DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

UNIVERSIDAD CARLOS III DE MADRID

TRABAJO FIN DE GRADO

CREACIÓN DE UN SISTEMA DE MEDICIÓN DE FELICIDAD USANDO TÉCNICAS DE VISIÓN ARTIFICIAL

Tutor: Jorge García Bueno

Autor: Álvaro Navarro Bendito

LEGANÉS

SEPTIEMBRE 2012

"La planificación a largo plazo no es pensar en decisiones futuras, sino en el futuro de las decisiones presentes."

P. **Drucker.**

A mi YAYO y a aquellas personas que me apoyan y me quieren.

Índice general

Lista de Figuras

Resumen

En este proyecto se ha realizado un análisis de los gestos de los rostros humanos, centrándonos en la expresión de la zona de la boca para determinar el estado de ánimo de la persona cuyo rostro está siendo sometido a análisis.

Para conseguir determinar el estado de ánimo se han empleado diferentes técnicas para disminuir la influencia del ruido (datos erróneos) y de la iluminación.

Para reducir la influencia al ruido hemos utilizado 3 espacios de color con un estudio de cada uno de ellos por separado, siendo estos: El espacio de color RGB, HSV y escala de Grises.

Para una visualización amigable por parte del usuario, se ha optado por introducir emoticonos que ilustren el estado de ánimo del rostro sometido a estudio.

Cabe destacar que en este proyecto se ha conseguido localizar y diferenciar 4 estados de ánimo diferentes: Normal, Feliz, Muy Feliz y Totalmente Feliz.

Palabras clave:

OpenCV, visión artificial, visión por computador, rostro, cara, boca.

Abstract

In this project we have analyzed the gestures of human faces, focusing on the expression of the mouth area to determine the frame of mind of the person whose face is undergoing analysis.

For determining the frame of mind, we have used different techniques to reduce the influence of noise (bad data) and lighting.

To reduce the influence of noise we used 3 color spaces with a study of each of them separately, namely: The color space RGB, HSV and Gray scale.

For viewing by the user friendly, has chosen to introduce emoticons to illustrate the frame of mind of the face under study.

Note that this project has been able to locate and differentiate 4 different frames of mind: Normal, Happy, Very Happy and Completely Happy.

Keywords:

OpenCV, artificial vision, computer vision, face, mouth.

Capítulo 1

Introducción

En este capítulo explicaremos los objetivos que se pretendían conseguir en un principio, y los que se han conseguido, dando un breve repaso a los objetivos que han surgido a raíz de los objetivos principales de este proyecto.

También daremos un breve repaso a la visión artificial explicando que es, así como a sus precedentes e historia y aplicaciones, es decir, para que se utiliza.

1.1. Objetivos

El objetivo principal para el que se ideó este proyecto era desarrollar un software, utilizando el lenguaje de programación C/C++ (que explicaremos más adelante) y las bibliotecas de visión artificial de OpenCV, para la determinación del grado de felicidad de un conjunto de usuarios mediante técnicas basadas en los algoritmos de cascadas de Haar o redes Neuronales.

Para llevar a cabo lo anterior se han ido tomando una serie de subobjetivos que se mencionan a continuación:

- Reconocimiento de los rostros de los usuarios en el entorno global.
- Reconocimiento de la zona de la boca en un entorno segmentado.
- Tratamiento y transformaciones de color sobre la imagen segmentada con los criterios anteriores.
- Reconocimiento del grado de felicidad según la morfología de la boca.

Todos estos objetivos secundarios, así como los primarios y algunos que no hemos mencionado que están integrados en las partes mencionadas serán explicados más en profundidad más adelante.

1.2. Visión por computador

La visión por computador o visión artificial es una rama de investigación de la inteligencia artificial, y consiste en extraer datos del entorno por medio de dispositivos que sean capaces de captar la información visual y analizar esta información en busca de los datos que nos interesen obtener para según qué aplicaciones.

Existen múltiples aplicaciones de la visión artificial, en este caso la hemos utilizado para el análisis de rostros. El reconocimiento de rostros es una tarea que a pesar de ser relativamente sencilla para las personas, que son capaces de reconocer un número determinado de rostros a pesar de las variaciones en la luminosidad, en la rotación y en la escala (a distintas distancias),

es una tarea complicada para un ordenador o máquina en general, no obstante contamos con una serie de características comunes a todos los rostros que no son habituales en el entorno, teniendo esto en cuenta se puede empezar a trabajar sobre ello y conseguir unos resultados considerablemente aceptables.

1.3. Antecedentes

La visión artificial como tal es un concepto relativamente nuevo, sin embargo el reconocimiento y estudio de los rostros es un tema que se lleva tratando desde los años 50 por los psicólogos. Estos estudios realizados por los psicólogos llegaron a manos de los ingenieros años más tarde y fué cuando se comenzaron a realizar estudios de como reconocer rostros de manera automática con una máquina.

En los años 80 ante la impotencia de conseguir que una máquina pudiera funcionar como la visión humana, se detuvieron los estudios sobre visión artificial, todo apunta a que esta frustración se debía en gran medida a que no existía la tecnología necesaria para llevar a cabo los estudios de una manera consistente, además de la impotencia de extraer información de un mundo tridimensional en una imagen bidimensional.

Fue en los años 90 con los nuevos procesadores y los avances tecnológicos en general, cuando se retomaron los estudios sobre visión artificial y reconocimiento de rostros. Y durante los últimos 15 años la investigación sobre este asunto se ha centrado en intentar conseguir crear sistemas totalmente automatizados capaces de reconocer rostros, con independencia de los problemas derivados de esta operación, y extrayendo las diferentes características de los gestos faciales (de los ojos, boca, cejas, etc....).

1.4. Aplicaciones

El reconocimiento y análisis de rostros es uno de los campos de la visión artificial que más interés despierta ya que el rostro es la zona del cuerpo en la cual se reflejan las emociones, y

por tanto es la zona del cuerpo humano de la cual se puede extraer más información emocional.

El análisis de los rostros humanos con técnicas de visión artificial tiene múltiples aplicaciones:

- Seguridad en los aeropuertos, lugares públicos, calles principales, carreteras. . .Como se muestra en las imágenes 1.1 y 1.2.
- Identificación rápida de la policía o el ejército de personas potencialmente peligrosas.
- Los sistemas preventivos para controlar el acceso a computadoras o sistemas personales.

Figura 1.1: Reconocimiento facial para vigilancia en edificios.

Figura 1.2: Cámaras para vigilancia perimetral.

- La solución basada en Web como un motor de búsqueda de Internet imágenes.
- Web basada en la solución como un servicio de entretenimiento.
- Inteligencia artificial en robots (interacción humano-robot).

Figura 1.3: Robot ABB con posibilidad de introducción de sistema con visión artificial para guiado.

Figura 1.4: Helicóptero con cámara integrada.

Figura 1.5: Coche con navegación por medio de visión artificial. Autopía (CSIC-UPM)

- Suplantación de los sistemas de identificación reales, tales como pasaportes o tarjetas de identificación.
- Disminuir la tasa de robo de identidad.
- Video juegos, realidad virtual,....

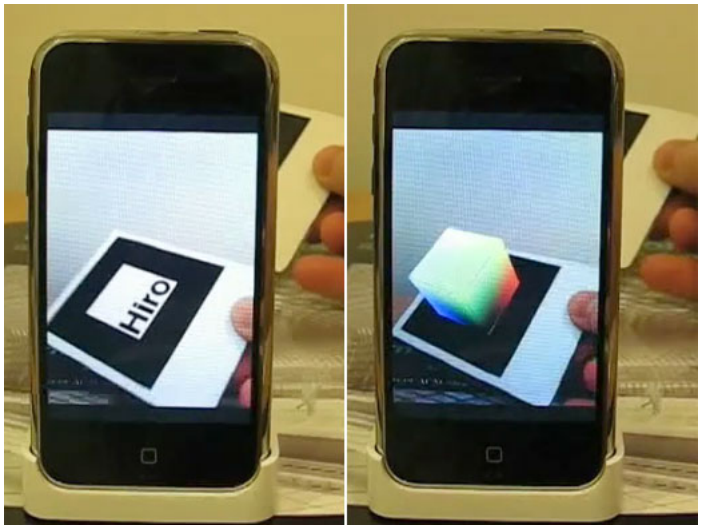


Figura 1.6: Simulación de realidad aumentada con las bibliotecas de ArtoolKit.

- Video vigilancia, control de edificios, etcétera. . .
- Control de calidad en la industria.

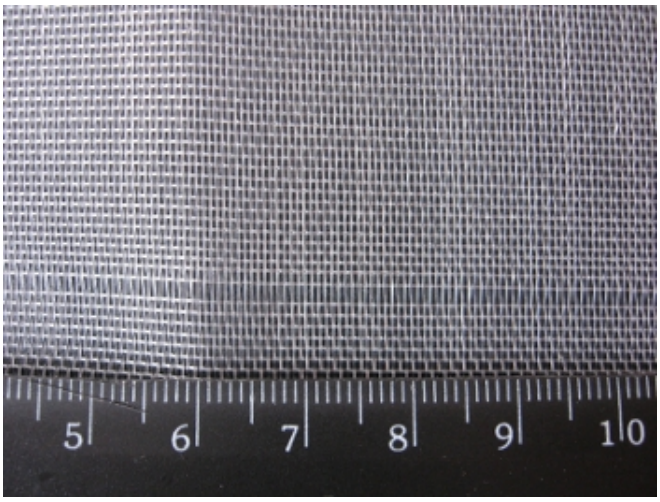


Figura 1.7: Ejemplo de control de calidad en productos textiles con visión artificial.

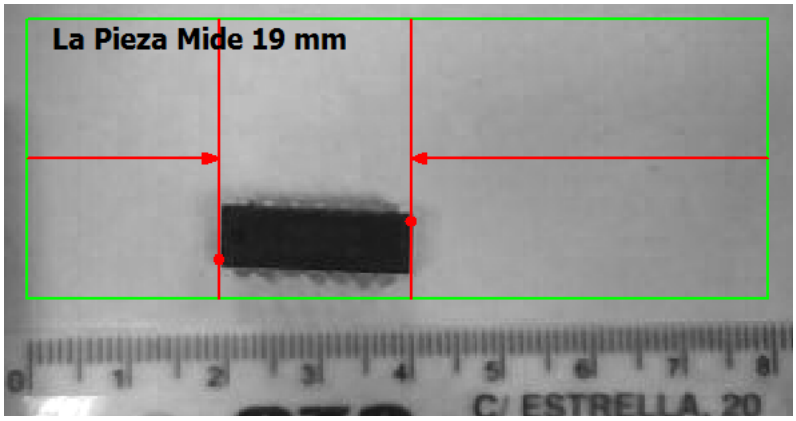


Figura 1.8: Ejemplo de control de calidad en componentes electrónicos con visión artificial.

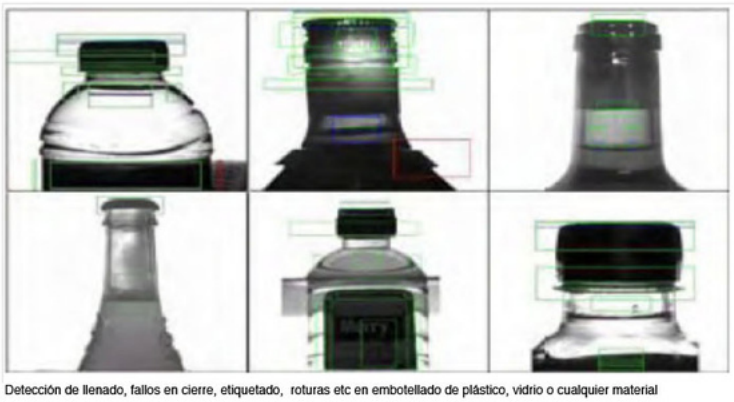


Figura 1.9: Ejemplo de control de calidad en productos plásticos con visión artificial.

- Análisis de estructuras biológicas.

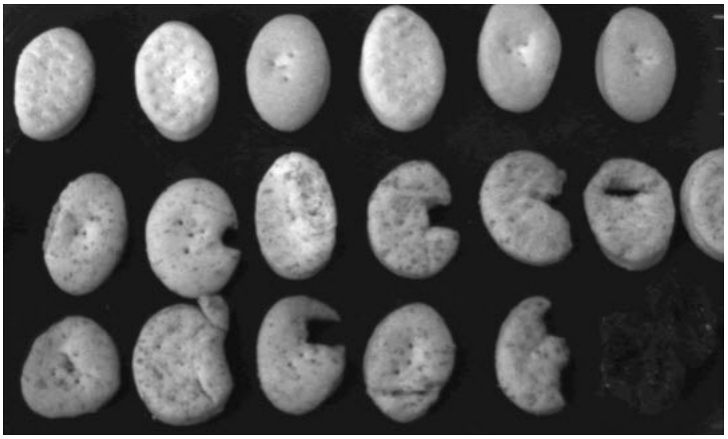


Figura 1.10: Tratamiento de imágenes para mejor diagnóstico de enfermedades.

Figura 1.11: Visualización celular con tratamiento previo sobre la imagen.

Sin duda de todas las aplicaciones anteriores la que prima es la seguridad y control, además de ser la aplicación más comercial y la cual mueve más dinero.

En nuestro caso si se decidiera realizar un estudio más amplio e intentar determinar estados de ánimo tales como el nerviosismo, el enfado, etc... o incluso detectar cuando una persona miente, serviría como sistema de seguridad ya que podríamos detectar sospechosos sólo con enfocar con la cámara a su rostro.

Capítulo 2

Aspectos teóricos

En este capítulo nos centraremos en explicar las bases teóricas sobre las que se asienta este proyecto.

Primero explicaremos que es una imagen digital, ya que es la herramienta sobre la que se aplican todas las operaciones necesarias para conseguir obtener la información requerida para la detección de los 4 estados de ánimo.

Después explicaremos que son los espacios de color, ya que es una base fundamental para el tratamiento de imágenes digitales y en concreto en los algoritmos de este proyecto.

Más tarde daremos un breve repaso sobre algunas de las operaciones que hemos realizado sobre la imagen, y de otras que no hemos utilizado y porque.

Por último nos centraremos en dar una base sobre que son las cascadas de Haar y para que se utilizan, explicando la base algorítmica.

2.1. Imagen digital

Una imagen digital es la manera en la que se le presenta la información visual extraída del entorno a una máquina o computador.

Esta información consta de un conjunto de píxeles los cuales contienen a su vez información de la posición del color en una matriz de dos dimensiones que representa la imagen. El color del píxel tiene 3 componentes que dependiendo del espacio de color que estemos manejando supondrán una cosa u otra.

La matriz que representa la información visual del entorno es bidimensional debido a que con una sola cámara convencional sólo se puede extraer información en dos dimensiones, por lo que se sufre una pérdida importante de la información del entorno y esto complica los algoritmos necesarios para el análisis del entorno.

$$f(x,y) = \begin{pmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{pmatrix}$$

De donde $f(x,y)$ es la matriz que define la imagen digital, y cada uno de los elementos constituyentes de la matriz son los píxeles.

2.2. Espacios de color

Existen varios tipos de espacios de color, cada uno de ellos tiene sus ventajas e inconvenientes que explicaremos a continuación:

- Espacio de color RGB

  Utiliza un concepto del color basado en la mezcla de Rojo (R = Red), Verde (G = Green) y Azul (B), en teoría con la mezcla de estos tres colores en distintas proporciones podemos conseguir cualquier otro color. Los valores de estas componentes van desde 0 hasta 255, siendo el RGB = 000 el Negro y RGB = 111 el blanco. En la figura 2.1 se muestra una ilustración de cómo sería el modelo de este espacio de color.

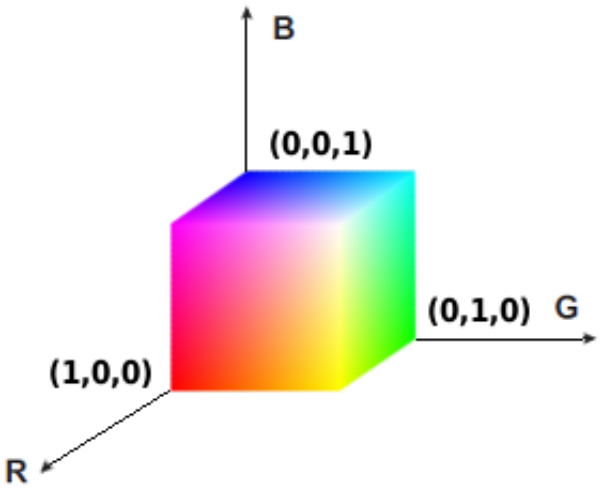


Figura 2.1: Espacio de color RGB

Este espacio de color tiene la ventaja de que es muy estable y gracias a la diferenciación de las componentes en Rojo, Verde y Azul se puede segmentar en principio fácilmente. Las desventajas que tiene este espacio es que es muy sensible a cambios en la iluminación del entorno, y que a efectos prácticos la segmentación en este espacio no da tan buenos resultados como en otros espacios debido a esto.

La ecuación que define este sistema es la siguiente:

$$P(x, y) = R + G + B$$

Siendo $P(x, y)$ el píxel correspondiente a las coordenadas (x,y) cuyo valor de color dependerá de las componentes R, G y B en el entorno real.

- Espacio de color HSV

  El espacio de color HSV está basado en un concepto del color bastante intuitivo. Al igual que los demás espacios de color consta de 3 componentes que lo definen, que son: H (Hue), S (Saturation), V (Value). En la figura 2.2 se muestra el modelo HSV.

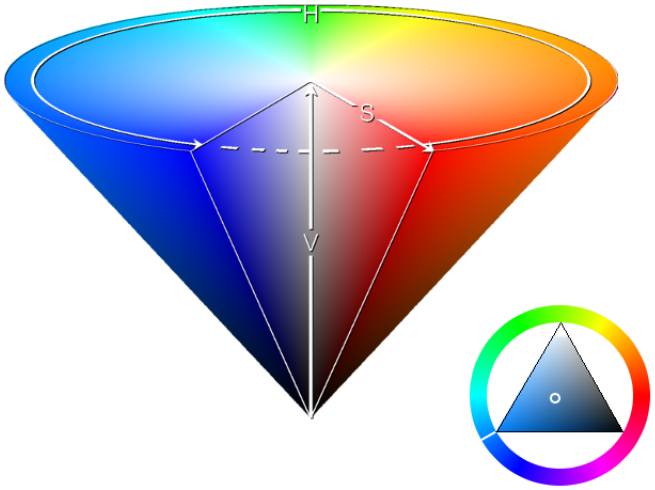


Figura 2.2: Espacio de color HSV

Véase que es muy parecido al espacio de color HSI donde la única diferencia relevante es la diferencia entre el ”Value” y la ”Intensity” o ”claridad” y ”brillo”. La diferencia es que el ”brillo” de un color puro es igual al brillo del blanco, mientras que la claridad de un color puro es igual a la claridad de un gris medio. A continuación, en la figura 2.3, se muestran las diferencias entre ambos espacios.

Este espacio de color tiene algunas ventajas con respecto al espacio de color RGB, como

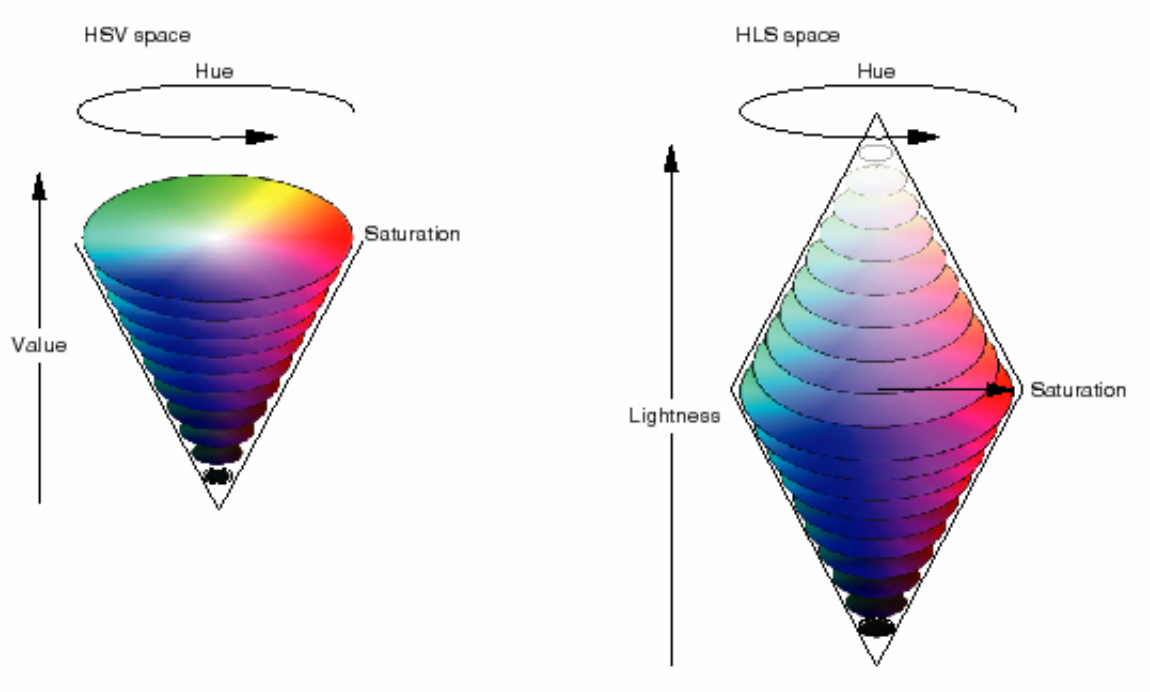


Figura 2.3: Espacios de color HSV y HSI

por ejemplo que al separar las componentes no en componentes de color sino de matiz, brillo y saturación se consigue una mayor independencia o al menos menor sensibilidad a los cambios de iluminación. A pesar de ser un espacio de color que cuenta con ventajas significativas existen algunos inconvenientes a la hora de elegir este espacio de color que hay que tener en cuenta, como por ejemplo su inestabilidad para colores con baja saturación, o lo que es lo mismo, para colores cercanos a los grises. Lo que sucede al trabajar con grises en HSV es que no distingue entre un rojo con baja saturación de un verde o azul con baja saturación, y por tanto el sistema se vuelve inestable en la componente H, da grandes saltos para pequeñas variaciones y de manera prácticamente aleatoria al no reconocer el gris que se le presenta.

- Escala de Grises

  Este espacio de color a pesar de estar compuesto por 3 componentes a efectos prácticos es como si se tratara de una sola componente cuya atenuación se encuentra más o menos acentuada. Esta componente de la que hablamos sería el blanco cuya máxima atenuación

daría como resultado el color negro, pasando por todos los niveles de gris anteriores (desde 0 hasta 255). Este color blanco no es más que la composición de los colores RGB pero en este caso no sólo con la misma proporción sino con el mismo valor de píxel para las 3 componentes, definiéndose por la siguiente ecuación:

$$P(x,y) = \frac{1}{3}R + \frac{1}{3}G + \frac{1}{3}B$$

Siendo $P(x,y)$ el píxel correspondiente a las coordenadas (x,y) cuyo valor de color dependerá de las componentes R, G y B en el entorno real, y teniendo en cuenta que el valor de R es igual al de G y al de B.

Puede observarse que este espacio de color conlleva una pérdida importante de información, para ser exactos dividimos la información relevante por 3. Esto puede resultar interesante para reducir la carga de proceso en algunos aspectos, pero hay que tener en cuenta que este hecho nos perjudica desde el punto de vista de pérdida de datos. En la figura 2.2 se muestra un ejemplo de la conversión de una imagen de RGB a escala de Grises.

(a) Imagen RGB (b) Imagen en escala de grises

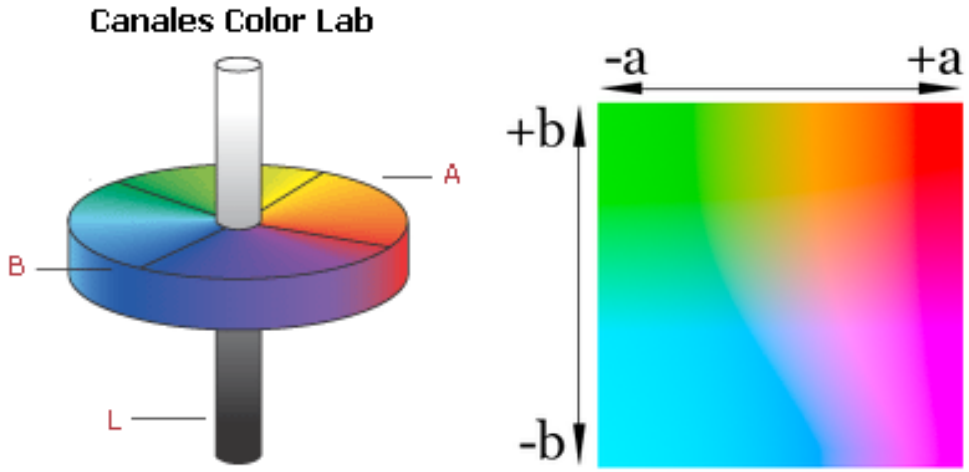
Figura 2.4: Transformación de la imagen a color a escala de grises.

- Color Lab

  El color Lab es una representación del color la cual intenta representar un espacio que sea capaz de definir unas variables cuya correspondencia entre las variaciones del nivel de estas variables y la variación visual perceptible del color sean similares, es decir, que sea más ”perceptiblemente lineal” que otros espacios de color.

  Se utiliza no sólo por ser más perceptibles los cambios realizados en las variables, si no porque al separar las componentes de la manera en que lo hace esto permite que este sistema sea mucho menos sensible a los cambios en la iluminación.

  Al igual que el resto de espacios de color que hemos definido anteriormente consta de 3 componentes L, A, B donde L es la Luminosidad, A es una variable cuyos valores positivos corresponden a las distintas tonalidades del color rojo y cuyos valores negativos corresponden a las distintas tonalidades del verde y B que es una variable cuyos valores positivos corresponden al color amarillo y cuyos valores negativos corresponden con las distintas tonalidades del azul. En la figura 2.2 se muestra el modelo del espacio de color LAB.



(a) Modelo Espacio de color LAB (b) Variables A y B

Figura 2.5: Espacio de color LAB

2.3. Limitaciones de la imagen digital

Una imagen digital como ya hemos dicho se puede definir como una matriz cuyos valores son los píxeles, pero ¿qué son los píxeles?

Al definir una imagen digital lo que se realiza es un muestreo espacial de manera discreta, es decir un muestreo no continuo o analógico, estableciéndose un mismo valor de color para un área de un tamaño determinado llamado píxel.

El área del píxel es determinante en la imagen digital ya que este tamaño de píxel determina la cantidad de información que tomamos del entorno o mejor dicho la cantidad de información que decidimos desechar o despreciar. Por ello una imagen digital sólo es un modelo estimado del entorno real cuya calidad dependerá de la tecnología con la extraigamos la información y de lo pixelada que se encuentre nuestra imagen digital.

Algunas de las limitaciones procedentes de la tecnología es el desbordamiento y "contagio" del valor del píxel cuando existe una iluminación excesiva, a este fenómeno se le conoce como *blooming*. Pero en nuestro caso esto no sucede por ser una cámara CMOS, que lo explicaremos más adelante en profundidad.

2.4. Transformaciones y algoritmos

Como ya hemos dicho una imagen es la manera en la que se le presenta la información visual extraída del entorno a una máquina o computador, pero para que el análisis de esta imagen en búsqueda de una información determinada sea óptimo, debemos realizar una serie de transformaciones y operaciones que simplifiquen el sistema sin destruir, eliminar o mezclar información de interés.

Todos los procesos de análisis de imágenes digitales llevan consigo, por tanto, un estudio previo y/o posterior del entorno a analizar, de tal forma que nos centremos en diseñar una serie de algoritmos que filtren la información que nos interesa para el caso en cuestión, o mejoren o resalten ciertas características para facilitar la búsqueda y extracción de la información objeto de la aplicación.

Dado que una imagen digital se basa en color del entorno, los algoritmos que se utilizan para el análisis de esta son operaciones que se realizan pensando en el color y la posición de este en el espacio bidimensional que compone la imagen.

A continuación explicaremos algunos de los algoritmos y herramientas más importantes a la hora de analizar una imagen, centrándonos en los algoritmos que se pueden utilizar para el reconocimiento facial, y en los algoritmos que son objeto de este proyecto.

A la hora de analizar una imagen se suelen realizar varias operaciones las cuales tienen como objetivo conseguir los mejores resultados posibles:

Primeramente se suele realizar una operación que elimine algún error o característica no relevante para el objeto del estudio de la imagen o que realce alguna característica que nos interese, estas operaciones pueden ser de filtrado o eliminación de ruido. En nuestro caso hemos realzado el color blanco para el observar si la persona que estaba siendo estudiada estaba sonriendo con más o menos intensidad (con dientes o sin dientes). La ecuación que definiría este algoritmo es la siguiente:

$$f(x,y) = \left\{ \begin{array}{ll} 255 & \text{si } f(x,y) \geq \text{T} \\ 0 & \text{si } f(x,y) < \text{T} \end{array} \right\}$$

Teniendo en cuenta que $f(x,y)$ es la variable estudiada y T es el valor de umbral elegido para la variable que está siendo estudiada. Esto no es más que una umbralización del color en el espacio que estemos trabajando que tiene como objetivo, en este caso en concreto, la ”binarización” que es convertir la imagen a otra en la cual sólo existen píxeles completamente negros (nivel de gris = 0) y completamente blancos (nivel de gris = 255) . Y en nuestro caso esta operación ha sido empleada para modificar valores de la ”Saturation” y del ”Value” en el espacio de color HSV, junto con una modificación del valor de Gris en el espacio de color de la Escala de Grises.

Pero dadas las características especiales de este proyecto, el orden de las operaciones que se suelen realizar para análisis de imágenes está bastante cambiado. Por lo que antes de realizar la umbralización por color hemos realizado una ecualización del histograma, pero ¿Qué es el histograma?.

El histograma es una representación gráfica del número de píxeles que corresponden a un nivel de color en una de las variables empleadas en el color. Para entenderlo mejor se muestra la figura 2.6 en la cual podemos observar a la derecha el histograma en RGB de la imagen de la flor.

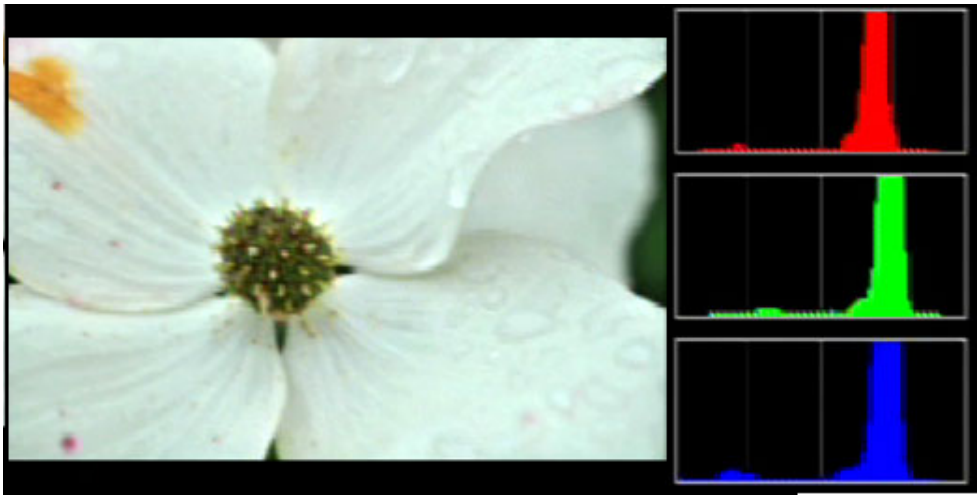
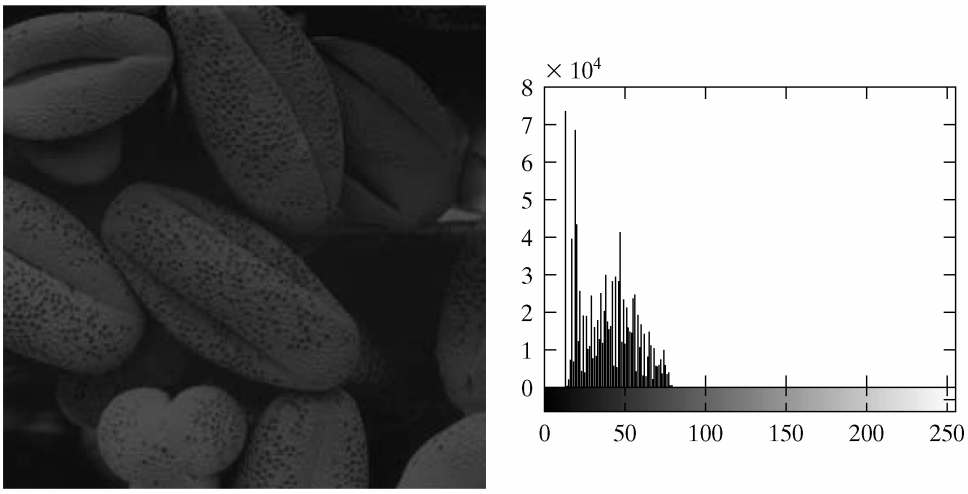
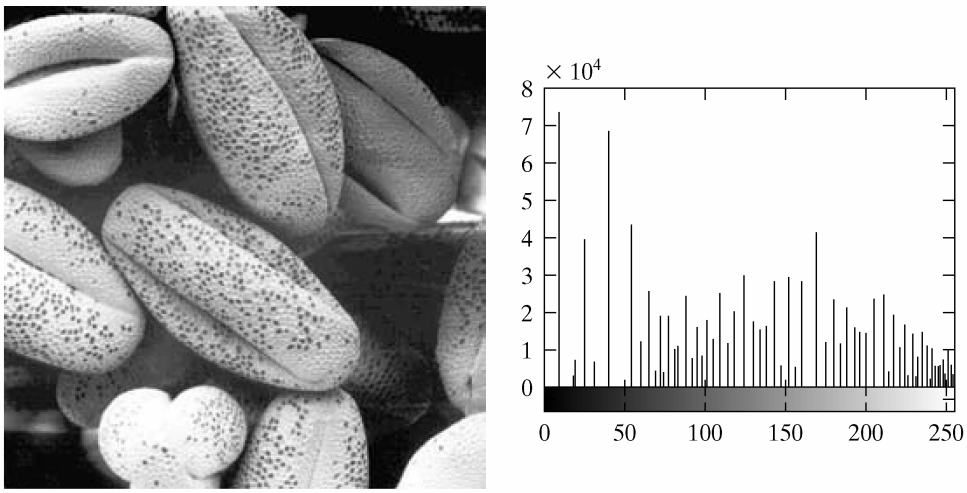


Figura 2.6: Histograma en RGB

La ecualización del histograma se realiza cuando el histograma de la imagen nos dice que el contraste en la imagen es inferior de lo que debería, es decir que los niveles de color están muy compactados en una zona. Se realizará una transformación sobre el histograma de tal forma que separemos los niveles de color, de manera proporcional, aumentando el contraste y la definición de la imagen. En la imagen 2.4 se muestra el histograma antes y después de realizarle una ecualización, junto con la imagen antes y después de realizarle esta transformación.



(a) Figura con bajo contraste



(b) Figura ecualizada

Figura 2.7: Ecualización de una imagen para aumentar su contraste, a la izquierda las imágenes donde se muestra el antes y el después de la ecualización y a la derecha los histogramas correspondientes.

Otro de los algoritmos que hemos utilizado para la segmentación de una porción del entorno, la cual nos interesaba tratar por separado, es el algoritmo ROI (Region Of Interest). Este algoritmo consiste en nuestro caso en extraer primero la porción correspondiente a los rostros de los usuarios para luego buscar las bocas en esas regiones y segmentar después las bocas con este mismo algoritmo para realizarle los tratamientos que hemos descrito anteriormente.

En la figura 2.8 se puede observar un ejemplo de este algoritmo que en nuestro caso utiliza un contorno cuadrado que inscriba la región de interés para recortar la imagen.

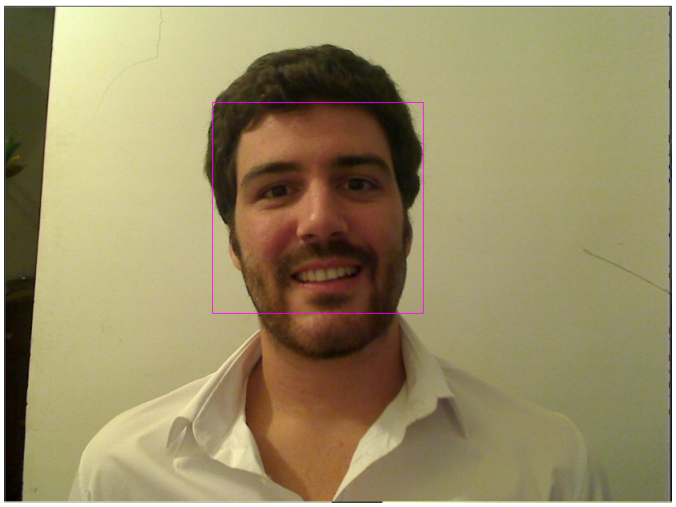


Figura 2.8: Region of interest

A continuación mencionaremos algunos de los algoritmos que barajamos en un principio y que finalmente no utilizamos por diversos motivos:

En un principio pensamos que el algoritmo SIFT (Scale Invariant Feature Transform) podría servir para definir la morfología de la boca debido a los puntos característicos que podemos encontrar en los alrededores de los labios de la boca, como se observa en la figura 2.9.

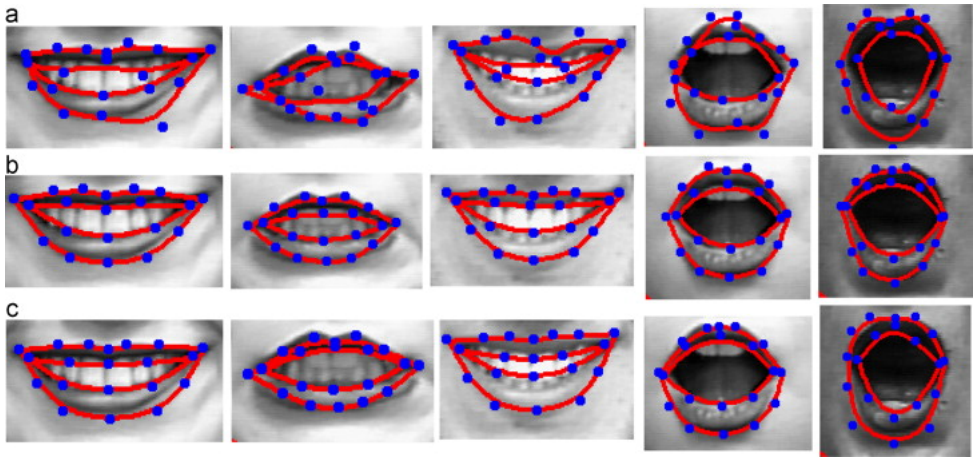


Figura 2.9: Algoritmo SIFT aplicado a bocas para determinar su morfología

Este algoritmo consiste en encontrar puntos con grandes diferencias con respecto al resto (o contraste) a los que llamamos Keypoints o puntos de interés, de tal forma que sean fáciles de encontrar en el siguiente "frame" de la imagen de tal forma que seamos capaces de localizar objetos o formas a pesar de que los rotemos o los alejemos (con un cierto límite) de la cámara.

Pero este algoritmo tiene el inconveniente de que si no se cuenta con suficiente contraste y unas condiciones de iluminación relativamente buenas, se pierden los Keypoints o puntos de interés que definen la boca, además debido a que sólo teníamos que reconocer 4 estados de ánimo no vimos necesaria la implementación de este algoritmo.

2.5. Cascadas de Haar

El reconocimiento de objetos o regiones relevantes de una imagen, se puede realizar de una manera mucho más eficiente si nos centramos en la detección de una serie de características específicas del objeto o la región estudiada. Cuantas más características otorguemos a la idea que tenemos de un objeto, más restrictivos seremos y menos posibilidades de que el sistema reconozca como objeto lo que no lo es obtendremos.

El rostro humano plantea más problemas que otros objetos que deseemos detectar ya que el rostro humano es un objeto dinámico, es decir, que viene dado de muchas formas y colores. Sin embargo, es muy interesante realizar la detección y seguimiento facial. El reconocimiento facial no es posible si la cara no se encuentra aislada del fondo, entendiendo como fondo todo lo que no sea un rostro.

Aunque existen varios algoritmos diferentes para llevar a cabo la detección de rostros, cada uno tiene sus propias ventajas e inconvenientes. Estos algoritmos suelen estar basados en la segmentación por tonos de la piel, contornos, y otros algoritmos más complejos que utilizan plantillas, redes neuronales, etc. Todos estos algoritmos tienen el mismo problema, que es que la carga computacional que tienen es demasiado alta para realizar un análisis en tiempo real. Esto se debe a que una imagen es un conjunto de valores de color y de intensidad, y el análisis de estos píxeles (donde se encuentran almacenados los valores de color, intensidad y posición) para la detección de la cara lleva consigo una carga computacional demasiado alta.

Viola y Jones ideó un algoritmo, llamado Clasificador de Haar, que fue pensado para detectar rápidamente cualquier objeto, incluyendo las caras humanas, usando AdaBoost, que se basan en Haar-like features y no en el análisis de todos y cada uno de los píxeles, lo que en principio reduciría significativamente la carga de proceso.

La base fundamental sobre la que se apoya la detección con Cascadas de Haar son los llamados Haar-like features. Estas características, en lugar de utilizar los valores de intensidad de un píxel, utilizan el cambio de los valores de contraste entre grupos rectangulares adyacentes de píxeles. Las desviaciones de contraste entre los grupos de píxeles se utilizan para determi-

nar la iluminación relativa y las zonas oscuras. Dos o tres grupos de píxeles adyacentes con una variación del contraste similar forman lo que denominamos como Haar-like feature. Las Haar-like features se utilizan para detectar una imagen. Estas características se pueden escalar (cambiar de tamaño de manera proporcional) fácilmente aumentando o disminuyendo el tamaño del grupo de píxeles que está siendo examinado. Esto permite podamos detectar el mismo objeto para diferentes tamaños en la imagen.

Las características rectangulares de una imagen se calculan utilizando una representación intermedia de una imagen, llamada imagen integral. La imagen integral es una matriz que contiene las sumas de los valores de intensidad de los píxeles situados a la izquierda y encima del píxel (x,y). Así, si $A[x, y]$ es la imagen original y $AI[x, y]$ es la imagen integral, la imagen integral se calcula como se muestra en la siguiente ecuación y se ilustra en la 2.10.

$$AI(x_1, y_1) = \sum_{(0,0)}^{(x,y)} A(x, y) \tag{2.1}$$

Siendo $x <= x_1$ y $y <= y_1$.

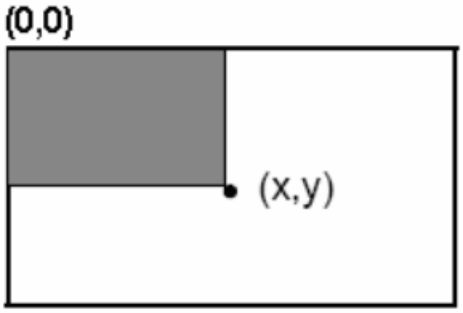


Figure 2 Summed area of integral image

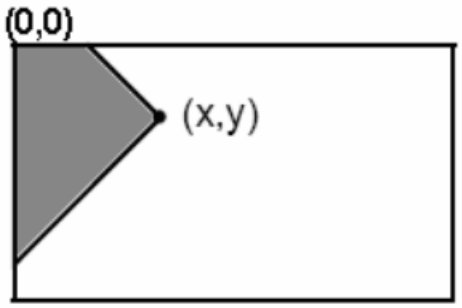


Figure 3 Summed area of rotated integral image

Figura 2.10: Representación del cálculo de las imágenes integrales para el algoritmo de Cascadas de Haar basado en Haar-like features.

Sólo se necesitan dos pasos para calcular ambos conjuntos integrales de la imagen, uno para cada conjunto. Tomando la imagen integral adecuada y tomando la diferencia entre seis u ocho elementos de la matriz que forman dos o tres rectángulos conectados, se puede calcular una característica para cualquier escala. De esta forma, el cálculo de la función es extremadamente rápido y eficiente.

En la imagen 2.11 se muestran las imágenes integrales existentes:

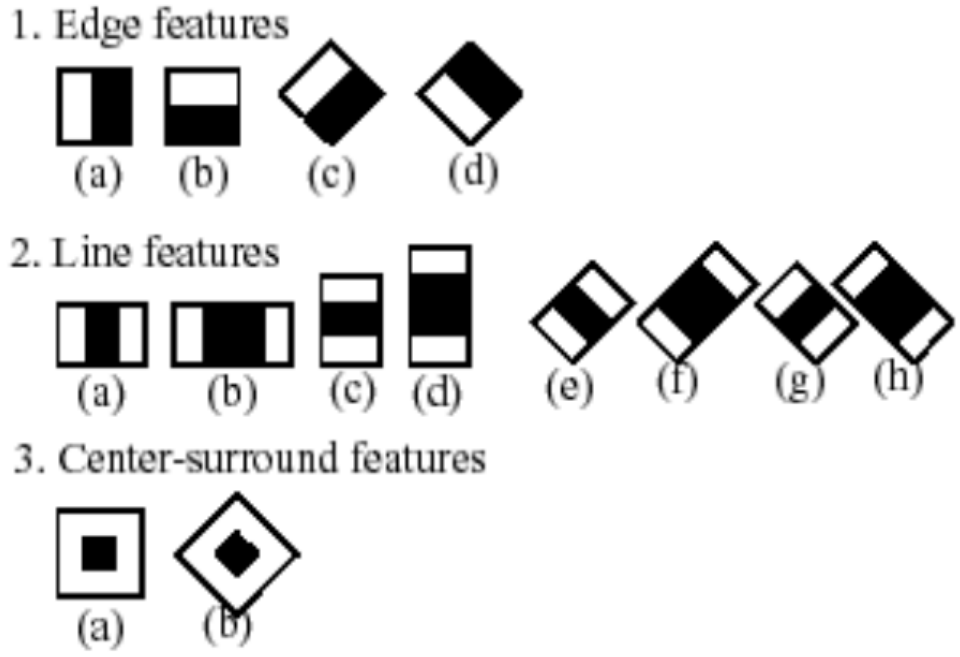


Figura 2.11: Imágenes integrales para el clasificador basado en Haar-like features.

Centrándonos en nuestro caso las Cascadas de Haar en el caso de OpenCV, funcionan de la siguiente forma. Primero se entrena el sistema mostrándole unos pocos cientos de casos que llamaremos ”positivos”, que tienen que estar escalados de tal forma que todos tengan el mismo tamaño y en los cuales sabemos que se encuentra el objeto o región de interés que someteremos a estudio (por ejemplo en nuestro caso se entrenaría con ejemplos positivos en los cuales hay rostros para el primer algoritmo y bocas para el segundo). Más tarde someteremos al sistema a entrenamiento con imágenes que no contengan ”positivos”, a estas imágenes las llamaremos ”negativos”.

Una vez está entrenado el clasificador, ya puede ser utilizado sobre una imagen, y lo que hace es pasar la ventana del tamaño de entrenamiento (en principio) por toda la imagen de manera iterativa, y devolverá un ”1” si se considera que la probabilidad de que la región que se acaba de estudiar es lo suficientemente grande como para ser considerado como ”positivo”, y un ”0” en caso contrario. Con el fin de que este algoritmo sea más eficiente, está implementado de tal forma que el tamaño de la ”máscara” que pasamos por toda la imagen es variable y prueba con varios tamaños, si no lo hiciera de este modo, el sistema sólo reconocería los objetos que tuvieran por casualidad el mismo tamaño en la imagen que la máscara, y esto reduciría significativamente la calidad y utilidad del clasificador.

Se dice que este clasificador es en ”cascada” porque el clasificador ”global” se compone de varios clasificadores simples o etapas de clasificación, que se aplican varias veces sobre la misma región hasta que la región candidata de ser positiva es rechazada o los clasificadores no la rechazan y es considerado como positivo.

En cuanto a lo que hemos explicado antes sobre las máscaras que se pasan sobre la imagen, existen algoritmos que realizan operaciones similares, uno de estos métodos, que es bastante más simple que las Cascadas de Haar, es la correlación.

La búsqueda de correlación en lo puntos de una imagen se realiza con un patrón que tiene una escala determinada, y que se superpone a la imagen donde se está buscando.

Debido a que al igual que en una simple resta de la máscara con la imagen se provocaría en ciertos casos desbordamiento de los valores de niveles de color de los píxeles (ya que si el valor que restamos de la máscara a la imagen es mayor nos dará un valor negativo, y esto no es posible representarlo ya que los valores sólo van desde 0 hasta 255), en la correlación como tal también sucedería esto, por ello se estableció la correlación normalizada, de tal forma que nunca podamos desbordar los valores de los píxeles, pero a la vez que podamos determinar si el píxel estudiado es el que corresponde con su dual en la máscara de manera sencilla.

A continuación se muestran las ecuaciones que rigen la correlación normalizada:

$$C_n = \frac{1}{\sigma_I \sigma_h} \sum \frac{(I(x,y) - m_I)(h(x,y) - m_h)}{N} \tag{2.2}$$

Donde C_n es la correspondencia de la correlación, cuyo valor de correlación va desde -1 hasta 1, n es el número de píxel.

$$m_I = \frac{\sum I(x,y)}{N} \tag{2.3}$$

$$\sigma_h^2 = \frac{\sum (I(x,y) - m_h)^2}{N} \tag{2.4}$$

$$\sigma_I^2 = \frac{\sum (I(x,y) - m_I)^2}{N} \tag{2.5}$$

Donde $I(x,y)$ es la imagen objeto de estudio, y $h(x,y)$ es la máscara que pasamos por toda la imagen.

$$I(x,y) = \begin{pmatrix} I(0,0) & I(0,1) & \cdots & I(0,N-1) \\ I(1,0) & I(1,1) & \cdots & I(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ I(M-1,0) & I(M-1,1) & \cdots & I(M-1,N-1) \end{pmatrix}$$

$$h(x,y) = \begin{pmatrix} h(0,0) & h(0,1) & \cdots & h(0,A-1) \\ h(1,0) & h(1,1) & \cdots & h(1,A-1) \\ \vdots & \vdots & \ddots & \vdots \\ h(B-1,0) & h(B-1,1) & \cdots & h(B-1,A-1) \end{pmatrix}$$

Hay que tener en cuenta que el tamaño de la máscara ($h(x,y)$), para que funcione el algoritmo, tiene que ser menor que el de la imagen ($I(x,y)$) y en una proporción determinada.

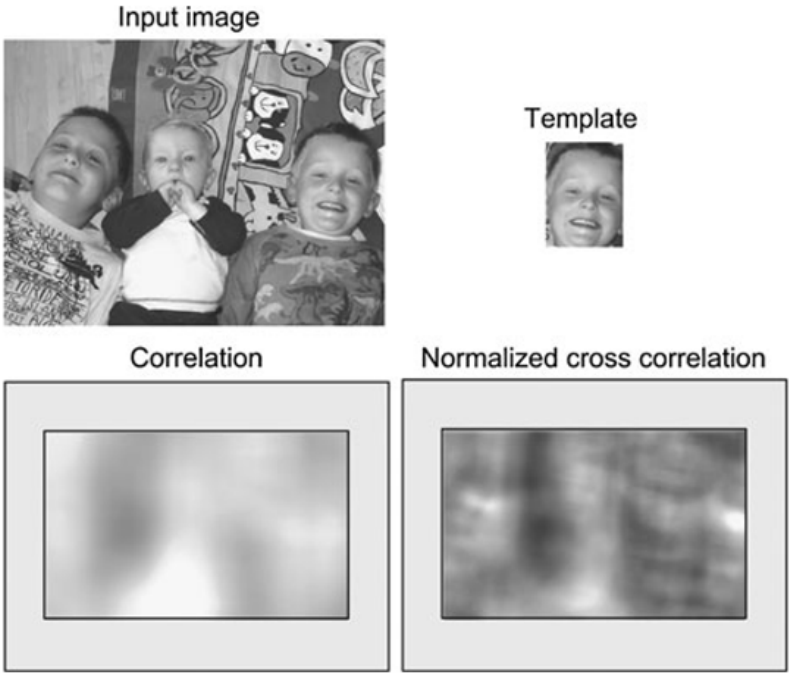


Figura 2.12: Correlación y correlación normalizada cruzada: A la derecha se muestra la máscara que pasamos por la imagen de la izquierda de manera iterativa. Los resultados del valor de la correlación y de la correlación normalizada cruzada se muestran justo debajo de estas imágenes. Las zonas más blancas son las de mayor correlación y las zonas más oscuras por el contrario son las de menos correlación.

En la imagen 2.12 se muestra un ejemplo de correlación y correlación normalizada cruzada.

Nótese que el tamaño de la imagen donde se muestra el resultado de la operación de correlación es inferior al tamaño de la imagen base que introdujimos para estudiarla. Esto se debe a que la máscara tiene un área efectiva que no cubre toda la imagen, por limitaciones de su propio tamaño.

En los resultados de la imagen puede observarse que las zonas con mayor correlación no corresponden con la zona en la que se encuentra el patrón, mientras que en la correlación normalizada si corresponde.

Capítulo 3

Plataforma de la aplicación

En este capítulo explicaremos la plataforma software y hardware que hemos utilizado y las razones de haberlas elegido.

En cuanto al software nos centraremos en comentar el lenguaje de programación que hemos usado (comentando sus precedentes), el Sistema Operativo elegido y las bibliotecas utilizadas para la programación de la aplicación.

En lo referente al hardware daremos un repaso al dispositivo que hemos utilizado, con una breve explicación de sus características comentando las ventajas de utilizar este dispositivo frente a otros parecidos.

3.1. Lenguaje de programación

El lenguaje de programación que hemos utilizado es el Lenguaje C/C++. El lenguaje C surgió en los laboratorios Bell de ATT ha sido asociado con el sistema operativo UNIX. Su eficiencia y facilidad de uso han hecho que el lenguaje ensamblador apenas haya sido utilizado en UNIX. Este lenguaje ha evolucionado paralelamente a UNIX. Una muestra de esto es que en 1980 se añaden al lenguaje C nuevas funcionalidades como clases, conversión de tipo y chequeo del tipo de argumentos de una función, entre otras, a este desarrollo del lenguaje C se le denominó lenguaje C con Clases.

En 1983, el lenguaje C con Clases sufrió una evolución y con ella fue rediseñado, extendido y nuevamente implementado. A esta nueva evolución se le denominó Lenguaje C++. Ahora las extensiones principales eran funciones virtuales, funciones sobrecargadas (un mismo identificador puede representar distintas funciones), y operadores sobrecargados (un mismo operador puede utilizarse en distintos contextos y con distintos significados).

Existen varios motivos, además de los anteriores, por los cuales hemos elegido usar este lenguaje de programación y no otro.

El primero de los motivos es el conocimiento previo que tenía sobre este lenguaje cursado en dos asignaturas diferentes a lo largo de la carrera. Pero hay otros motivos de peso que nos inclinan a sentirnos a gusto programando en este lenguaje ya que es un leguaje superior, intuitivo y sencillo de programar (ya que es de alto nivel):

- Programación orientada a objetos.
- Portabilidad.
- Brevedad.
- Programación modular.
- Compatibilidad con C.
- Velocidad.

Pero la razón más importante para haber escogido este lenguaje de programación es que las bibliotecas de visión artificial que hemos utilizado se encuentran en este lenguaje. Si bien es cierto que nuestras bibliotecas se encuentran en otros lenguajes, el Lenguaje C/C++ es el lenguaje, de los que permite utilizar, en el cual sabemos programar.

3.2. Sistema Operativo

El Sistema Operativo que hemos decidido utilizar para la realización de este proyecto es Linux, esto se debe a que este entorno posee una serie de características que lo hacen más rápido y estable que otros sistemas operativos. También es cierto que la utilización de este sistema Operativo no sería posible para la realización de este proyecto si no existiera compatibilidad con las bibliotecas y hardware utilizados.

En concreto nos hemos decantado por Ubuntu 12.04 de 32 bits, que se trata de una distribución GNU/Linux, cuya licencia es totalmente libre.

Otras ventajas con las que cuenta este sistema operativo son:

- Multiplataforma, multitarea, multiprocesador y multiusuario.
- Uso de bibliotecas enlazadas estática y dinámicamente
- Protección de la memoria, haciéndolo más estable frente a caídas del sistema.
- Carga selectiva de programas según la necesidad.

3.3. Bibliotecas

En nuestro proyecto hemos decidido utilizar las bibliotecas de OpenCV (Open Computer Vision) que es una herramienta con múltiples bibliotecas y funciones para tratamiento de imágenes en 2 dimensiones.

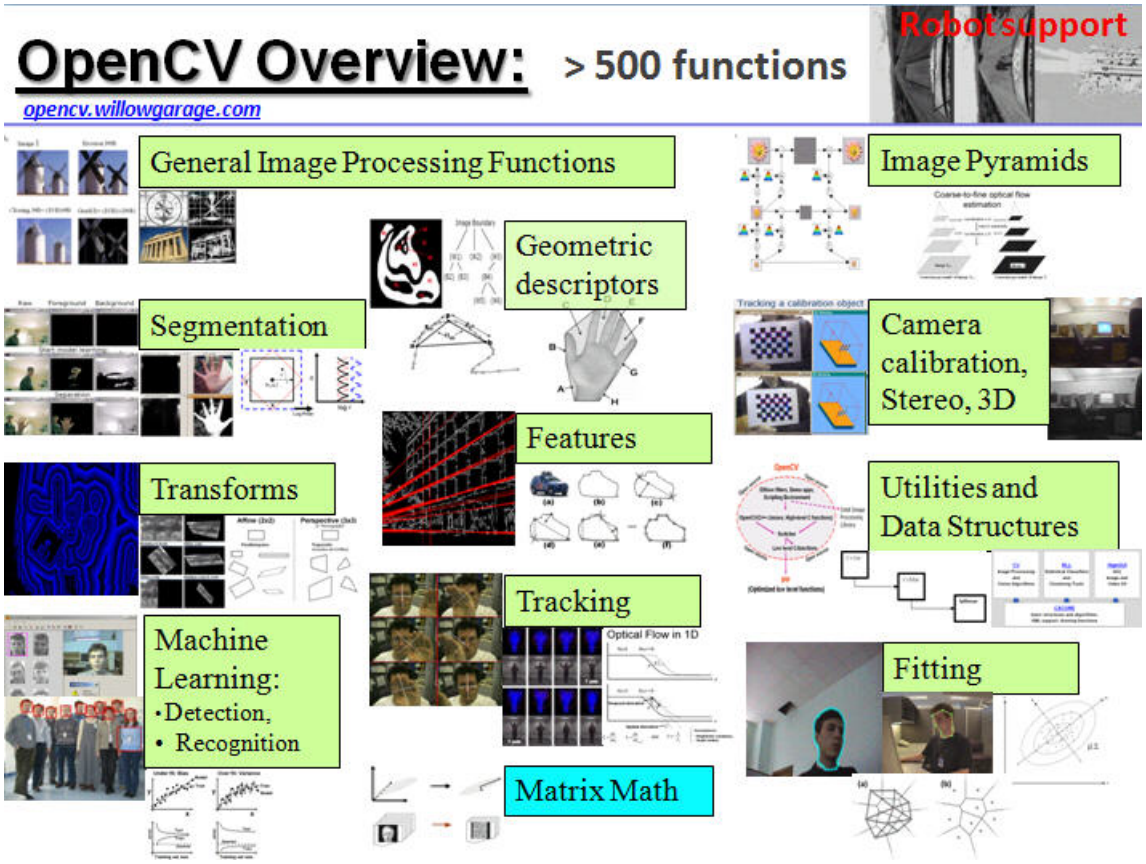


Figura 3.1: Funcionalidades de OpenCV

Como ya hemos dicho es compatible con GNU/Linux pero también con otros Sistemas Operativos como Microsoft Windows, Android y Mac, y no sólo eso, sino que además es compatible con los lenguajes de programación en C, C++, Python y Java, por tanto podemos decir que es multiplataforma.

Su código es totalmente abierto y libre para su utilización comercial y para fines de investigación, ya que se encuentra bajo los términos de la licencia BSD.

OpenCV cuenta con más de 2500 algoritmos optimizados para preprocesamiento (filtrado), transformaciones, búsqueda de características, etc, es decir para la programación y análisis de imágenes y en concreto cuenta con funciones y algoritmos preestablecidos capaces de realizar reconocimiento de objetos y reconocimiento facial. Además el uso de OpenCV como plataforma para análisis de imágenes se encuentra muy extendido por todo el mundo (más de 40000

usuarios). Lo que ha hecho que la elección de estas bibliotecas sea acertada.

En concreto hemos utilizado la versión OpenCV 2.3.1. para el Sistema Operativo Linux que cuenta con numerosos ejemplos de programas en los cuales se muestran las funcionalidades de los algoritmos propios de esta herramienta.

Toda la información sobre OpenCV puede consultarse en su página:
http://opencv.willowgarage.com/wiki/

3.4. Plataforma Hardware

Para la realización de este proyecto hemos utilizado el dispositivo Pro 9000 de Logitech, a continuación explicaremos la razón de su elección, las ventajas y desventajas con respecto a otros dispositivos y las características técnicas.

Antes de introducirnos más en profundidad en lo referente a la parte técnica del dispositivo, vamos a explicar la diferencia entre una cámara CCD y una cámara CMOS comentando las ventajas e inconvenientes de ambas tecnologías.

Tanto la tecnología CMOS como la CCD se basan en el efecto fotoeléctrico. En el caso de la tecnología CMOS el sensor está formado por una serie de fotositos y cada uno de ellos compone lo que denominamos como pixel, que produce una señal eléctrica en consecuencia a la intensidad luminosa recibida.

A diferencia del sensor CCD en el CMOS se incorpora un amplificador de señal eléctrica en cada uno de los fotositos, en el CCD se amplifica una vez extraída y enviada al exterior la información relativa a cada fotosito.

Una de las ventajas más significativas de los sensores CMOS con respecto a los sensores CCD es que debido a que la electrónica puede leer directamente la señal de cada píxel, evitamos el contagio del desbordamiento producido por una cantidad elevada de luminosidad

conocido como *blooming*.

En ambas tecnologías los fotositos sólo captan intensidad lumínica (no distinguen el color), por lo que se emplea un filtro conocido como máscara de Bayer, la cual hace que una serie de fotositos se especifiquen en la detección de la intensidad lumínica de un sólo color (rojo, verde o azul).

Ventajas de la tecnología CMOS:

- Menor consumo eléctrico.
- Más baratas que las cámaras CCD
- Lectura simultánea de mayor número de pixeles
- El conversor digital puede estar integrado en el mismo chip.
- No le afecta el *blooming*.
- Mayor flexibilidad en la lectura de imágenes.
- Los pixeles pueden ser expuestos y leídos simultáneamente.
- Distintos tipos de píxeles (según tamaño y sensibilidad) combinables.
- Muy alta frecuencia de imagen en comparación a un CCD del mismo tamaño.

Desventajas de la tecnología CMOS:

- Menor superficie receptora de la luz por píxel.
- Menor uniformidad de los píxeles (mayor ruido de patrón fijo-FPN).
- Efecto ”jelly” o inestabilidad en la imagen con movimientos rápidos o flashes.

En nuestro caso la cámara Pro 9000 de Logitech utiliza la tecnología CMOS. Un resumen de sus características técnicas más importantes son:

- Resolución máxima de la imagen: 8 megapíxeles.

- Velocidad máxima de adquisición de imágenes: 30 fps (”frames” por segundo).
- Cuenta con un ”Balance de blanco Automático”.
- Cuenta con un Micrófono Incorporado.
- La Conexión se realiza a través del puerto serie USB.
- Característica Adicional: Adaptador de vídeo en color de 16 bits.

Podemos destacar que la Pro 9000 de Logitech es un producto que tiene como punto fuerte la calidad en relación al precio, es decir, que su calidad es relativamente alta en proporción a su precio. Esta cámara cuenta con componentes ópticos de Carl Zeiss que son de gran calidad.

En cuanto al diseño de la cámara podemos decir que es fácilmente adaptable al ordenador, ya que cuenta con un soporte antideslizante y ajustable, aunque no fijo, por lo que debemos tener cuidado.

Sus dimensiones son de 10 cm de ancho, 4cm de alto y 3 cm de profundidad aproximadamente (sin tener en cuenta el soporte de la misma). Esta cámara nos permite un rotación vertical de unos 180^{o} sobre su eje, lo que le otorga cierta movilidad y flexibilidad a la hora de extraer imágenes. La WebCam Pro 9000 de Logitech cuenta con una zona redondeada la cual se enciende en color rojo para saber que cuentas con conexión entre la cámara y el PC.

En la figura 3.2 se muestra una imagen del dispositivo.

Figura 3.2: Webcam Pro 9000 de Logitech

El formato de las imágenes extraídas desde la Webcam Pro 9000 de Logitech va desde los 320x240 hasta los 3264x2448 píxeles (8 megapíxeles) para el formato 4:3 y desde los 320x180 hasta 1280x720 píxeles en formato 16:9. Es de destacar que en formato 4:3 la calidad de la imagen es notablemente de bastante más calidad.

En cuanto a la calidad de las imágenes para la grabación de video está muy por debajo, como es lógico, de la calidad de las imágenes obtenidas de las fotos. En concreto la calidad baja de 8 megapíxeles que teníamos en las fotos a 2 megapíxeles para los vídeos, de tal forma que la resolución para el formato 4:3 pasaría a estar en 1600x1200 píxeles como máximo y en 1280x720 píxeles para el formato 16:9.

En cuanto al micrófono que lleva incorporado, además de las funcionalidades propias de un micrófono asociado a una Webcam, cabe destacar que gracias a la tecnología que lleva integrada, que se llama RightSound y que pertenece a Logitech, se consigue evitar problemas asociados al "eco" de la sala, ya que además lleva asociado un ecualizador automático, y también elimina o atenúa los ruidos procedentes del acoplo de señales al micrófono.

Capítulo 4

Sistema propuesto

En este capítulo nos centraremos en explicar las funcionalidades del sistema y los entresijos de cada uno de los ”módulos” de que se compone la aplicación.

Nuestro sistema cuenta con varios algoritmos, los cuales se basan en distintas técnicas de análisis de imágenes, para conseguir realizar un programa capaz de detectar el grado de felicidad de una persona según la morfología de su boca (expresión bucal).

Este sistema es capaz de detectar hasta cuatro estados de ánimo, siendo estos: Normal, Feliz, Muy Feliz y Totalmente Feliz.

4.1. Adquisición de datos

La adquisición de datos se ha sido realizada por el dispositivo Webcam Pro 9000 de Logitech que ya hemos explicado en el apartado de plataforma hardware. Para realizar la adquisición con éxito hemos tenido que realizar una programación de tal forma que eligiera esta cámara para realizar las capturas de imágenes. Esto se debe a que el portátil con el que contábamos para realizar esta aplicación ya disponía de una Webcam incorporada y por defecto todo programa que requiriera de toma de imágenes accedía directamente a esta cámara y no a la Pro 9000 que habíamos enchufado vía USB. Y como ya hemos explicado, la calidad de imagen es muy superior de esta forma.

4.2. Detección de Rostros

Para conseguir nuestro objetivo final, primeramente hemos realizado un análisis global del entorno de la imagen y hemos decidido emplear un método que vaya disminuyendo la región de búsqueda para así ir especificando cada vez más la zona y de esta forma mejorar significativamente la efectividad, eficiencia, calidad y rapidez de los clasificadores.

Para ello hemos utilizado un clasificador en Cascada de Haar que introduciéndole los archivos .htm adecuados a lo que queramos buscar, realiza el entrenamiento, el testeado y la aplicación del clasificador de manera muy sencilla.

Introduciendo dos archivos o conjuntos diferentes de rostros (Uno de rostros frontales y otro de rostros frontales con gafas), con 3500 datos ”positivos” cada uno y contando con un total, entre los dos, de 7000 datos ”negativos” conseguimos unos resultados realmente buenos.

Este subsistema inicial es capaz de realizar una detección múltiple para un entorno arbitrario de la imagen, habiéndose probado con hasta 6 caras a la vez.

4.3. Segmentación de Rostros

Una vez detectados los rostros de la imagen hemos realizado la segmentación de este con un recorte ROI (Region of Interest), en principio recorta el mínimo cuadrado, es decir, el cuadrado con mínima área que sea capaz de inscribir todo el rostro, y así con todos los rostros. Además se señaliza en la imagen origen, con un rectángulo de color Rosa oscuro, el rectángulo donde se encuentra la zona reconocida como rostro, que es la misma que hemos recortado como ROI.

Una vez hecho esto se realiza una operación sobre la imagen ROI del rostro de tal forma que obtenemos otra ROI que contiene sólo la información relativa al tercio inferior de la ROI del rostro completo (En realidad la primera imagen ROI, que es la que contiene a todo el rostro, no se llega a generar, sólo se cogen los parámetros que servirían para generarla y se le realizan las operaciones pertinentes). Realizando esta simple operación evitamos muchos errores de reconocimiento en la fase posterior.

4.4. Detección de bocas

Tomando las imágenes transformadas de los rostros, que contienen sólo la información relativa al tercio inferior de la cara, realizamos un análisis específico sobre ellas. Esto consiste en utilizar un nuevo clasificador, que tiene el mismo funcionamiento que el clasificador en cascada de Haar que utilizamos en un principio para reconocer rostros, con la diferencia de que los conjuntos de datos que le introducimos como archivos .htm son ahora de bocas.

Debemos decir que en este caso también introducimos dos conjuntos de datos diferenciados, uno con 3500 datos ”positivos” de bocas con una morfología correspondiente a un estado de ánimo neutro, y otro con 3500 datos de bocas sonrientes pero en distintos grados (unas bocas sonríen más que otras), de manera arbitraria (con la restricción de que sonrían), haciendo un total, como en el caso anterior, de 7000 datos ”negativos”, es decir, casos en los cuales, no hay boca alguna.

4.5. Segmentación de bocas

Con todo lo anterior, somos capaces de detectar sólo dos estados de ánimo de manera estable, que serían: Un estado de ánimo neutro y un estado de ánimo alegre.

Por ello y con el fin de conseguir detectar un total de 4 estados de ánimo para múltiples usuarios al mismo tiempo, se ha optado por segmentar la boca, en el caso de que esta se encuentre en un estado de ánimo alegre (da igual el grado de alegría). En caso de que se detecte como estado de ánimo neutro explicaremos más adelante como se trataría ese caso.

La operación que se realiza es la misma que realizábamos con los rostros en la primera etapa, es decir, creamos una ROI cuya área sea la mínima posible y que inscriba el total de los píxeles pertenecientes al conjunto detectado como boca (entendiendo que se realiza para cada boca por separado y de manera independiente).

4.6. Tratamiento de las imágenes ROI de las bocas

Tomando las imágenes ROI de las bocas, y con el fin de facilitar lo máximo posible un posterior análisis y reconocimiento de 3 estados diferentes a partir de estas imágenes. Se ha decidido realizar una serie de transformaciones sobre esta imagen.

Lo que pretendemos con estas transformaciones es encontrar y segmentar fácilmente el color blanco, característico de los dientes.

La primera transformación que realizamos es una ecualización del histograma de la imagen de tal forma que el blanco ahora resaltará más sobre el resto. Después pasamos la imagen del espacio de color RGB al espacio de color HSV, de esta forma cualquier operación que realicemos se verá menos influenciada con respecto a la luminosidad del entorno.

Una vez que pasamos la imagen a espacio de color HSV, nos centramos en las variables S (Saturation) y V (Value) ya que para valores cercanos al blanco la variable H se vuelve inesta-

ble. Umbralizamos ambas variables para binarizar la imagen.

Por otro lado cogemos la imagen ROI ecualizada y la pasamos de RGB a escala de grises de tal forma que sea fácil segmentar el color blanco. Umbralizamos los niveles de gris, de tal forma que obtenemos una nueva imagen binarizada.

Empleando una operación AND para las 2 imágenes binarizadas, se consigue una imagen binarizada bastante restrictiva, que consigue unos resultados relativamente buenos.

Aplicamos una transformación más para eliminar el error procedente de la iluminación lateral del rostro. Esta transformación consiste en tomar una línea central de la ROI binarizada final, cuya altura es la misma que la de la imagen ROI y cuya anchura es de 10 píxeles, de esta forma es como si nos centraremos en 4 ”keypoints” o puntos de interés de la boca.

4.7. Reconocimiento del estado de ánimo

Como ya hemos dicho con el clasificador en cascada de Haar no existía ningún problema a la hora de reconocer el estado de ánimo neutro, más bien el problema nos lo encontrábamos al intentar discernir entre los distintos estados de alegría. Por lo que la clasificación del estado neutro es inmediata.

Ahora bien para el reconocimiento de los otros 3 estados que deseamos reconocer nos basaremos en la línea central binarizada que hemos segmentado antes. Para ello haremos uso del histograma como herramienta.

Primero calcularemos el histograma de la imagen de la línea central binarizada, con una función auxiliar, de tal forma que sólo obtendremos 2 niveles de gris, en el 0 (negro) y en el 255 (blanco), lo cual es lógico, ya que es una imagen binarizada. Hecho esto realizaremos un recuento de los píxeles que son blancos de tal forma que según el número que exista de ellos, fijaremos las fronteras entre los distintos estados de ánimo.

4.8. Simbolización del estado de ánimo

Una vez hemos establecido las fronteras entre un estado de ánimo y otro (Feliz, Muy Feliz y Totalmente Feliz) y sabiendo que teníamos ya asegurada la detección de un estado de ánimo neutro (Normal). Simbolizaremos estos cuatro estados con 4 emoticonos, que expresen el estado de ánimo en sí.

Este emoticono se posicionará en la frente del usuario al cual le hemos realizado el estudio de su estado de ánimo. Para ello utilizaremos herramientas anteriores, en concreto la ROI de los rostros la cual tiene información muy interesante para poder posicionar el emoticono, de tal forma que, hemos decidido posicionar el emoticono a $1/2$ en "X" y a $1/8$ en "Y", teniendo en cuenta que el eje de coordenadas es la esquina superior izquierda de la ROI y que el eje Y es creciente hacia abajo.

Además con el fin de obtener unos resultados más profesionales, hemos decidido que el tamaño del emoticono vaya en función al tamaño del rostro del usuario en cuestión, de tal forma que su tamaño comprende $1/6$ del tamaño del rostro.

En la figura 4.1 se muestra lo anteriormente descrito.

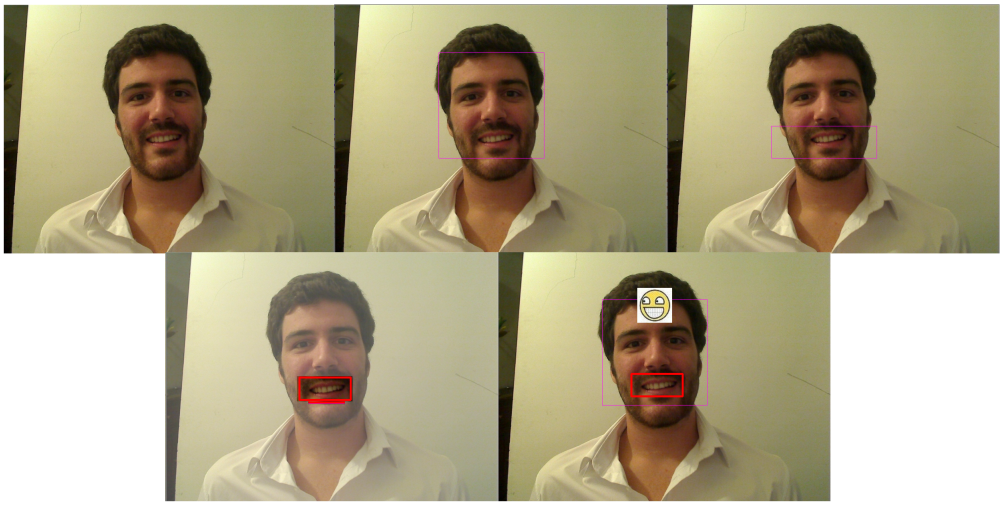
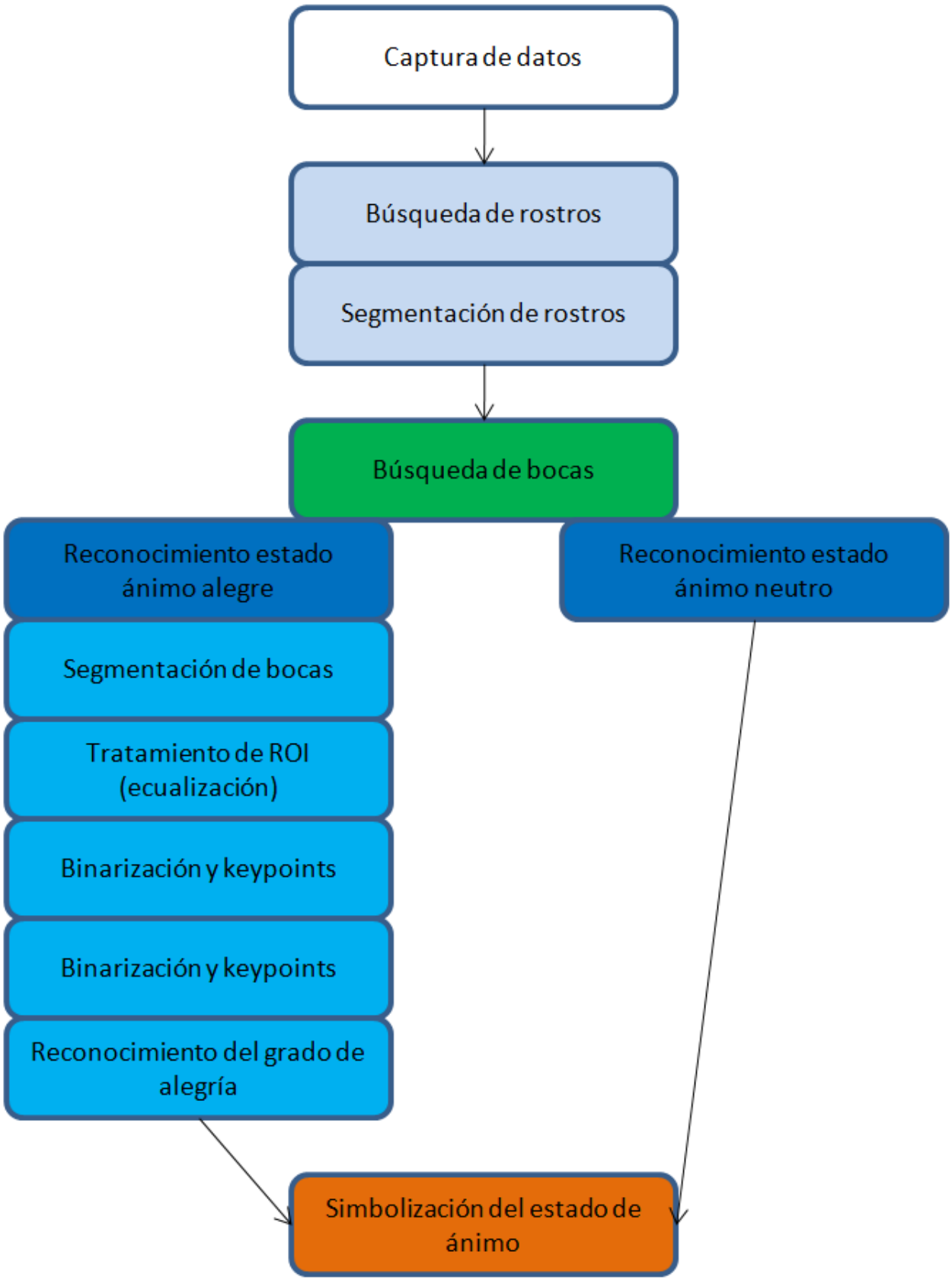


Figura 4.1: Figura ilustrativa del funcionamiento del sistema


Captura de datos
Búsqueda de rostros
Segmentación de rostros
Búsqueda de bocas
Reconocimiento estado ánimo alegre
Reconocimiento estado ánimo neutro
Segmentación de bocas
Tratamiento de ROI (ecualización)
Binarización y keypoints
Binarización y keypoints
Reconocimiento del grado de alegría
Simbolización del estado de ánimo

Capítulo 5

Experimentos

En este capítulo explicaremos los diferentes experimentos que hemos realizado para la consecución del proyecto.

Explicaremos las ventajas y desventajas de los diferentes algoritmos en la práctica. Centrándonos en los problemas que hemos tenido y la manera por la cual los hemos resuelto.

También comentaremos el grado de optimización y carga de proceso que lleva consigo la aplicación.

Todo ello lo ilustraremos con imágenes y gráficas que amenicen la explicación de los algoritmos a la vez que simplifiquen su entendimiento.

5.1. Experimento 1. Programación para configuración del hardware

Estableciendo como base OpenCV surgió un problema de reconocimiento de la cámara Pro 9000 de Logitec debido a que el ordenador en el cual íbamos a programar ya tenía una webcam integrada.

El problema principal era que la webcam integrada tenía una calidad de imagen muy inferior a la cámara Pro 9000 de Logitec y esto hacía que cualquier programación posterior se viera afectada.

Además del problema de la calidad de imagen, teníamos uno añadido y era la velocidad a la cual procesaba las imágenes, que distaba mucho del tiempo real.

5.2. Experimento 2. Búsqueda y reconocimiento global de bocas

En esta parte se intentó realizar una búsqueda y reconocimiento sólo de las bocas de los usuarios en el entorno global. Por medio de Cascadas Haar se probaron varias alternativas, pero los resultados dejaban mucho que desear. El problema no procedía de la falta de reconocimientos positivos de bocas, sino de la inmensa cantidad de datos erróneos localizados como positivos, es decir, falsos positivos. En la imagen 5.1 se muestra el error producido con este sistema de búsqueda en estas condiciones.

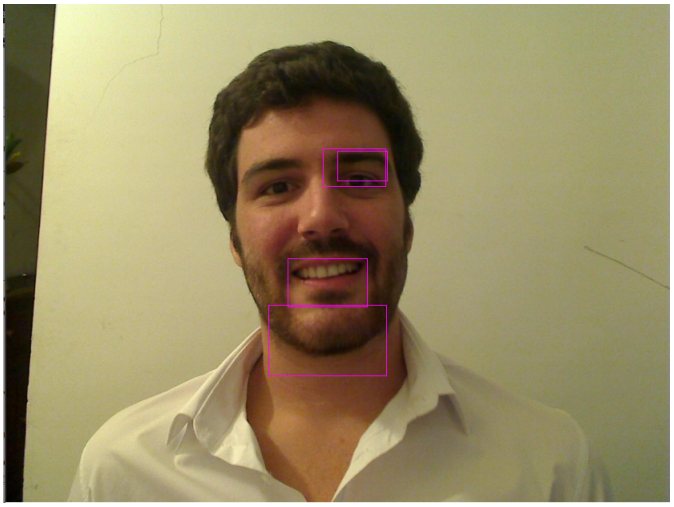


Figura 5.1: Error búsqueda de bocas global

Debido a este hecho, y observando que los falsos positivos no sucedían con frecuencia en el área del propio rostro, decidimos realizar una búsqueda específica.

5.3. Experimento 3. Búsqueda y reconocimiento facial

Para realizar una búsqueda de bocas específica, es decir en un entorno limitado o restringido según nuestras restricciones, decidimos realizar una búsqueda global de rostros primero, observando que esta búsqueda de rostros, tiene una tasa de falsos positivos muy baja.

Para reconocer los rostros se pueden utilizar muchas técnicas pero en nuestro caso hemos utilizado las Cascadas de Haar que es una técnica muy utilizada en este campo. Como diseñar nuestro propio archivo para las Cascadas de Haar llevaría mucho tiempo ya que para que funcione adecuadamente se necesita entrenar el sistema con 3000 casos positivos y 3000 casos negativos, hemos decidido utilizar archivos ya existentes.

Estos archivos constan de 7000 ejemplos cada uno, y en nuestro caso hemos utilizado 2 uno para rostros frontales normales y otro para rostros frontales con gafas. Dando los siguientes resultados.

A continuación se muestra en la imagen 5.2 los resultados de este experimento.

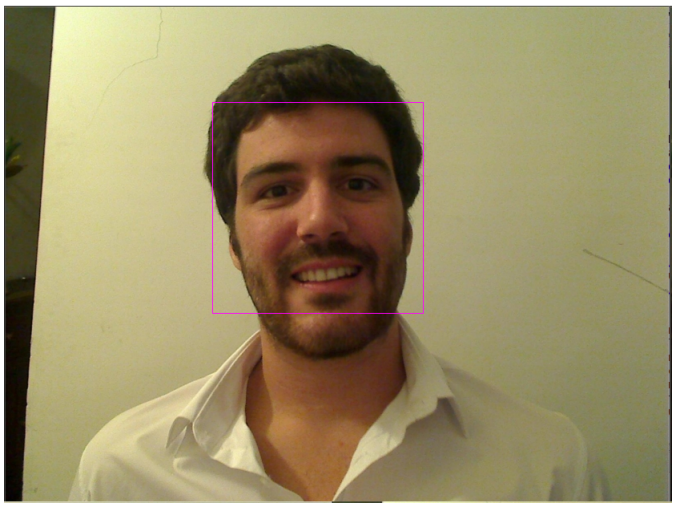


Figura 5.2: Reconocimiento facial con cascadas de Haar

5.4. Experimento 4. Limitación del área de búsqueda

Observamos que aún restringiendo el área de búsqueda del algoritmo de reconocimiento de bocas, se ocasionaban en algunas ocasiones especiales de iluminación no óptima o posición girada de la cara, algunos falsos positivos en la parte de los ojos y de la nariz.

Teniendo lo anterior en cuenta con el fin de optimizar nuestro sistema lo más posible, restringimos aún más el área de búsqueda obteniéndose una ROI (Region of Interest) del tercio inferior de los rostros. Este hecho no sólo evita gran número de falsos positivos si no que al reducir el área de búsqueda, disminuye el tiempo de cómputo del algoritmo de búsqueda.

5.5. Experimento 5. Búsqueda y reconocimiento específico de la boca

Cuando realizamos la búsqueda en la imagen ROI (Region of Interest) extraída de la región correspondiente al tercio inferior de la parte reconocida como rostro, los resultados mejoran

considerablemente como se muestra en la figura 5.3.

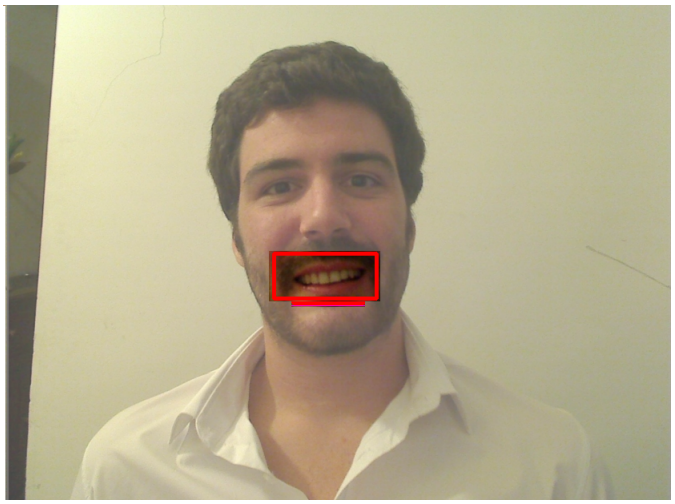


Figura 5.3: Reconocimiento local de la boca

Este método anula los errores que se producían en la nariz y los ojos, que eran reconocidos en ocasiones como bocas.

5.6. Experimento 6. ROI de la boca

Una vez reconocida la región de la boca se recorta esta zona exclusivamente para realizarle una serie de transformaciones posteriores.

Con ánimo de realizar el posterior reconocimiento del Grado de Felicidad del usuario se realiza una operación sobre el ROI de la boca, y esto es extraer sólo la parte central de esta localización tomando sólo los píxeles que se encuentren en una región comprendida entre los 5 píxeles anteriores al centro y los 5 píxeles posteriores al centro, teniendo la misma altura que el ROI de la boca.

Con todo esto evitamos errores derivados de la iluminación variable entre las dos mitades de la cara, para su posterior binarización.

En la figura 5.4(a) se muestra la zona del ROI de la boca y el trozo central en la figura 5.4(b).

(a) ROI de la boca (b) Segmentación del trozo central de la boca

Figura 5.4: Preparación para posterior análisis del estado de felicidad a partir de la apertura y posición de la boca

5.7. Experimento 7. Binarización por color

Las pruebas realizadas en este aspecto dieron resultados buenos en un inicio pero mucho mejores con el ROI y segmentación del centro del ROI de la boca.

En un inicio se realizó sólo la segmentación del blanco con la imagen transformada a escala de grises, pero los resultados mejoraron al realizarlo en paralelo en HSV y unir con una función ambos resultados, ya que de esta forma somos más restrictivos.

En la figura 5.5 se muestra la binarización del color.

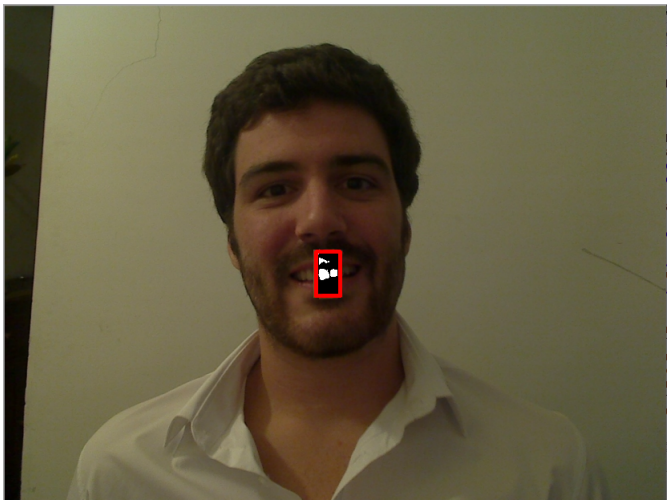


Figura 5.5: Binarización zona central de la boca

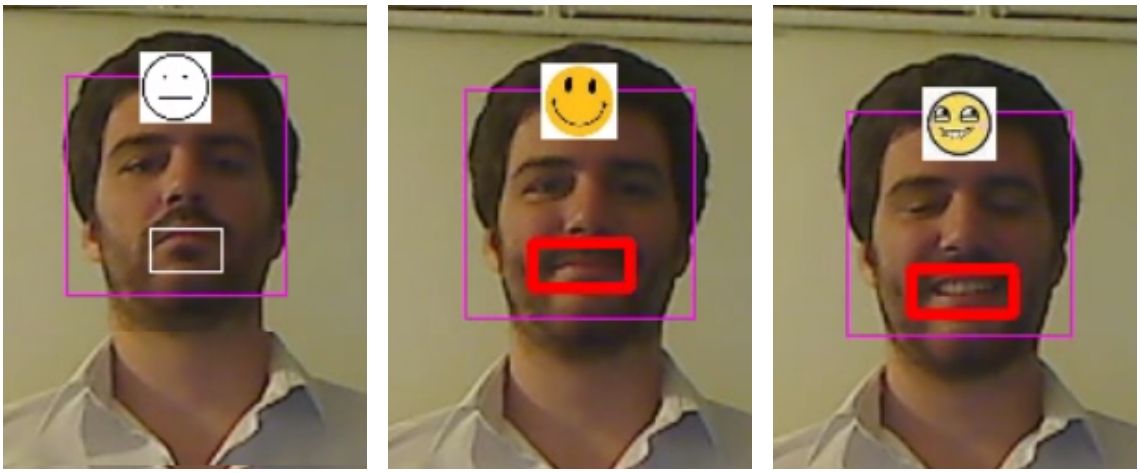
5.8. Experimento 8. Reconocimiento Grado de felicidad

En cuanto al reconocimiento del Grado de felicidad, se realiza por cuantificación del número de píxeles blancos de la imagen segmentada del centro de la imagen ROI de la boca, dando unos resultados relativamente buenos.

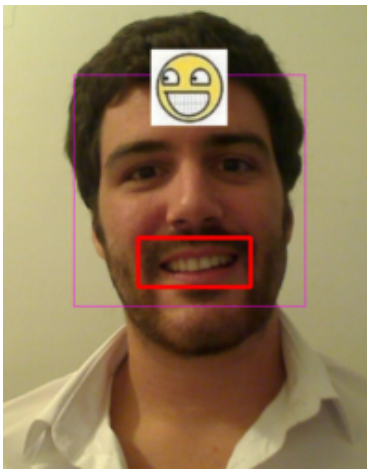
Este sistema es capaz de reconocer hasta 4 estados que se simbolizarán posteriormente.

5.9. Experimento 9. Simbolización del grado de Felicidad

Para que el usuario de la aplicación observe los 4 estados de una forma amena y fácil se ha optado por simbolizar el estado con emoticonos que se colocan en la frente dando los siguientes resultados que se muestran en la figura 5.9.



(a) Estado ″Normal″ (b) Estado ″Feliz″ (c) Estado ″Muy Feliz″



(d) Estado ″Totalmente Feliz″

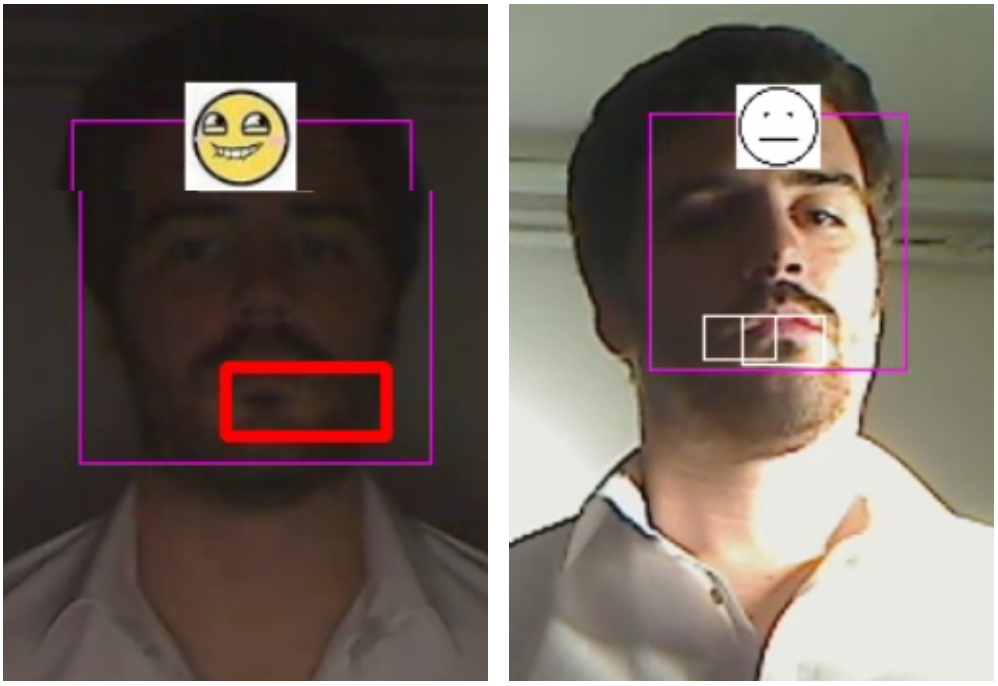
Figura 5.6: Simbolización con emoticonos del estado de felicidad

.

5.10. Experimento 10. Resultados vs Iluminación

Debido a las limitaciones que poseen todos los dispositivos ópticos (que ya hemos introducido anteriormente) que se basan en la captura de la luz visible para obtener información del entorno, una baja iluminación puede hacer que todos los algoritmos fallen, y por muy complejo que sea el programa siempre existirá esta limitación debido a la tecnología con la que trabajamos. Esto se debe a que en este caso la iluminación es sinónimo de información, pero cuidado porque una iluminación excesiva también conlleva pérdida de datos (aunque no por *blooming* en este caso, ya que trabajamos con tecnología CMOS).

En las siguientes imágenes se muestran los errores acontecidos cuando la iluminación no es suficiente (figura 5.7(a)) o es excesiva (figura 5.7(b)) y la pérdida de datos es significativa.



(a) Error producido por falta de iluminación (b) Error producido por exceso de iluminación

Figura 5.7: Errores derivados de la iluminación del entorno

5.11. Experimento 11. Resultados vs Tiempos

Ahora veremos la carga computacional que tienen cada uno de los algoritmos que hemos utilizado, y explicaremos la razón por la cual cada uno tiene esa carga de proceso determinada.

El primer algoritmo que se realiza en la aplicación es el de búsqueda y detección de rostros en el entorno global de la imagen de entrada. La carga computacional de este algoritmo se muestra en la figura 5.8.

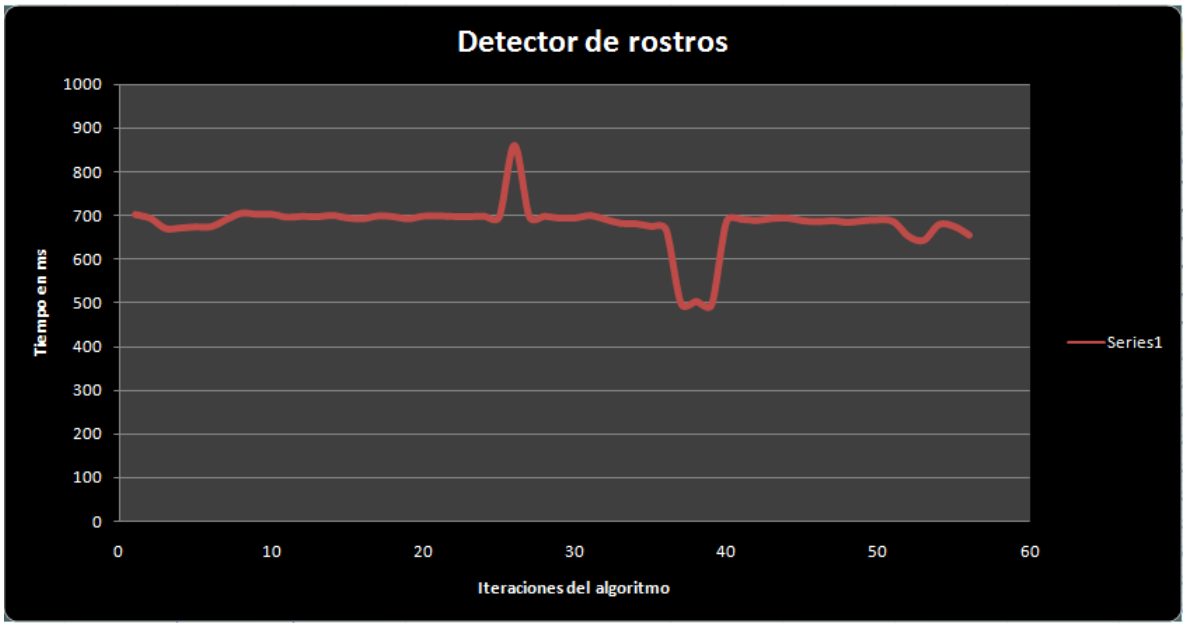


Figura 5.8: Carga computacional del algoritmo de búsqueda y reconocimiento facial, llevada a cabo en el entorno global de la imagen de entrada, para varios entornos, iluminación, número de rostros y movimiento de la Webcam Pro 9000 de Logitech.

Como se observa en la figura 5.8 la media del tiempo que tarda en completar la búsqueda y reconocimiento además de la segmentación de los rostros, se encuentra en torno a los 700 ms observándose que existe una desviación de 150 ms por encima y por debajo de este valor para algunos casos en los que facilitamos o entorpecemos, según las condiciones del entorno, el desarrollo del proceso.

Nos encontramos con que cuando aumentamos el número de usuarios a los que tenemos que reconocer y segmentar el rostro, el tiempo que tarda el algoritmo, no es ni mucho menos el doble, sino que sólo sube entre 100 ms y 200 ms el tiempo. Por lo que podemos deducir que la mayor parte del tiempo no se debe al reconocimiento en sí, de los rostros.

Sin embargo nos encontramos con que cuando la iluminación es desfavorable (por exceso o por defecto), el tiempo que tarda el proceso es indeterminado, ya que es probable, que en condiciones en las que las características que definen el rostro son inestables, no encuentre el rostro ”nunca”.

Pero no podemos atribuir la carga del proceso a la iluminación, al menos en su totalidad, ya que para una misma iluminación la carga de proceso varía y no sólo por cambiar el número de usuarios, si no por condiciones pseudoaleatorias (determinadas pero en principio muy complejas de predecir, y a efectos de simplificar el sistema los consideramos fuera de nuestro alcance y por tanto aleatorias).

Cambiando de algoritmo, el segundo algoritmo que realizamos es el de búsqueda de bocas, diferenciando entre sonrientes y neutras. A continuación en la imagen 5.9 se muestra la carga computacional de este algoritmo.

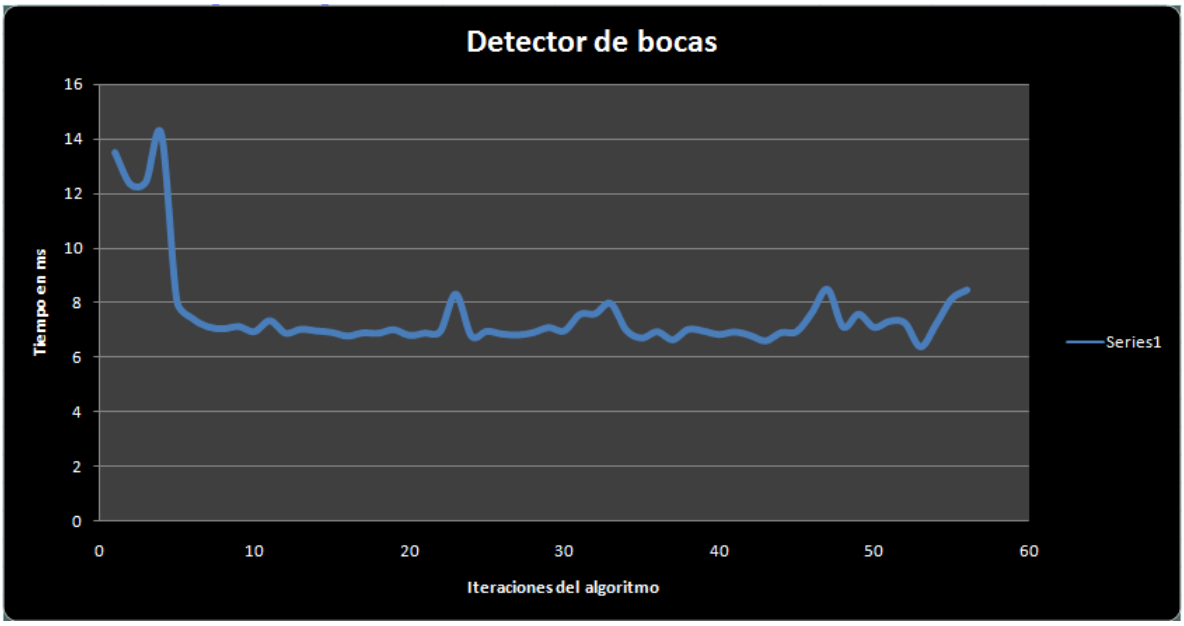


Figura 5.9: Carga computacional del algoritmo de búsqueda y reconocimiento de bocas, llevada a cabo en el entorno específico de la imagen segmentada del rostro (ROI) tomada para varios entornos, iluminación, número de rostros y movimiento de la Webcam Pro 9000 de Logitech.

En este algoritmo observamos que la carga de proceso es muy inferior a la del algoritmo anterior, ya que la carga computacional del algoritmo anterior era del orden de 100 ms y ahora manejamos unas cifras del orden de casi los 10 ms, es decir 10 veces inferior.

Si evaluamos los motivos de este suceso observamos que no se debe al número de usuarios ya que el incremento de usuarios produce un incremento de la carga computacional en ambos algoritmos. La complejidad morfológica podría ser un motivo lógico, ya que la cara tiene una forma más compleja que la boca (ya que a la complejidad de la boca se le suma la complejidad de las otras partes de la cara, por lo que siempre va a ser más compleja o al menos con más características morfológicas la cara que la boca), pero esto no parece ser una razón suficiente para explicar el gran abismo que separa a la carga computacional del algoritmo de la cara y de

la boca.

El tercero de los algoritmos cuya carga computacional es relevante para el estudio, es el algoritmo del reconocimiento del grado de sonrisa. En la imagen 5.10 se muestra la carga computacional de este proceso.

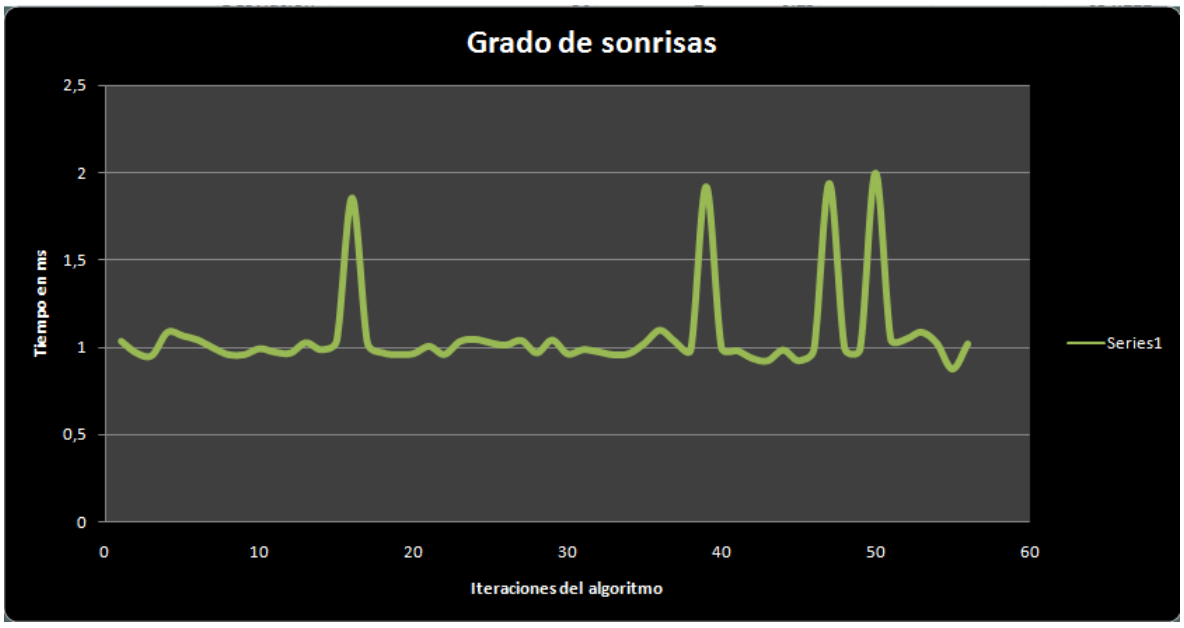


Figura 5.10: Carga computacional del algoritmo de reconocimiento del grado de felicidad de la boca, llevada a cabo en el entorno específico de la imagen segmentada de la boca (ROI) tomada para varios entornos, iluminación, número de rostros y movimiento de la Webcam Pro 9000 de Logitech.

En este caso podemos observar que nos encontramos en el orden de 1 ms, es decir, nuevamente se ha reducido la carga computacional unas 10 veces del algoritmo que está por encima.

Analizando la proporcionalidad de los órdenes de magnitud temporales que manejamos en los 3 algoritmos, nos encontramos con que el tiempo de cómputo es directamente proporcional al tamaño, o mejor dicho, al área de la imagen en la cual realizamos la búsqueda. Entrando en detalle, el primer algoritmo realizaba una búsqueda global en toda la imagen por lo que el tamaño de la imagen es relativamente grande. En el segundo algoritmo sólo buscamos en el

tercio inferior del rostro, es decir el área de búsqueda se ha reducido en un orden de magnitud, aproximadamente, ya que depende de la distancia del usuario a la cámara (escala). Y en el tercer algoritmo, que es el de reconocimiento del grado de felicidad de la boca, sólo trabajamos con una fraja central de la boca que es un orden de magnitud inferior en tamaño que la boca.

Si mostramos los resultados de la carga computacional en porcentaje de la carga computacional del proceso completo, quedaría algo como lo de la figura 5.11.

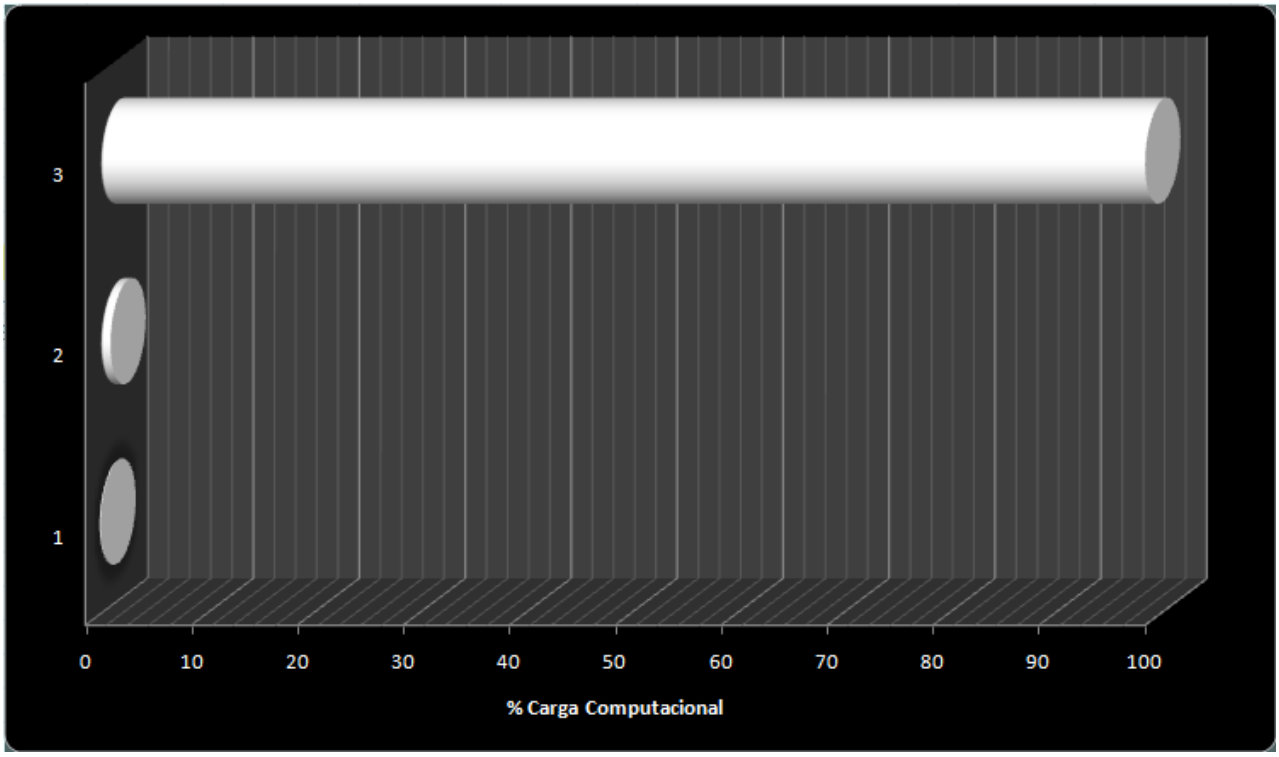


Figura 5.11: Carga computacional de cada uno de los algoritmos de la aplicación, llevada a cabo en el entorno específico de la imagen segmentada de la boca (ROI) tomada para varios entornos, iluminación, número de rostros y movimiento de la Webcam Pro 9000 de Logitech.(1) Grado de felicidad; (2) Búsqueda y reconocimiento de bocas; (3) Búsqueda y reconocimiento de rostros.

Como se puede observar se realiza una búsqueda progresiva piramidal, cada vez más restrictiva, específica y rápida. En la figura 5.12 se ilustra las zonas segmentadas por cada algoritmo y como se sigue una estructura ”piramidal”.

Figura 5.12: Regiones de búsqueda y segmentación de cada algoritmo de manera progresiva y piramidal.

Capítulo 6

Conclusiones

Nuestro sistema cuenta con varios algoritmos, los cuales se basan en distintas técnicas de análisis de imágenes, para conseguir realizar un programa capaz de detectar el grado de felicidad de una persona según la morfología de su boca (expresión bucal).

Este sistema es capaz de detectar hasta cuatro estados de ánimo, siendo estos: Normal, Feliz, Muy Feliz y Totalmente Feliz.

Para conseguir nuestro objetivo primeramente hemos realizado un análisis global del entorno de la imagen mediante cascadas de Haar para la detección de rostros, con un archivo con 7000 datos referentes a esto.

Una vez detectados los rostros de la imagen hemos realizado una transformación de esta, eliminando todos los datos que se encuentren por encima del tercio inferior de la imagen de esta forma evitamos muchos errores para la fase posterior.

Tomando las imágenes transformadas de los rostros realizamos un análisis específico sobre ellas buscando las bocas en esta porción de los rostros. Para detectar las bocas hemos utilizado cascadas de Haar con 7000 datos de bocas para las bocas en estado Normal, y de otros 7000 datos para la boca sonriendo.

Si realizamos la detección de boca Normal nos situaría un emoticono en la frente de la per-

sona cuya boca ha sido detectada como tal, que ilustraría su estado normal o falto de felicidad y volvería al principio del sistema.

En el caso de realizar la detección de "bocas felices" pasamos a una fase de ecualización de la imagen y procesado en paralelo para obtener los mejores resultados posibles. Consistiendo esta fase en ecualizar la imagen para reducir la influencia de la iluminación sobre los resultados posteriores y realizar 2 tareas simultáneas: Transformar la imagen a espacio de color HSV y por otro lado a Escala de Grises. Binarizamos la imagen HSV en sus parámetros S (Saturation) y V (Value) con umbrales calculados por prueba y error o método Heurístico, por otro lado, binarizamos también la imagen en Escala de Grises y unificamos ambos resultados con una función AND, que lo que hace es que si un pixel es blanco en las dos imágenes, en la nueva imagen también lo será y para cualquier otra combinación en la imagen resultante será negro, de esta forma somos muy restrictivos a la hora de decidir que un pixel sea blanco.

Más tarde realizamos una nueva transformación que es simple en esencia pero imprescindible en el sistema. Lo que hacemos es tomar los píxeles centrales de la imagen binarizada resultante del proceso anterior, tomando los píxeles que se encuentren a una distancia en X de +-5 píxeles de la posición central y de altura, la altura del cuadrado que señala la detección de la boca.

Tomando la imagen anterior realizamos el cálculo del histograma, al ser una imagen binaria sólo tendrá dos valores 0 y 255 que contienen un número de píxeles determinado, de esta forma contando el número de píxeles blancos decidiremos el grado de felicidad. Una vez decidido el grado de felicidad se situará el emoticono correspondiente en la frente del usuario detectado.

Capítulo 7

Trabajos Futuros

Algunos de los trabajos futuros que podríamos añadir a este proyecto para aumentar las aplicaciones y utilidades del mismo del mismo, podrían ser los siguientes:

- Aumentar el número de archivos, funciones y algoritmos para poder reconocer más zonas de la cara junto con más facciones, de tal forma que seamos capaces de reconocer más estados de ánimo, siendo consecuentes con una optimización de la carga del proceso de la nueva aplicación.

- Como una funcionalidad añadida, podríamos, aprovechando que la Webcam Pro 9000 de Logitech tiene un micrófono incorporado, reconocer el estado de ánimo o incluso la identidad de un determinado usuario por el tono de la voz.

- Incluir una etapa de filtrado que simplifique de alguna forma la búsqueda y reconocimiento de rostros, para así disminuir la gran carga computacional de este algoritmo que es el más costoso de todos.

- Escaneo de la cara en 3 dimensiones con la nueva tecnología de los sensores como el de Microsoft, que es el sensor Kinect para Xbox, de esta forma podríamos centrarnos en la morfología de la cara y, al tener información de profundidad, podríamos realizar un

estudio mucho más profundo y exacto sobre los estados de ánimo y reconocimiento en general.

Figura 7.1: Mapa de profundidad obtenido del sensor Kinect de Microsoft.

Bibliografía

[1] Phillip Ian Wilson, Dr. John Fernandez. *Facial feature detection using Haar Classifiers*. Texas AM University, 2006.

[2] Antonio Rama, Francesc Tarrés. *Un nuevo método para la detección de caras basado en Integrales Difusas*. TDept. Teoria del Senyal i Comunicacions - Universitat Politècnica de Catalunya, Barcelona, Spain.

[3] Bradski, G. and Kaehler, A. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, 2008.

[4] Bishop, C.M. *Pattern Recognition And Machine Learning*. Springer, 2006.

[5] Szeliski, R. *Computer vision: Algorithms and applications*. Springer-Verlag New York Inc, 2010.

[6] Duda, R.O. and Hart, P.E. and Stork, D.G. Pattern classification. *Pattern Classification and Scene Analysis: Pattern Classification*, Wiley, 2001.

[7] Ranga Rodrigo *Image Processing and Computer Vision.*