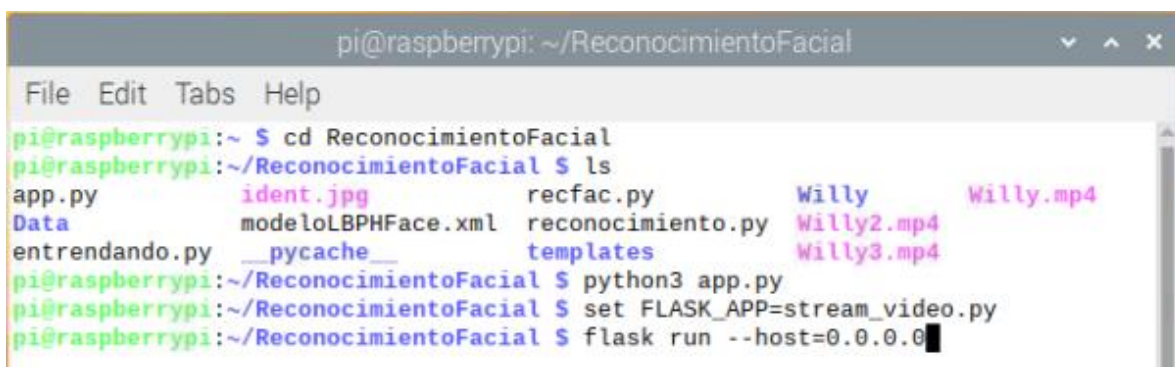
A diferencia de los scripts descritos anteriormente para poner en funcionamiento el streaming hay que crear una aplicación de Python desde el programa y ejecutara de la misma manera que los demás.

Una vez realizado este paso tendremos que indicarle a la raspberry que queremos definir que haremos con la app con el comando

```
$ set Flask_APP=stream_video.py
```

Posterior a esto tendremos que poner en marcha nuestro servidor con el siguiente comando

```
$ flask run --host=0.0.0.0
```



```
pi@raspberrypi: ~/ReconocimientoFacial
File Edit Tabs Help
pi@raspberrypi:~ $ cd ReconocimientoFacial
pi@raspberrypi:~/ReconocimientoFacial $ ls
app.py          ident.jpg      recfac.py      Willy          Willy.mp4
Data            modeloLBPHFace.xml  reconocimiento.py  Willy2.mp4
entrendando.py __pycache__    templates      Willy3.mp4
pi@raspberrypi:~/ReconocimientoFacial $ python3 app.py
pi@raspberrypi:~/ReconocimientoFacial $ set FLASK_APP=stream_video.py
pi@raspberrypi:~/ReconocimientoFacial $ flask run --host=0.0.0.0
```

Una vez introducidos estos códigos ya estará en línea nuestro servidor en la red de área local por lo que únicamente los dispositivos conectados a nuestra red podrán acceder a el straming.

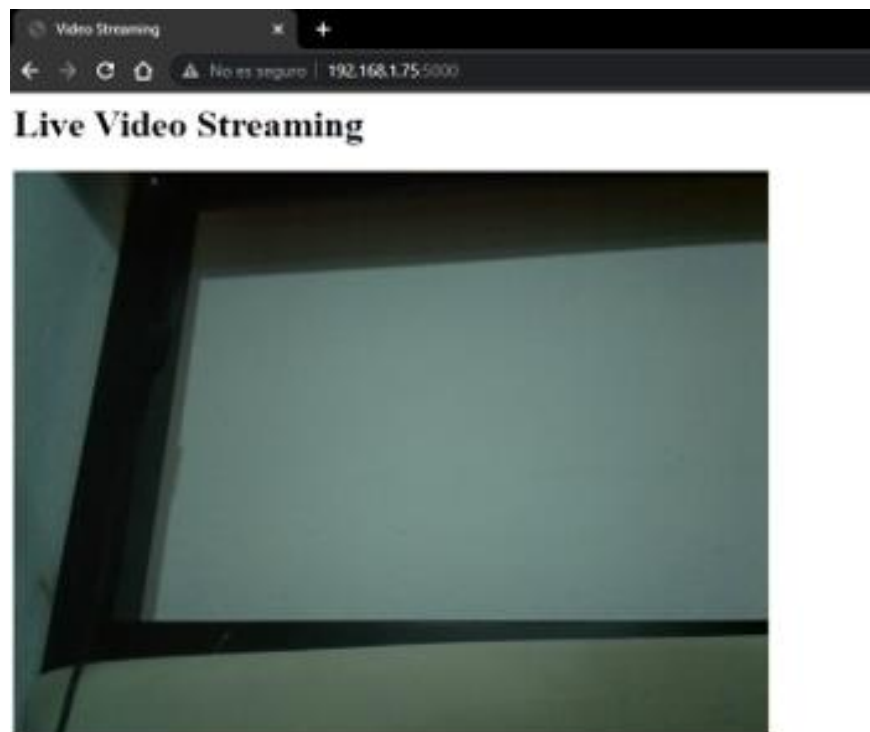
```
pi@raspberrypi: ~/ReconocimientoFacial
File Edit Tabs Help
pi@raspberrypi:~ $ cd ReconocimientoFacial
pi@raspberrypi:~/ReconocimientoFacial $ ls
app.py          ident.jpg      recfac.py      Willy          Willy.mp4
Data           modeloLBPHFace.xml  reconocimiento.py  Willy2.mp4
entrenando.py  __pycache__    templates      Willy3.mp4
pi@raspberrypi:~/ReconocimientoFacial $ python3 app.py
pi@raspberrypi:~/ReconocimientoFacial $ set FLASK_APP=stream_video.py
pi@raspberrypi:~/ReconocimientoFacial $ flask run --host=0.0.0.0
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
```

Para acceder a nuestra página web bastara con ingresar en el navegador la dirección ip de nuestro raspberry seguido del puerto donde se puede ver.

De manera predeterminada nuestra app de Flask se configura en el puerto 5000 por lo que la manera de ingresar será

0.0.0.0:5000

(se intercambian los 0 por la dirección de la raspberry)

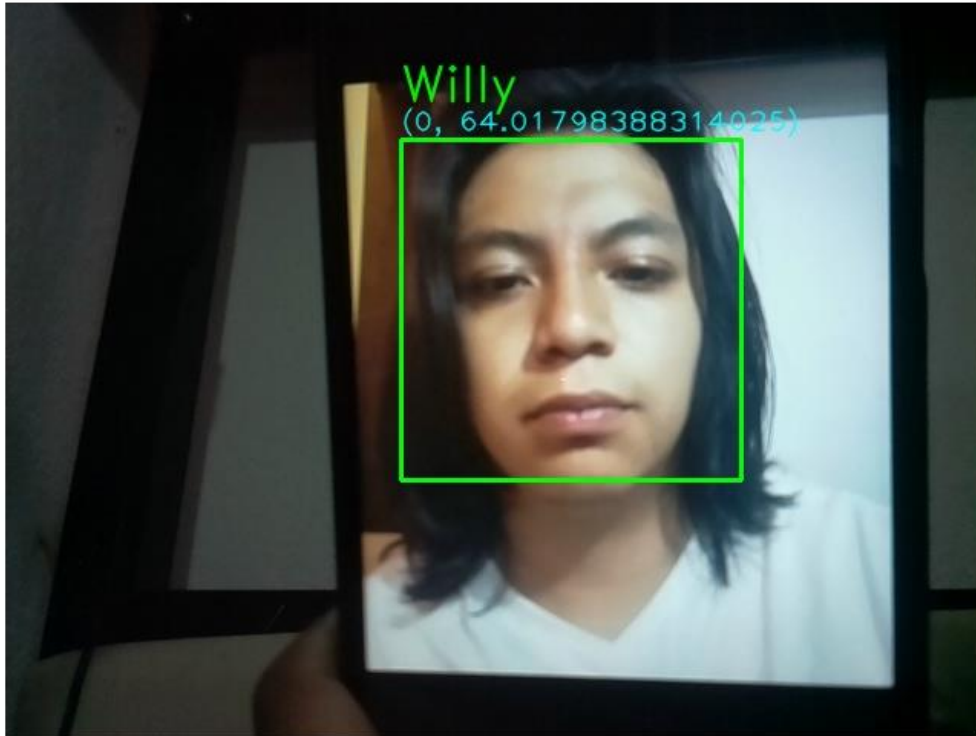


Finalmente podremos visualizar que mediante este programa podremos implementar el reconocimiento fácil y de la misma manera enviarlo mediante internet, en este caso para poder monitorear la cerradura eléctrica lo cual se puede traducir en monitorear la seguridad de nuestro hogar, oficina, etc.

## Live Video Streaming



## Live Video Streaming



### Observaciones

Es importante recalcar la importancia de unificar cada una de las tierras

Debido a que se necesitan unificar cada una de las tierras y estas serán nuestra referencia se presentó la problemática de que al ser una carga inductiva nos distorsiona los caracteres en la comunicación LCD por lo que para solucionar esta problemática se conectó un filtro consistente en un capacitor de acoplamiento para atenuar este ruido y de esta manera poder mostrar los textos en el display.

Otra de las cosas notables es que este tipo de cerraduras si se mantienen demasiado tiempo abiertas se calientan demasiado por lo que es importante delimitar un tiempo sobre el cual estará abierto para que no se sobrecaliente previniendo que se dañe.

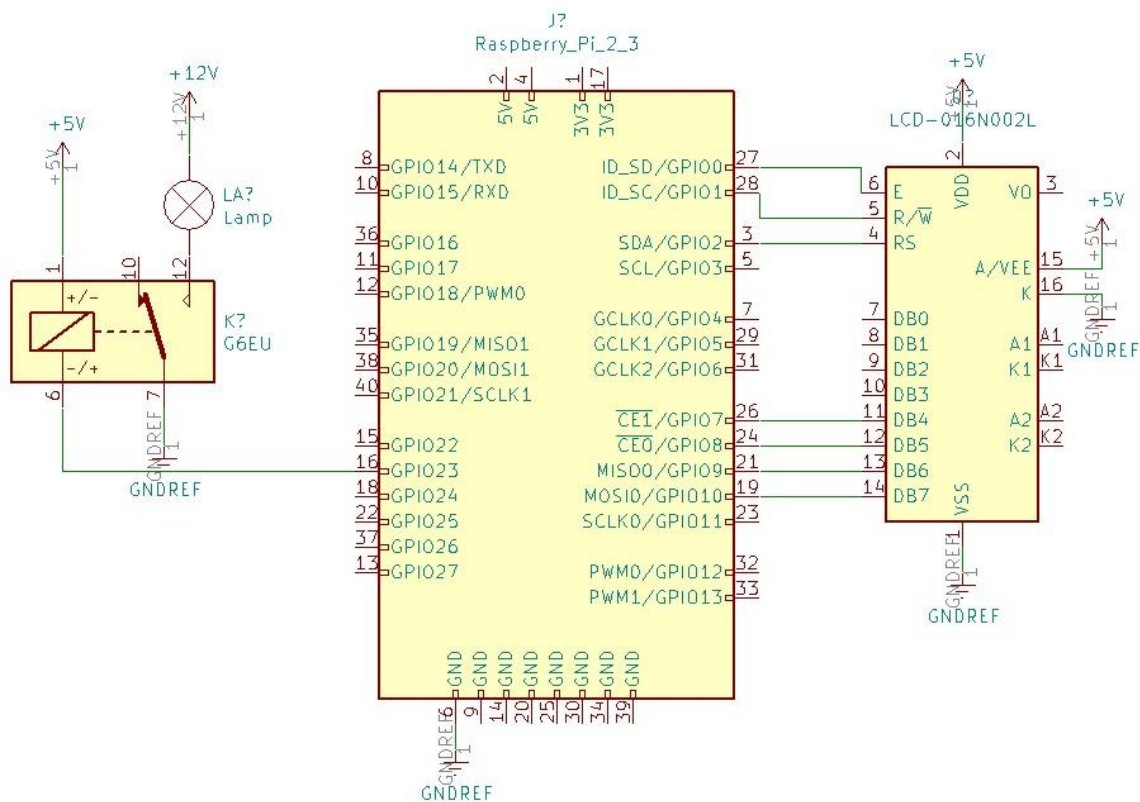
Respecto al web streaming fue una de las cosas más complicadas de manejar e integrar con respecto a todo lo demás ya que el primer script utilizaba una condición para realizar las funcionalidades adicionales como la pantalla lcd y el control del módulo de relevadores, sin embargo nos encontramos con que debido a se utiliza un ciclo "while" que siempre es verdadero dentro de del buffer para poder tomar las muestras, por lo que si se mantenía de la manera que se escribió la primera versión del programa cada vez que se necesitara ejecutar una de estas instrucciones se repetiría tantas veces como frames fueran "reconocidos" por nuestro algoritmo.



Debido a esta situación se tuvo que reescribir el programa en una segunda versión con todas las funcionalidades anidadas en funciones, con lo que logramos que se ejecutaran solo cuando eran necesarias y no siempre que tuviéramos un nuevo frame.

Dado esto último se encontró otra problemática ya que una vez abierta la cerradura por el reconocimiento ya no se cierra, dado que al hablar de una cerradura podemos generalizar que será utilizada para una puerta en la mayoría de los casos esto se solucionó implementando un pulsador que saca del estado de la función al programa y lo regresa al ciclo original.

El diagrama eléctrico completo luce de la siguiente manera:



## Conclusiones

De este proyecto lo que puedo concluir es lo imperativo que es conocer las herramientas de desarrollo, así como sus fundamentos tanto en hardware como de software, ya que con la creciente globalización con la que vivimos hoy en día es necesario estar a la vanguardia ya que los problemas han cambiado a ser de índole mayormente tecnológico y es importante estar prepara para el desarrollo de nuevas herramientas que nos permitan solventar las problemáticas actuales.

## Fuentes y referencias:

- [1] <https://github.com/GabySol/OmesTutorials2020/tree/master/6%20RECONOCIMIENTO%20FACIAL>
- [2] [https://docs.opencv.org/4.2.0/da/d60/tutorial\\_face\\_main.html](https://docs.opencv.org/4.2.0/da/d60/tutorial_face_main.html)
- [3] <https://www.raspberrypi.org/documentation/configuration/camera.md>
- [4] <https://rplcd.readthedocs.io/en/stable/usage.html>
- [5] <https://flask.palletsprojects.com/en/2.0.x/>
- [5] <https://github.com/NakulLakhotia/Live-Streaming-using-OpenCV-Flask>
- [5] [https://docs.opencv.org/3.0-beta/doc/py\\_tutorials/py\\_gui/py\\_image\\_display/py\\_image\\_display.html](https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_gui/py_image_display/py_image_display.html)
- [5] <https://towardsdatascience.com/video-streaming-in-web-browsers-with-opencv-flask-93a38846fe00>