FACULTAD DE INGENIERÍAS Y CIENCIAS AGROPECUARIAS

# PROCESAMIENTO DE IMÁGENES USANDO RASPBERRY PI.

Trabajo de Titulación presentado en conformidad con los requisitos establecidos para optar por el título de Ingenieros en Redes y Telecomunicaciones

Profesor Guía

MSc. Jorge Wilson Granda Cantuña

Autores

Verónica Raquel Gordillo Gordillo

Carlos Francisco Nacimba Coello

Año

2016

## DECLARACIÓN DEL PROFESOR GUÍA

“Declaro haber dirigido este trabajo a través de reuniones periódicas con las estudiantes, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación”

_____

Jorge Wilson Granda Cantuña
Master en Ingeniería Eléctrica
C.I.1708594187

## DECLARACIÓN DE AUTORÍA DE LAS ESTUDIANTES

“Declaramos que este trabajo es original, de nuestra autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes.”

| | |
|---|---|
| _____ | _____ |
| Verónica Raquel Gordillo Gordillo | Carlos Francisco Nacimba Coello |
| C.I. 1003519707 | C.I: 1716535982 |

## AGRADECIMIENTOS

A Dios de quien proviene toda bendición material y espiritual, sin la cual no hubiese sido posible la realización de ningún proyecto. A mi madre porque su esfuerzo es impresionante y su amor para mi invaluable, quien junto con mi padre me han educado y me han proporcionado todo y cada cosa q he necesitado para culminar la carrera, a mis hermanos quienes siempre están conmigo. A mi compañero de tesis y a todos quienes me han apoyado para lograr este objetivo. Muchas gracias.

Verónica Gordillo

**DEDICATORIA**

A las mujeres de mi vida: mis hermanas Salome y Fernanda y muy en especial a mi mami por contagiarme siempre de su alegría, su energía y su pasión por hacer las cosas y hacerlas bien, sin pretextos.

Verónica Gordillo

**AGRADECIMIENTOS**

A mis padres Fausto y Fanny quienes me han apoyado en todo este proceso, que con sus palabras de aliento me dieron fuerzas para avanzar en un escalón más de mi vida; a mis hermanos Daniel y Mary que son un ejemplo y una bendición en mi vida.

Carlos Nacimba.

**DEDICATORIA**

Dedico el presente proyecto de titulación a mis abuelitos Oswaldo y Maruja quienes me han inculcado los valores con los que guío mi vida.

Carlos Nacimba

## RESUMEN

El presente proyecto desarrolla el prototipo para la implementación de un sistema capaz de captar imágenes tomadas desde una cámara y procesarlas en una plataforma Raspberry pi. El procesamiento que se llevó a cabo es el reconocimiento e identificación de rostros, así como también el reconocimiento de dos gestos faciales, para finalmente transmitir las imágenes de un punto a otro, todo este proceso realizado en tiempo real. El proyecto utiliza dispositivos tales como Raspberry pi, una cámara y un computador para conseguir el funcionamiento del sistema de procesamiento.

Para presentar el desarrollo del proyecto se ha conformado la parte escrita de cuatro capítulos, distribuidos de la siguiente manera:

Capítulo 1: Se detallan los temas de investigación en los cuales se basa el desarrollo del proyecto; en particular se detalla el estudio de las señales análogas y digitales como base del proyecto.

Capítulo 2: Aquí se realiza la explicación del diseño; y como fue implementado el sistema de procesamiento de señales (imágenes), a partir de los diagramas y las investigaciones anteriormente realizadas.

Capítulo 3: en este capítulo se realiza las pruebas de funcionamiento del sistema, obtenidas una vez que se ha implementado el sistema de procesamiento.

Capítulo 4: Finalmente en este capítulo se redactan las conclusiones y recomendaciones, las cuales se basan en la experimentación con el sistema de procesamiento de señales (imágenes).

En la sección comprendida de anexos, se recopila los datasheet de los dispositivos más importantes utilizados en la construcción del sistema de procesamiento de imágenes, así como el código fuente del programa.

# ABSTRACT

This project develops a prototype for implementing a system capable of capturing images taken from a camera and processed on a platform Raspberry pi. The processing was carried out is the recognition and identification of faces, as well as the recognition of two facial gestures, to finally transmit images from one point to another, this whole process done in real time. The project uses devices such as Raspberry Pi, a camera and a computer for the operation of the processing system.

To show in detail, as the objective of this project is demonstrated; the written document consists of four chapters, they are distributed as follows:

Chapter 1: research topics in which the project is based are detailed; in particular, the study of analog and digital signals as detailed project basis.

Chapter 2: Here the explanation of the design is done; and as implemented the system signal processing (images) from diagrams and research previously conducted.

Chapter 3: This chapter tests system operation is performed, obtained once it has been implemented processing system.

Chapter 4: Finally, in this chapter are drawn conclusions and recommendations, which are based on experimentation with the signal processing system (images).

The annex section consists of a collection of technical information of the devices used and the program source code.

# ÍNDICE

# ÍNDICE DE FIGURAS

# ÍNDICE DE TABLAS

# INTRODUCCIÓN

En la actualidad, el procesamiento de imágenes es una de las herramientas más usadas para la recopilación de datos en diferentes áreas de investigación científica, tales como monitoreo de cultivos, investigaciones marinas, cartografía del suelo, captura de infracciones en semáforos inteligentes, etc.

Dependiendo del uso y del propósito de las imágenes, se pueden aplicar diferentes tipos de filtros y fases de procesamiento, esto con el fin de mostrar particularidades que se deseen resaltar y mostrar para su respectivo análisis, tal es el caso del RPA (vehículo aéreo no tripulado), que actualmente se desea implementar con diferentes tipos de cámaras comunicadas entre sí, generando de esta manera una conexión integrada de dispositivos que obtendrán información específica para ser procesada en tiempo real.

La evolución del software ha permitido un mejor desempeño y aprovechamiento del hardware, a tal punto que cada vez los dispositivos de hardware tienden a ser más pequeños, tal es el caso del Raspberry pi que es un minicomputador diseñado con todos los componentes y puertos que podemos encontrar en placas de ordenadores de nueva generación; su sistema operativo es una versión de la distribución de Linux llamada Raspbian, pero también soporta hasta una versión de Windows 10. Sus aplicaciones con la programación correcta pueden ser innumerables; a pesar de tener un limitado rango de procesamiento, logra realizar las operaciones que se pueden ejecutar en un PC normal como ofimática, Internet, multimedia; de esta manera se convierte en un elemento muy valioso para el aprendizaje del mundo de la informática.

Por lo que en este proyecto se desarrolla la implementación de un prototipo capaz de captar imágenes tomadas desde una cámara y procesarlas en una plataforma Raspberry pi usando Python como lenguaje de programación, para luego transmitir las imágenes de un punto de interés hacia otro.

**Marco Referencial**

El Raspberry Pi es un computador que posee las funciones principales de un computador PC, creado por la fundación Raspberry Pi, cuyo objetivo principal es la enseñanza de los sistemas de computación a la sociedad. Su nombre fue tomado con la decisión de crear una combinación entre una fruta y un computador, como ya había sucedido con Apple o BlackBerry, su concepto es la de un computador que puede ser programado usando Python, actualmente se encuentra disponible en dos modelos conocidos como el Modelo A y el Modelo B, ambos modelos con un procesador Broadcom BCM2835.

Una de las diferencias más significativas entre los PC`s que usamos para nuestras tareas básicas de estudio o trabajo en relación con el Raspberry pi, es su sistema operativo al contario de usar Windows o Apple OS x esta pequeña placa usa una distribución de Linux con base en Debian, llamada Raspbian, el cual es de código abierto lo que nos permite hacer cambios en el código fuente del sistema.

**Alcance**

El documento plantea la implementación de un sistema capaz de captar imágenes tomadas desde una cámara y procesarlas en una plataforma Raspberry pi usando Python como lenguaje de programación. El procesamiento que se realiza es el reconocimiento e identificación de rostros, así como también el reconocimiento de dos gestos faciales (sonrisa y seriedad), para finalmente transmitir las imágenes de un punto a otro, utilizando la señal WiFi del computador, todo este proceso realizado en tiempo real.

**Justificación**

El análisis de señales (imágenes) es una herramienta útil y eficiente a la hora de extraer información del entorno, incluyendo gran parte de la robótica que lleva a cabo múltiples actividades en la industria, la investigación y en los hogares familiares. Estos presentan un módulo principal para el procesamiento de imágenes, que brinda al sistema una información primaria acerca de las

cualidades del entorno en el cual desarrolla su actividad. La necesidad de que estos sistemas lleven a cabo tareas más complicadas cada vez, lleva a mejorar casi exponencialmente el procesamiento de imágenes, para que éstas también sean transmitidas en tiempo real. El diseño y construcción del sistema de procesamiento de imágenes usando Raspberry pi permitirá:

- Servir como base para futuras investigaciones e implementaciones de sistemas de comunicación autónomos, capaces de transmitir imágenes procesadas en tiempo real.

- Transmitir información crítica desde el sistema en movimiento de reconocimiento, hasta el punto en el cual se realizará la toma de decisiones; lo que puede ser usado en servicios de socorro de desastres naturales, policía, fuerza civil, cruz roja, que necesitan herramientas que los ayuden en sus operaciones.

- Obtener datos más precisos sobre eventos críticos que ocurren a nivel mundial, usando sensores de percepción remota, lo cual ayuda a la toma de decisiones de manera eficaz.

## OBJETIVOS

### General

Implementar un sistema de procesamiento de imágenes, usando el Raspberry pi y transmitir las imágenes de un punto de interés a hacia otro.

### Específicos

- Analizar las funcionalidades de la plataforma Raspberry pi.
- Estudiar el lenguaje de programación Python conjuntamente con Opencv para realizar el procesamiento de las imágenes.
- Identificar técnicas de procesamiento de señales análogas y digitales.
- Diseñar e implementar el prototipo para el sistema de procesamiento digital de imágenes.
- Realizar las pruebas de transmisión y verificar el funcionamiento del sistema.

# Capítulo I. Marco Teórico

En este capítulo se presenta información sobre los conocimientos teóricos indispensables, en los cuales se fundamenta el proyecto.

## 1.1. Señales Análogas

Una señal análoga es aquella que se representa como una variación continua en el tiempo; si se toma un segmento de dicha señal y se realiza un recorrido en el tiempo, se debe pasar por todos los valores que componen este segmento de la señal pasando de un valor a otro en forma continua.

Este tipo de señales se las encuentra en todos los escenarios de la vida diaria, la luz que recibe el sentido de la visión, los sonidos que son percibidos por el sentido de la audición, la electricidad alterna con los que funcionan los equipos electrónicos etc. De esta forma a las señales análogas se las puede clasificar en dos grupos:

- Periódicas: Aquellas en las cuales sus valores se repiten en un determinado ciclo de tiempo, por lo que se puede predecir su comportamiento.
- No Periódicas: Aquellas que no repiten sus valores en un periodo de tiempo, con lo que no se puede realizar una predicción de su comportamiento.


Figura 1. Señal Análoga

Cualquier señal variable en el tiempo, también puede ser representa en el ámbito de sus componentes de frecuencia, es así que, para toda señal es factible que pueda ser representada como una composición entre su frecuencia fundamental y sus armónicos correspondientes. El análisis de Fourier es el proceso matemático que permite realizar la descomposición de las señales análogas.

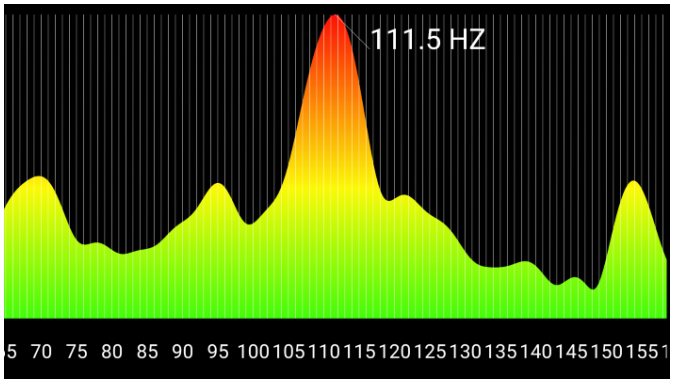



Figura 2.Espectro de frecuencia de un tambor acústico.

### 1.1.1 Parámetros

Las señales análogas tienen las siguientes características:

- Valor Pico: Se define como el valor más alto que puede alcanzar la señal; también se lo denomina como amplitud.
- Periodo: Es el tiempo que se demora una señal para volver a repetir sus valores de un ciclo.
- Frecuencia: Para una señal periódica la frecuencia es el número de veces que se repite un ciclo.

### 1.1.2. Ventajas y Desventajas

Las señales análogas tienen las siguientes ventajas:

- Una de sus principales ventajas con respecto a las señales digitales es su fidelidad ya que una señal análoga contiene mayor información que una digital.
- El tratamiento para este tipo de señales es considerablemente más sencillo con respecto a las señales digitales
- Entre las principales desventajas se encuentra el hecho de que para su trasmisión se necesita un mayor ancho de banda.
- Otra de las desventajas es su comportamiento ante la introducción de ruido en la señal; ya que este tipo de señales son muy susceptibles al ruido y es muy difícil que se elimine en este tipo de señales.
- Los sistemas de control no tienen capacidad para trabajar con señales análogas, por lo que es necesario transfórmalas en señales digitales para poder trabajar con ellas.

## 1.2. Señales Digitales

A diferencia de una señal análoga la cual varia con valores continuos en el tiempo, una señal digital toma solo ciertos valores discretos en el tiempo; quiere decir que entre una variación de tiempo la señal no va a tomar valores intermedios entre los cambios de un estado a otro. Un ejemplo ilustrativo son los interruptores de dos estados en los cuales se pasa de encendido o apagado, sin niveles intermedios. La forma representativa de una señal digital es un tren de pulsos de onda cuadrada en el dominio del tiempo; en el cual se pueden observar valores 1L 0L.

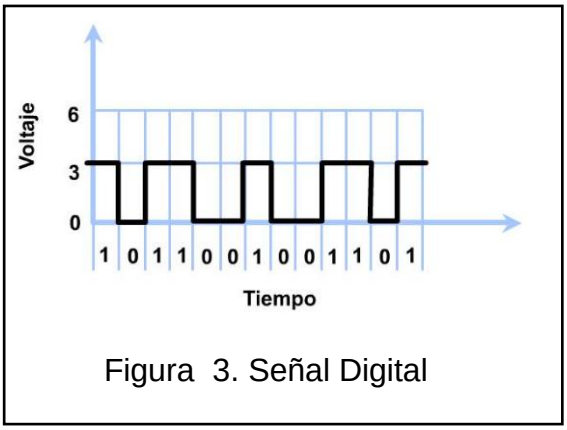



Figura 3. Señal Digital

La señal digital descrita en la figura 3, muestra un tren de pulsos de estado binario, para el cual se tienen los estados de 1L como 3V y 0L como 0V, es necesario tomar en cuenta que esto funciona para una señal digital con dos estados

### 1.2.1. Ventajas

Las ventajas de las señales digitales son las siguientes:

- En cualquier punto de la transmisión de una señal digital, si se detecta que la señal es muy baja, esta puede ser reconstruida y enviada hacia el punto de destino.

- Cuando un receptor recibe la señal desde un punto de origen, este puede realizar la detección y corrección de errores.

- Facilita el manejo de distintas operaciones, al ser señales con estados discretos el procesamiento lógico y matemático es mucho más sencillo, esta característica se ha convertido en uno de los fundamentos para el avance en áreas de tecnología.

- Permite realizar copias idénticas de la señal sin que se pierdan las características de la misma.

- El ruido ambiental influye de manera menos perjudicial en los sistemas digitales.

### 1.2.2. Desventajas

- Ya que una señal digital trata de representar a un mundo de señales análogas, es necesario conversión análoga-digital en la parte transmisora y un bloque de decodificación en el segmento receptor.

- Al estar una señal digital compuesta por valores discretos con estados definidos, se necesita una señal de sincronismo, la cual ayuda a los sistemas transmisores y receptores, a saber, en qué puntos inicia o termina un mensaje.

  Una señal digital se compone por muestras de una señal análoga por lo que mientras la cantidad de muestras tomadas sea menor, menor será la calidad de la señal resultante

### 1.2.3. Teorema del muestreo de Nyquist

Para el año de 1928 el investigador y científico Sueco Harry Nysquist, formuló el teorema de muestreo; pero no fue sino hasta 1949 que este pudo ser demostrado matemáticamente por el Estadounidense Claude Elwood Shannon; actualmente los avances en la mayoría de los campos en las telecomunicaciones tienen como punto base este teorema.

Básicamente lo que Shannon demostró a partir de fórmulas matemáticas; es que cualquier señal periódica en el tiempo, en banda base puede ser reconstruida a partir de muestras; con la condición que las muestras hayan sido tomadas a una razón igual o mayor al doble de la frecuencia del ancho de banda.

De esta manera si la frecuencia más alta contenida en la señal análoga de la que se pretende realizar muestras es $Fmax$, la señal con la que se deben realizar las muestras debe cumplir la siguiente condición $Fm$ debe ser mayor a $2Fmax$.

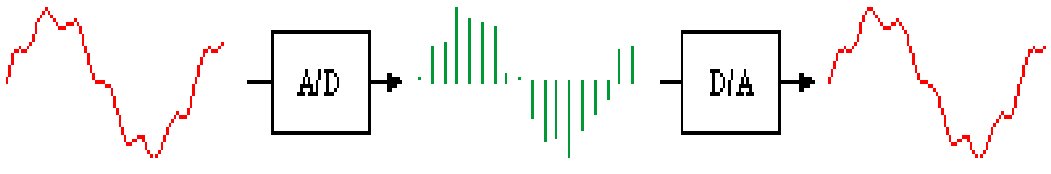



Figura 4. Teorema de Nyquist

### 1.2.3.1 Descripción Matemática

La teoría del muestreo define que, para una señal de ancho de banda limitado, la frecuencia de muestreo, $f_m$, debe ser mayor que dos veces su ancho de banda [B] medida en Hertz [Hz].

$$fm > 2 \cdot B \qquad \text{(Ecuación 1)}$$

La frecuencia resultante del producto 2·B se denomina razón de muestreo, y la mitad de este valor se le denomina frecuencia de Nyquist.

$$s_s(t) = s(t)\,\delta_{T_s}(t)$$

$$\delta_{T_s}(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT_s) \qquad \text{(Ecuación 2)}$$

$$T_s = \frac{2\pi}{w_s}$$

$$w_s = 2\pi f_s$$

### 1.2.4. Digitalización

La digitalización permite que un mensaje en formato analógico se convierta en una sucesión de impulsos eléctricos, los mismos que equivalen a dígitos de estado binario, para poder ser transmitidos, a estos dígitos combinados se los llama bits y corresponden a el 0 ó el 1. De esta manera, todo mensaje que se transforme en señal eléctrica y posteriormente sea codificado de manera digital puede ser almacenado o en su defecto, ser transmitido como un tren de pulsos por diferentes tipos de líneas como hilo telefónico, microondas, fibra óptica(FO), cable coaxial, etc.

La digitalización presenta varias beneficios, ya que a partir de ella se puede dar un tratamiento especial a la información, por ejemplo: permite el almacenamiento masivo de información en dispositivos de tamaño pequeño y lo que es más importante permite la manipulación de la información contenida dentro de estas señales a un nivel de conformación básico, por ejemplo en imágenes digitales se pueden realizar procesos a niveles de manipulación por pixeles, lo mismo se aplica para señales de video digital, lo cual no era posible que se realice con señales análogas.

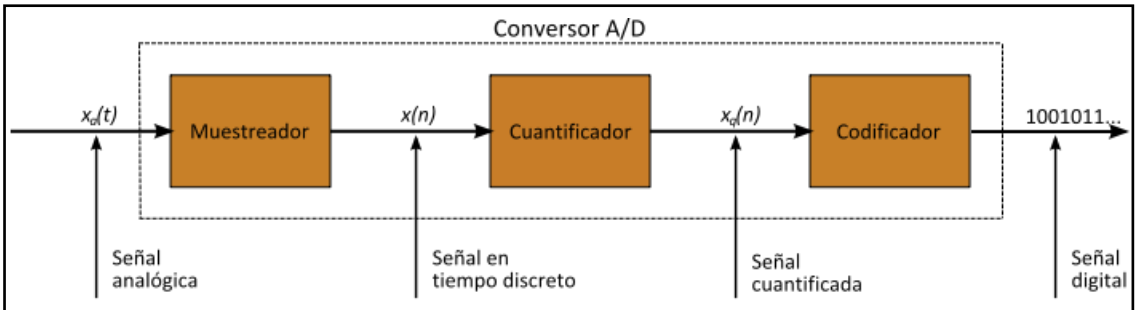



Figura 5. Conversor análogo-digital

En la figura 5 se observa un diagrama en bloques de un conversor de análogo a digital el cual consta de tres módulos, y que son los que tiene que recorrer una señal para ser convertida de un formato análogo a digital.

### 1.2.4.1 Muestreo

El muestro es básicamente en tomar muestras de la amplitud de la señal análoga, dichas muestras deben ser tomadas a un intervalo periódico. La frecuencia o también llamada tasa de muestreo determina el número de muestras que se toma de la señal análoga, en un cierto intervalo de tiempo, teóricamente la tasa de muestreo debe ser igual a dos veces la frecuencia fundamental de la señal.

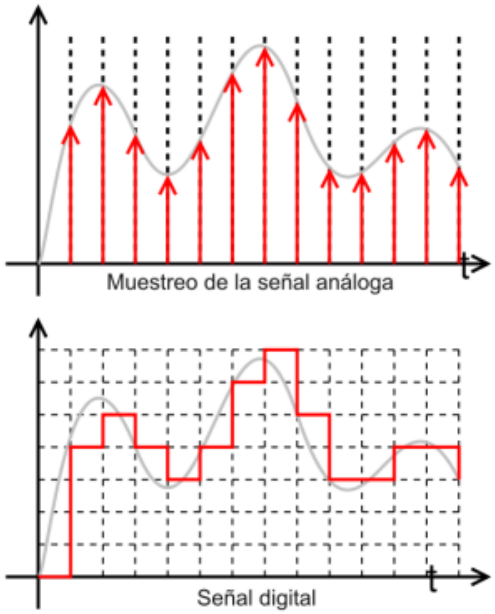



Figura 6. Muestreo de Señales análogas

### 1.2.4.2 Cuantificación

Luego del proceso de muestreo el siguiente paso a realizarse para la digitalización de la señal análoga es el proceso de cuantificación.

Una vez que se tienen las muestras de la señal, la cuantificación genera un número finito de valores para cada muestra, de esta manera los valores resultantes del muestreo se convierten en valores discretos.

Para realizar la cuantificación lo primero que se realiza es la medición del valor de tensión con la que llegan cada una de las muestras obtenidas anteriormente

en el primer bloque, a estos valores se les coloca un valor discreto, seleccionado por aproximación y que se encuentran dentro del margen de niveles previamente establecidos. Si algunos de los niveles obtenidos no coinciden directamente, se debe tomar como valor el inferior más próximo.

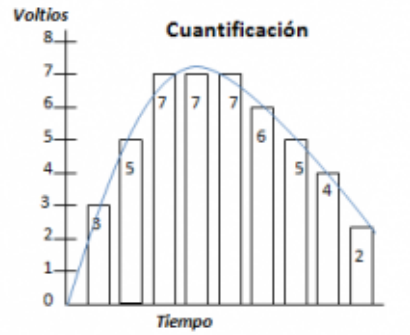



Figura 7. Cuantificación de Señales
Tomado de (Vignolo, J.)

Los tipos de cuantificación más usados son:

- Cuantificación uniforme
- Cuantificación no uniforme
- Cuantificación logarítmica
- Cuantificación vectorial

#### 1.2.4.3 Codificación

Luego del proceso de cuantificación, el siguiente proceso a ejecutarse es el proceso de codificación, el cual a su vez es el último proceso de la digitalización.

La codificación consiste en realizar una interpretación de los valores obtenidos luego de la cuantificación con un código binario establecido anteriormente, de esta manera se obtiene una señal digital como tren de pulsos, el cual posteriormente para ser decodificado debe se debe usar la misma secuencia de código usado en el proceso de codificación.


Figura 8. Codificación de Señales
Tomado de (Vignolo, J.)

Para el ejemplo de la figura 8. Se toma como base un sistema de código binario natural, para realizar la codificación de esta señal se observa que el código debe tener al menos 8 valores distintos por lo que se usa el código de la siguiente tabla:

Tabla 1. Código binario natural

| ***VALOR DISCRETO*** | ***CODIGO BINARIO*** |
|---|---|
| 0 | 000 |
| 1 | 001 |
| 2 | 010 |
| 3 | 011 |
| 4 | 100 |
| 5 | 101 |
| 6 | 110 |
| 7 | 111 |

## 1.3 Procesamiento de Imágenes

El procesamiento digital de imágenes es una ciencia desarrollada a partir de los años 60 y consiste en aplicar de técnicas a las imágenes digitales con el objetivo de mejorar la calidad de las imágenes o a su vez o facilitar la búsqueda de información, es decir se busca, emplear algoritmos sobre la imagen, para reconocer datos implícitos. Gracias a los avances de los dispositivos tales

como microprocesadores, se logró impulsar el procesamiento de imágenes con niveles altos de eficiencia. Hoy en día se encuentran diversos programas dedicados esencialmente para el procesamiento de imágenes digitales.

### 1.3.1 Procesamiento Digital de Imagen

#### 1.3.1.1 Imagen

Se defina a una imagen como la representación de objetos reales que se encuentran en tres dimensiones, en el plano de dos dimensiones, desde el punto de vista de la física, una imagen también se la considera con un objeto cuya intensidad luminosa varía de un punto a otro. Las imágenes en blanco y negro (monocromáticas) también se las representa como una función continua representada como una función f(x,y), en donde sus coordenadas son (x,y) y el valor de f es un proporcional a la intensidad de la luz ( intensidad luminosa , nivel gris) justamente en ese punto. En la figura 9 se indica el procesamiento de una imagen.

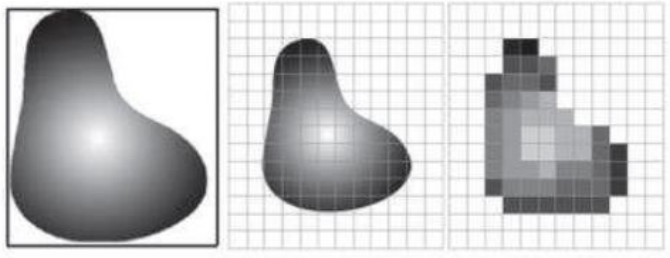


Figura 9. Digitalización de un objeto
Tomado de (García, P.)

Para realizar el procesamiento de una imagen es necesario que esta sea tratada por un computador, ya que es necesario someter la función f (x,y) a un proceso de discretización , en el cual se obtienen un número finito de valores los cuales puedan ser aumentados para poder aumentar la resolución de dicha imagen.

### 1.3.1.2 Digitalización en el procesamiento de imágenes

La digitalización en el procesamiento de imágenes como ya se ha indicado consiste básicamente en la descomposición de una matriz de M x N, donde cada una posee un valor proporcional a su nivel de gris, tal como se muestra en la figura 10. Dichos valores pueden ser cualquiera dentro de un rango continuo, para este procedimiento es correcto dividir dicho rango en una serie de k intervalos, para que de esta de forma el nivel de gris correspondiente a cada punto sea asignado en una serie de k intervalos, de esta forma cada nivel gris de cada punto será asignado a uno de los valores que representa dicho intervalo. Actualmente los sistemas de procesadores más eficientes trabajan con 256 niveles de gris.

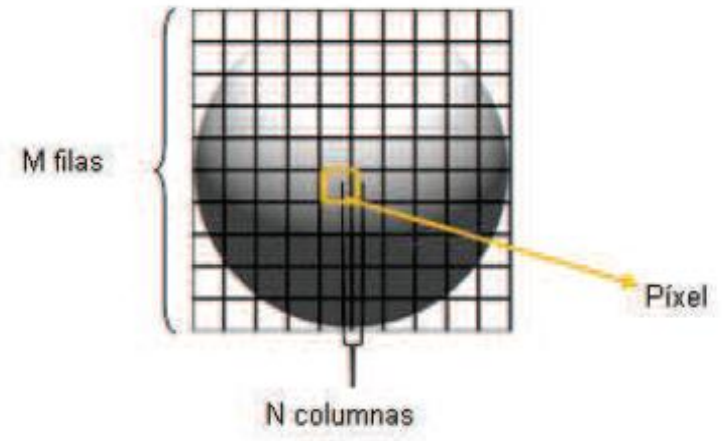



Figura 10. Descomposición de un objeto en una matriz MxN
Tomado de (García, P.)

Cada elemento resultante de la descomposición de la imagen recibe el nombre de pixel, en donde el número de niveles de gris y las dimensiones de la matriz condicionan la resolución de la imagen digital en otras palabras mientras la matriz de como resultado una mayor división de la imagen, mayor será el número de pixeles por lo que la resolución aumenta. Generalmente, se utiliza un octeto para almacenar cada uno de los pixeles, por lo que las intensidades de las imágenes se cuantifican en 256 niveles es decir con 8 bits.

Dependiendo del tamaño, a una imagen digital que contenga 256 niveles de gris y una dimensión de 256 x256 pixeles, ocupará 64 KB de memoria.

### 1.3.2 Etapas para el procesamiento digital de imágenes

Las etapas descritas son en la figura 11, describen básicamente las etapas realizadas con el software de procesamiento de imágenes.

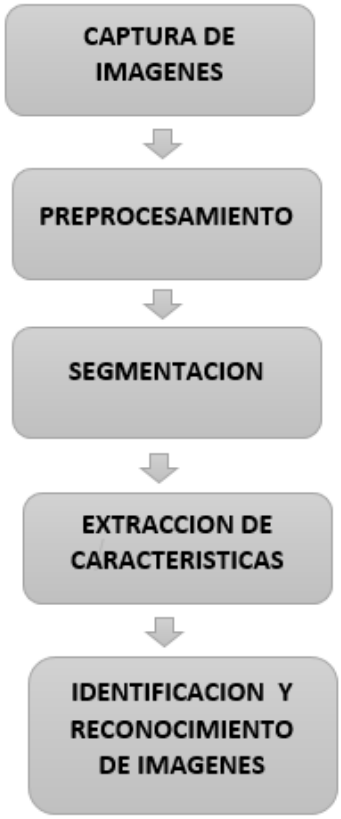



Figura 11. Etapas del procesamiento Digital de imágenes

#### 1.3.2.1 Captura de imágenes

Para el presente proyecto la captura se realiza por medio de una cámara digital. Este es el proceso por el cual se trata de convertir un objeto u imagen en una imagen digital, para posteriormente ser procesada de acuerdo a las

necesidades del usuario en donde se destacará algún aspecto de la información.

#### 1.3.2.2 Re-procesamiento de la imagen

El reprocesamiento es un conjunto de métodos que usan el filtrado de imágenes a nivel de pixeles para mejorar una imagen; mediante este proceso se optimizan zonas de las cuales se desea resaltar sus características; entre las principales operaciones matemáticas que se usan tenemos:

- Transformada Rápida de Fourier
- Dispersiones
- Convolución
- Desconvolución

A continuación, se detallan Las principales técnicas de filtrado:

- El filtrado por eliminación de ruido consiste en eliminar los pixeles cuyo nivel de intensidad es distinto al de sus pixeles vecinos.
- Al suavizar la imagen se reduce las diferentes variaciones de intensidad entre pixeles contiguos.
- El filtrado por detección de bordes consiste básicamente en la detección de los pixeles en donde se producen cambios perceptibles de la función intensidad.
- Al realzar los bordes se acentúa los bordes que se localizan en una imagen.

#### 1.3.2.3 Segmentación

En esta etapa de segmentación se divide la imagen en regiones las cuales van a ser manipulables a escala de grises, de esta manera las regiones que tienen los niveles de grises similares forman parte de un segmento, es decir se asocian por segmentos dependiendo de los noveles de grises que presentan, lo que conlleva a detectar los bordes de la imagen.

Uno de las etapas más críticas dentro del procesamiento de imágenes es el proceso de segmentación ya que de esta depende en gran parte la interpretación de resultados.

#### 1.3.2.4 Extracción de características

El proceso de extracción de características es un proceso que consiste básicamente en el reconocimiento de las características más relevantes de las imágenes encontrados en el proceso de segmentación, para de esta manera obtener la información a partir de dichas características exclusivas, como, por ejemplo: sus características a nivel de figura, tamaño, forma etc., mediante este proceso se le diferenciará de otras que son parte del mismo grupo o conjunto. Es decir, en esta etapa esencialmente se extraen las características más relevantes para la identificación de las imágenes deseadas.

#### 1.3.2.5 Identificación de imágenes

La identificación de imágenes es el proceso de reconocimiento o interpretación de manera automatizada, en donde se analiza las características de imagen con las etapas antes desarrolladas, para ser comparadas con una base de datos o; a su vez con algoritmos de toma de decisiones, los cuales están esencialmente diseñados para realizar el objetivo final del procesamiento de imágenes, que en nuestro caso en la detección e identificación de rostros y gestos faciales.

# Capítulo II. Diseño e implementación del Prototipo

Para el siguiente capítulo se describe el diseño y la implementación del prototipo de procesamiento de imágenes usando el Raspberry pi.

## 2.1. Diseño de sistema de procesamiento de señales

Se inicia con un diagrama de flujo en el cual se observa los principales módulos con los que se realizó el diseño del prototipo para procesamiento de imágenes.

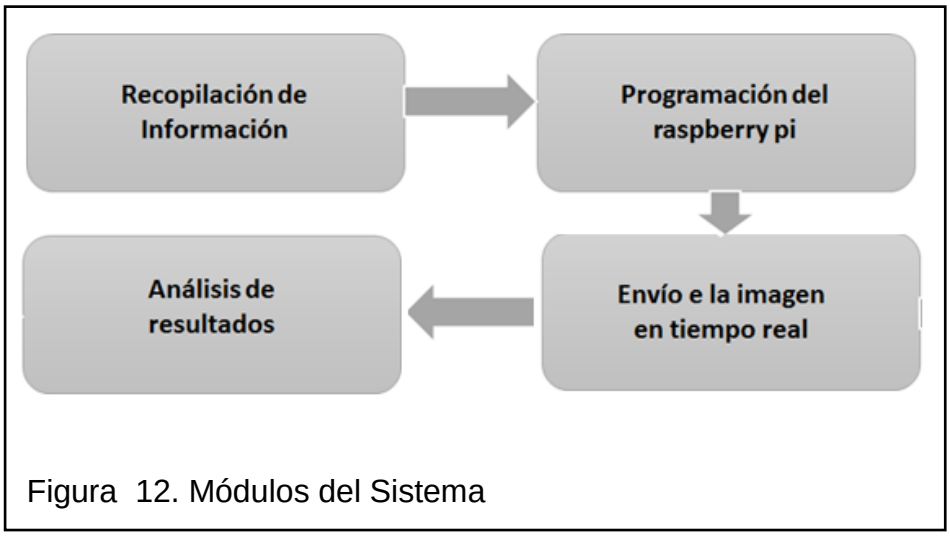



Figura 12. Módulos del Sistema

La cámara web es el periférico que se conecta al Raspberry y forma parte del primer módulo encargado de capturar las imágenes para luego ser procesadas, en el siguiente módulo se realiza todo el procesamiento mediante la programación adecuada y la inclusión de las librerías de procesamiento de imágenes correspondientes, una vez realizado el proceso se realiza una transmisión hacia un servidor FTP configurado previamente en el sistema operativo del Raspberry pi, al cual podemos acceder desde cualquier dispositivo conectado dentro de la red LAN.

En el módulo de programación se definen las tareas que se va a realizar para el caso del prototipo se definen:

- Detección de rostros
- Identificación de Personas
- Detección de sonrisas

### 2.1.1. Diagrama de interconexión

En la Figura 13, se observa el diagrama físico de conexiones entre los diferentes elementos, iniciando con la cámara la cual está conectada vía USB hacía el Raspberry; para que el Raspberry pueda acceder a una red local se interconecta a un access marca TP-LINK vía cable de red RJ-45; la particularidad de este access es que está conectado en forma de repetidor de otro router en nuestro caso el TP-LINK TL-WDR3600, con lo que tenemos una conexión inalámbrica hacía el router principal, y de esta manera se obtiene una salida a internet y una conexión inalámbrica para otros dispositivos como laptops.

.

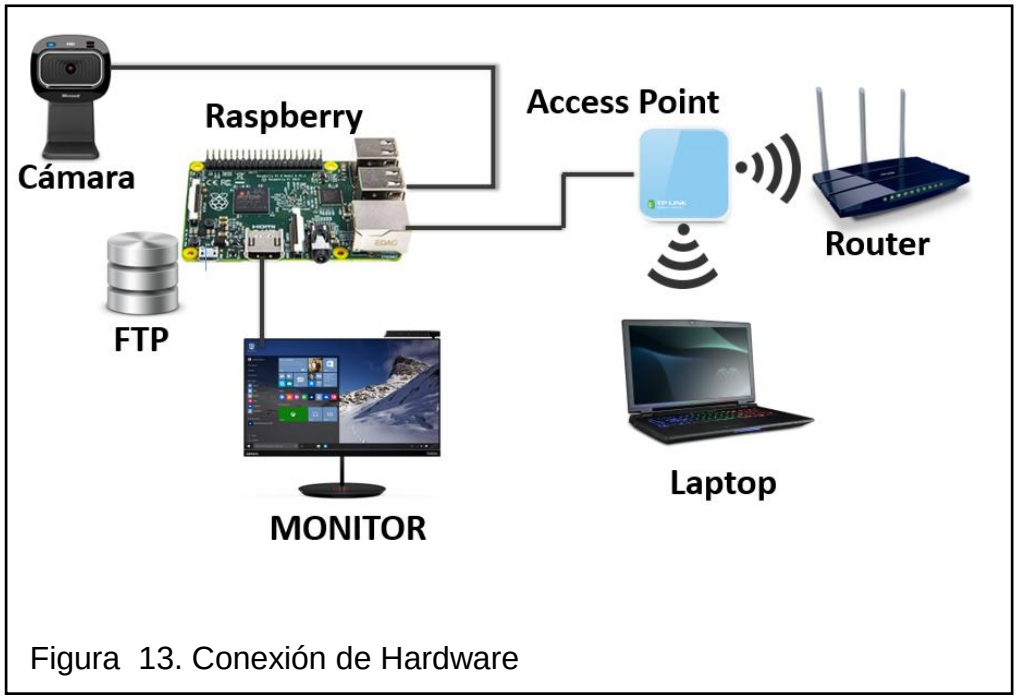



Figura 13. Conexión de Hardware

La importancia de tener un computador conectado a la red radica en que se realiza un acceso remoto al sistema operativo de Raspberry a través de Windows, con esto se facilita la programación y acceso.

### 2.1.2. Lista de componentes

De acuerdo con el diagrama de interconexión los componentes físicos o hardware constan de:

- Raspberry Pi 2
- Cámara Web
- Router nano TPLink
- Micro SD 16Gb
- Adaptador de Voltaje

## 1.2 Raspberry Pi

### 2.2.1 Definición

El Raspberry Pi es un computador on board del tamaño de una tarjeta de crédito, el cual incorpora todas las funciones de un PC de tamaño normal esto quiere decir procesamiento de hojas de textos, hojas de cálculo, navegadores, reproducción de vídeo de alta calidad, juegos y también brinda la posibilidad de programación en lenguaje Python y C; fue creado por la fundación Raspberry Pi, cuyo objetivo principal es la enseñanza de los sistemas de computación a la sociedad. Su nombre fue tomado con la decisión de crear una combinación entre una fruta y un computador, como ya había sucedido con Apple o Blackberry, su concepto es la de un computador que puede ser programado usando Python.

El Raspberry ha venido evolucionando a través de los años, el proyecto fue iniciado desde el 2006 pero su lanzamiento oficial al mercado se realizó en febrero de 2012, en sus primeros años de vida el Raspberry se basaba en un microcontrolador ATMEL ATmega664.

Para mayo de 2009 se crea la fundación Raspberry Pi en el Reino Unido con el objetivo de desarrollar el uso y entendimiento de los ordenadores a los niños.

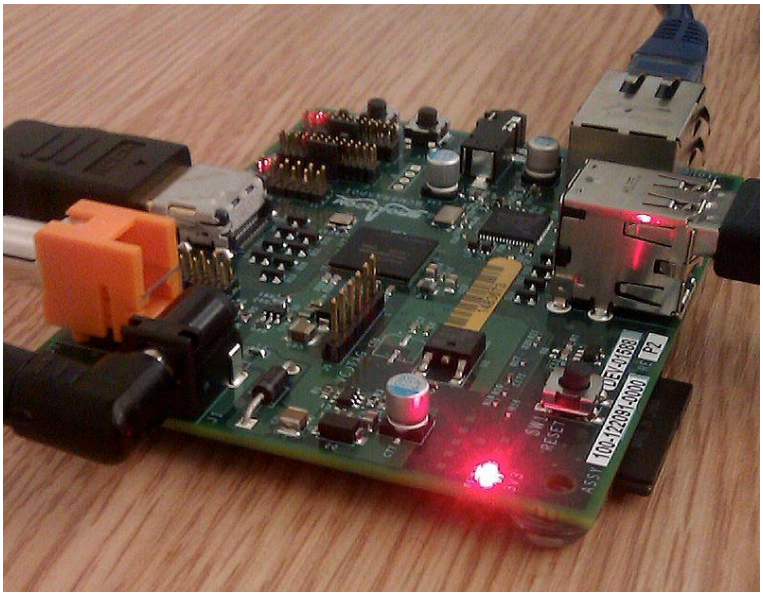


Figura 14. Prototipo inicial del Proyecto Raspberry
Tomado de (Horan, B.)

La fundación Raspberry brinda soporte para las descargas de las distribuciones de los sistemas operativos compatibles, y promueve el aprendizaje de Python lenguaje de programación incorporado por defecto.

*"La Fundación Raspberry Pi es una organización educativa benéfica registrada con sede en el Reino Unido. El objetivo de nuestra Fundación es promover la educación de los adultos y los niños, en particular en el campo de los ordenadores, informática y temas relacionados". [ ]*

La ventaja más significativa a parte de su reducido tamaño radica en el uso de software libre, con un sistema operativo basado en Linux con esto se facilita la configuración de servidores para diferentes tipos de aplicaciones.

### 2.2.2. Placa Raspberry pi

Con dimensiones de 8.5 x 5.6 [cm] y un peso aproximado de 45 [g], convierten al Raspberry en un computador de fácil portabilidad.

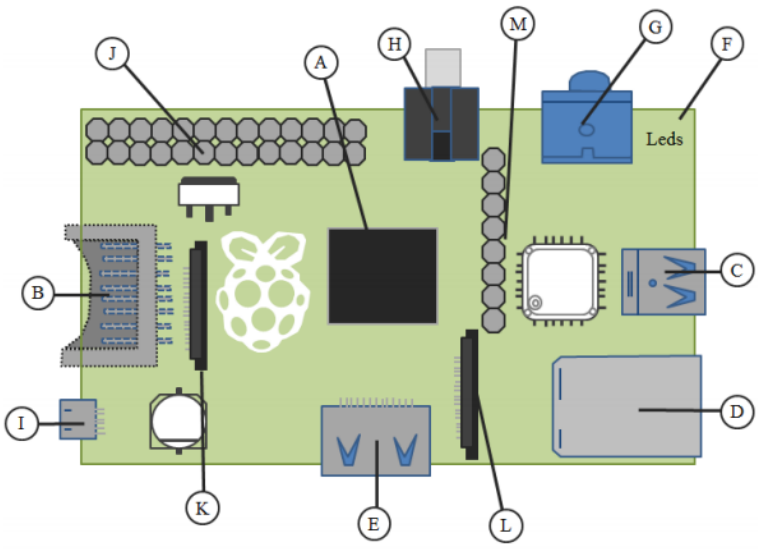



Figura 15. Esquema de la placa Raspberry Pi
Tomado de (Raspberry Org.)

En la figura 15 se muestra la distribución de los principales componentes de la placa y los cuales se describen a continuación.

**A-El Procesador**

Como en todo computador el procesador es el elemento más importante, encargado de interpretar las funciones de los programas y entornos almacenados y configurados en el computador.

**B-Ranura para microSD**

Ya que se necesita un dispositivo de almacenamiento como HDD se incorpora una ranura para microSD.

**C-Puerto USB**

Se trata de puertos que cumplen las especificaciones USB 2.0 de uso general para comunicación con periféricos, en los primeros modelos se tenía un consumo restringido a 100[mA] lo cual limitaba el uso de periféricos.

**D-Puerto Ethernet**

Para las conexiones hacia la red, el Raspberry cuenta con un puerto ethernet RJ45, ahora bien, para la conexión con los dispositivos como routers, switches o PC, la raspberry cuenta con sistema auto-MDI lo que le permite sensar previamente el dispositivo conectado y ajustarse con configuraciones de cable cruzado o normal.

**E-Puerto HDMI**

El puerto HDMI incorporado en la placa dispone de video digital y salida de audio, la cual soporta 14 diferentes resoluciones, es compatible únicamente con dispositivos que tengan incorporado este puerto para la conexión de video.

**F-Led Status**

Identifican el estado de la placa cuando se encuentra en funcionamiento

| LED | COLOR | SIGNIFICADO |
|---|---|---|
| ACT | Verde | Enciende cuando hay transmisión de la MicroSD |
| PWR | Rojo | Muestra que el sistema este alimentado con 3.3 [v] |

**G-Salida de audio análoga**

La placa también dispone de un conector de audio estándar de 3.3mm diseñado para manejar cargas de alta impedancia como bocinas, existe un problema con el controlador de audio que provoca bajo sonido cuando se conectan audífonos.

**H-Puerto RCA**

Proporciona una salida de la señal de video en formato PAL o NTSC para la conexión con dispositivos que dispongan de esta entrada, sin embargo la resolución en comparación con la salida HDMI es muy baja.

**I-Alimentación de energía**

Consta de un puerto micro USB el cual funciona como entrada de corriente, se recomienda el uso de fuentes iguales o superiores a 1[A].

**J-Entradas y salidas de propósito general (GPIO)**

Se trata de un conjunto de pines los cuales nos abren las puertas para la conexión y control de dispositivos eléctricos a través de programación mediante Python.

El Puerto Gpio es un conjunto de 26 pines usados con diferentes propósitos, algunos de estos pines llevan voltaje para alimentar leds, y de igual manera para cerrar el circuito se dispone de pines de tierra, es necesario identificar cada uno de estos pines para realizar una conexión correcta.

**K-Interfaz de monitor serial (DSI)**

Se trata de un puerto con 15 pines que sirven para la conexión con pantallas LCD, con esto se puede visualizar el sistema en pantallas touch.

**L-Interfaz de cámara serial (CSI)**

Diseñado para la conexión de la Pi Camera, cámara onboard diseñada por la fundación Raspberry

Se pueden definir dos modelos en la distribución de las placas Raspberry pi, el modelo A y el modelo B los cuales han evolucionado durante estos años incorporando cambios y mejoras en el hardware.

Tabla 2. Tabla comparativa A modelos Raspberry

| |  |  |  |
|---|---|---|---|
| Raspberry Pi | Modelo A | Modelo A+ | Modelo B |
| SoC | Broadcom BCM2835 | Broadcom BCM2835 | Broadcom BCM2835 |
| CPU | 700MHz ARM1176JFZ-S | 700MHz ARM1176JFZ-S | 700MHz ARM1176JFZ-S |
| GPU | VideoCore IV | VideoCore IV | VideoCore IV |
| RAM | 256Mb | 256Mb | 512Mb |
| USB | 1 | 1 | 2 |
| Video | RCA, HDMI | Jack, HDMI | RCA, HDMI |
| Audio | Jack, HDMI | Jack, HDMI | Jack, HDMI |
| Boot | Memoria SD | Memoria microSD | Memoria SD |
| Wireless | No tiene | No tiene | No tiene |
| Red Ethernet | No tiene | No tiene | Ethernet 10/100 |
| Alimentación | 5V / 2Amp | 5V / 2Amp | 5V / 2Amp |
| GPIO | 26 pines GPIO | 40 pines GPIO | 26 pines GPIO |
| Tamaño | 85,6 x 53,98 mm | 65 x 56 mm | 85,6 x 53,98 mm |

Tabla 3. Tabla comparativa B modelos Raspberry

|  |  |  |
|---|---|---|
| Modelo B+ | RPi V2 modelo B | RPi 3 modelo B |
| Broadcom BCM2835 | Broadcom BCM2836 | Broadcom BCM2837 |
| 700MHz ARM1176JFZ-S | 900MHz Quad-core ARM Cortex-A7 | 1.2Ghz Quad Cortex A53 |
| VideoCore IV | 250Mhz VideoCore IV | 400Mhz VideoCore IV |
| 512Mb | 1Gb | 1Gb |
| 4 | 4 | 4 |
| Jack, HDMI | Jack, HDMI | Jack, HDMI |
| Jack, HDMI | Jack, HDMI | Jack, HDMI |
| Memoria microSD | Memoria microSD | Memoria microSD |
| No tiene | No tiene | 802.11n / Bluetooth 4.1 |
| Ethernet 10/100 | Ethernet 10/100 | Ethernet 10/100 |
| 5V / 2Amp | 5V / 2Amp | 5V / 2,5Amp |
| 40 pines GPIO | 40 pines GPIO | 40 pines GPIO |
| 85 x 56 x 17 mm | 85 x 56 x 17 mm | 85 x 56 x 17 mm |

### 2.2.3. Software

El Raspberry está diseñado para correr con distribuciones de Linux modificadas desde su núcleo para ser usadas en la placa, ya que se debe tomar en cuenta la limitación de la memoria RAM y del nivel de procesamiento.

En la página oficial del Raspberry se encuentran las opciones de descarga de las imágenes ISO para la instalación del sistema, la distribución de Linux recomendada es Raspbian, aunque también se encuentran disponibles otras distribuciones incluyendo una imagen para desarrolladores basada en Windows 10, en la figura 16 se muestran las diferentes distribuciones que se pueden instalar en el Raspberry.

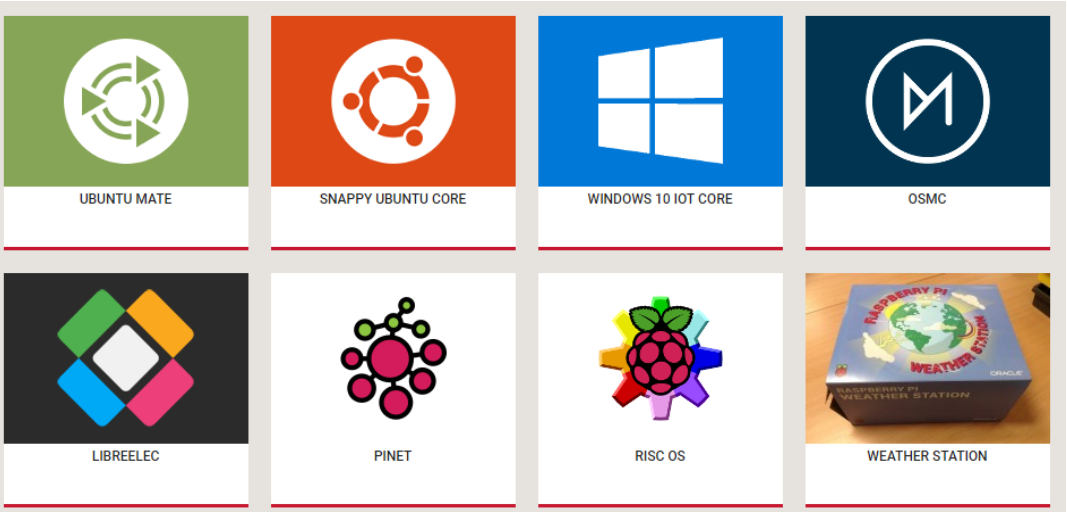



Figura 16. Distribuciones disponibles para el Raspberry
Tomado de (Cox, T.)

### 2.2.4. Aplicaciones

Las aplicaciones en las que se desee usar el Raspberry van a depender del alcance del proyecto que se proponga a realizar. Esto se debe a que se trata de una placa de aprendizaje y a que se tiene la limitación de algunos

elementos como la memoria RAM, las más importantes aplicaciones son las que se describen a continuación:

**Educación**

El principal uso para el que fue desarrollado el Raspberry fue para la educación, para que todas las personas desde niños hasta personas adultas se puedan interesar en el campo de la informática y tengan acceso a las herramientas informáticas actuales.

**Servidor**

Ya que el sistema operativo que maneja el Raspberry es una distribución de Linux es posible instalar varios aplicativos de servidores como DNS o FTP para y de esta manera obtener un mayor provecho de este mini computador.

**Programación**

Se tiene la capacidad de realizar programación en varios lenguajes como Java, C, C++, Python, etc. el lenguaje de programación por defecto es Python, lenguaje con un gran potencial y muy fácil de usar en cualquier tipo de aplicación.

## 2.3. Componentes Adicionales

### 2.3.1. Cámara Web/ PiCamera

El dispositivo encargado de captar las imágenes que luego serán procesadas es la cámara web; para el prototipo se usa una cámara web marca Microsoft de 5MPX; el Raspberry al ser un computador onboard dispone de varios puertos e interfaces para la conexión de dispositivos, la cámara web se conecta vía USB, pero también dispone de un puerto único para la conexión de la PiCamera llamado CSI que está diseñado específicamente para la conexión de la PiCamera, la cual es onboard diseñada específicamente para el Raspberry y que se conecta al puerto CSI a través de un bus de cable flexible, también tiene

una resolución de 5Mpx y soporta video de 1080p, esencialmente es análoga a la cámara de nuestro smartphone.

Figura 17. Cámara Pi


Figura 18. Cámaras Web

### 2.3.2. Micro SD 16GB class 10

De igual manera que un PC, el Raspberry necesita un dispositivo de almacenamiento en el cual se grabe el sistema operativo de arranque; para el Raspberry se trata de una micro SD de 16GB clase 10, estas memorias pueden venir en formato de clase 10 y clase 4, lo que nos indica la clase es la velocidad a la que van a trabajar, una memoria clase 10 trabaja a mayor velocidad que una de clase 4. La capacidad mínima de almacenamiento requerida es de 4GB.


Figura 19. Micro SD

### 2.3.3. Adaptadores de corriente 1.5 A

La fuente de alimentación se conecta a través de un puerto micro USB que se encuentra en la parte izquierda de la placa, como mínimo por especificaciones técnicas se recomienda usar un adaptador de 5V] a 1500mA; a medida que usemos una mayor cantidad de periféricos USB se debe contemplar un adaptador de mayor capacidad, con lo cual no tendremos problemas de intermitencia en nuestra placa.

### 2.3.4. Router Nano TP-Link TL-WR702N

Se trata de un mini router el cual tiene varias formas de configuración para su uso, y que trabaja a una velocidad de 150Mbps; consta de un puerto Ethernet, un puerto micro USB para la alimentación de 5v, con lo cual puede ser alimentado con el USB de la laptop sin inconvenientes.

Las formas de trabajo que posee este router son Access Point AP (default), Client, Router, Repetidor y Puente, soporta el estándar 802.11n y tiene compatibilidad con los viejos estándares 802.11b/g.

Su frecuencia de trabajo está en la banda de 2.4GHZ y soporta seguridades con encriptación WEP, WPA/WPA2, WPA-PSK/WPA2-PSK para brindar seguridad a la red que se esté configurando.


Figura 20. Router nano TPLINK

## 2.4. Software de Gestión

Al ser el sistema operativo una distribución de Linux se realiza el estudio de paquetes de software y bibliotecas compatibles con el sistema operativo, de esta manera se verifica compatibilidad en Software.

### 2.4.1. Python

Es el lenguaje de programación incorporado en el sistema Raspbian, está diseñado para ser un lenguaje de programación de uso fácil.

Fue desarrollado a finales de 1980, desde sus inicios la filosofía de Python ha sido hacer que el código sea legible y accesible, usa tipiado dinámico lo cual significa que una variable puede tomar varios valores en diferentes momentos.

Raspbian, sistema operativo del Raspberry tiene incorporado una versión de PYTHON 2.7 la cual puede ser actualizada a las versiones 3.0 o superior, la estructura que se maneja en Python es de alto nivel, con lo que se ve un enfoque a simplificar las sentencias y la orientación a objetos.

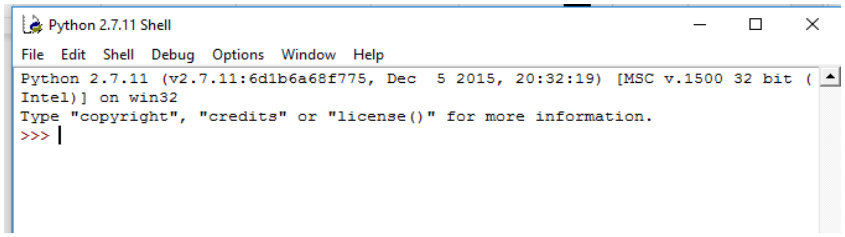



Figura 21. IDLE de Python

Aunque se puede utilizar versiones superiores de Python como la 3.0, para el prototipo se trabajó con la versión 2.7 que es más estable y cuya documentación de uso de sentencias se encuentran en su página oficial:

https://www.python.org/

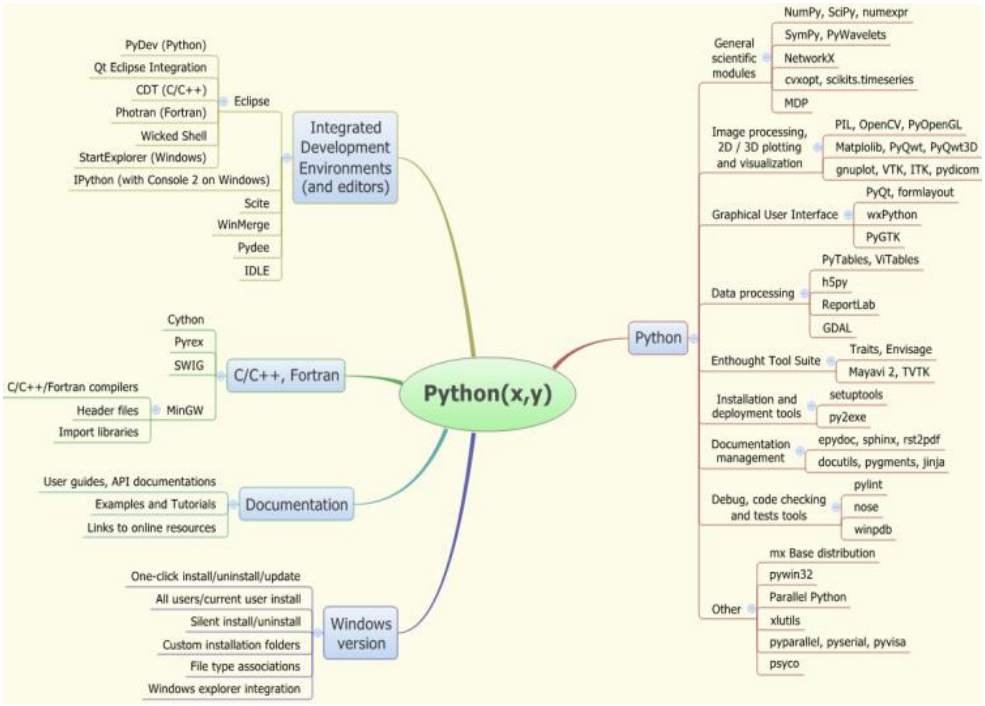



Figura 22. Principales bibliotecas para Python
Tomado de (Norris, D.)

En la Figura 22, se observa la potencialidad que se obtiene al usar Python con la implementación de las diferentes librerías, para el prototipo se emplearán la

mayoría de la parte derecha de la figura las cuales serán descritas posteriormente.

Para reducir la complejidad de las sentencias escritas en Python no se emplean llaves para cerrar las sentencias, por el contrario, para saber que estamos escribiendo dentro de una sentencia Python usa la posición del tabulador para saber que estamos en el flujo correcto de la sentencia.

### 2.4.2. Librerías

#### 2.4.2.1. OpenCv

Open Source Computer Vision Library, se trata de una biblioteca de código abierto creada por Intel y Willow Garage, en la cual se encuentran diferentes algoritmos que facilitan el procesamiento de imágenes para sistemas de visión artificial. Se encuentra disponible para lenguajes de programación C, C++, Python, Java etc. y soportado en diferentes plataformas de sistemas operativos como Window, Linux, OsX, Android, iOS etc.

Opencv está creado de una manera modular, lo cual significa que un paquete incluye varias sub librerías.

Los principales módulos para procesamiento son:

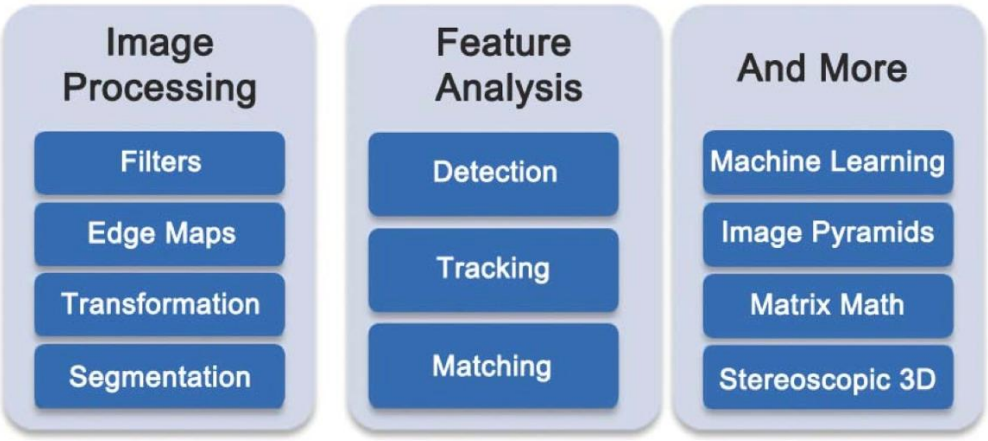



Figura 23. Módulos de Procesamiento para OPENCV

#### 2.4.2.2. QT

Se trata de una biblioteca multiplataforma la cual es muy usada para el desarrollo de aplicaciones en las cuales se incluyen interfaces gráficas de usuario. QT está desarrollado como software libre y de código abierto.

PyQt es una extensión de la biblioteca QT el cual es usado para la programación en lenguaje Python, y está disponible para varios sistemas operativos y en diferentes versiones de licencias.

A continuación, se escribe un pequeño ejemplo para la creación de pantallas en QT.

```
import sys
from PyQt import QtGui
class ventanaprincipal(QtGui.QMainWindow):
        def__init__(self):
                super(ventanaprincipal, self).__init__()
                self.setWindowTitle("hola mundo")
app=QtGui.QApplication(sys.argv)
ventana=ventanaprincipal()
ventana.show()
sys.exit(app.exec_())
```

El resultado es el siguiente:

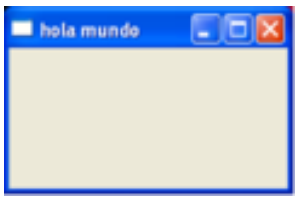



*Figura 24. Pantalla Programada con QT*

#### 2.4.2.3. Fisherfaces

El reconocimiento de rostros es un método biométrico usado para identificar a individuos, con varios objetivos los cuales pueden ser de índole comercial o de seguridad, dentro de las primeras se puede distinguir para uso de tarjetas de crédito, pasaportes, licencias, y controles de acceso, las aplicaciones de seguridad pueden distinguirse reconocimientos para retratos policiales, vigilancia por video, etc.

Existen varios métodos para poder realizar el reconocimiento de rostros, en cualquiera de ellos el proceso que se debe seguir es tomar una imagen en dos dimensiones (filas, columnas), la cual es transformada en un vector unitario a su vez contenido en un espacio n-dimensional(n=filasxcolumnas). El paso siguiente es extraer la imagen promedio y se proyecta un vector resultante en un espacio de menor dimensión que la imagen original; esta proyección es la que se va a comparar con un conjunto de imágenes compuesta en una base, el resultado más similar corresponde al resultado de reconocimiento.

El método conocido como fisherfaces usa el discriminante lineal de Fisher (FLD), para la reducción de dimensión, es un método de reconocimiento facial en el cual se toma en cuenta cómo se va reflejando la luz en el rostro.

Para el cálculo de los fisherfaces se define una matriz de distribución entre clases (SB) y la matriz de distribución intraclases (SB), es el resultado de X1 las cuales son el conjunto de imágenes y Ni es el número de imágenes de Xi.

$$S_B = \sum_{i=1}^{c} N_i (u_i - u)(u_i - u)^T$$

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in \mathrm{X}} N_i (x_k - u_i)(x_k - u)^T \qquad \text{(Ecuación 3)}$$

### 2.4.3. Raspbian

Una de las diferencias más significativas entre los PC que usamos para nuestras tareas básicas de estudio o trabajo en relación con el Raspberry, es su sistema operativo, al contrario de usar Windows o Apple OS x esta pequeña placa usa una distribución de Linux con base en Debian, llamada Raspbian, el cual es de código abierto lo que nos permite hacer cambios en el código fuente del sistema. Es el sistema oficial y recomendado para la Raspberry pi y lo podemos descargar de la siguiente dirección:

https://www.raspberrypi.org/downloads

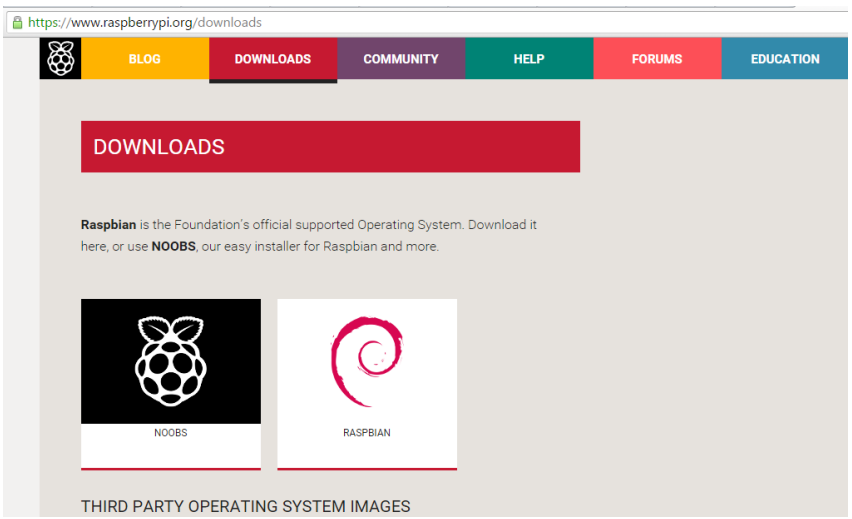



Figura 25. Página de descarga para el sistema operativo Raspbian
Tomado de (Raspberry, Org.)

Los paquetes preinstalados en el sistema son totalmente compatibles con el software libre que se encuentra en cualquier distribución de Linux, como pueden ser: procesadores de texto (open office), software para cálculo y gráficos, programación (Python).

Para esta distribución se usa LXDE *"Lightweight X11 Desktop Environment"* como escritorio, y proyecta un entorno de escritorio ligero y rápido, otra de las

mejoras es el menú raspi-config con el que se pueden realizar configuraciones generales del sistema sin tener que realizar cambios manuales en los archivos de configuración.

## 2.5 Implementación del sistema de procesamiento

Con el conocimiento de todas las herramientas que forman parte del prototipo, tanto de hardware como software de debe iniciar con el proceso de implementación, el cual consiste en juntar tanto las partes físicas como la configuración y programación.

### 2.5.1 Instalación del sistema operativo

El primer paso para el proceso de instalación es preparar el Raspberry, para lo cual debemos realizar la instalación del sistema operativo; como ya se explicó anteriormente se trata del Raspbian Jessie, lo primero que se realiza es descargar la imagen ISO del sistema que está disponible en la página web, como en toda instalación se necesita un dispositivo de almacenamiento en el caso del prototipo es la tarjeta micro SD, y con ayuda del programa Win32Disk se graba la imagen en la micro SD, de esta manera la SD se convierte en un dispositivo booteable y cargado con el Raspbian Jessie.

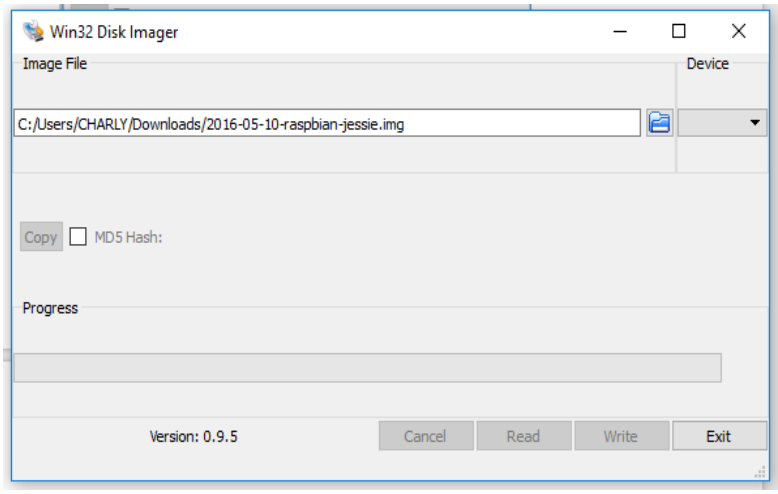



Figura 26. Win32 Disk Manager

Se inserta la SD en la ranura del Raspberry y se conectan los periféricos necesarios como pantalla, mouse y teclado como se muestra en la figura. Finalmente se energiza la placa con el adaptador de voltaje, y se deben realizar las configuraciones como nombre de usuario, password, idioma, red etc. La instalación brinda las opciones de instalar el sistema en modo consola o modo gráfico; para una mayor potencialidad se escoge la opción del sistema en modo gráfico.

Figura 27. Conexión de Periféricos

Una vez finalizada la instalación del sistema tenemos un computador operativo y listo para ser empleado dependiendo de las necesidades de su usuario. Al finalizar la instalación el primer paso que debemos realizar es la expansión de la micro SD, este paso se lo realiza para que la capacidad total sea reconocida en el sistema de esta manera tenemos un total de 4GB usado para el sistema y el resto para aplicaciones que se deseen instalar.

Figura 28. Escritorio Raspbian

### 2.5.2 Configuración del router nano TPLink

Para la configuración del Router nano TPLINK se emplea el modo de trabajo tipo repetidor, esto quiere decir que el router nano va a replicar la señal Wireless de otro router que se encuentre en el rango de cobertura.

El primer paso como se muestra en la figura 29, es la conectar el router, de cualquiera de las dos formas con un adaptador o a través del USB de la laptop.

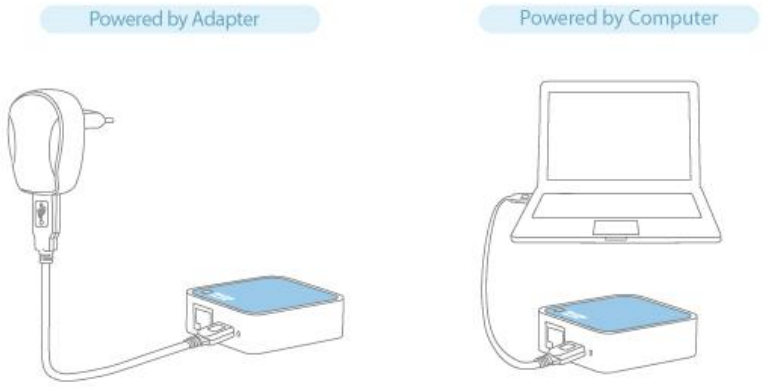



Figura 29. Conexión de Fuente de Poder Router Nano
Tomado de (TP-LINK)

Para acceder a la página de configuración se conecta la laptop a la red que da por defecto el router, y acceder a la dirección 192.168.0.254 en el navegador, en este punto la página de presentación nos solicita un usuario y contraseña que por defecto son: admin en los dos casos.

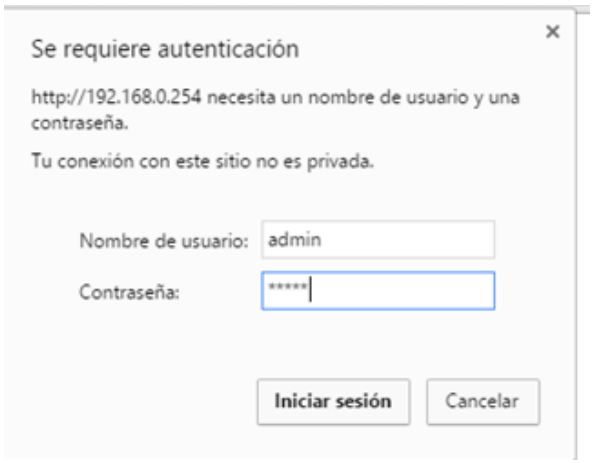



Figura 30. Autenticación para ingresar al Router

Una de las ventajas de este router es su sistema gráfico el cual permite realizar una configuración fácil y rápida gracias a su asistente de configuración, con el cual en pocos pasos podemos realizar la conexión deseada. Se debe recordar que con este tipo de conexión el nano router lo va a extender la señal de un router primario al que está conectado.

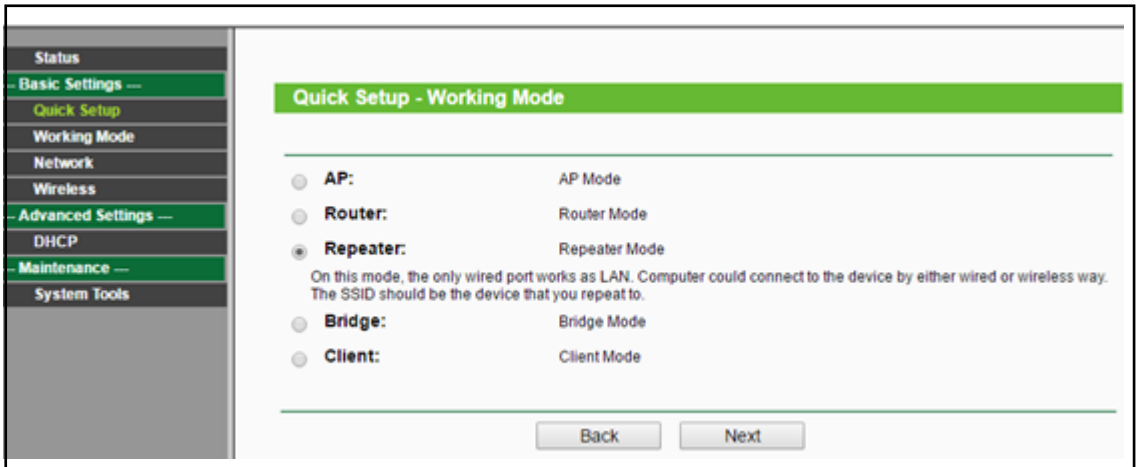



Figura 31. Menú de modo de trabajo del Router

Finalmente se realiza un barrido de redes, en el panel de configuración se muestran las redes que se encuentran al alcance del nano router, una vez

seleccionada la red se llena el formulario de configuración con los datos de la red principal como se muestra en la figura 32.

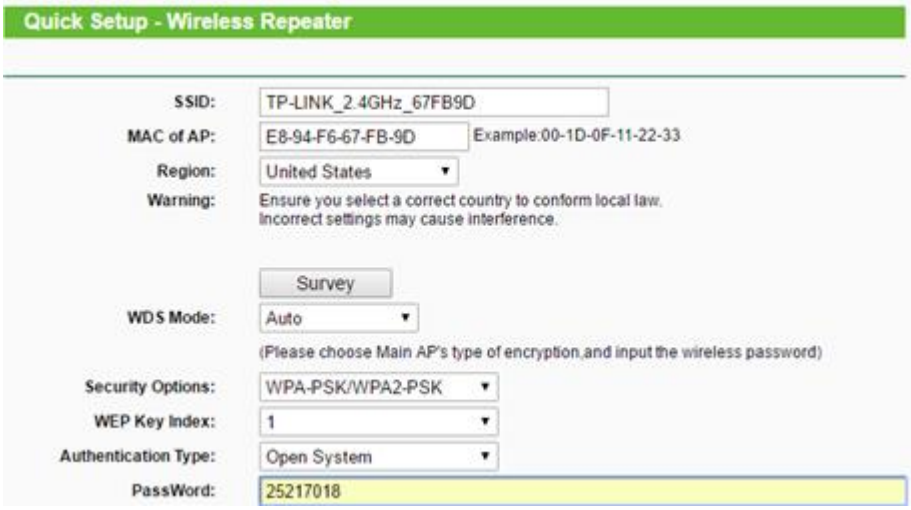



Figura 32. Configuración del Router como repetidor

Siguiendo el diagrama de conexión se conecta el Raspberry con un patch cord hacia el nano router, el sistema está configurado por defecto para recibir la configuración de red mediante DHCP como se muestra en la figura 33.


Figura 33. Asignación DHCP para el Raspberry

Al tener configuración mediante DHCP cada vez que se desconecte la red, por algún motivo o al desconectar el Raspberry, la asignación de IP va a cambiar, para evitar esto tenemos dos opciones, la primera es asignar una IP estática en el Raspberry, o reservar una IP en el router principal mediante la MAC. En el diseño se contempla la segunda opción (se debe verificar si el router a usar tiene esta opción), la configuración es sencilla y se muestra en la figura 29, en

el formulario se deben colocar la Mac del Raspberry, la IP que se va a reservar (192.168.0.106) y habilitar la función.

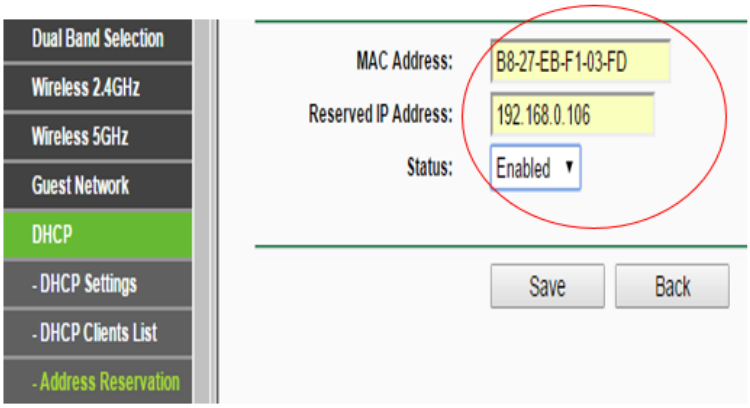



Figura 34. Configuración DHCP permanente para el Raspberry

Por último, se realiza el reset del router para que la configuración sea aplicada. Se verifica si los cambios fueron aplicados, como se puede observar en la figura 35, el Raspberry cuenta con su asignación de IP permanente, de esta manera por más que se desconecte el Router o el Raspberry la dirección IP asignada no va a cambiar.

| | | | | |
|---|---|---|---|---|
| 7 | raspberrypi | B8-27-EB-F1-03-FD | 192.168.0.106 | Permanent |

Figura 35. Comprobación DHCP permanente

### 2.5.3 Configuración de Escritorio remoto

Para obtener visualización del escritorio de Raspbian se conectó un monitor con puerto HDMI, pero una forma de optimizar este proceso es instalar el paquete de Escritorio Remoto en el sistema Raspbian a través de línea de comando con lo cual se instala el paquete necesario para realizar la conexión mediante escritorio remoto.

*sudo apt-get install rdesktop*

Para acceder remotamente desde un sistema operativo como Windows se necesita conocer, la IP (192.168.0.106) del Raspberry, así como el usuario y contraseña (pi, Raspberry); se coloca esta información como se muestra en la figura 37 y se logra acceder al escritorio principal. La ventaja de esta conexión radica en que se simplifica el uso de periféricos ya que no se utiliza ni teclado ni mouse ni pantalla, toda la programación e instalación de paquetes se la realiza mediante conexión remota.

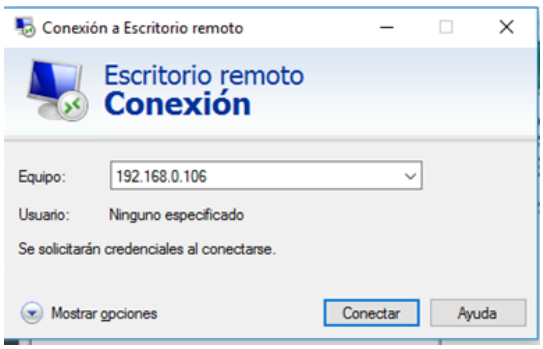



Figura 36. Inicio de Conexión Remota

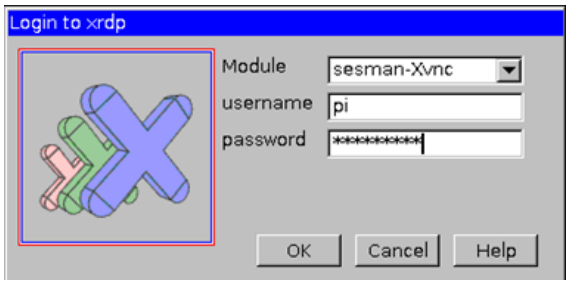



Figura 37. Autenticación con el Raspberry

Como se observa en la figura 36, se accede al escritorio de Raspbian mediante la conexión remota hacia la IP 192.168.0.106, de esta manera se puede acceder al Raspberry para iniciar con la instalación de los paquetes y aplicativos según el proyecto a desarrollar.

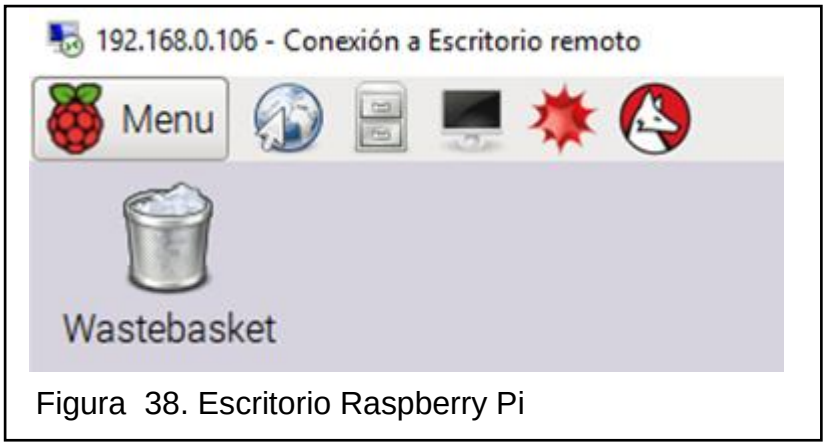



Figura 38. Escritorio Raspberry Pi

### 2.5.4 Instalación de dependencias

A pesar que con las versiones preinstaladas de Python en Raspbian se puede iniciar con diferentes proyectos de programación, para el prototipo de procesamiento digital de imágenes es necesaria la instalación de diferentes librerías de procesamiento de imágenes compatibles con Python.

#### 2.5.4.1. Instalación de PIP (Python Package Index)

Se inicia con la instalación del paquete PIP (Python Package Index), herramienta que ayuda a realizar la instalación de paquetes directamente del sitio web oficial de Python sin necesidad de ingresar más líneas de comando, para instalarlo se ingresa mediante línea de comandos:

*sudo apt-get install python-pip*

Una vez que finaliza la instalación debemos realizar el upgrade de la herramienta; mediante línea de comando

*sudo pip install -U pip*

#### 2.5.4.2. Instalación de Opencv

Con PIP se realiza la instalación de Opencv en su versión 2.4 con la siguiente línea de comando:

*sudo pip install opencv*

Una vez finalizada la instalación verificamos que la librería está incluida y que es la versión 2.4 ya que versiones superiores aún se encuentran en versiones beta; accedemos al IDLE de Python e importamos la biblioteca de cv2, se observa que no hay errores y una de las propiedades de la biblioteca cv2 es la versión la cual se verifica que es la 2.4.12, es importante que sea esta versión ya que es compatible con todas las herramientas adicionales para el prototipo.

*>>>import cv2*
*>>>cv2.__version__*
*'2.4.12'*
*>>>*

#### 2.5.4.3. Instalación de extensiones para imágenes

En el prototipo es necesario trabajar con diferentes formatos de imágenes como jpg, gif, png, etc; para esto es necesario instalar paquetes especiales de imágenes esta instalación se realiza mediante línea de comando con la siguiente instrucción.

*$ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev*

De igual manera que se necesitan paquetes I/O para imágenes, también se necesita paquetes para video, y de igual manera que en el caso anterior se instalan mediante línea de comando con la instrucción.

*$ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev*
*$ sudo apt-get install libxvidcore-dev libx264-dev*

#### 2.5.4.4. Instalación de Fisherfaces y librerías especiales

El prototipo de procesamiento de imágenes se basa en el método de reconocimiento llamado Fisherfaces, el cual puede instalado ejecutando el archivo setup.py el cual se puede descargar de la página del proyecto de reconocimiento facial Facerec.

https://github.com/bytefish/facerec/tree/master/py/facerec

Para poder usar el método de reconocimiento Fisherfaces es necesario la instalación de cuatro librerías adicionales; las mismas que incluyen métodos de procesamiento matemático para la manipulación de las imágenes.

**PIL**

Python Image Library (PIL) añade habilidades de procesamiento de imágenes para el intérprete de Python, es compatible con los principales formatos de imágenes; la versión en la que se encuentra actualmente es la 1.1.7 y es compatible con versiones de Python 1.5 hasta 2.7; la instalación se la realiza mediante línea de comando:

*sudo pip install pil*

**NUMPY**

Agrega soporte a Python para procesamiento de vectores y matrices a través de una biblioteca de funciones matemáticas de alto nivel; su instalación se la realiza mediante línea de comando con la siguiente instrucción:

*sudo pip install numpy*

Verificamos la instalación abriendo el IDLE de Python e importando la biblioteca con su versión.

```
>>> import numpy
>>> numpy.__version__
'1.11.0'
>>>
```

**SCIPY**

Es una biblioteca open source de algoritmos matemáticos se encuentra actualmente en la versión 0.9.0; su instalación se realiza mediante línea de comando con la siguiente instrucción:

*sudo pip install scipy*

De la misma manera comprobamos la instalación importando la biblioteca y verificando su versión.

```
>>> import scipy
>>> scipy.__version__
'0.9.0'
```

**MATPLOTLIB**

Es una biblioteca que se usa para la generación de gráficos partiendo de listas de vectores o arrays en lenguaje de programación Python, trabaja conjuntamente con Numpy; su instalación se realiza mediante línea de comando con la siguiente instrucción:

*sudo apt-get install python-matplotlib*

Verificamos que se encuentre instalado y que la versión sea la correcta

```
>>> import matplotlib
>>> matplotlib.__version__
'1.5.1'
```

### 2.5.5. Instalación del servidor FTP

Con la finalidad de almacenar las imágenes procesadas, es necesario la instalación de un servidor FTP en el Raspberry, con la finalidad de acceder a las imágenes desde cualquier dispositivo conectado a la red en la que se encuentra configurado el servidor. Para esto se inicia con la instalación del paquete del servidor mediante líneas de comando

*sudo apt-get install proftpd*

A continuación, se muestra la pantalla de configuración para la instalación escogemos la opción:

*Ejecutar proftpd: independiente*

Esperamos que la instalación finalice y realizamos la configuración; para esto cambiamos parámetros en el archivo de configuración ubicado en la siguiente dirección nano /etc/proftpd/proftdp.conf.

Añadimos las siguientes etiquetas para la configuración y reiniciamos el servidor con lo que ya se tiene la comunicación con el servidor FTP.

Servername “FTPDebian”
DefaultRoot /ftp
<Limit WRITE > AllowAll

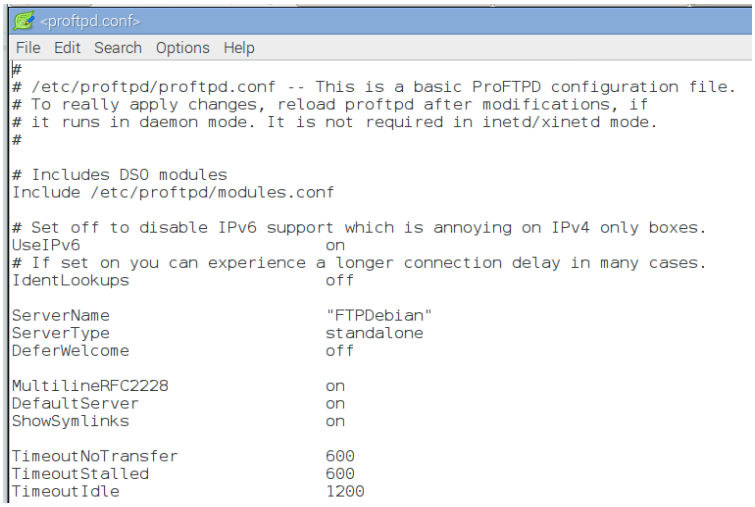



Figura 39. Archivo de configuración Proftdp

Finalmente se dan los permisos a la carpeta que va a servir como repositorio de las imágenes; se ubica la carpeta ftp y se escribe el comando

chmod 777 /ftp/ -R

Change mode “chmod” permite cambiar los permisos de acceso a un directorio o a un archivo, 777 se da como resultado de una combinación octal la cual otorga permisos de lectura, escritura y ejecución.

Se realiza una copia de imágenes en la carpeta ftp del servidor para realizar las pruebas; estos archivos posteriormente van a ser visualizados desde los dispositivos

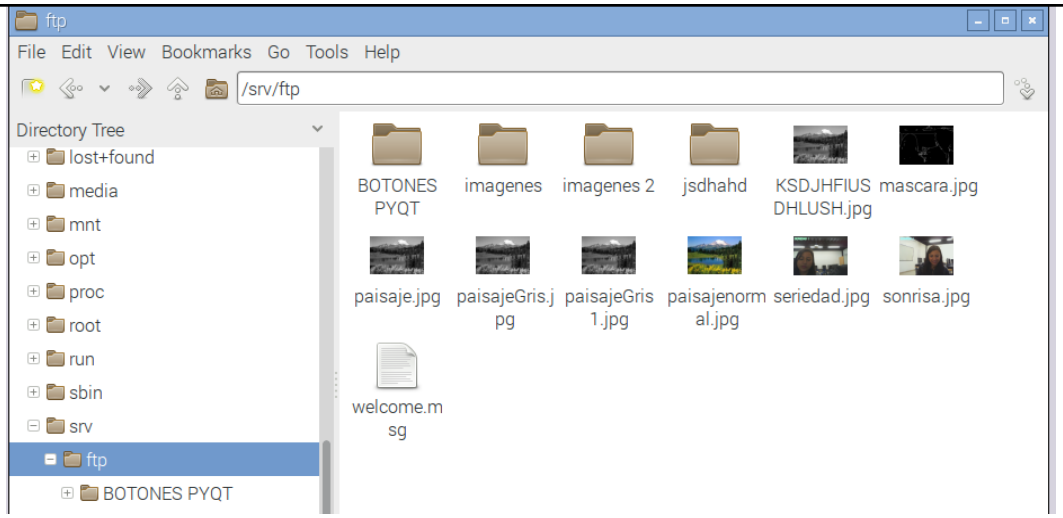



Figura 40. Carpeta de almacenamiento servidor FTP

Verificamos que se tiene acceso al servidor mediante un ping al servidor en este caso ping 192.168.0.106 y se comprueba que existe comunicación.

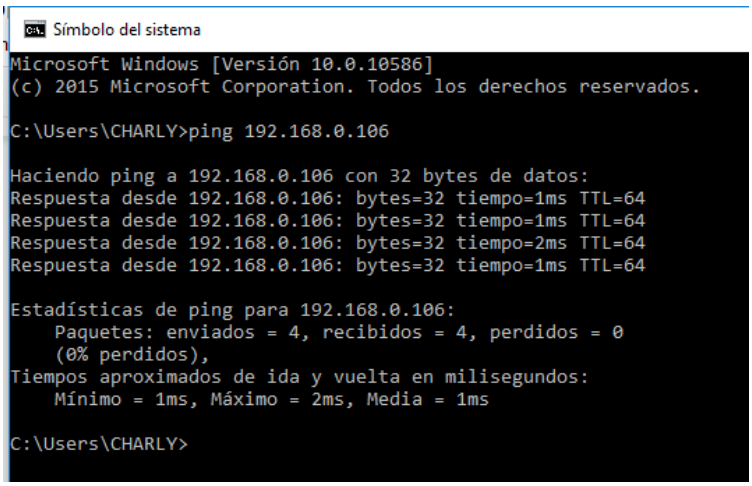



Figura 41. Comunicación al servidor FTP

Finalmente se comprueba que el servidor está activo accediendo desde diferentes dispositivos, se realizan dos pruebas; la primera desde el navegador de una laptop conectada en la red, y desde un dispositivo android mediante la aplicación File Comander.

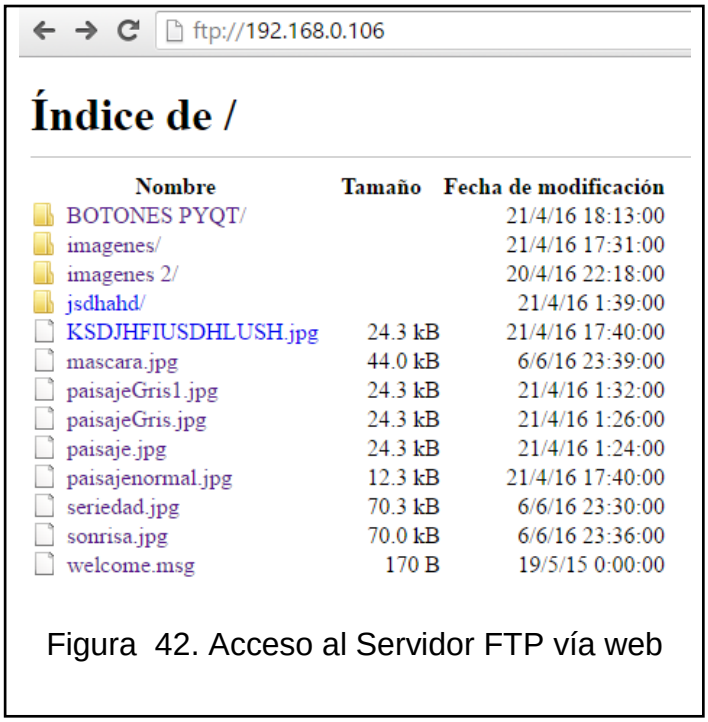



Figura 42. Acceso al Servidor FTP vía web

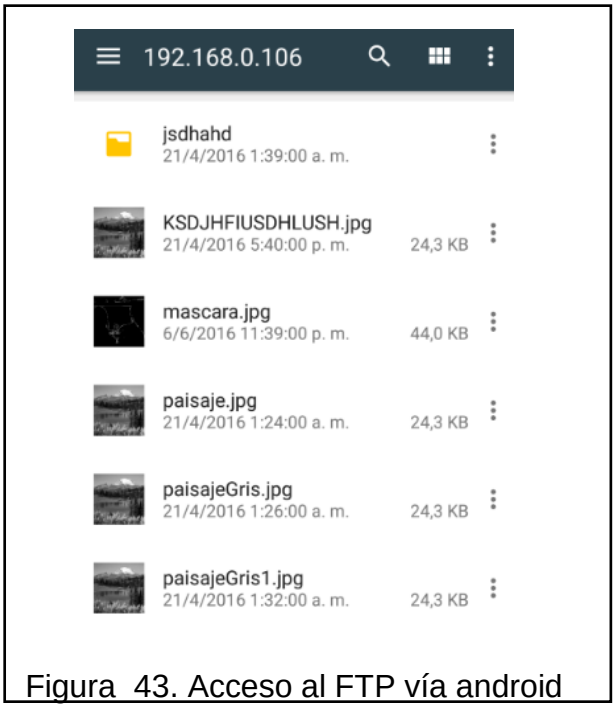



Figura 43. Acceso al FTP vía android

De esta manera se comprueba que el servidor FTP funciona y se puede acceder desde cualquier dispositivo conectado a la red, la funcionalidad del

servidor FTP en el prototipo es la de almacenar las imágenes procesadas en los diferentes módulos del prototipo.

### 2.5.6. Programación

La programación se realiza con lenguaje de programación Python, para lo cual creamos un archivo *.py; esto se lo realiza mediante el menú del IDLE de Python con la pestaña File-New, File de esta manera se crea el archivo ejecutable.

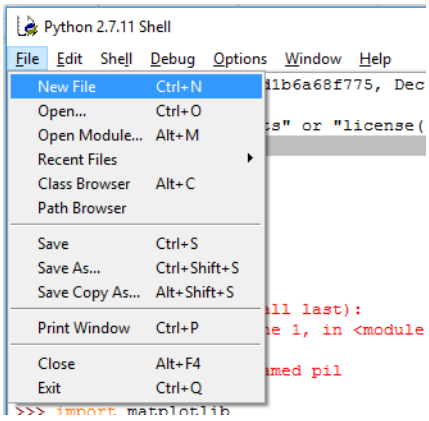



Figura 44. Creación Archivo .py

Con el archivo en blanco y las herramientas instaladas se puede programar la lógica que debe seguir el prototipo; de esta manera se puede identificar los segmentos más importantes y comunes entre los diferentes procesos, como son, librerías, paquetes adicionales, etc.

#### 2.5.6.1. Importación de Librerías

Como todas las librerías fueron instaladas en la primera parte del programa se realiza la importación de todas las librerías que se usan a lo largo del programa.

```
from PyQt4.QtCore import pyqtSlot
from PyQt4.QtGui import *
import cv2
```

```
import numpy as np
from Tkinter import *
from ftplib import FTP
from facerec.feature import Fisherfaces
from facerec.classifier import NearestNeighbor
from facerec.model import PredictableModel
from PIL import Image
import sys, os
import time
import multiprocessing
```

#### 2.5.6.2. Creación de la pantalla principal y botones

Una vez importada la biblioteca QT es posible la creación de una interfaz gráfica de usuario GUI, para lo cual se establece la lógica de una pantalla principal dentro de la cual se programan los botones que acceden a las diferentes opciones del prototipo.

Para crear la pantalla principal

```
app = QApplication(sys.argv)
w = QWidget()
w.setWindowTitle('PROCESAMIENTO DE IMAGENES ')
w.resize(500, 300)
w.show()
app.exec_()
```

Para crear un botón dentro de la pantalla:

```
btndet = QPushButton('DETECTAR ROSTRO, w)
btndet.move(10,150)
def on_click():
btndet.clicked.connect(on_click)
```

El resultado final se muestra en la figura 45, como se observa se puede añadir más botones dentro de la pantalla principal siguiendo la lógica de la programación, para que cada botón acceda a una parte determinada del programa.

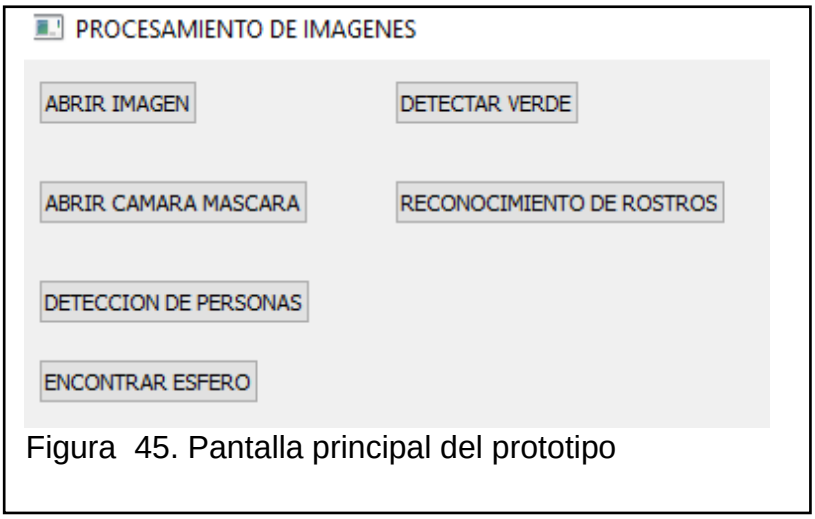



Figura 45. Pantalla principal del prototipo

#### 2.5.6.3. Conexión con el servidor FTP

Establece la comunicación entre el servidor y el programa escrito en Python, de esta manera en la parte seleccionada del proceso se envían frames o capturas de la imagen que está siendo procesada. Los principales argumentos que se deben conocer son la dirección IP del servidor, las ubicaciones en las cuales se están almacenando las imágenes que luego son exportadas al servidor, y el formato de la imagen.

```
ftp=FTP("192.168.0.106")
ftp.login()
fp=open('PATH IMAGEN.jpg','rb')
ftp.storbinary('STOR IMAGEN.jpg', fp)
```

#### 2.5.6.4. Manipulación de imágenes

A través de las bibliotecas incluidas en Opencv se puede realizar varias operaciones para manipular las imágenes, las básicas comprenden lectura de imágenes, captura de video, cambio de filtros, mostrar en pantalla, colocar etiquetas; con estas primeras operaciones se puede realizar programas básicos que sirven como plantillas para desarrollar programas más complejos, de esta manera las sentencias básicas escritas en Python corresponden a:

```
captura=cv2.Videocapture(1)
hsv=cv2.cvtColor(imagen,cv2.COLOR_BGR2HSV)
```

*gray=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)*
*cv2.imshow(img, img)*

#### 2.5.6.5. Archivo Haar Cascade

Uno de los objetivos principales del prototipo, es la detección y reconocimiento de rostros; en base a esto se utilizó un clasificador en cascada el cual se entrena con una cierta cantidad de muestras de un objeto en particular, las cuales se identifican como muestras positivas que se encuentran a escala original, y negativos del mismo objeto. Son plantillas de reconocimiento de un determinado objeto, el adjetivo de cascada indica que el clasificador resultante se compone a partir de varios clasificadores más sencillos aplicados a la región de interés; para el caso del prototipo se emplearon Haar Cascade para el reconocimiento de rostros y para el reconocimiento de los ojos; el archivo plantilla es de uso libre y se puede descargar en formato XML, para importar este archivo se lo realiza mediante las siguientes funciones

*face_cascade=cv2.CascadeClassifier(haarcascade_frontalface_default.xml)*
*face_cascade=cv2.CascadeClassifier(haarcascade_eye.xml)*

Estos archivos son de uso libre y se pueden descargar como plantillas predefinidas para el uso de la detección de un objeto de interés, pero también es posible realizar, los archivos deben estar en la misma carpeta en la que se encuentra el programa fuente de esta manera se los importa mediante su nombre.

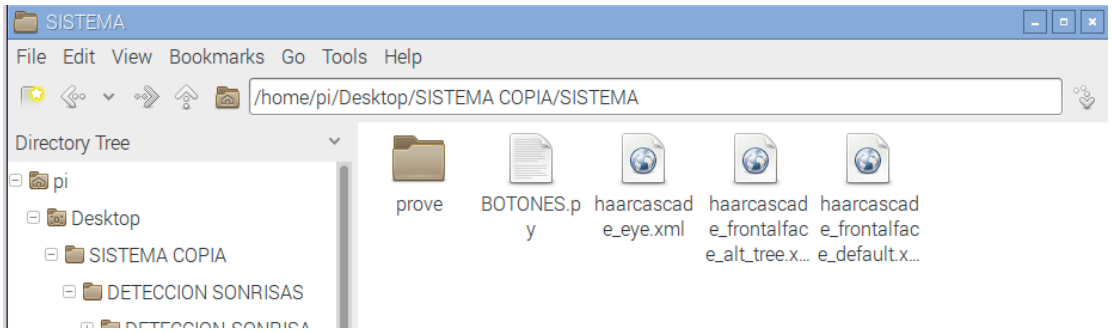



Figura 46. Archivos Haar Cascade

# 3. Capítulo III. Pruebas de funcionamiento

En este capítulo se realiza los análisis de los diferentes resultados obtenidos por medio de la implementación del sistema de procesamiento de imágenes propuesto en este proyecto.

## 3.1 Escenario de pruebas

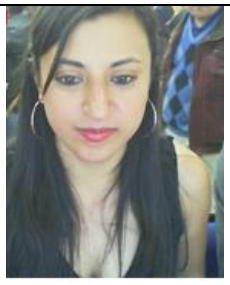
Las pruebas se las realizaron con cinco sujetos de prueba, realizando los diferentes procesos un total de cinco oportunidades; con esto se evalúa el porcentaje de efectividad para el prototipo.

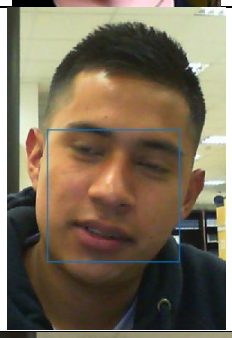
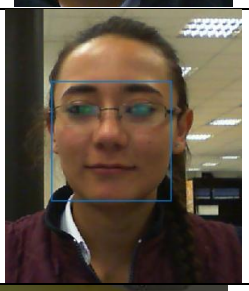
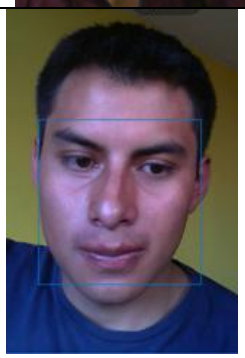
La primera prueba consiste en verificar que el prototipo pueda realizar la detección de rostros de los sujetos de prueba y de sus ojos; en la segunda parte se verifica que se realice el reconocimiento de los sujetos de prueba; para finalmente realizar la detección de dos parámetros de gestos (sonrisa y seriedad).

## 3.2 Sujetos de prueba

A continuación, se detallan los datos de las personas con las que se realizaron las pruebas de funcionamiento del prototipo.

Tabla 4. Datos de los Sujetos de prueba

| NOMBRE | APELLIDO | CI | TELEFONO | PERFIL | |
|---|---|---|---|---|---|
| Verónica | Gordillo | 100259707 | 992966231 |  | |

| | | | | | |
|---|---|---|---|---|---|
| Carlos | Nacimba | 1716535982 | 99020233 |  | |
| Paul | Ordoñez | 1735382926 | 9736273622 |  | |
| Johanna | Maldonado | 1723062319 | 0987026020 |  | |
| Daniel | Nacimba | 1726813650 | 0987757000 |  | |

## 3.3. Inicio de las pruebas

Los datos se tabularon con una escala porcentual, en la que se establece el porcentaje de efectividad para cada proceso del prototipo. A continuación, se detallan las pruebas obtenidas para cada sujeto de prueba.

**Verónica Gordillo**

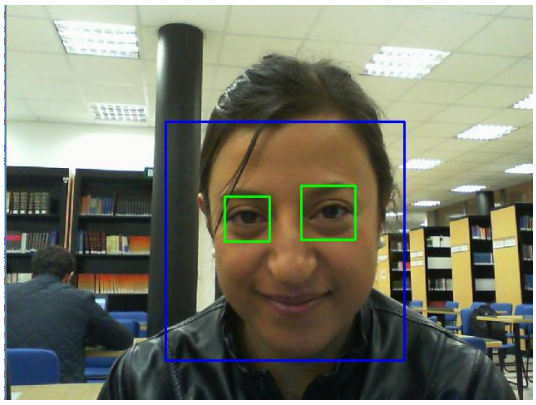


Figura 47. Reconocimiento de rostros Verónica

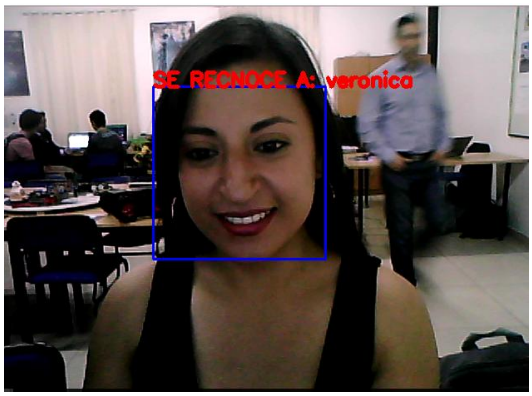



Figura 48. Identificación de Rostro Verónica

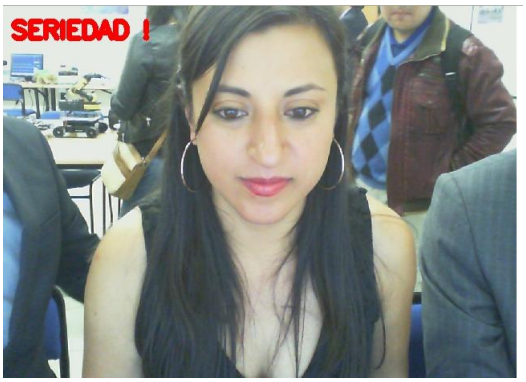



Figura 49. Detección de Seriedad Verónica

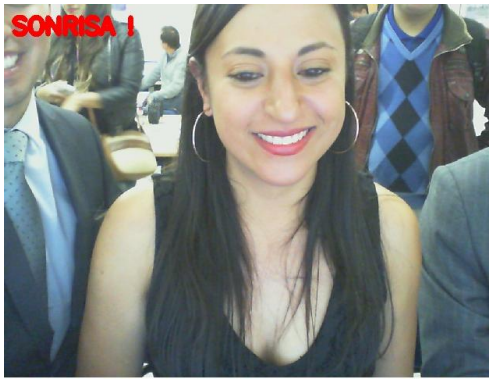



Figura 50. Detección de Sonrisa Verónica

Tabla 5. Resultados Verónica Gordillo

| Verónica Gordillo | | | | | | |
|---|---|---|---|---|---|---|
| | **PRUEBA 1** | **PRUEBA 2** | **PRUEBA 3** | **PRUEBA 4** | **PRUEBA 5** | **RESULTADO %** |
| RECONOCIMIENTO ROSTRO | SI | SI | SI | NO | SI | 80% |
| DETECCION DE ROSTRO | SI | SI | SI | SI | SI | 100% |
| DETECCION DE SONRISAS | SI | SI | SI | NO | SI | 80% |
| DETECCION DE SERIEDAD | SI | SI | SI | SI | SI | 100% |

**Carlos Nacimba**

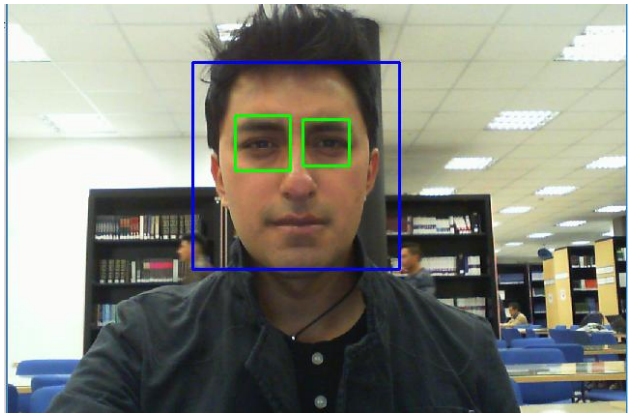


*Figura 51. Reconocimiento de rostros Carlos*

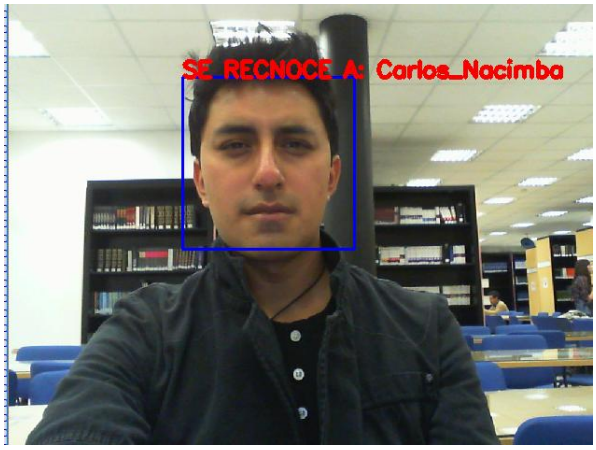



Figura 52. Identificación de Rostro Carlos


*Figura 53. Detección de Seriedad Carlos*


*Figura 54. Detección de Sonrisa Carlos*

Tabla 6. Resultados Carlos Nacimba

| Carlos Nacimba | | | | | | |
|---|---|---|---|---|---|---|
| | **PRUEBA 1** | **PRUEBA 2** | **PRUEBA 3** | **PRUEBA 4** | **PRUEBA 5** | **RESULTADO %** |
| RECONOCIMIENTO ROSTRO | SI | SI | SI | SI | SI | 100% |
| IDENTIFICACION ROSTRO | SI | SI | SI | SI | SI | 100% |
| DETECCION DE SONRISAS | SI | SI | NO | NO | SI | 60% |
| DETECCION DE SERIEDAD | SI | SI | SI | SI | SI | 100% |

**Paul Ordoñez**

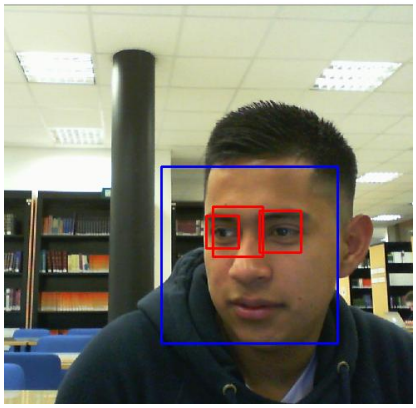


Figura 55. Reconocimiento de rostros Paul

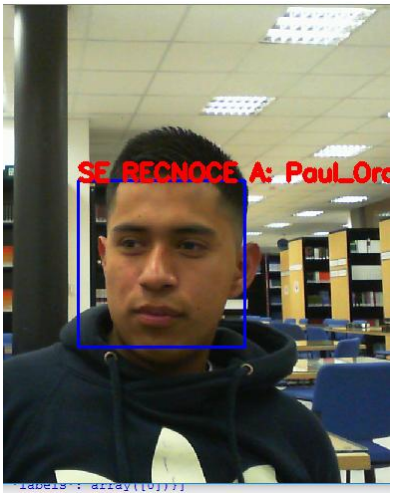



*Figura 56. Identificación de Rostro Paul*


Figura 57. Detección de Seriedad Paul


Figura 58. Detección de Sonrisa Paul

Tabla 7. Resultados Paul Ordoñez

| Paul Ordoñez | | | | | | |
|---|---|---|---|---|---|---|
| | **PRUEBA 1** | **PRUEBA 2** | **PRUEBA 3** | **PRUEBA 4** | **PRUEBA 5** | **RESULTADO %** |
| RECONOCIMIENTO ROSTRO | SI | SI | SI | SI | SI | 100% |
| DETECCION DE ROSTRO | SI | SI | SI | SI | SI | 100% |
| DETECCION DE SONRISAS | SI | SI | SI | NO | SI | 80% |
| DETECCION DE SERIEDAD | SI | SI | SI | SI | SI | 100% |

**Johanna Maldonado**

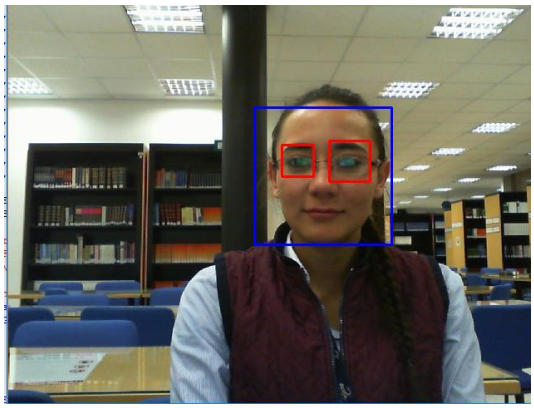


Figura 59. Reconocimiento de rostros Johanna

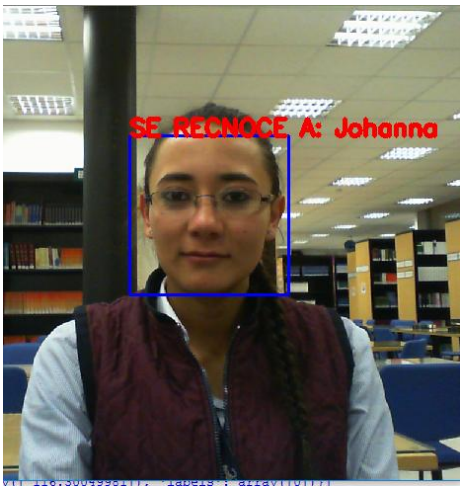



*Figura 60. Identificación de Rostro Johanna*


Figura 61. Detección de seriedad Jhoanna


Figura 62. Detección de sonrisa Jhoanna

Tabla 8. Tabla de resultados Johana Maldonado

| Jhoanna Maldonado | | | | | | |
|---|---|---|---|---|---|---|
| | **PRUEBA 1** | **PRUEBA 2** | **PRUEBA 3** | **PRUEBA 4** | **PRUEBA 5** | **RESULTADO %** |
| RECONOCIMIENTO ROSTRO | SI | SI | SI | NO | SI | 80% |
| DETECCION DE ROSTRO | SI | SI | SI | SI | SI | 100% |
| DETECCION DE SONRISAS | SI | SI | NO | NO | NO | 40% |
| DETECCION DE SERIEDAD | SI | SI | SI | SI | SI | 100% |

**Daniel Nacimba**

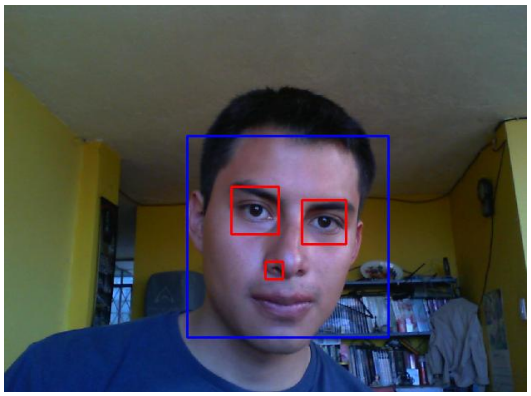


Figura 63. Reconocimiento de rostros Daniel


Figura 64. Identificación de Rostro Daniel

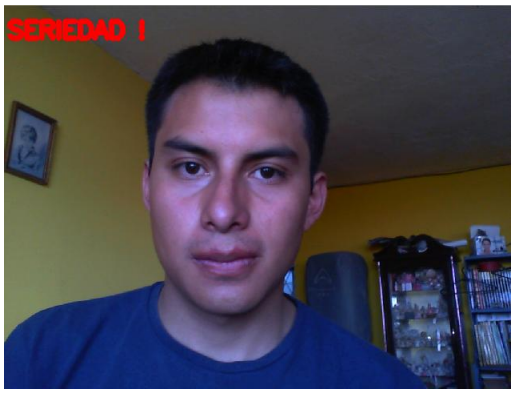



Figura 65. Detección de seriedad Daniel


Figura 66. Detección de sonrisa Daniel

Tabla 9. Resultados Daniel Nacimba

| | PRUEBA 1 | PRUEBA 2 | PRUEBA 3 | PRUEBA 4 | PRUEBA 5 | RESULTADO % |
|---|---|---|---|---|---|---|
| RECONOCIMIENTO ROSTRO | SI | SI | SI | SI | SI | 80% |
| DETECCION DE ROSTRO | SI | SI | SI | SI | SI | 100% |
| DETECCION DE SONRISAS | SI | SI | SI | NO | SI | 80% |
| DETECCION DE SERIEDAD | SI | SI | SI | SI | SI | 100% |

## 3.4. Resultados Finales

Se establece que el prototipo tiene una efectividad del 100% para realizar la detección de rostros y de ojos; esto se debe al uso de las plantillas para detección Haar Cascade.

Para el reconocimiento de rostros en las diferentes pruebas se tiene una efectividad en un rango del 80%-100%; esto se debe al uso de Fisherfaces como método de reconocimiento, con lo cual las variaciones de luz influyen en los resultados de las diferentes pruebas.

Finalmente se comprobó que las imágenes seleccionadas fuesen enviadas y almacenadas en el servidor FTP.

Las últimas pruebas de funcionamiento para el prototipo se realizaron en el Día de las Telecomunicaciones UDLA con la presencia de profesores y alumnos, a los cuales se les demostró el funcionamiento y uso del prototipo.

# Capítulo IV. Conclusiones y Recomendaciones

## 4.1 Conclusiones

Una de las diferencias más significativas entre los PC`s que usamos para nuestras tareas básicas de estudio o trabajo en relación con el Raspberry pi, es su sistema operativo al contario de usar Windows o Apple OS x esta pequeña placa usa una distribución de Linux con base en Debian, llamada Raspbian, el cual es de código abierto lo que nos permite hacer cambios en el código fuente del sistema.

La versión del sistema operativo Raspbian, tiene incorporado una versión de PHYTON 2.7 sin embargo esta versión puede ser actualizada a las versiones 3.0 o superior. Recordando que Python es un lenguaje de programación de alto nivel creado para facilitar el aprendizaje de sistemas de programación, en el cual las sentencia o bucles se cierran con la alineación del tabulador.

La interfaz del prototipo desarrollado se ajusta a parámetros simples de uso para facilitar su entendimiento y manipulación por diferentes usuarios, con lo que se sienta las bases para próximas investigaciones referentes a temas de procesamiento de imágenes.

De igual manera que un Pc, el Raspberry necesita un dispositivo de almacenamiento en el cual se grabe el sistema operativo de arranque, para el caso el Raspberry se trata de una micro SD de 16GB clase 10, estas memorias pueden venir en formato de clase 10 y clase 4, lo que nos indica la clase es la velocidad a la que van a trabajar una memoria clase 10 trabaja a mayor velocidad que una de clase 4.

## 4.2 Recomendaciones

Aunque el Raspberry es computador onboard con una gran capacidad para desarrollo de proyectos, se debe tomar en cuenta las limitaciones que pueden presentarse, como la memoria RAM, se deben diseñar los proyectos de manera que se pueda aprovechar el máximo de recursos del Raspberry.

Según las especificaciones técnicas se recomienda usar un adaptador de 5v a 1500mA, sin embargo, a medida que usemos una mayor cantidad de periféricos USB se debe contemplar un adaptador de mayor capacidad, con lo cual no tendremos problemas de intermitencia en nuestra placa.

Opencv, biblioteca que facilita el procesamiento de imágenes tiene algunas versiones, se debe tener cuidado en la versión que se esté instalando ya que hay sentencias y clases que no constan en las versiones más actuales como la 3.0, pero que si están presentes en la versión 2.4.

Ya que estamos usando el método de reconocimiento para rostros Fiserfaces es importante tomar en cuenta que este método se basa en la detección de luz reflejada en el rostro por lo que se deben realizar las pruebas con entornos de luz natural y fondos preferentemente claros y de un solo color.

Para un mejor entendimiento del lenguaje de programación se recomienda revisar la estructura de sentencias y clases en Python.

# REFERENCIAS

Boylestad, R, & Nashelsky, L. (2009). Teoría de circuitos y dispositivos electrónicos (10ª. ed.) , México DF, México: Pearson Educación.

Cabello, F. (2016). Tutorial: Instalar un servidor FTP en un Raspberry. Recuperado el 20 de abril de 2016, de: http://movilforum.com/instalar-servidor-ftp-raspberry/

Cox, T., (2014). Raspberry Pi Cookbook for Python Programmers, Birmingham, Reino Unido: Packt Publishing Ltd.

García, P. (2013). Reconocimiento de imágenes utilizando redes neuronales artificiales. Universidad Complutense de Madrid, Madrid. España.

Harris, C, & Jones, E. (2016). Scipy: Scientific Library for Python. Recuperado el 6 de marzo 2016, de: https://sourceforge.net/projects/scipy/files/scipy/0.9.0/

Horan, B. (2015). Practical Raspberry. Reino Unido: Tecnology in Action.

Llulluna, F. (2014). Procesamiento de imágenes mediantes Software Libre Python para el análisis metalográfico en aceros de bajo contenido de carbono. Escuela Politécnica Nacional, Quito, Ecuador.

López, A, & Mendoza, C. (2015). Sistema de Autenticación Facial mediante la Implementación del algoritmo PCA modificado en Sistemas embebidos con arquitectura ARM. Recuperado el 6 de marzo de 2016 de http://www.mecamex.net/revistas/LMEM/revistas/LMM-V04-N02-02.pdf

Marina, M. (2016). Detección de figuras geométricas. Recuperado el 23 de mayo de 2016, de http://acodigo.blogspot.com/2014/05/deteccion-de-figuras-geometricas.html

Mendoza, C. (2009). Procesamiento y análisis digital de imágenes mediante dispositivos lógicos programables (tesis de pregrado). Universidad Tecnológica de la Mixteca, Oaxaca, México.

Muja, M. (2010). Recognition pipeline and Object Detection Scalability. Recuperadoel 20 de abril de 2016, de: http://www .willowgarage. com/blog/2010/09/20/scalable-object- recognition

Norris, D. (2014). Raspberry Pi Projects for the Evil Genius (2a. ed.). Estados Unidos: McGraw Hill Education.

Raspberry Org. Installyn Phyton Packages. Recuperado el 12 mayo de 2016, de: https://www.raspberrypi.org/documentation/linux/software/python.md

Upton, E, & Halfacree, G. (2014). Raspberry Pi Guía de usuario. (2a. ed.). Recuperado el 20 de marzo de 2016 de https: //es.scribd. com/doc /225682222/Raspberry-Pi-Guia-Del-Usuario-2da-Ed-en-Espanol

Vignolo, J., (2008). Introducción al Procesamiento de señales digitales, Valparaíso, Chile: Ediciones Universitarias de Valparaíso.

# ANEXOS

**ANEXO A**

**Características Técnicas Raspberry Pi 2 Model B**

adafruit learning system

# Introducing the Raspberry Pi 2 - Model B

Created by lady ada


Last updated on 2015-02-09 03:00:41 PM EST

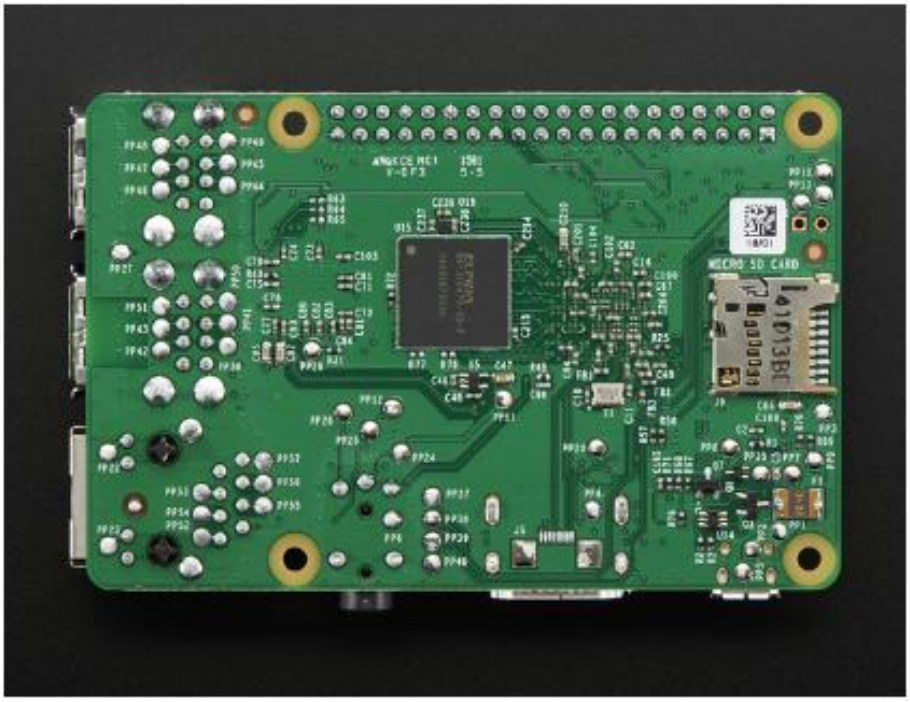
# Overview


Didn't think the Raspberry Pi could get any better? You're in for a big surprise! The Raspberry Pi 2 Model B is out and it's amazing! With an upgraded ARMv7 multicore processor, and a full Gigabyte of RAM, this pocket computer has moved from being a 'toy computer' to a real desktop PC

The big upgrade is a move from the BCM2835 (single core ARMv6) to BCM2836 (quad core ARMv7). The upgrade in processor types means you will see ~2x performance increase just on processor-upgrade only. For software that can take advantage of multiple-core processors, you can expect 4x performance on average and for really multi-thread-friendly code, up to 7.5x increase in speed!

That's not even taking into account the 1 Gig of RAM, which will greatly improve games and web-browser performance!

Best of all, the Pi 2 keeps the same shape, connectors and mounting holes as the Raspberry Pi B+. That means that all of your HATs and other plug-in daughterboards will work just fine. 99% of cases and accessories will be fully compatible with both versions



**Please note:** The new processor on the Pi 2 means that you will need to update your existing SD card or create a new SD card with your operating system (Raspbian, Arch, XBMC, NooBs, etc) because you cannot plug in olderscards from a Pi 1 into a Pi 2 without upgrading with **sudo apt-get upgrade** on the Pi 1 first.

Also, any precompiled software will not work at full speed (although supposedly the processor will be able to run it). Still, you'll likely want to have it recompiled for the new processor! For many people, this isn't a big deal, but if you have a pre-created Pi 1 Model A+B+ card image, just be aware it won't work without performing an 'sudo apt-get upgrade' on the *older Pi 1* before installing on the Pi 2!

## What to watch out for!

Watch out, the Raspberry Pi 2 Model B is VERY different from the Raspberry Pi Model B - check for that "2" when checking accessories and compatibility!

The Raspberry Pi 2 Model B looks *a lot* like a Raspberry Pi Model B+! Look for the chip on the bottom to identify the Pi 2

## What hasn't changed

**The basic form-factor of the PI 2 Model B is nearly 100% the same as the PI model B+**

- The shape and size of the PCB is the same
- The 4 mounting holes are in the same location and are the same size
- The USB, Ethernet, A/V, HDMI, micro SD and microUSB connectors are int the same exact locations and are the same size
- The Camera, Display and 40-pin GPIO connectors are in the same exact locations and are the same size

**The physical changes include:**

- Processor chip is larger, has moved slightly
- The RAM is now soldered onto the bottom of the board (no longer PoP)
- Other components and chips are moved around slightly to make space for the larger processor and RAM chip on bottom

This means that 99% of cases designed for the Raspberry Pi Model B+ will work with the Raspberry Pi 2 Model B. This includes the Adafruit B+ Pi cases (http://adafru.it/2258)

One exception is some Pibow cases which have a layer that has cutouts for the specific location of the processor. (http://adafru.it/epX)Pimoroni has informed us that they will have a new case design that is compatible with both. Check the description of any case to make sure it is compatible with both **Raspberry PI Model B+** *and* **Raspberry PI 2 Model B**

# What has changed, watch out!

## New Processor

The processor has completely changed on the new RaspberryPi 2, instead of a ARM v6 core chip (arm6l) the BCM2836 has been upgraded to an ARM v7 core which is a much more powerful core

**However, your existing Raspberry PI SD card images may not work because the firmware and kernel must be recompiled/adapted for the new processor.**

If you have a Raspberry Pi 2, and you are trying to upgrade your existing SD card, you will need to upgrade your installation. To do that, log into your Pi 1 and at a console or terminal type in **sudo apt-get upgrade** to perform the upgrade procedure. You'll need your Pi to be on the Internet to do this. Once upgraded, the card will work on both Pi 1 and Pi 2 computers

If you have any pre-compiled binaries that you are downloading, those will need updating too, in order to take advantage of the speed increase. Anything where you have access to source code can be recompiled and ought to work just fine. Supposedly any ARMv6 software is forwards compatible with ARMv7 but we haven't tested it for sure yet

## Power Draw

Quad-core ARMv7 processor means higher current draw.

Just having the Pi 2 Model B running idle (no HDMI, graphics, ethernet or wifi just console cable) the Pi 2 draws **200mA**

With WiFi running, that **adds another 170mA**

If you have Ethernet instead, that **adds about 40mA**

When doing the heaviest computational tasks, we added about 200-250mA more current. So if you're really using your Pi 2 and you have a WiFi dongle, **expect to need 650mA @ 5V, at least**. More if you have stuff connected to the GPIO connector, other USB devices, Ethernet as well, etc!

If you're still running with a cheap 5V 700mA power supply, we really recommend upgrading to a 5V @ 2000mA!

## How to tell if you have a Pi 2

Since the Raspberry Pi 2 Model B looks *a lot* like the Raspberry Pi Model B+ you'll want to get good at identifying which you have.

## PCB Silkscreen Name

First up, you can look for the name which is near the GPIO connector:


Look for the "Raspberry Pi 2" text and you're golden!

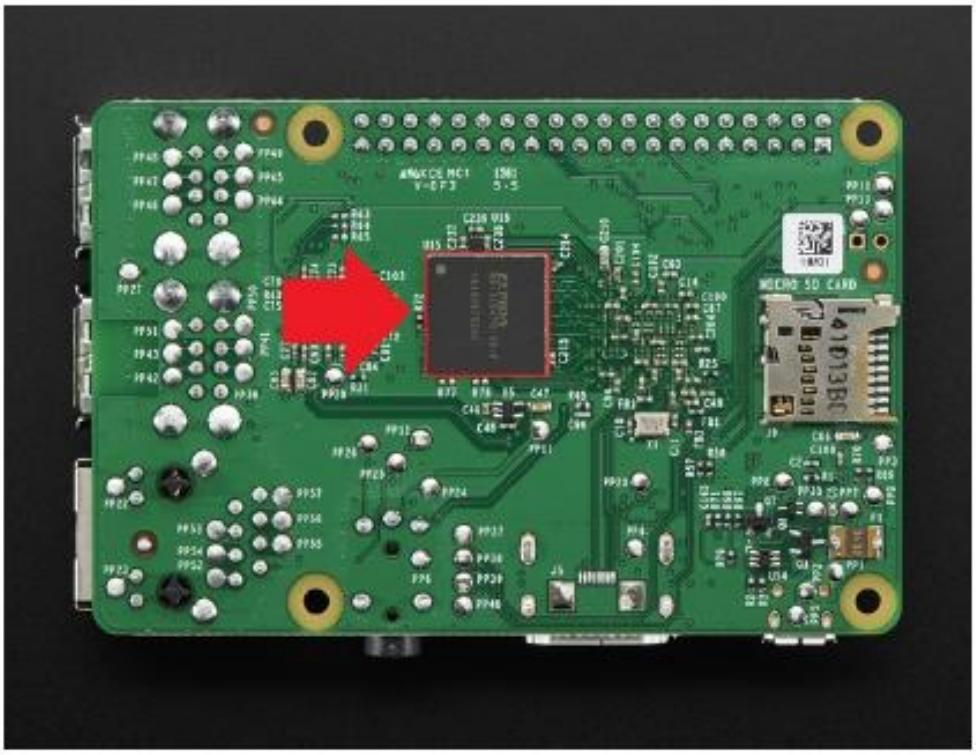
## Broadcom Logo on Processor

Since the Pi 2 does not use 'package-on-package', assembling the RAM on top of the processor, you can easily spot the Broadcom logo on top of the quad processor


If you don't see the logo, or you see something like "Samsung" or "Hynix" it's a B+

## RAM chip on bottom

The Pi 2 has a RAM chip that is soldered onto the bottom of the Raspberry Pi's circuit board. The B+ does not have one at all, the RAM chip is soldered directly on the processor. So just look for a black square chip on the bottom of the PCB. The naming and logo on the RAM may vary depending on what company supplied the memory.


MICRO SD CARD

# ANEXO B

## Cámara  LifeCam Hd 3000

Microsoft®
LifeCam
HD-3000


| | |
|---|---|
| **Version Information** | |
| Product Name | Microsoft® LifeCam HD-3000 |
| Product Version | Microsoft LifeCam HD-3000 |
| Webcam Version | Microsoft LifeCam HD-3000 |
| **Product Dimensions** | |
| Webcam Length | 1.55 inches (39.3 millimeters) |
| Webcam Width | 1.75 inches (44.5 millimeters) |
| Webcam Depth/Height | 4.28 inches (109 millimeters) |
| Webcam Weight | 3.17 ounces (89.9 grams) |
| Webcam Cable Length | 59.1 inches (1500 millimeters) |
| **Compatibility and Localization** | |
| Interface | High-speed USB compatible with the USB 2.0 specification |
| Operating Systems[1] | • Microsoft Windows® 8.1, Windows 8, Windows RT 8.1, Windows RT 8, and Windows 7<br>• Macintosh OS X v10.7-10.9<br>• Android 3.2 and 4.2<br>[1]Advanced functionality not available with all devices and/or operating systems. See compatibility information at: www.microsoft.com/hardware/compatibility. |
| Top-line System Requirements | Requires a PC that meets the requirements for and has installed one of these operating systems:<br>• Microsoft Windows 8.1, Windows 8, or Windows 7<br>**For VGA video calling:**<br>• Intel Dual Core 1.6 GHz or higher<br>• 1 GB of RAM<br>• 1.5 GB<br>• USB 2.0 required<br>• Windows-compatible speakers or headphones<br>**For 720p HD recording:**<br>• Intel Dual Core 3.0 GHz or higher<br>• 2 GB of RAM<br>• 1.5 GB<br>• USB 2.0 required<br>• Windows-compatible speakers or headphones<br>You must accept License Terms for software download. Please download the latest available software version for your OS/Hardware combination.<br>Internet access may be required for certain features. Local and/or long-distance telephone toll charges may apply.<br>Software download required for full functionality of all features.<br>Internet functions (post to Windows Live™ Spaces, send in e-mail, video calls), also require: Internet Explorer® 6/7/8 browser software required for installation; 25 MB hard drive space typically required (users can maintain other default Web browsers after installation) |
| Compatibility Logos | • Compatible with Microsoft Windows 8 and Windows RT<br>• Optimized for Microsoft Lync<br>• Skype Certified |
| Software Localization | Microsoft LifeCam software version 3.0 may be installed in Simplified Chinese, Traditional Chinese, English, French, German, Italian, Japanese, Korean, Brazilian Portuguese, Iberian Portuguese, Russian, or Spanish. If available, standard setup will install the software in the default OS language. Otherwise, the English language version will be installed. |

| | |
|---|---|
| **Windows Live™ Integration Features** | |
| Video Conversation Feature | Windows Live call button delivers one touch access to video conversation |
| Call Button Life | 10,000 actuations |
| Webcam Controls & Effects | LifeCam Dashboard provides access to animated video effects and webcam controls |
| Windows Live Integration Features | Windows Live Photo Gallery integration - Take a photo with LifeCam Software, then with one click open Photo Gallery to edit, tag and share it online<br>Windows Live Movie Maker integration - Record a video with LifeCam Software and start a movie project on Movie Maker with just one click to then upload it to your favorite networking site |
| **Imaging Features** | |
| Sensor | CMOS sensor technology |
| Resolution | • Motion Video: 1280 X 720 pixel resolution*<br>• Still Image: 1280 X 800 |
| Imaging Rate | Up to 30 frames per second |
| Field of View | 68.5° diagonal field of view |
| Imaging Features | • Digital pan, digital tilt, vertical tilt, swivel pan, and 4x digital zoom<br>• Fixed focus from 0.3m to 1.5m<br>• True Color - Automatic image adjustment with manual override<br>• 16:9 widescreen<br>• 24-bit color depth |
| **Product Feature Performance** | |
| Audio Features | Integrated microphone |
| Microphone Technology | Omni directional microphone |
| Frequency Range | Frequency range 200Hz – 20kHz |
| Mounting Features | Flexible universal attachment base |
| Storage Temperature & Humidity | -40 °F (-40 °C) to 140 °F (60 °C) at <5% to 65% relative humidity (non-condensing) |
| Operating Temperature & Humidity | 32° F (0° C) to 104 °F (40 °C) at <5% to 80% relative humidity (non-condensing) |
| **Certification Information** | |
| Country of Manufacture | People's Republic of China |
| ISO 9001 Qualified Manufacturer | Yes |
| ISO 14001 Qualified Manufacturer | Yes |
| Restriction on Hazardous Substances | This device complies with all applicable worldwide regulations and restrictions including, but not limited to: EU directive 2002/95/EC on the Restriction of the Use of Certain Hazardous Substances in Electrical and Electronic Equipment and EU Registration Evaluation and Authorization of Chemicals (REACH) regulation regarding Substances of Very High Concern. |
| FCC ID | This device complies with part 15 of the FCC Rules and Industry Canada ICES-003. Operation is subject to the following two conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation. Tested to comply with FCC standards. For home and office use. Model number: 1492, LifeCam HD-3000. |
| Agency and Regulatory Marks | • ACMA Declaration of Conformity (Australia and New Zealand)<br>• ICES-003 report on file (Canada)<br>• EIP Pollution Control Mark, EPUP (China)<br>• CE Declaration of Conformity (European Union)<br>• WEEE (European Union)<br>• VCCI Certificate (Japan)<br>• KCC Certificate (Korea)<br>• GOST Certificate (Russia)<br>• CITC Letter (Kingdom of Saudi Arabia)<br>• UkrSEPRO Certificate (Ukraine)<br>• FCC Declaration of Conformity (USA)<br>• UL and cUL Listed Accessory (USA and Canada)<br>• CB Scheme Certificate (International) |
| Windows Certification Kit (WCK) | ID: 1608508 (32-bit) and 1608509 (64-bit) Microsoft Windows 8.1 |

* One megapixel = 1,000,000 pixels. Lower resolution available when sending video via instant messaging.

## ANEXO C

### Código fuente del programa de procesamiento de imágenes

### SISTEMA DE DETECCION Y RECONOCIMIENTO DE ROSTROS

```
from PyQt4.QtCore import pyqtSlot
from PyQt4.QtGui import *
import cv2
import numpy as np
from Tkinter import *
from ftplib import FTP
from facerec.feature import Fisherfaces
from facerec.classifier import NearestNeighbor
from facerec.model import PredictableModel
from PIL import Image
import sys, os
import time
import multiprocessing

ftp=FTP("192.168.0.106")
ftp.login()

# crear nuestra pantalla
app = QApplication(sys.argv)
w = QWidget()
w.setWindowTitle('PROCESAMIENTO DE IMAGENES ')
w.resize(500, 300)
##########################################################################
##########################################################################
btn2 = QPushButton('ABRIR CAMARA MASCARA', w)
btn2.move(10,60)

@pyqtSlot()
def on_click():

    cap = cv2.VideoCapture(0)

    while(1):

        _, frame = cap.read()
        hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

        lower_red = np.array([30,150,50])
        upper_red = np.array([255,255,180])

        mask = cv2.inRange(hsv, lower_red, upper_red)
```

```
        res = cv2.bitwise_and(frame,frame, mask= mask)

        cv2.imshow('Original',frame)
        edges = cv2.Canny(frame,100,200)
        cv2.imshow('Edges',edges)
        cv2.imwrite('/home/pi/imagenes/mascara.jpg',edges)
        fp=open('/home/pi/imagenes/mascara.jpg','rb')
        ftp.storbinary('STOR mascara.jpg', fp)

        k = cv2.waitKey(5) & 0xFF
        if k == 27:
            break

    cv2.destroyAllWindows()
    cap.release()

btn2.clicked.connect(on_click)
#############################################################################

btn3 = QPushButton('DETECTAR VERDE', w)
btn3.move(210,10)
@pyqtSlot()
def on_click():


    #Iniciamos la camara
    captura = cv2.VideoCapture(0)

    while(1):

        _, imagen = captura.read()
        hsv = cv2.cvtColor(imagen, cv2.COLOR_BGR2HSV)

        verde_low = np.array([49,50,50], dtype=np.uint8)
        verde_hi = np.array([80, 255, 255], dtype=np.uint8)

        maskver = cv2.inRange(hsv, verde_bajos, verde_altos)

        moments = cv2.moments(mask)
        area = moments['m00']

        if(area > 2000000):

            x = int(moments['m10']/moments['m00'])
            y = int(moments['m01']/moments['m00'])
```

```
            print "x = ", x
            print "y = ", y

            #Dibujamos una marca en el centro del objeto
            cv2.rectangle(imagen, (x-5, y-5), (x+5, y+5),(0,0,255), 2)
            cv2.putText(imagen, "pos:"+ str(x)+","+str(y), (x+10,y+10),
cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255), 2)

        cv2.imshow('mask', mask)
        cv2.imshow('Camara', imagen)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            captura.release()
            break

    cv2.destroyAllWindows()

btn3.clicked.connect(on_click)

##############################################################################
#
btn4=QPushButton('RECONOCIMIENTO DE ROSTROS',w)
btn4.move(240,60)
@pyqtSlot()
def on_click():
    #IMPORTAMOS LA BASE PARA RECONOCIMIEMTO
    rostro_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    ojo_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
    cap = cv2.VideoCapture(0)
    while 1:
        ret, img = cap.read()
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        rostro = rostro_cascade.detectMultiScale(gray, 1.3, 5)
        for (x,y,w,h) in faces:
            cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
            roi_gray = gray[y:y+h, x:x+w]
            roi_color = img[y:y+h, x:x+w]
            ojo = ojo_cascade.detectMultiScale(roi_gray)
            for (ex,ey,ew,eh) in eyes:
                cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)
        cv2.imshow('img',img)
        k = cv2.waitKey(30) & 0xff
        if k == 27:
            break

    cap.release()
```

```
    cv2.destroyAllWindows()
btn4.clicked.connect(on_click)
################################################################################
#

btn5=QPushButton('DETECCION DE PERSONAS',w)
btn5.move(10,110)
@pyqtSlot()
def on_click():
    model = PredictableModel(Fisherfaces(), NearestNeighbor())
    vc=cv2.VideoCapture(0)
    face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_alt_tree.xml')
    def read_images(path, sz=(256,256)):
        c = 0
        X,y = [], []
        folder_names = []
        for dirname, dirnames, filenames in os.walk(path):
            for subdirname in dirnames:
                folder_names.append(subdirname)
                subject_path = os.path.join(dirname, subdirname)
                for filename in os.listdir(subject_path):
                    try:
                        im = cv2.imread(os.path.join(subject_path, filename),
cv2.IMREAD_GRAYSCALE)
                        if (sz is not None):
                            im = cv2.resize(im, sz)
                        X.append(np.asarray(im, dtype=np.uint8))
                        y.append(c)
                    except IOError, (errno, strerror):
                        print "I/O error({0}): {1}".format(errno, strerror)
                    except:
                        print "Unexpected error:", sys.exc_info()[0]
                        raise
                c = c+1
        return [X,y,folder_names]
    pathdir='prove/'
    quanti = int(raw_input('CUAL WEBCAM SE USARA? \n NUMERO DE FUENTES:'))
    for i in range(quanti):
        nome = raw_input('HOLA USUARIO '+str(i+1)+' CUAL ES SU NOMBRE?\n NOMBRE:')
        if not os.path.exists(pathdir+nome): os.makedirs(pathdir+nome)
        print ( 'ESTAS LISTO PARA TOMAR ALGUNAS FOTOS ? \n')
        print ( 'POR FAVOR ESPEAR  10 SEGUNDOS PULSE "S" CUANDO ESTES EN EL CENTRO')
        while (1):
            ret,frame = vc.read()
            gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            faces = face_cascade.detectMultiScale(gray, 1.2, 3)
```

```
            for (x,y,w,h) in faces:
                cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
            cv2.imshow('Reconocimiento',frame)
            if cv2.waitKey(10) == ord('s'):
                break
        cv2.destroyAllWindows()
        start = time.time()
        count = 0
        while int(time.time()-start) <= 35:
            ret,frame = vc.read()
            gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            faces = face_cascade.detectMultiScale(gray, 1.2, 3)
            for (x,y,w,h) in faces:
                cv2.putText(frame,'CAPTURANDO!', (x,y),
cv2.FONT_HERSHEY_SIMPLEX,0.8,(0,0,255),3,1)
                count +=1
                cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
                resized_image = cv2.resize(frame[y:y+h,x:x+w], (273, 273))
                if count%5 == 0:
                    print  pathdir+nome+str(time.time()-start)+'.jpg'
                    cv2.imwrite( pathdir+nome+'/'+str(time.time()-start)+'.jpg', resized_image );
            cv2.imshow('Reconocimiento',frame)
            cv2.waitKey(10)
        cv2.destroyAllWindows()
    [X,y,subject_names] = read_images(pathdir)
    list_of_labels = list(xrange(max(y)+1))
    subject_dictionary = dict(zip(list_of_labels, subject_names))
    model.compute(X,y)
    while (1):
        rval, frame = vc.read()
        img = frame
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray, 1.2, 3)

        for (x,y,w,h) in faces:
            cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
            sampleImage = gray[y:y+h, x:x+w]
            sampleImage = cv2.resize(sampleImage, (256,256))
            [ predicted_label, generic_classifier_output] = model.predict(sampleImage)
            print [ predicted_label, generic_classifier_output]
            if int(generic_classifier_output['distances']) <=  700:
                cv2.putText(img,'SE RECNOCE A: '+str(subject_dictionary[predicted_label]), (x,y),
cv2.FONT_HERSHEY_SIMPLEX,0.8,(0,0,255),3,1)
        cv2.imshow('RECONOCIMIENTO',img)
        if cv2.waitKey(10) == 27:
            break
```

```
    cv2.destroyAllWindows()
    vc.release()
btn5.clicked.connect(on_click)

# mostramos la aplicaion
w.show()
app.exec_()
```

## SISTEMA DE DETECCION DE SONRISAS

```
from PyQt4.QtCore import pyqtSlot
from PyQt4.QtGui import *
import cv2
import numpy as np
from Tkinter import *
from ftplib import FTP
from facerec.feature import Fisherfaces
from facerec.classifier import NearestNeighbor
from facerec.model import PredictableModel
from PIL import Image
import sys, os
import time
import multiprocessing

import cv2
from cv2 import cv
import time
from PIL import Image
import numpy as np
import csv
import logistic
import mouthdetection as m
import RPi.GPIO as GPIO
import time



# crear nuestra pantalla
ftp=FTP("192.168.0.106")
ftp.login()
app = QApplication(sys.argv)
w = QWidget()
w.setWindowTitle('DETECTOR DE SONRISAS')
w.resize(220, 150)

btndet = QPushButton('INICIAR PROGRAMA', w)
```

```
btndet.move(30,10)
# ceramos las acciones
################################################################################
###
@pyqtSlot()
def on_click():
   WIDTH, HEIGHT = 28, 10
   dim = WIDTH * HEIGHT

   def show(area):
      cv.Rectangle(img,(area[0][0],area[0][1]),
                   (area[0][0]+area[0][2],area[0][1]+area[0][3]),
                  (255,0,0),2)
      cv.NamedWindow('Face Detection', cv.CV_WINDOW_NORMAL)
      cv.ShowImage('Face Detection', img)
      cv.WaitKey()

   def crop(area):
      crop = img[area[0][1]:area[0][1] + area[0][3], area[0][0]:area[0][0]+area[0][2]] #img[y: y +
h, x: x + w]
      return crop

   def vectorize(filename):
      size = WIDTH, HEIGHT # (width, height)
      im = Image.open(filename)
      resized_im = im.resize(size, Image.ANTIALIAS) # resize image
      im_grey = resized_im.convert('L') # convert the image to *greyscale*
      im_array = np.array(im_grey) # convert to np array
      oned_array = im_array.reshape(1, size[0] * size[1])
      return oned_array


   if __name__ == '__main__':

      smilefiles = []
      with open('smiles.csv', 'rb') as csvfile:
         for rec in csv.reader(csvfile, delimiter='    '):
            smilefiles += rec



      neutralfiles = []
      with open('neutral.csv', 'rb') as csvfile:
         for rec in csv.reader(csvfile, delimiter='    '):
            neutralfiles += rec

      phi = np.zeros((len(smilefiles) + len(neutralfiles), dim))
```

```
labels = []

PATH = "../data/smile/"
for idx, filename in enumerate(smilefiles):
   phi[idx] = vectorize(PATH + filename)
   labels.append(1)

PATH = "../data/neutral/"
offset = idx + 1
for idx, filename in enumerate(neutralfiles):
   phi[idx + offset] = vectorize(PATH + filename)
   labels.append(0)

lr = logistic.Logistic(dim)
lr.train(phi, labels)

cv2.namedWindow("PREVIA")
vc = cv2.VideoCapture(0)

if vc.isOpened(): # try to get the first frame
   rval, frame = vc.read()
else:
   rval = False

print "\n\n\n\n\nPRESIONAR ESPACIO PARA RELAIZAR LA CAPTURA O ESC PARA SALIR "

while rval:
   cv2.imshow("PREVIA", frame)
   rval, frame = vc.read()
   key = cv2.waitKey(40)
   if key == 27: # exit on ESC
      break
   if key == 32: # press space to save images
      cv.SaveImage("webcam.jpg", cv.fromarray(frame))
      img = cv.LoadImage("webcam.jpg") # input image
      mouth = m.findmouth(img)
      # show(mouth)
      if mouth != 2: # did not return error
         mouthimg = crop(mouth)
         cv.SaveImage("webcam-m.jpg", mouthimg)
         # predict the captured emotion
         result = lr.predict(vectorize('webcam-m.jpg'))
         if result == 1:
            print "SE DETECTA FELICIDAD SONRISA "
```

```
            font=cv2.FONT_HERSHEY_SIMPLEX
            cv2.putText(frame,'SONRISA !',(10,40), font, 1, (0,0,255), 5)
            cv2.imwrite("FELICIDAD.jpg",frame)
            cv2.imshow("felicidad.jpg",frame)
            cv2.imwrite('/home/pi/imagenes/sonrisa.jpg',frame)


            fp=open('/home/pi/imagenes/sonrisa.jpg','rb')
            ftp.storbinary('STOR sonrisa.jpg', fp)
            GPIO.setmode(GPIO.BOARD)
            GPIO.setup(7,GPIO.OUT)
            GPIO.output(7,True)
            time.sleep(5)
            GPIO.output(7,False)
            GPIO.cleanup()



        else:
            print "SE DETECTA SERIEDAD"
            font=cv2.FONT_HERSHEY_SIMPLEX
            cv2.putText(frame,'SERIEDAD !',(10,40), font, 1, (0,0,255), 5)
            cv2.imwrite("SERIEDAD.jpg",frame)
            cv2.imshow("SERIEDAD.jpg",frame)
            cv2.imwrite('/home/pi/imagenes/seriedad.jpg',frame)

            fp=open('/home/pi/imagenes/seriedad.jpg','rb')
            ftp.storbinary('STOR seriedad.jpg', fp)
            GPIO.setmode(GPIO.BOARD)
            GPIO.setup(11,GPIO.OUT)
            GPIO.output(11,True)
            time.sleep(5)
            GPIO.output(11,False)
            GPIO.cleanup()


    else:

        print "ERROR AL LEER LOS GESTOS "


cv2.destroyWindow("PREVIA")


##########################################################################
```

```
# conectamos las aciones a los botones
btndet.clicked.connect(on_click)
w.show()
app.exec_()

import cv2
from cv2 import cv

def findmouth(img):

 haarFace = cv.Load('haarcascade_frontalface_default.xml')
 haarMouth = cv.Load('haarcascade_mouth.xml')
 # running the classifiers
 storage = cv.CreateMemStorage()
 detectedFace = cv.HaarDetectObjects(img, haarFace, storage)
 detectedMouth = cv.HaarDetectObjects(img, haarMouth, storage)

 maxFaceSize = 0
 maxFace = 0
 if detectedFace:
  for face in detectedFace: # face: [0][0]: x; [0][1]: y; [0][2]: width; [0][3]: height
   if face[0][3]* face[0][2] > maxFaceSize:
    maxFaceSize = face[0][3]* face[0][2]
    maxFace = face

 if maxFace == 0: # did not detect face
  return 2

 def mouth_in_lower_face(mouth,face):
  # if the mouth is in the lower 2/5 of the face
  # and the lower edge of mouth is above that of the face
  # and the horizontal center of the mouth is the center of the face
  if (mouth[0][1] > face[0][1] + face[0][3] * 3 / float(5)
   and mouth[0][1] + mouth[0][3] < face[0][1] + face[0][3]
   and abs((mouth[0][0] + mouth[0][2] / float(2))
    - (face[0][0] + face[0][2] / float(2))) < face[0][2] / float(10)):
   return True
  else:
   return False

 filteredMouth = []
 if detectedMouth:
  for mouth in detectedMouth:
   if mouth_in_lower_face(mouth,maxFace):
    filteredMouth.append(mouth)
```

```
maxMouthSize = 0
for mouth in filteredMouth:
  if mouth[0][3]* mouth[0][2] > maxMouthSize:
    maxMouthSize = mouth[0][3]* mouth[0][2]
    maxMouth = mouth

try:
  return maxMouth
except UnboundLocalError:
  return 2
```