MÁSTER UNIVERSITARIO DE INVESTIGACIÓN EN INGENIERÍA DE SOFTWARE Y SISTEMAS INFORMÁTICOS

TRABAJO FINAL DE MÁSTER

DESARROLLO Y EVALUACIÓN DE TÉCNICAS DE VISIÓN ARTIFICIAL APLICADAS A UN PROBLEMA DE RECONOCIMIENTO FACIAL

31105151 – PERCEPCIÓN VISUAL

AUTOR: José Luis Carvajal

DIRECTOR: Carlos Cerrada Somolinos

Curso 2018 – 2019 – Convocatoria Septiembre

MÁSTER UNIVERSITARIO DE INVESTIGACIÓN EN INGENIERÍA DE SOFTWARE Y SISTEMAS INFORMÁTICOS

TRABAJO FINAL DE MÁSTER

DESARROLLO Y EVALUACIÓN DE TÉCNICAS DE VISIÓN ARTIFICIAL APLICADAS A UN PROBLEMA DE RECONOCIMIENTO FACIAL

31105151 – PERCEPCIÓN VISUAL

AUTOR: José Luis Carvajal

DIRECTOR: Carlos Cerrada Somolinos

DECLARACIÓN JURADA DE AUTORÍA DEL TRABAJO CIENTÍFICO, PARA LA DEFENSA DEL TRABAJO FIN DE MASTER

Fecha: 04/09/2019

Quién suscribe:

Autor: José Luis Carvajal
D.N.I: 0802476374

Hace constar que es el autor del trabajo:

Título completo del trabajo.

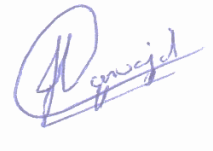
DESARROLLO Y EVALUACIÓN DE TÉCNICAS DE VISIÓN ARTIFICIAL APLICADAS A UN PROBLEMA DE RECONOCIMIENTO FACIAL

En tal sentido, manifiesto la originalidad de la conceptualización del trabajo, interpretación de datos y la elaboración de las conclusiones, dejando establecido que aquellos aportes intelectuales de otros autores, se han referenciado debidamente en el texto de dicho trabajo.

DECLARACIÓN:

- ✓ Garantizo que el trabajo que remito es un documento original y no ha sido publicado, total ni parcialmente por otros autores, en soporte papel ni en formato digital.
- ✓ Certifico que he contribuido directamente al contenido intelectual de este manuscrito, a la génesis y análisis de sus datos, por lo cual estoy en condiciones de hacerme públicamente responsable de él.
- ✓ No he incurrido en fraude científico, plagio o vicios de autoría; en caso contrario, aceptaré las medidas disciplinarias sancionadoras que correspondan.

Fdo.



Impreso TFdM05_AutorPbl. Autorización de publicación y difusión del TFM para fines académicos

Autorización

Autorizo/amos a la Universidad Nacional de Educación a Distancia a difundir y utilizar, con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firma del/los Autor/es

RESUMEN

El reconocimiento de rostros ha sido un tema de auge en los últimos años [1], su importancia y relevancia ha sido enfocada principalmente en medidas de seguridad para los sistemas informáticos. Este tipo de sistemas ha permitido una gama amplia de aplicaciones en ingeniería, en salud, economía, entre otras que han requerido la autenticación de usuarios o el reconocimiento de los mismos para diferentes fines [2] [3]. En el proceso de reconocimiento de rostro ha sido posible la implementación de numerosas técnicas computacionales que han servido para realizar procesos eficaces específicos [4] [5]. Una de las herramientas más comúnmente utilizada para estos fines han sido las de computación inteligente, que comprende un número amplio de herramientas como las redes bayesianas, los mapas auto organizativos, aproximaciones probabilísticas, entre otras.

Palabras claves: reconocimiento de rostro, visión por computador, algoritmos.

ÍNDICE

Lista de ilustraciones

Lista de tablas

CAPÍTULO 1

INTRODUCCIÓN Y JUSTIFICACIÓN

1.1. Introducción y justificación

Una de las ramas de la Inteligencia Artificial que ha experimentado un mayor crecimiento en estos últimos años es la Visión por Computador. Esta es la disciplina que estudia cómo procesar, analizar e interpretar imágenes de forma automática. Estas técnicas tienen aplicaciones en muchos ámbitos, como la seguridad, la medicina, la navegación automática o los entornos museísticos, por poner algunos ejemplos.

El reconocimiento facial es una tarea muy fácil para los seres humanos, y un ejemplo muy claro de esto es el caso de los niños quienes rápidamente aprenden a reconocer el rostro de sus padres. Esto podría suponer que enseñarles a las computadoras a reconocer personas es una tarea fácil, pero lastimosamente no es así.

El reconocimiento de rostros es un problema que fue considerado desde las primeras etapas de visión por computador. Este problema ha sido estudiando más a fondo en los últimos años, gracias a los avances computacionales que han permitido implementar algoritmos super complejos usando técnicas como por ejemplo la técnica de eigenface y las basadas en redes neuronales.

A lo largo de esta investigación, se podrá conocer los métodos para el reconocimiento y aprendizaje de rostros humanos, además de las técnicas de identificación de patrones de reconocimiento de rostros en una imagen a través de los rasgos de características aprendidas.

Además, se propone el desarrollo de una aplicación de reconocimiento de rostros que detecte caras de la imagen, así como también qué parte de la cara de una persona se selecciona (ojos, oídos, nariz, entre otras).

CAPÍTULO 2

ESTADO DEL ARTE

Luego de haber realizado una búsqueda bibliográfica del tema en estudio, se encontraron varios artículos relacionados con la investigación:

En 1987 Sirovich y Kirby realizaron el primer trabajo de reconocimiento facial, en donde se utilizaba el análisis de componentes principales (PCA), para generar unas imágenes semejantes a caras llamadas eigenpictures. El objetivo principal de este trabajo de investigación fue el de caracterizar un conjunto de caras con un número mínimo de parámetros para solucionar el problema de la caracterización e identificación de los patrones para la representación de caras extraídas de las imágenes [6].

Respecto a los sistemas basados en PCA, tienen un problema que se basan en la apariencia de las imágenes para proceder a la comparación, esto quiere decir que influye en gran medida el gesto que tenga la cara de la persona en el instante que se realice el reconocimiento.

En el año de 1991, Turk y Pentland demostraron que el error residual de codificar las Eigenface se podía utilizar para detectar caras en las imágenes, a través de un descubrimiento que permitió sistemas automatizados de reconocimiento facial en tiempo real [7].

En 1997 B. Moghaddam y A Pentland realizaron un estudio que en vez de utilizar PCA crearon una técnica no supervisada para el aprendizaje visual. Esta se basaba en la estimación de la densidad en espacios de alta dimensión, utilizando la descomposición por medio de eigenspaces. Estos métodos generan una representación lineal reducida de las imágenes de las caras, en donde cada cara proyectada tiene menor dimensión antes de realizarse el reconocimiento [8].

La tecnología capturó la atención del público a partir de la reacción de los medios a una prueba de implementación en el Super Bowl de la NFL en enero de 2001, la cual capturó imágenes de vigilancia y las comparó con una base de datos de fotos digitales. Esta demostración inició un muy requerido debate sobre cómo usar la tecnología para

satisfacer necesidades nacionales, mientras se tomaban en consideración las preocupaciones sociales y de privacidad del público.

Javier Calli, en su trabajo de investigación titulado “Reconocimiento facial basado en Algoritmo Eigenface”, propuso un algoritmo con la capacidad de localizar y reconocer rostros de imágenes digitales de frente con variación en escala. También se describe la construcción de un sistema de adquisición de imágenes para generar una base de dato de fotografías que permitan se comparadas contra imágenes con cambio de tamaño. Además en este trabajo se presenta como crear la base de datos de imágenes de prueba, así como los resultados en la localización de ojos, la extracción de la región elíptica de la cara, la normalización, el suavizado con variación total y la aplicación de la técnica de análisis de componentes principales para la generación del espacio de Eigenface y obtención del rostro promedio [9].

CAPÍTULO 3

MARCO TEÓRICO

3.1. Definiciones generales

3.1.1. Pixel

Un pixel es el menor elemento de una imagen el cual posee una intensidad. Pixel es una abreviatura inglesa que significa “picture element”.

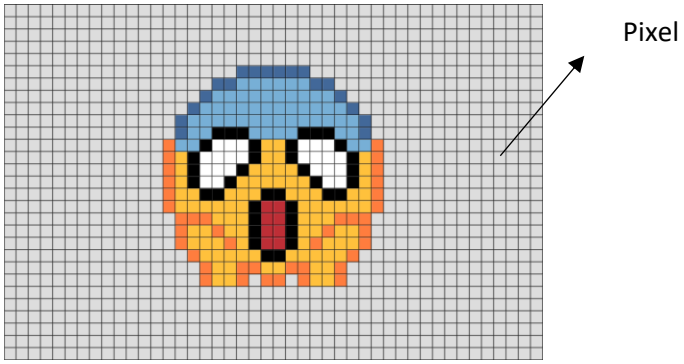


Ilustración 1: Pixel.

3.1.2. Imagen

La agrupación o el conjunto de pixeles forman la imagen, cada una con un valor de intensidad o brillo asociado. Una imagen se representa a través de una matriz, ya sea bidimensional o tridimensional, de manera que cada elemento corresponda a la intensidad del pixel de la imagen.

Ilustración 2: Imagen Original.

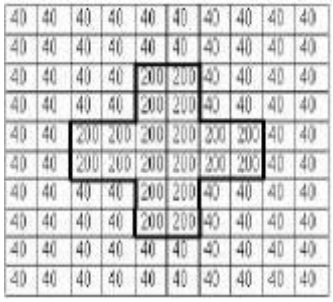


Ilustración 3: Estructura matricial de la Imagen.

3.1.2.1. Tipos de Imágenes

3.1.2.1.1. Color verdadero o RGB

En este tipo de imágenes cada pixel tiene un color particular, en donde dicho color está formado por una cantidad de color rojo, por una cantidad de color verde y por una cantidad de color azul. Cada componente tiene un valor de intensidad cuyo rango está entre 0 y 255, formando un total de 256^3 = 16777216 diferentes colores posibles dentro de una misma imagen.

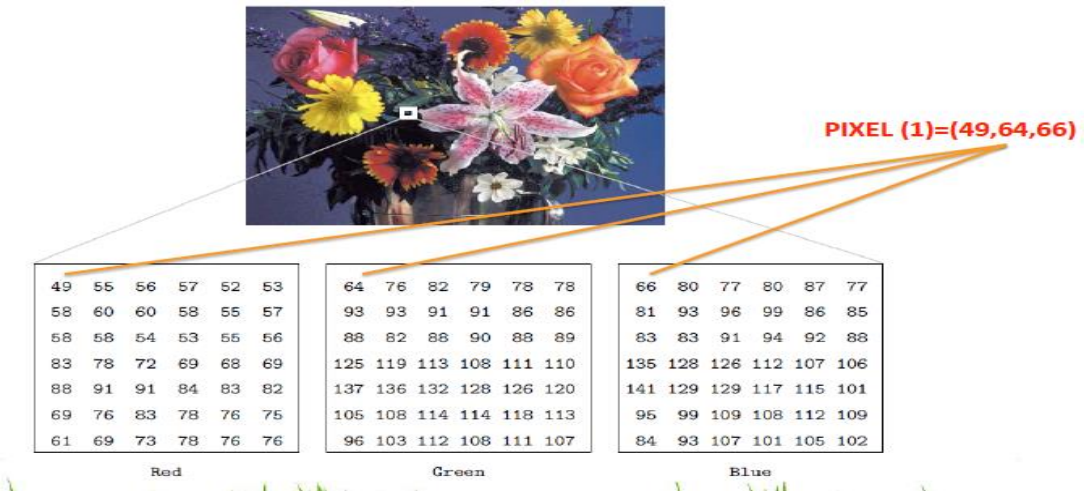


Ilustración 4: Imagen RGB:

3.1.2.1.2. Escala de grises

En una imagen a escala de grises cada pixel es una sombra de gris, en donde la intensidad normalmente va desde 0 (color negro) hasta 255 (color blanco). Este rango de intensidad puede ser representado por 8 bits o exactamente 1 byte. Este tipo de imágenes es muy común encontrarlas en imágenes de rayos X, o en imágenes de texto impresos.

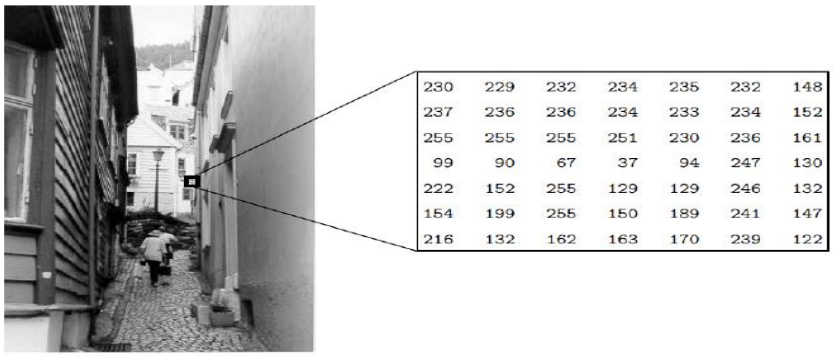


Ilustración 5: Imagen escala de grises.

3.1.2.1.3. Blanco y negro

Este tipo de imágenes o también denominadas imágenes binarias, cada pixel tiene un color que puede ser blanco (1) o negro (0), por lo que se necesita un bit por pixel. Es muy común encontrar estas imágenes en textos, formas y planos arquitectónicos.

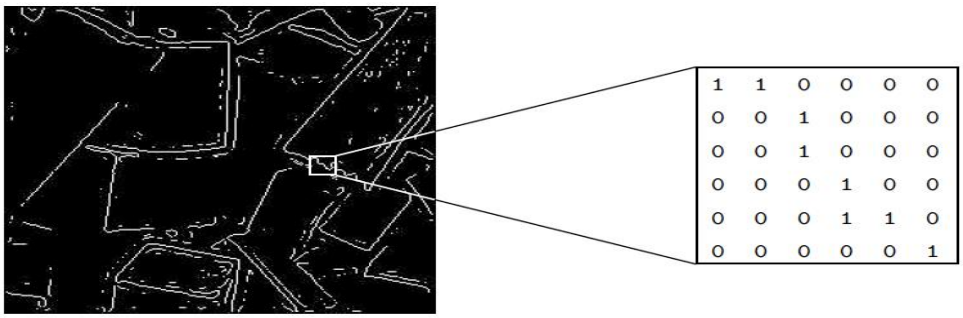


Ilustración 6: Imágenes binarias.

3.1.2.1.4. Indexadas

La mayoría de las imágenes solo tienen un pequeño subconjunto de colores de los más de 16 millones de colores posibles. Para conveniencia de manipulación y almacenamiento, la imagen tiene un mapa de colores asociados llamado paleta de colores. En este tipo de imágenes cada pixel tiene un valor que no corresponde al color directamente como en las de RGB, sino que corresponde al índice del color dentro del mapa de colores.

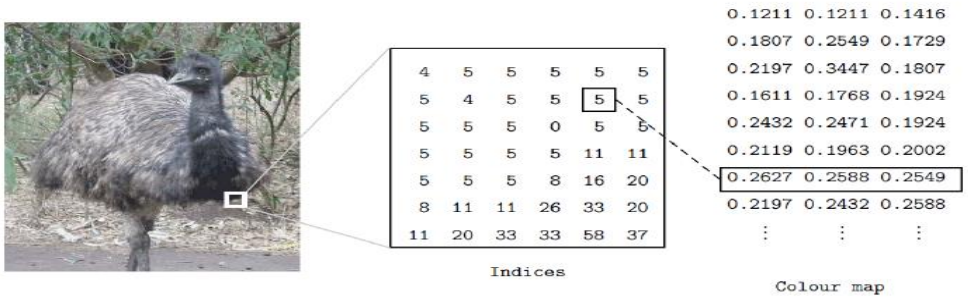


Ilustración 7: Imagen indexada.

3.2. Reconocimiento de facial

En muchos casos las condiciones bajo las que se obtiene la imagen son controladas, por ejemplo, las fotografías obtenidas por la policía o las obtenidas en el laboratorio. Por lo tanto, la localización de la cara en la escena puede ser fácilmente determinada. En otros casos la localización de la cara de la imagen no es conocida a priori. El primer paso, por lo tanto, es determinar si en la escena existen caras y si una cara está presente, localizarla e la imagen.

El reconocimiento facial tiene como objetivo identificar a las personas que están presentes en una imagen.

El problema de reconocimiento se debe a muchos factores. Uno de los inconvenientes a la hora de detección de cara en una imagen es el vello facial, es decir, bigote, barba, entre otros. Otro problema es la orientación de la cara en la imagen. Además, existen otros factores que dificultan de una u otra manera la detección, como lo es la iluminación y la calidad de las imágenes.

También se vuelve un problema el reconocimiento de una persona cuando existen cambios en el rostro debido a enfermedades, accidentes, tratamientos o cirugías estéticas o simples cambios en el maquillaje, además, el reconocimiento sobre fotografías antiguas en las que las personas a identificar tienen una complexión distinta y también se debe tener en cuenta el parecido entre padres/madres y sus hijas/hijos, hermanos, o gemelos.

3.2.1. Preprocesamiento de la imagen

Como se puede saber las imágenes que son de una misma persona pueden ser tomadas en diferentes momentos, lo que finalmente resultaría que cada imagen sea diferente debido a la variación de la iluminación, el ángulo de enfoque y el tamaño del rostro de la persona. Por este motivo es importante procesar la imagen.

Entre las tareas más importantes en el preprocesamiento de la imagen se puede nombrar:

- Extraer la información del rostro.
- Cambiar el tamaño de la imagen a un tamaño similar con todos los rostros.
- Aplicar un método de filtrado.

3.2.2. Extracción de Características

El objetivo principal de esta fase es extraer la información más discriminante de un rostro, eliminando toda la información innecesaria. La extracción de características se la utiliza para obtener la información que resulta más relevante de la cara con la finalidad de realizar una comparación.

Las técnicas más usadas son:

- Extraer componentes básicos del rostro.
- Relacionar la distancia entre ojos, nariz y boca.
- Descriptores invariantes y vectores de característica.

3.2.3. Entrenamiento

Esta etapa consiste en usar alguna forma de aprendizaje que le permita al sistema aprender rostros.

Para esto se utiliza la metodología de redes neuronales. Esto consiste en obtener los valores correspondientes a cada una de las conexiones de las cuales forman la red neuronal.

3.2.4. Reconocimiento

Esta es la última etapa de reconocimiento facial. Esta etapa se basa en alimentar al sistema con imágenes de rostros diferentes utilizadas en la fase anterior (entrenamiento), con la

finalidad de obtener un resultado alguna forma de codificación que permita saber de qué persona se trata ese rostro.

3.3. Detección de rostros

La detección de rostros es el proceso para buscar rostros humanos automáticamente en los medios sociales (imágenes o videos digitales). Cuando se detecta un rostro, se informa en una posición y se asocia con un tamaño y una orientación. Además, se puede buscar por puntos de referencia, como los ojos y la nariz. [10]

Ilustración 8: Detección de Rostros.

Los métodos de detección de caras tienen como objetivo dos tareas diferentes:

- Detección de objetos: la cual consiste en determinar si en una imagen está presente algún objeto de los que el sistema pueda identificar.
- Localización: consiste en encerrar el objeto dentro de un área para su procesamiento. Con esto obtendremos una zona de interés (ROI - Region of interest). [11]

Tradicionalmente la detección de objetos se realiza en base a la obtención de características de la imagen, lo que permite centrar el interés en regiones concretas y de esta forma se puede procesar un menor volumen de información. Existen varios métodos que se basan en la obtención de características de la imagen:

- En base al color
- A la Forma

- Movimiento en vídeos
- O una combinación de todas las anteriores

Sin embargo, aunque los mecanismos anteriores ofrecen soluciones factibles, en la actualidad el reconocimiento de objetos se realiza mediante redes neuronales, que, aunque también se basen en la detección de características, éstas no están prefijadas en la red, siendo éstas aprendidas por la misma y permitiendo un continuo refinamiento del algoritmo interno. [11]

En el momento de detectar un rosto se debe tomar en cuenta algunos inconvenientes que pueden afectar de manera negativa a la detección como por ejemplo la inclinación de la cabeza de las personas, la colocación de gafas o sombreros, e incluso algunos gestos como son la sonrisa, guiños, entre otros.

Para la detección de rostros se deben de seguir los siguientes pasos:

- Detectar si existe alguna cara en la imagen sin identificarla. Si se tratase de un video se puede hacer un seguimiento del rostro. A través de esto, se proporciona la localización y la escala de donde encontramos la cara.
- Localizar los componentes del rostro, a través de transformaciones geométricas. Para normalizar las imágenes de rostros se pueden utilizar diferentes métodos como, por ejemplo, la distancia entre las pupilas, la posición de la nariz, entre otras.
- Proporcionar información para distinguir entre los rostros de diferentes personas según las variaciones geométricas.

El patrón facial de características extraído se compara con los vectores de características extraídos de las caras de la base de datos. Si se encuentra un gran porcentaje de similitud, retorna la identidad del rostro de la persona, caso contrario, se indica que la cara es de una persona desconocida.

Ilustración 9: Proceso para detectar rostros.

3.4. Métodos de aprendizaje de rostros

3.4.1. Redes neuronales

Las redes neuronales es una parte muy relevante en la Inteligencia Artificial. Están inspiradas en el funcionamiento del cerebro humano, ya que trata de crear modelos artificiales con la finalidad de solucionar problemas complicados de resolver, por lo que las redes neuronales tratan de emular el comportamiento del cerebro humano.

Una red neuronal está compuesta por una serie de neuronas interconectadas entre sí mediante enlaces que imitan los componentes neuronales biológicos. Las redes neuronales al igual que las redes biológicas aprenden por repetición y cuando más información se tenga para entrenar y más veces se entrena a la red, se obtendrán mejores resultados.

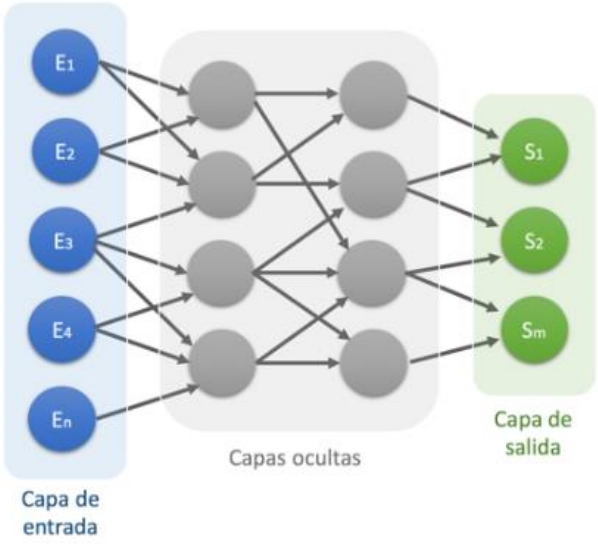


Ilustración 10: Modelo de una Red Neuronal.

Las características principales de las redes neuronales son:

- Su topología.
- El mecanismo de aprendizaje.
- Tipo de asociación realizada entre la información de entrada y salida.
- La forma de representación.

Existen diferentes arquitecturas de redes neuronales entre las cuáles se puede mencionar:

- Redes alimentadas hacia delante de capa simple.
- Redes alimentadas hacia delante de multicapas. Redes recurrentes.

Una de las arquitecturas más utilizadas para el reconocimiento de rostros es la red alimentada hacia delante multicapas o red de perceptrones multicapas (MLP), ya que una red multicapa es capaz de resolver problemas más complejos que una red monocapa. [12]

Las redes neuronales presentan un sinnúmero de características semejantes a las del cerebro humano. Son capaces de aprender de la experiencia, abstraer características esenciales a partir de la entrada de información. Las ventajas de las redes neuronales son:

- Tienen la habilidad de aprender mediante una etapa de aprendizaje.
- Crea su propia representación de la información en su interior.
- Debido a que una red neuronal guarda información en forma redundante, ésta puede seguir respondiendo de manera aceptable.
- Puede manejar cambios no relevantes en la información de entrada, como señales con ruido u otros cambios en la entrada.
- La estructura de una red neuronal es paralela, es decir que, si es implementado con dispositivos electrónicos o en computadores, se pueden obtener resultados en tiempo real.

3.4.1.1. Redes neuronales convolucionales (CNN)

Las redes neuronales convolucionales (CNN), nacen con la necesidad de procesar imágenes de una manera efectiva y eficiente. Hoy en día se utilizan también para lo que es procesamiento de textos, pero su fuerte es para procesamiento de imágenes, ya sean en fotografías o videos.

Estas redes cuentan con una capa de entrada, la cual se encarga de recibir todos los pixeles de la imagen. Por ejemplo, si se tiene una imagen de 10 x 10 se va a tener 100 entradas para recibir la imagen. Después la capa de entrada va a pasar toda esa información a las

capas ocultas. Dentro de las capas ocultas se van a tener dos operaciones importantes: el *pooling* o agrupación, que es una operación que se utiliza para reducir el tamaño de la imagen, es decir que la información procesada en las siguientes capas es de una imagen más pequeña que la original, con la que será menos pesado para la computadora. Y la otra son las convoluciones, que consiste en pasar una serie de filtros a la imagen, con la finalidad de detectar ciertos patrones relevantes dentro de ella.

Por ejemplo, si se tiene una imagen muy sencilla y al lado un kernel o filtro que tiene una orientación vertical de tamaño 3 x 3, por lo general el filtro tiene un tamaño mucho menor al de la imagen. La idea de la convolución es que ese filtro se vaya desplazando por la imagen y a medida que se va desplazando va realizando una serie de operaciones, en donde es un proceso iterativo. En la primera iteración el filtro está ubicado en la esquina superior izquierda de la imagen y lo que hace la convolución es multiplicar punto a punto los coeficientes del filtro con la porción de la imagen que está debajo de ese filtro con los pixeles seleccionados de la imagen. Se hace una multiplicación punto a punto con cada uno de los coeficientes de la imagen y se suman los valores para generar el resultado o la imagen resultante o píxel correspondiente en la imagen de salida. En la segunda iteración se realiza exactamente la misma operación con la diferencia de que el filtro se desplaza una posición a la derecha, se repite el procedimiento y se obtiene el resultado, y así sucesivamente. Cuando llegue al extremo derecho el kernel se desplaza una posición hacia abajo y vuelve al lado izquierdo y se realizan las mismas operaciones. Se repite el procedimiento hasta que el kernel hace un barrido de toda la imagen. La imagen resultante va a ser más pequeña que la imagen de entrada.

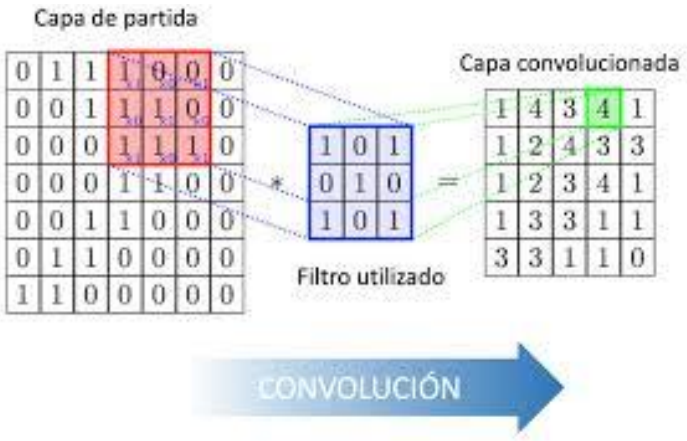


Ilustración 11: Convolución.

3.4.1.2. Redes neuronales recurrentes (RNN)

Las redes neuronales recurrentes (RNN) son capaces de procesar diferentes tipos de secuencias como textos, conversaciones, música, videos, y además de eso, no sólo clasifican los datos como lo hacen las redes neuronales convolucionales, sino que también están en capacidad de generar nuevas secuencias. Si a una red neuronal o convolucional se le presenta por ejemplo una imagen, con el entrenamiento adecuado estas arquitecturas lograrán clasificar un sinnúmero de datos logrando a la vez una alta precisión. Pero qué sucede si en lugar de una única imagen se introduce a la red una secuencia de imágenes, es decir un video. En este caso las RCN no serán capaz de procesar los datos, en primer lugar, porque esta arquitectura está diseñada para que los datos de entrada y de salida siempre tengan el mismo tamaño. Sin embargo, un video o una conversación se caracterizan por ser un tipo de dato con un tamaño variable. En segundo lugar, en un video los datos están correlacionados, esto quiere decir que la siguiente imagen en la secuencia de video dependerá de la imagen anterior, incluso las imágenes estarán relacionadas con las que se presenten más adelante en la secuencia. Una red neuronal convolucional no está en capacidad de analizar la relación entre varias imágenes de la secuencia. Las redes neuronales recurrentes no presentan este inconveniente y por tanto son capaces de analizar secuencias.

Una secuencia es por ejemplo un texto escrito, en donde su contenido no basta con que leamos cada palabra de manera individual ya que nuestro cerebro concatena todas las palabras leídas hasta el momento permitiéndonos comprender la idea central del texto. Así que una secuencia es una serie de datos, imágenes, palabras, etc., que siguen un orden específico y que tienen significado cuando se analiza en conjunto y no de manera individual.

Una red neuronal recurrente no tiene una estructura de capas definida, sino que permiten conexiones arbitrarias entre las neuronas, incluso pudiendo crear ciclos, con esto se consigue crear la temporalidad, permitiendo que la red tenga memoria.

Las redes neuronales recurrentes son muy potentes para todo lo que tiene que ver con el análisis de secuencias, como puede ser el análisis de textos, sonido o video [13].

3.4.2. Algoritmos genéticos

Un algoritmo genético es un método de optimización basado en los mecanismos de la evolución natural para resolver problemas de búsqueda y aprendizaje. [12]

Un algoritmo genético es una técnica de búsqueda basada en la teoría de la evolución, donde se intenta replicar el comportamiento biológico de la selección natural y la genética.

Son procedimientos o algoritmos que intentan emular el proceso de evolución. El objetivo consiste en conseguir la mejor solución por comparación con un conjunto de soluciones, en donde, se generan soluciones a partir del cruzamiento de las soluciones obtenidas y se comparan si son mejores que las anteriores.

Entre las fases de un algoritmo genético se tiene:

1. Codificar la información del problema.
2. Generar aleatoriamente la población inicial. Cada individuo necesita tener codificada su información en forma de cromosoma. Cada uno de estos cromosomas son posibles soluciones del problema a estudiar.
3. Evaluación de la población. A cada cromosoma se le aplica la función de aptitud que ha elegido para su estudio.
4. Selección y generación de nueva población.
    - Se repiten los pasos 3 y 4 un número de veces y finalmente se elige la solución de mayor puntuación.
    - Se genera una nueva población usando cromosomas con mayor puntuación y añadiendo nuevos cromosomas mediante:
        - Cruce: Se eligen aleatoriamente los individuos que se van a reproducir y el cruce también se hace de forma aleatoria.
        - Mutación: Contribuye a introducir diversidad en el proceso. Consiste en utilizar los cromosomas de mayor puntuación e introducir un cambio aleatorio en sus elementos.

Los algoritmos genéticos son de probada eficacia en caso de querer calcular funciones no derivables, aunque su utilización es posible con cualquier función.

Entre las limitaciones de los algoritmos genéticos se pueden mencionar las siguientes:

- Para un sistema que está compuesto por demasiadas variables, componentes o elementos su espacio de búsqueda aumenta de manera exponencial. Esto se debe a las relaciones que pueden surgir.
- Para problemas que poseen una alta complejidad, la función de evaluación puede ser demasiada costosa en relación con el tiempo y recursos.
- No se recomienda utilizarlos en problemas en donde se buscan soluciones a problemas que convergen en resultados simples como por ejemplo correcto o incorrecto, ya que el algoritmo difícilmente convergerá y la solución será tan válido como escoger a azar.
- Puede aparecer casos en los cuales dependiendo de los parámetros que se usen para evaluar el algoritmo, podría no converger en una solución óptima o terminar en resultados no satisfactorios.
- La solución óptima se obtiene a través de la comparación a otras soluciones, por lo que no se tiene demasiado claro un criterio de cuándo se debe detener ya que no se toma en cuenta con una solución específica.

3.5. Técnicas aplicadas al reconocimiento facial

3.5.1. Técnicas basadas en apariencia

La técnica basada en experiencia tiene como objetivo convertir el problema de reconocimiento facial en un problema de análisis de espacio, en donde se puedan aplicar diferentes técnicas estadísticas. Uno de los problemas de esta técnica es que para obtener buenos resultados se requieren un conjunto de muestras considerable para la fase de entrenamiento. Otros inconvenientes son los aspectos de iluminación, la pose o la expresión de la cara, lo que posee un gran impacto al momento de generar los resultados finales. Dependiendo del método utilizado, estos inconvenientes pueden tener mayor o menor impacto.

3.5.1.1. Análisis de Componente Principal (PCA). Eigenfaces

En visión por computador, se le conoce como eigenfaces al conjunto de vectores propios para el reconocimiento de la cara humana. El método de eigenfaces fue planteado por Sirovich y Kirby [14], para el reconocimiento y desarrollado unos años después en la clasificación de la cara por Matthew Turk y Alex Pentland [15].

El PCA es una técnica muy difundida para la parametrización de: formas, apariencia y movimiento, la misma que ha sido desarrollada como una especie de paradigma en la visión artificial. Las representaciones que realiza PCA han demostrado ser útiles para la resolución de problemas como reconocimiento de objetos y rostros, seguimiento, detección y modelado de fondos. Los datos de entrenamiento para PCA son pre – procesados de alguna manera o son generados por algún otro algoritmo de visión [16].

Antes de realizar alguna tarea de reconocimiento de un rostro dentro de una imagen, es importante convertir esta imagen a un formato que sea útil para generar patrones. Una de esas técnicas es la de eigenfaces, la cual simplifica la imagen y aísla las características de interés del rostro. Esta técnica es utilizada para la obtención de rostros en una imagen.

El algoritmo de eigenface, consiste en los siguientes pasos:

1. En el entrenamiento, cada imagen es organizada como un vector bidimensional R^2, donde cada valor se obtiene de la concatenación de cada una de las filas de la imagen, obteniendo como resultado una matriz de dimensiones R^2 x M. El rostro promedio ω se genera según la siguiente fórmula:

$$\Psi = \frac{1}{M} \sum_{n=1}^{M} \Gamma_n$$

Γ_i con i = 1, 2, ……. M

2. Este rostro promedio es restado a cada una de las imágenes $\neg_i$. La variable i debe estar entre 1 y M, donde se obtiene un nuevo conjunto de vectores.

$$\Phi = \Gamma_i - \Psi$$

3. Después de aplicar la fórmula anterior, se obtiene como resultado la matriz D = $[\Phi_1, \Phi_2, \ldots . \Phi_M]$ de dimensiones N^2 x M.

4. Aquí es donde se buscan los autovectores de la matriz de covarianza de D y la matriz traspuesta de D. Aplicando la siguiente fórmula:

$$C = \frac{1}{N^2} D.D'$$

Se obtiene una matriz de N^2 x N^2. Los vectores resultantes son vectores ortonomarles que se usan para construir la representación de imágenes. El tamaño de la matriz C hace intratable este paso. Para resolver este inconveniente, se realiza la aproximación de dichos vectores.

- Primero de obtiene la matriz de covarianza reducida: L = $\frac{1}{M}$ D'.D de dimensiones M x M.
- Se generan los autovectores de L, los cuáles se ordenan de mayor a menor según sus correspondientes autovalores, los que conforman la matriz v.
- Se aproximan los autovectores C: u = D v. Donde cada columna de u representa un vector propio.

5. Se obtiene un patrón:
   Ω_i^T = [ω_1, ω_2, …….. ω_M], donde el valor de i va desde 1 hasta M.
   $\omega_k = u_i^T (\Gamma_i - \Psi)$ con k = 1 hasta M

Dada como imagen de entrada del sistema, el rostro de una persona, el proceso de reconocimiento intenta buscar en la base de datos de las imágenes, aquella que corresponde al rostro dado, a través del cálculo de su patrón Ω usando el proceso que se expuso anteriormente y se busca la distancia mínima.

$$\min(|\Omega - \Omega_i|^2) \text{ donde } i = 1, 2, \ldots\ldots M$$

Una vez hallada la distancia mínima se indica cuál es la imagen correspondiente.

3.5.1.2. Análisis Lineal Discriminante (LDA). Fisherfaces

Una variación del eigenface o análisis de componentes principales es el fisherface, la cual es una mejora de la primera. Este es un método cuyo objetivo principal es el reconocimiento de caras, teniendo en cuenta como se refleja la luz y las expresiones faciales. Teniendo en cuenta el que el conjunto de entrenamiento está etiquetado, es posible usar esa información para disminuir la dimensionalidad del espacio de características. Aplicando el método de Fisherface es probable construir una matriz de proyección en la cual la relación entre la dispersión intra-clase y la inter-clase sea máxima.

Sin embargo, según experimentos realizados muestran que entre los resultados existen variaciones en cada caso en cuanto a las condiciones de iluminación y gestos (15.3% de reconocimiento incorrecto para el Eingfaces frente al 7.3% para Fisherfaces) [17].

Para obtener los fisherfaces se define una matriz de distribución inter - clase (S_B) y la matriz de distribución inter - clase (S_W). Sea X_i (conjunto de N imágenes) y N_i el número de imágenes de X_i, dando como resultado la imagen u_i.

S_B y S_w se obtiene de las fórmulas:

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

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

La máxima relación de la matriz de distribución entre clases de imágenes proyectadas y las proyecciones futuras que habrá dentro de éste mismo determinante, es una matriz ortonormal denominada W_{opt}

$W_{opt} = \arg \max_w \frac{|W^T S_B W|}{|W^T S_W W|} = [W_1 W_2 \quad W_m]$

La siguiente matriz tiene c – 1 valores propios diferentes de cero. Por lo tanto, el límite superior de m es c – 1 (c es el número de clases).

$S_B \omega_i = S_W \omega_i Y_i$

En lo que tiene que ver con el reconocimiento de caras la matriz S_W es N – c (donde N es el número de imágenes de entrenamiento). Por ende, se puede elegir una matriz W en la que la distribución intra - clases de las imágenes proyectadas pueda ser 0.

Usando el método de PCA (eigenfaces) se disminuye la dimensión del espacio de características a N – c. A continuación, aplicamos la técnica FLD que reduce la dimensión a c – 1.

$W_{pca}^T = W_{fla}^T \, W_{pca}^T$

$W_{pca}^T = \arg \max_w |W^T S^T W|$

$W_{fld} = \arg \max_w \frac{|W^T W_{pca}^T S_B W_{pca} W|}{|W^T W_{pca}^T S_w W_{pca} W|}$

3.5.1.3. Independent Component Analysis (ICA)

Esta técnica es una generalización del método PCA. El objetivo fundamental de este método es descomponer una señal observada en una combinación lineal de fuentes independientes. La diferencia con el PCA es que el PCA decorrelaciona las señales de entrada usando estadísticos de segundo orden, mientras que el ICA minimiza mayores órdenes de dependencia.

Sea una matriz de variables independientes: $S = (S_1 \ldots\ldots S_n)$ y una matriz de observaciones X, en donde cada columna es el resultado de un experimento aleatorio, y en cada fila se tiene el valor de una prueba de ese experimento. Como es de conocimiento el método ICA trata de descomponer la señal observada en una combinación de fuentes independientes. La matriz de combinación se llamará A: $X = AS$

Con el algoritmo ICA se busca una matriz W que cumpla con la condición de que $U = WX = WAS$, en donde U es la estimación de máxima probabilidad de las componentes independientes.

Existen dos métodos de implementar el método ICA para el reconocimiento de caras:

- Se puede colocar en cada fila de la matriz X una imagen diferente, de esta manera se tendrá que cada imagen es una variable aleatoria y los pixeles son pruebas.
- Otra opción es trasponer la matriz X y tener en cada columna una imagen, de manera que, en este caso, los píxeles son variables aleatorias y cada imagen una prueba.

Al aplicar la primera opción en el ICA, se encuentra una matriz W tal que las columnas de U son estadísticamente independiente.

Las imágenes fuentes estimadas por las columnas de U son usadas como imágenes base para la representación de caras.

Los que se suele realizar para reducir la complejidad computacional es aplicar ICA sobre un conjunto de m (cuya condición debe ser $m < n$ filas) combinaciones lineales de las imágenes, obteniendo como resultado una matriz de m imágenes fuentes independientes en las columnas U.

3.5.1.4. Métodos basados en kernels

Estos métodos son una generalización de los tres métodos anteriores (PCA, LDA, ICA). Para la aplicación de estos métodos se debe realizar lo siguiente:

1. Se mapean los vectores de entrenamiento a través de una función no lineal que lleva a los puntos a un espacio de mayor dimensión.
2. Se plantea un problema equivalente al PCA, LDA o ICA en dicho espacio.
3. Se soluciona el problema equivalente usando el kernel trick, que es una manera reducida de resolver el problema de PCA, LDA o ICA en el espacio de mayor dimensión. Si cumple con las condiciones, se pueden realizar todos los cálculos de la resolución de problemas equivalente sin necesidad de mapear los vectores en el espacio de mayor dimensión. Para ello existen varias funciones llamadas núcleos (kernels), que lo hacen posible.

3.6. Análisis de sistemas actuales

3.6.1. Algoritmo de Viola – Jones

El algoritmo de Viola-Jones es un método de detección de objetos que se usa ampliamente en la detección de caras en imágenes y video. El algoritmo se basa en la comparación entre las intensidades luminosas de regiones rectangulares de las imágenes denominadas Haar-like features que calcula empleando una imagen integral [18].

En MATLAB el toolbox de Visión por computador donde es posible utilizar el detector de objetos en cascada (vision.CascadeObjectDetector) utiliza el algoritmo de Viola – Jones para detectar las caras, narices, ojos o la parte superior del cuerpo de las personas. También se puede utilizar la función Image Labeler para entrenar un clasificador personalizado. Para detectar rasgos faciales o la parte superior del cuerpo de una imagen se debe realizar lo siguiente:

1. Cree el objeto vision.CascadeObjectDetector y establezca sus propiedades.
2. Llame al objeto con los argumentos como si éste fuese una función.

Un ejemplo de la implementación del algoritmo de Viola – Jones en MATLAB se puede apreciar a continuación:

```
faceDetector = vision.CascadeObjectDetector;
I = imread ('visionteam.jpg');
bboxes = faceDetector (I);
IFaces = insertObjectAnnotation (I, 'rectangle' , bboxes, 'Face' );
figura
imshow (IFaces)
título ('caras detectadas'); [19]
```

Ilustración 12: Detección de Caras. Viola - Jones.

3.6.2. YOLO

El detector de objetos v2 you-only-look-once (YOLO) usa una red de detección de objetos de una sola etapa. YOLO v2 es más rápido que otros detectores de objetos de aprendizaje profundo de dos etapas, como las regiones con redes neuronales convolucionales (R-CNN más rápidas) [20].

YOLO implementa una Red Neuronal Convolucional que tiene como objetivo la detección de objetos. Las pruebas realizadas por su autor en el reconocimiento de varios tipos de objetos reflejan una tasa de procesamiento de 30 imágenes por segundo, lo que la convierte en un buen candidato para el procesamiento de videos [21].

El modelo YOLO v2 ejecuta una CNN de aprendizaje profundo en una imagen de entrada para producir predicciones de red. El detector de objetos decodifica las predicciones y genera cuadros delimitadores [20].

```
videoFile = 'highway_lanechange.mp4';
videoFreader =
vision.VideoFileReader(videoFile,'VideoOutputDataType','uint8');
depVideoPlayer =
vision.DeployableVideoPlayer('Size','Custom','CustomSize',[640 480]);
cont = ~isDone(videoFreader);
```

```
while cont
    I = step(videoFreader);
    in = imresize(I,[224,224]);
    out = yolov2_detect_mex(in);
    step(depVideoPlayer, out);
    cont = ~isDone(videoFreader) && isOpen(depVideoPlayer); % Exit the loop
if the video player figure window is closed
end
```

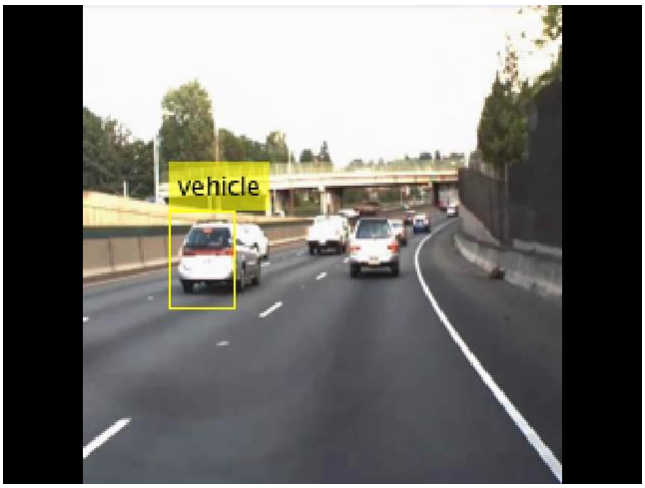


Ilustración 13: Detección de Objetos. YOLO.

3.6.3. HOG

HOG, cuyas siglas significan Histogramas de Gradientes Orientados, es un tipo de descriptor de características cuyo objetivo es generalizar el objeto de tal forma que el mismo objeto, en este caso un rostro, produzca lo más cerca posible del mismo descriptor de características cuando se lo vea bajo diferentes condiciones. Esto hace que la tarea de clasificación sea más fácil.

El detector de rostros HOG es bastante simple de entender. Una de las principales razones para esto es que utiliza una función "global" para describir un rostro en lugar de una colección de características locales. En pocas palabras, esto significa que todo rostro está representado por un único vector de características, a diferencia de muchos vectores de características que representan partes más pequeñas de ese rostro [22].

Para encontrar rostros en una imagen, comenzamos haciendo que la imagen sea en blanco y negro porque no se necesita datos de color para encontrar rostros.

Luego se observa cada píxel en nuestra imagen de a uno por vez. Para cada píxel, se quiere ver los píxeles que lo rodean directamente.

El objetivo es determinar cuán oscuro se compara el píxel actual con los píxeles que lo rodean directamente. Luego se quiere dibujar una flecha que muestre en qué dirección la imagen se vuelve más oscura.

Si se repite ese proceso para cada píxel de la imagen, se termina con cada píxel reemplazado por una flecha. Estas flechas se llaman gradientes y muestran el flujo de claro a oscuro en toda la imagen.

Esto puede parecer algo aleatorio, pero hay una buena razón para reemplazar los píxeles con gradientes. Si se analiza los píxeles directamente, las imágenes realmente oscuras y las realmente claras de la misma persona tendrán valores de píxeles totalmente diferentes. Pero al considerar solo la dirección en que cambia el brillo, tanto las imágenes en realidad oscuras como las realmente brillantes terminarán con la misma representación exacta [22].

Pero guardar el gradiente para cada píxel nos da demasiados detalles. Sería mejor si se pudiera ver el flujo básico de luminosidad / oscuridad en un nivel más alto para poder ver el patrón básico de la imagen.

Para hacer esto, dividimos la imagen en pequeños cuadrados de 16×16 píxeles cada uno. En cada cuadro, se va a contar cuántos puntos de gradientes en cada dirección principal (cuántos apuntan hacia arriba, cuantos apuntan hacia la derecha, etc.). Luego reemplazamos ese cuadrado en la imagen con las direcciones de flecha más fuertes [22].

El resultado final es que se convierte la imagen original en una representación muy simple que capta la estructura básica de una cara de una manera simple.

Para encontrar rostros en esta imagen de HOG, todo lo que se tiene que hacer es encontrar la parte de la imagen que se parece más a un patrón de HOG conocido que se extrajo de un gran conjunto de otras caras de entrenamiento: [22]

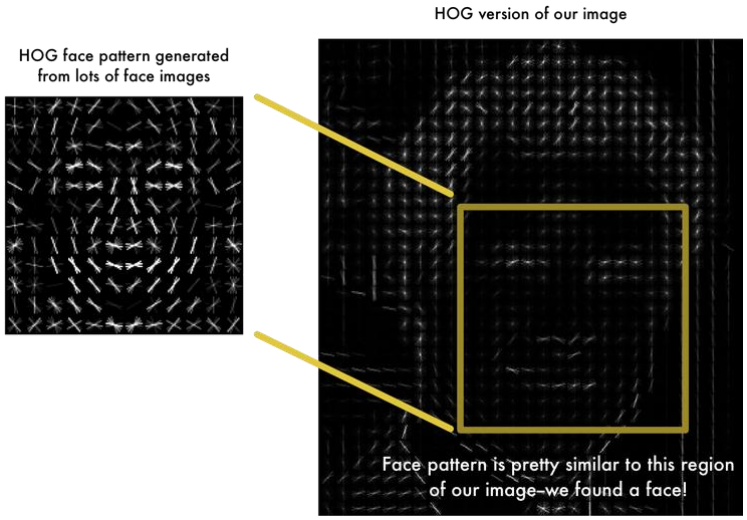


Ilustración 14: Aplicación de HOG.

Usando esta técnica, ahora podemos encontrar caras fácilmente en cualquier imagen:

Ilustración 15: Detección de caras. HOG.

3.6.4. Algoritmo KLT

En la visión por computador, el rastreador de características de Kanade – Lucas – Tomasi (KLT) es un enfoque para la extracción de características. Se propone principalmente con el fin de abordar el problema de que las técnicas tradicionales de registro de imágenes son generalmente costosas. KLT utiliza la información de intensidad espacial para dirigir la búsqueda de la posición que produce la mejor coincidencia. Es más rápido que las técnicas tradicionales para examinar muchas menos coincidencias potenciales entre las imágenes.

En el siguiente ejemplo se muestra cómo detectar y rastrear automáticamente una cara usando puntos de características. En este ejemplo, el enfoque hace un seguimiento de la cara incluso cuando la persona inclina la cabeza o se mueve hacia la cámara o hacia ella [23].

```
% Create a cascade detector object.
faceDetector = vision.CascadeObjectDetector();

% Read a video frame and run the face detector.
videoFileReader = vision.VideoFileReader('tilted_face.avi');
videoFrame      = step(videoFileReader);
bbox            = step(faceDetector, videoFrame);

% Draw the returned bounding box around the detected face.
videoFrame = insertShape(videoFrame, 'Rectangle', bbox);
figure; imshow(videoFrame); title('Detected face');
```

Ilustración 16: Detección de rostros. Algoritmo KLT.

El algoritmo KLT rastrea un conjunto de puntos de características a través de los cuadros de video. Una vez que la detección localiza la cara, el siguiente paso en el ejemplo identifica los puntos de características que se pueden rastrear de manera confiable. Este ejemplo utiliza el estándar, "buenas características para rastrear" propuesto por Shi y Tomasi [23].

```
points = detectMinEigenFeatures(rgb2gray(videoFrame), 'ROI', bbox);

% Display the detected points.
figure, imshow(videoFrame), hold on, title('Detected features');
plot(points);
```

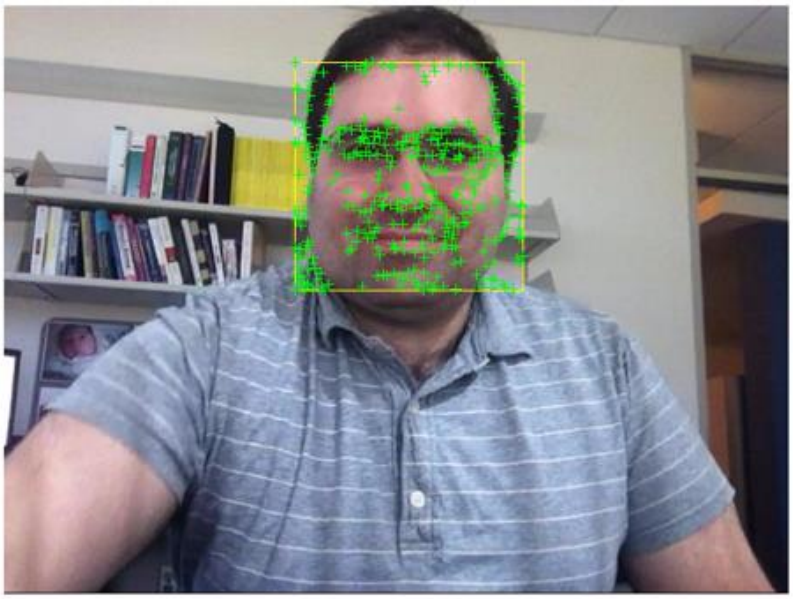


Ilustración 17: Detección de puntos.

CAPÍTULO 4

DESALLORRO DE LA APLICACIÓN

4.1. Herramientas utilizadas para el reconocimiento de rostros

4.1.1. Matlab

MATLAB, cuyas siglas significan Matrix LABoratory, es un entorno de desarrollo IDE con un lenguaje de programación propio, con extensión .m. Está disponible para varias plataformas como son Linux, Windows, Mac OS.

MATLAB permite la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de GUI (interfaces de usuario) y la comunicación de programas en otros lenguajes. El paquete de MATLAB dispone de dos herramientas que son Simulink y GUIDE. MATLAB posee cajas de herramientas también denominadas toolboxes y Simulink paquetes de bloques llamados blockset.

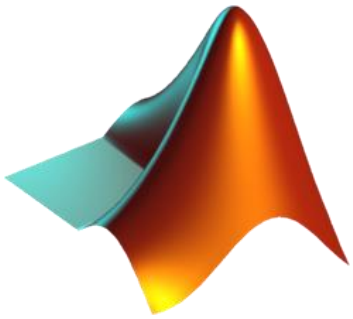


Ilustración 18: MATLAB.

4.1.2. Toolbox Visión por Computador

El Toolbox de Visión por Computador que viene incorporado en MATLAB, proporciona algoritmos, funciones y aplicaciones que permiten diseñar y probar sistemas de visión de video, visión 3D y procesamiento de video. Estas herramientas permiten la detección y seguimiento de objetos, así como la detección, extracción y comparación de características.

Puede entrenar detectores de objetos personalizados utilizando algoritmos de aprendizaje profundo y automático tales como YOLO, Faster R – CNN y ACF. La mayoría de los

algoritmos de la caja de herramientas son compatibles con la generación de código C / C++.

4.2. Desarrollo

Para realizar la fase de desarrollo se ha implementado una aplicación que corre bajo el sistema operativo Windows 8.x o superior, la cual permite abrir una imagen almacenada en el computador y aplicando técnicas de detección de rostros reconoce las partes de una persona, como son la nariz, boca, ojos. Además, se tiene almacenado en un archivo. mat una base de datos con imágenes de personas ingresadas previamente a través de la aplicación. Esta base de datos permitirá la identificación de personas por medio de la comparación de las caras detectadas con las caras almacenadas en la base de datos.

Para el desarrollo de la aplicación se basó en el algoritmo de Viola – Jones, ya que es un algoritmo muy fácil de entender, además las versiones nuevas de MATLAB traen incorporado el Toolbox de Visión por Computador lo cual hace más sencillo la realización de la aplicación.

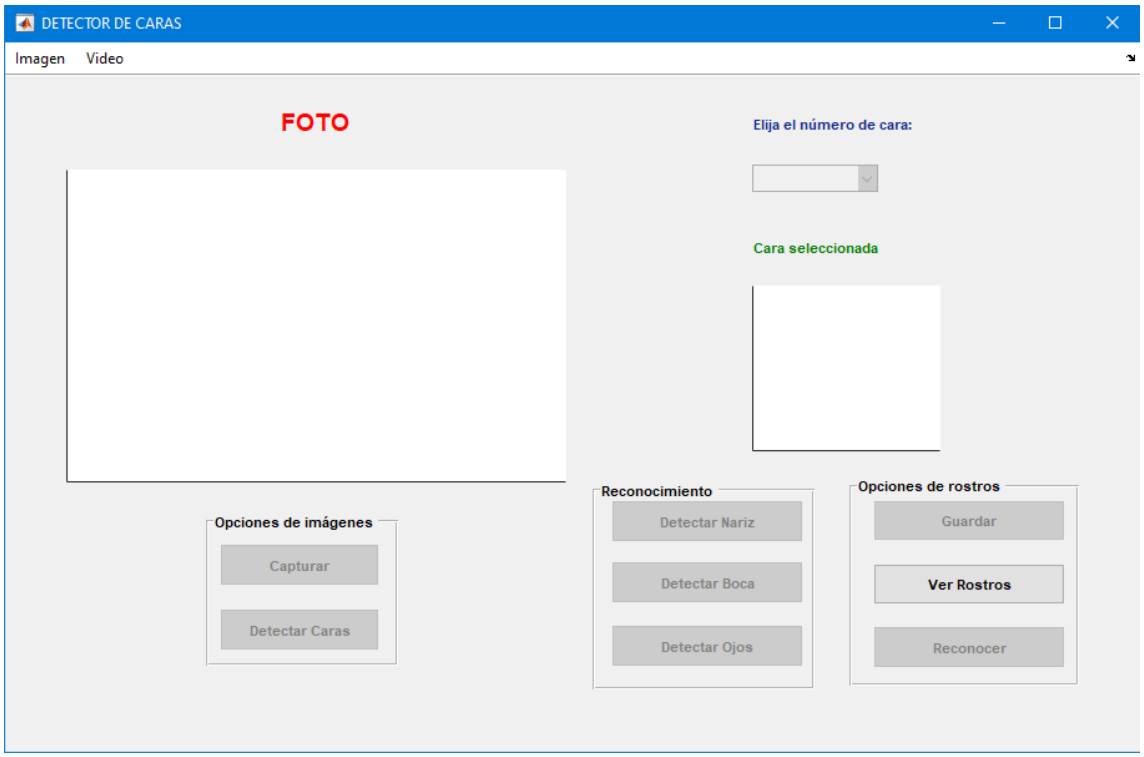


Ilustración 19: Pantalla Principal.

El algoritmo del funcionamiento paso a paso de la aplicación fue el siguiente:

1. Inicio.
2. Lee la imagen. Los puede hacer a través de una imagen o capturar la imagen desde un video.
3. Detecta Caras.
4. Si la cara es detectada entonces las caras se encierran en un rectángulo, y se activa un popup menú con el número de caras para que el usuario pueda seleccionar la cara, ir al siguiente paso. Caso contrario se visualiza un mensaje que no se ha detectado caras y finaliza el algoritmo.
5. Se elije la cara y se activa los botones de detección (nariz, ojos y boca) y se muestra en un rectángulo la parte seleccionada.
6. Luego el usuario tiene dos opciones que son, guardar la cara o reconocer la cara, Si el usuario desea almacenar la cara de la persona debe oprimir el botón guardar e ir al siguiente paso, caso contrario ir al paso 8.
7. Se deberá ingresar el nombre de la persona y esa información incluyendo el rostro se guardará en un archivo .mat.
8. Si el usuario desea reconocer la cara de una persona deberá hacer clic en el botón reconocer. La cara seleccionada se comparará con cada una de las caras guardadas en el archivo .mat. Si se encuentra la cara se mostrará un mensaje diciendo que se ha encontrado la cara y con el nombre de la persona correspondiente a esa cara. Caso contrario se visualizará un mensaje de que no se ha encontrado la cara y finaliza el algoritmo.

4.2.1. Lectura de la imagen

Cuando el usuario pulsa el menú Imagen, se despliegan dos submenús, uno de ellos es para Abrir la imagen. Al momento de hacer clic en el submenú Abrir, se accede a la ventana de buscador de archivos, donde el usuario debe seleccionar la carpeta donde se encuentra la imagen que se desea hacer el reconocimiento. El código en Matlab y resultados se muestran a continuación:

```
function mnuabrir_Callback(hObject, eventdata, handles)
global img;
[Filename Path] = uigetfile('*.jpg','Abrir Imagen');
if isequal(Filename,0)
   return;
else
```

```
    img=imread(strcat(Path,Filename));
    axes(handles.imgfoto);
    imshow(img);
    set(handles.btndetcaras,'enable','on');
end
```

El comando uigetfile permite desplegar el buscador de archivos en donde se mostrarán todas las imágenes en formato JPG. En la variable Path se almacena la ruta en la que se encuentra la imagen y en la variable Filename el nombre del archivo.
Luego se verifica si la variable Filename contiene o no una imagen seleccionada. Si esta contiene imagen se leerá la imagen a través del comando imread y se almacenará en la variable img, la cuál se mostrará en el axe imgfoto.

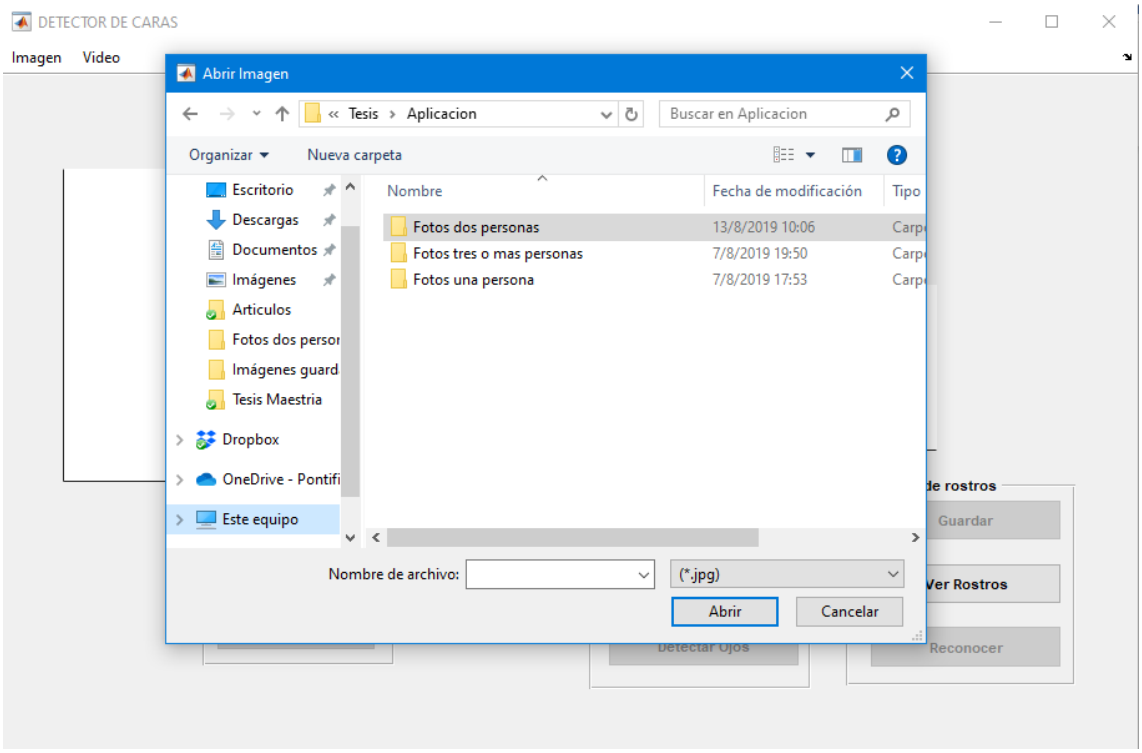


Ilustración 20: Opción de Abrir Imagen.

4.2.2. Captura de imagen por video

Otra de las opciones que poseen los usuarios es capturar una imagen a través de un video. Para esto el usuario debe ir al menú video, opción cámara. Cuando se desee tomar la foto sólo tendrá que hacer clic en el botón capturar.

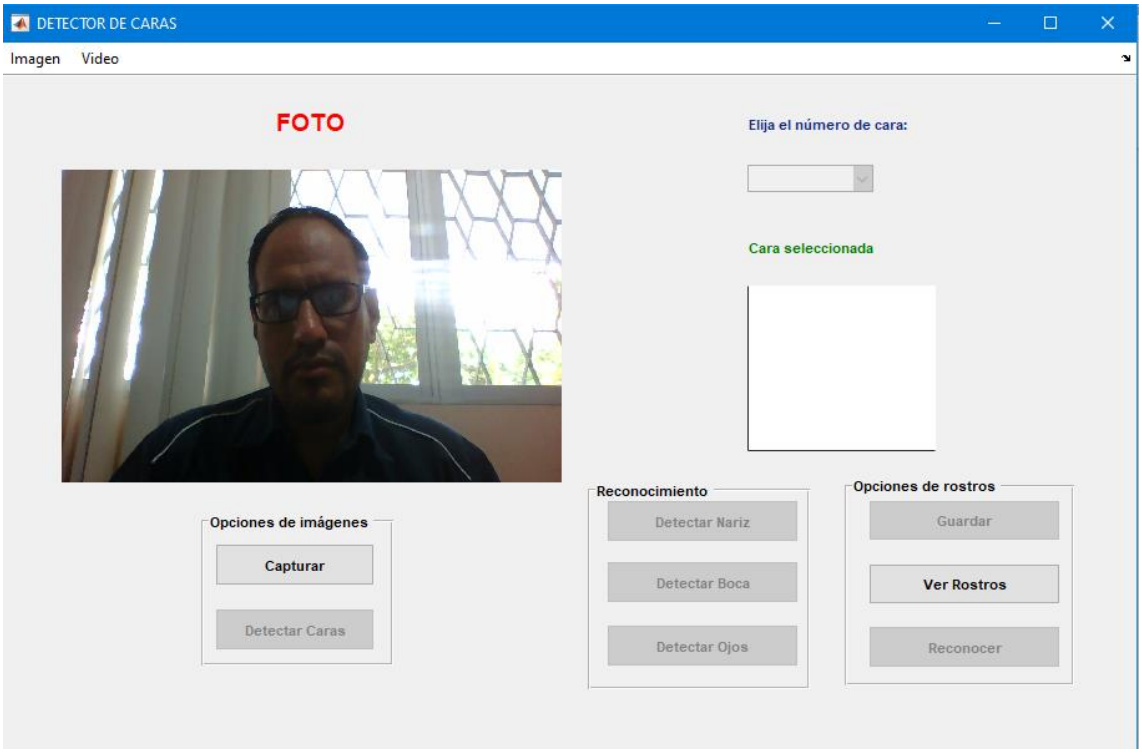


Ilustración 21: Video.

Para activar la cámara se utiliza el siguiente código:

```
function mnucamara_Callback(hObject, eventdata, handles)
global vid;
axes(handles.imgfoto);
vid=videoinput('winvideo',1);
vid.ReturnedColorSpace='rgb';
vidRes=get(vid,'VideoResolution');
nBands=get(vid,'NumberOfBands');
hImage=image(zeros(vidRes(2),vidRes(1),nBands));
preview(vid,hImage);
set(handles.btncapturar,'enable','on');
```

Luego de eso se procede a capturar el video:

```
function btncapturar_Callback(hObject, eventdata, handles)
global vid img;
img=getsnapshot(vid);
%close preview;
axes(handles.imgfoto);
imshow(img);
set(handles.btndetcaras,'enable','on');
```

Cabe recalcar que para un buen funcionamiento de la aplicación, el video debe realizárselo en un lugar iluminado, puesto que si es en un lugar oscura posiblemente no se detecte la cara.

4.2.3. Reconocimiento de caras

Luego que se lee y muestra la imagen, se abrirá la interfaz gráfica en donde se visualizará la foto seleccionada y una serie de botones, entre ellos uno que es para la detección de caras. El código usado fue:

```
function btndetcaras_Callback(hObject, eventdata, handles)
global img BB num;
face=vision.CascadeObjectDetector('FrontalFaceCART');
BB=step(face,img);
obj=insertObjectAnnotation(img,'rectangle',BB,'Cara');
imshow(obj);
hold on;
v=[];
if size(BB,1)==0
   msgbox('No se ha detectado ninguna cara','Mensaje');
else
 for i=1:size(BB,1)
      rectangle('position',BB(i,:),'Linewidth',4,'Linestyle','-
','Edgecolor','r');
      v=[v i];
 end
 msgbox(sprintf('Se han detectado: %d caras',i),'Detección');
 num=1;
 set(handles.btndetcaras,'enable','off');
 set(handles.lstcara,'string',v);
 set(handles.lstcara,'enable','on');
end
```

En este apartado se utiliza el clasificador vision.CascadeObjectDetector('FrontalFaceCART'), el cual permitirá detectar las caras que tiene la imagen.

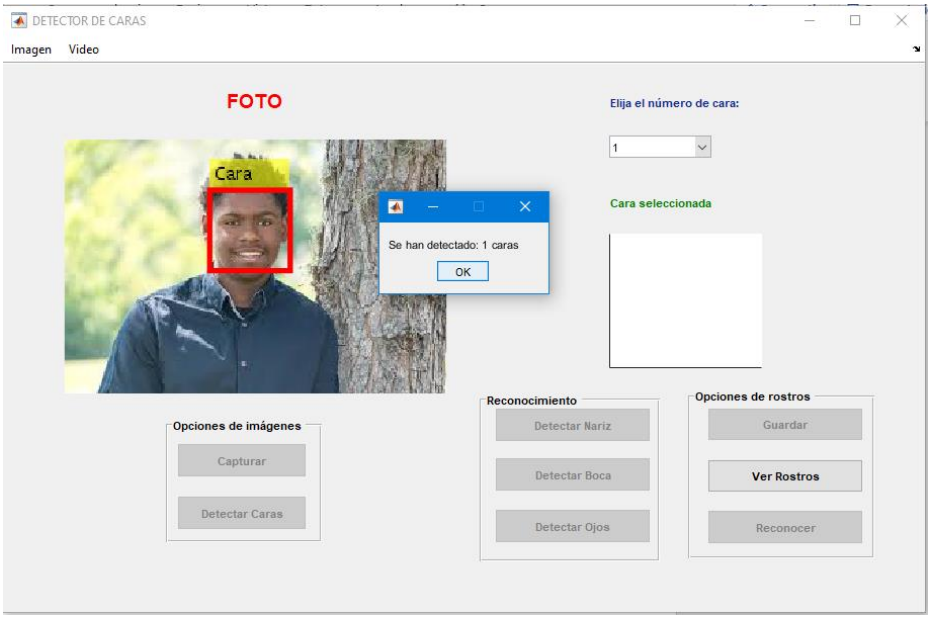


Ilustración 22: Detección de Caras.

En la variable BB se almacenan el número de caras detectadas. Si el tamaño de BB es igual a cero, quiere decir que en esa imagen no se detectaron caras y por ende se visualizará un mensaje de que no se ha detectado ninguna cara.

Si se han encontrado caras, se activará un popup menú el cual permitirá seleccionar la cara de la persona.

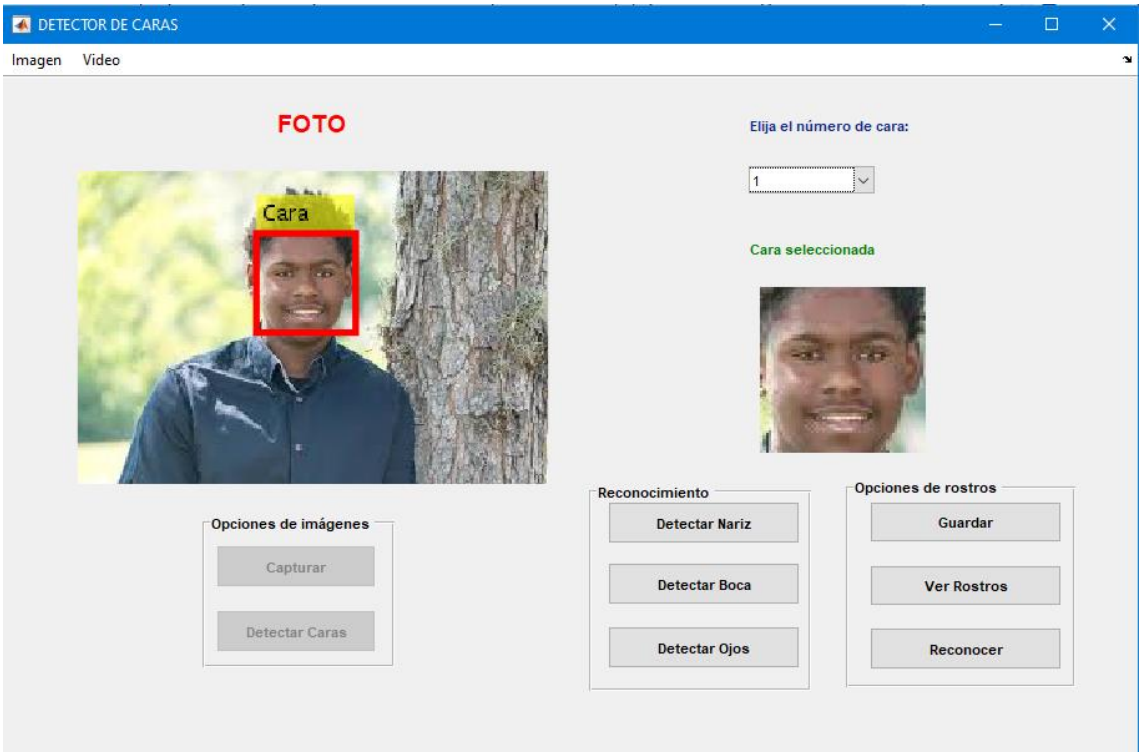


Ilustración 23: Selección de la Cara.

La aplicación también permite detectar la raza – etnia de la persona. La herramienta distingue entre tres razas que son la afro, mestiza y la blanca.

El código siguiente permite identificar la raza de la persona:

```
imagen=imresize(imgr,[40 40]);
imagen=imagen(1:5,16:23,1:3);
prom=(mean(mean(imagen(:,:,1)))+mean(mean(imagen(:,:,2)))+mean(mean(im
agen(:,:,3))))/3;
if (prom>=0 && prom<=110)
   msgbox('Es de raza afro','Raza');
elseif (prom>110 && prom<=160)
    msgbox('Es de raza mestiza','Raza');
else
     msgbox('Es de raza blanca','Raza');
end
```

En este caso lo primero que se hace es redimensionar la imagen a color a un tamaño de 40 x 40. Luego se extrae una porción de la piel de dicha imagen, para ello se hace un pequeño recorte a la imagen. Después de haber realizado el recorte se saca un promedio de la parte recortada, teniendo en cuenta que es una imagen RGB.

Si el promedio está entre 0 y 110 quiere decir que es de raza afro. Si está entre 110 y 160 es de raza mestiza y si es mayor a 160 quiere decir que es de raza blanca.

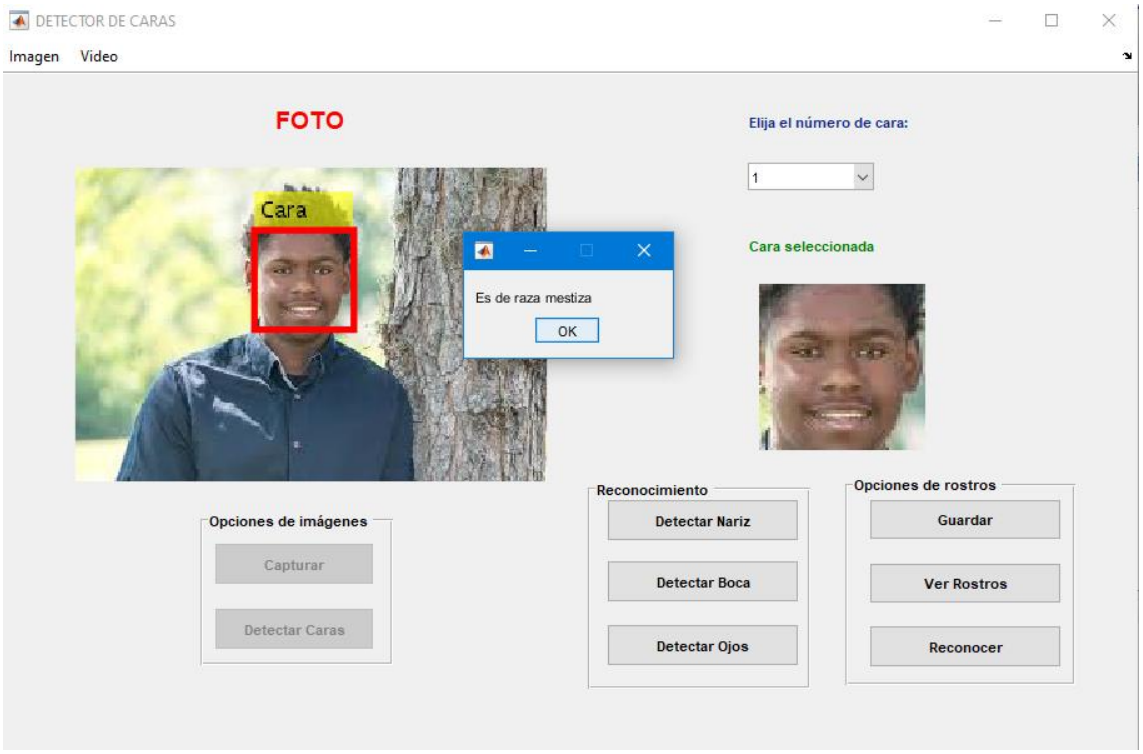


Ilustración 24: Raza de la persona.

4.2.4. Detección de nariz

Para la detección de la nariz se usó el siguiente código:

```
function btndetnariz_Callback(hObject, eventdata, handles)
global imgr num;
nariz=vision.CascadeObjectDetector('nose','MergeThreshold',num);
BBN=step(nariz,imgr);
if size(BBN,1)>0
axes(handles.imgcara);
rectangle('Position',BBN(1,:),'LineWidth',4,'LineStyle','-
','EdgeColor','b');
else
    msgbox('No se ha detectado la nariz','Detección');
end
```

Igual que en la detección de caras, se usa el clasificador vision.CascadeObjectDetector('nose','MergeThreshold',num), que permitirá detectar la nariz de la persona y si la encuentra almacenarla en la variable BBN. El resultado de aplicar este código generó lo siguiente:

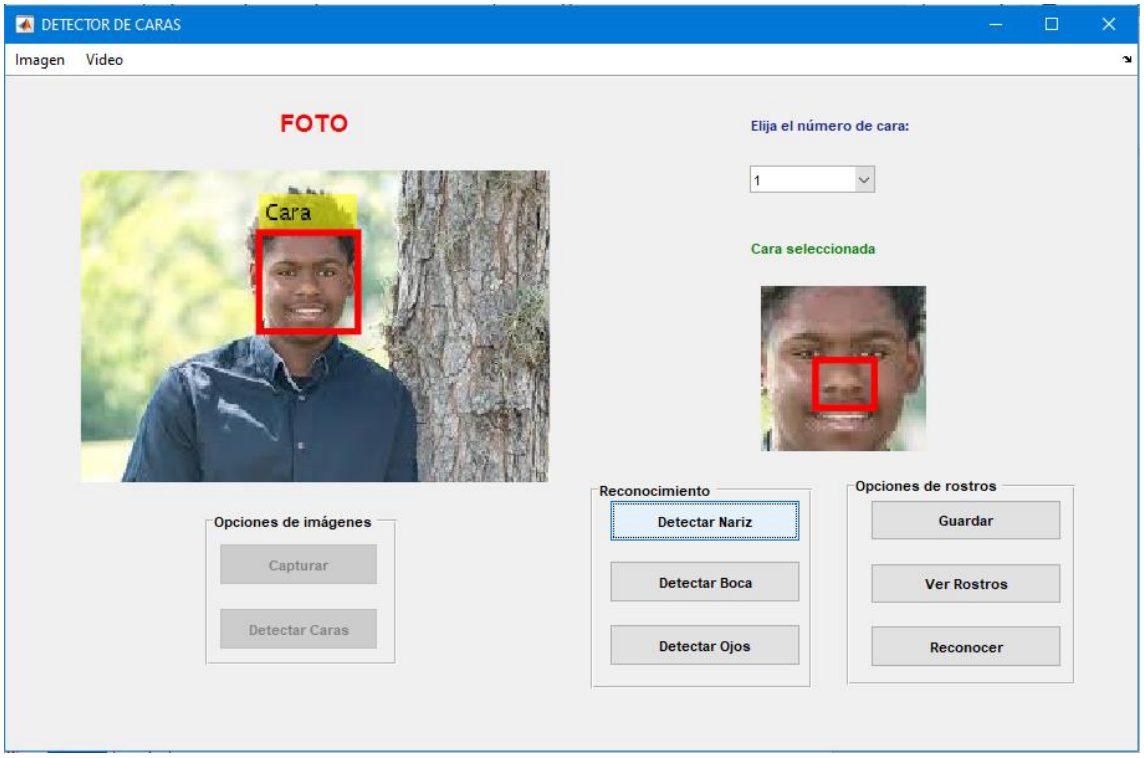


Ilustración 25: Detectar Nariz.

La variable num es global, correspondiente al valor del MergeThreshold y se le asignó previamente el valor de uno. El problema que se tuvo en esta opción es que, si la imagen de la cara seleccionada está muy pixeleada, no se va a poder localizar la nariz de la persona. Para ello habría que cambiar el valor del MergeThreshold.

4.2.5. Detección de boca

El código que se usó para la detección de boca fue el siguiente:

```
function btndetboca_Callback(hObject, eventdata, handles)
global imgr num;
boca=vision.CascadeObjectDetector('mouth','MergeThreshold',num);
BBN=step(boca,imgr);
if size(BBN,1)>0
axes(handles.imgcara);
max=1;
for i=1:size(BBN,1)
 if i>1
     if BBN(i,1)>BBN(i-1,1)
       max=i;
     end
```

```
 end

end
rectangle('Position',BBN(max,:),'LineWidth',4,'LineStyle','-
','EdgeColor','b');
else
    msgbox('No se ha detectado la boca','Detección');
end
```

En este caso se utiliza también el clasificador vision.CascadeObjectDetector, pero en esta ocasión se le envía el parámetro mouth que permite la detección de boca en las imágenes. Si detecta en la variable BBN se almacenará el valor de 1 y si no detecta el valor de 0.

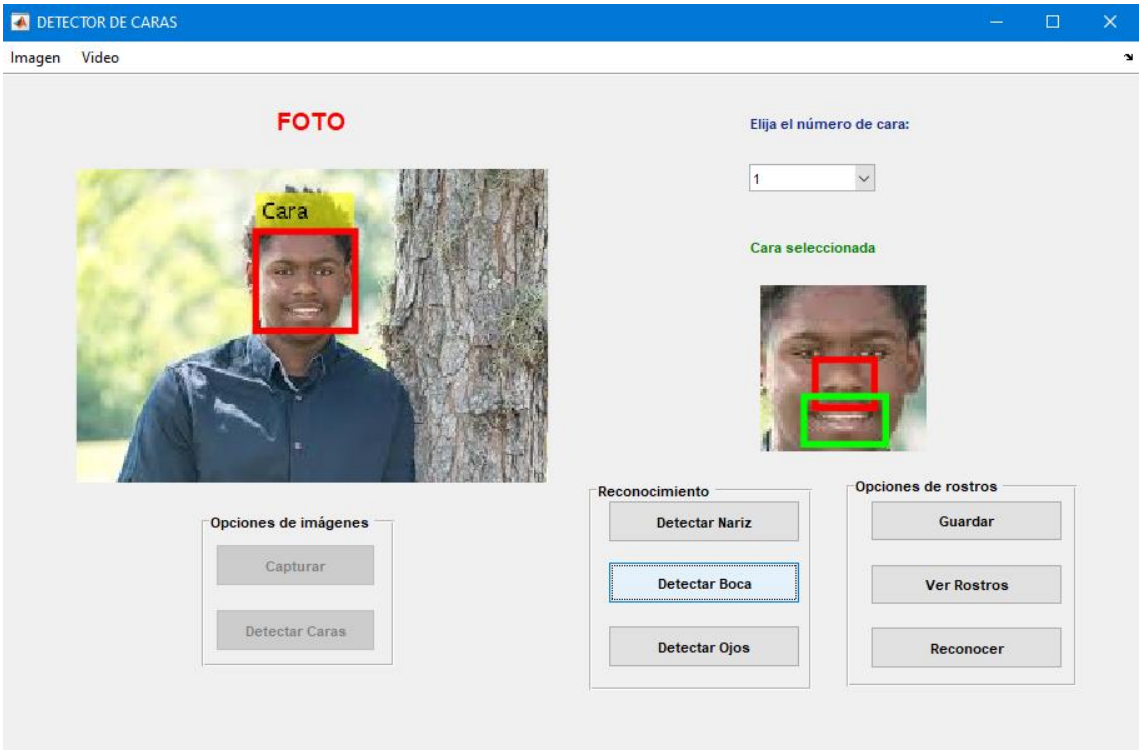


Ilustración 26: Detección de Boca.

En esta parte se pudo notar que en algunas imágenes se encierran partes que no corresponden a la boca de la persona. Para corregir eso, al momento de detectar la boca se coloca una condición, en donde el máximo valor corresponde a la boca de la persona.

4.2.6. Detección de ojos

Por último, se realizó la codificación para la detección de ojos. El código utilizado fue:

```
function btndetojos_Callback(hObject, eventdata, handles)
global imgr num;
Ojos = vision.CascadeObjectDetector ('EyePairBig','MergeThreshold',num
);
BBN = step (Ojos, imgr);
```

```
if size(BBN,1)>0
rectangle ( 'Position' , BBN, 'LineWidth' , 4, 'LineStyle' , '-'
,'EdgeColor' , 'b' );
else
     msgbox('No se ha detectado los ojos','Detección');
end
```

Para la detección de ojos se usa el clasificador visión.CascadeObjectDetector y se le envía el parámetro EyerPairBig para la detección del par de ojos que tiene una persona. El resultado que se pudo mostrar fue el siguiente:

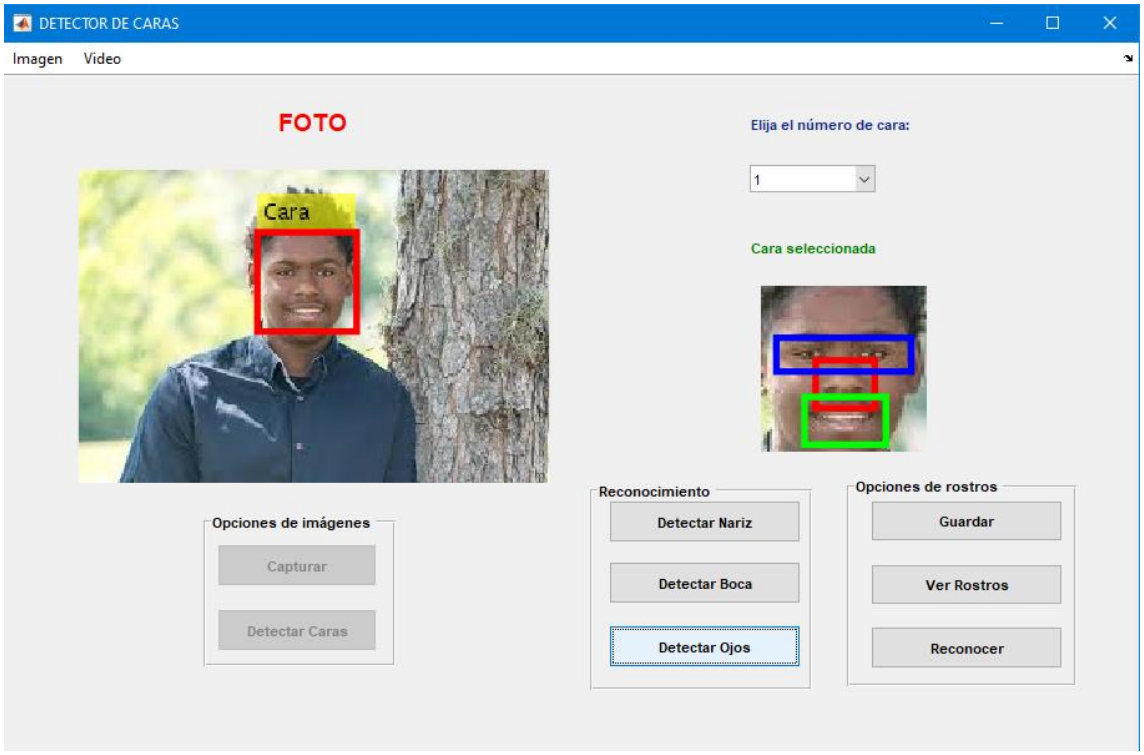


Ilustración 27: Detección de Ojos.

4.2.7. Guardar

Esta aplicación contiene una opción Guardar. Lo que permite esta opción es almacenar la cara detectada en una base da datos. Como se indicó anteriormente la base de datos es un archivo de formato mat en donde se almacenará los nombres y apellidos de la persona, así como la imagen de la cara a color con dimensiones 40 x 40 y la imagen de la cara en blanco y negro en 40 x 40. Esta imagen en blanco y negro posteriormente permitirá la comparación para el reconocimiento de persona.

```
function btnguardar_Callback(hObject, eventdata, handles)
global g imgr; %Ruta del directorio.
nm=inputdlg('Ingrese los nombres y apellidos:','Datos');
```

```
arch=exist(strcat(g,'bd.mat'));  %Comprueba si existe o no la base de
datos
if arch==0
    i=1;
else
    load(strcat(g,'bd.mat'));
    i=length(dper)+1;
end
dper(i)=nm;
imgper(:,:,:,i)=imresize(imgr,[40 40]);
imgperbn(:,:,i)=imresize(im2bw(imgr),[40 40]);
save(strcat(g,'bd.mat'),'dper','imgper','imgperbn');
msgbox('Se guardó un registro con éxito','Guardar');
```

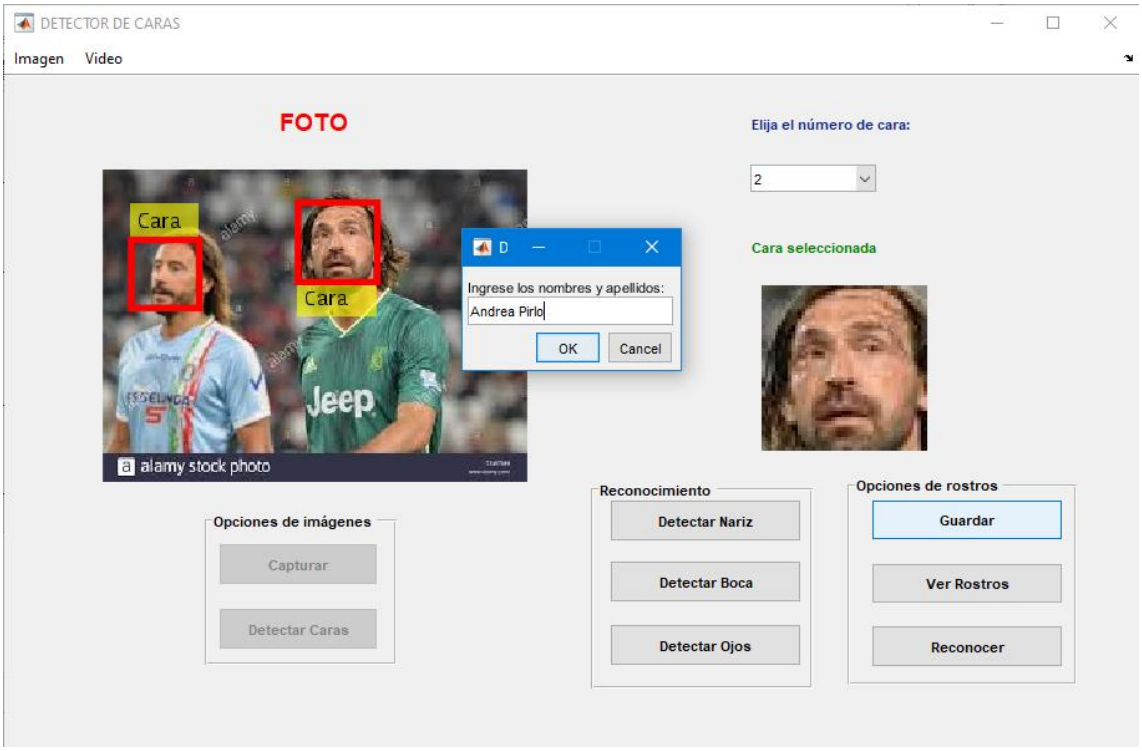


Ilustración 28: Guardar Rostros.

4.2.8. Ver Rostros

En esta opción el usuario tiene la posibilidad de visualizar los rostros de las personas que se han ido ingresando. La aplicación se ha realizado para visualizar un total de los 36 primeros rostros almacenados en la base de datos con los nombres de las personas ingresadas.

```
function btnverrostros_Callback(hObject, eventdata, handles)
global g;
arch=exist(strcat(g,'bd.mat'));  %Comprueba si existe o no la base de
datos
```

```
if arch==0
    msgbox('No existe imágenes en la base de datos','Mensaje');
else
     load(strcat(g,'bd.mat'));
     k=1;
     tam=length(dper);
     figure;
     set(gcf,'Name','Base de Datos de Imágenes Guardadas')
     for i=1:36
            if k>tam
               break;
            else
                subplot(5,5,k);
                imshow(imgper(:,:,:,i));
                title(dper(i),'FontSize',8);
            end
            k=k+1;
     end
end
```

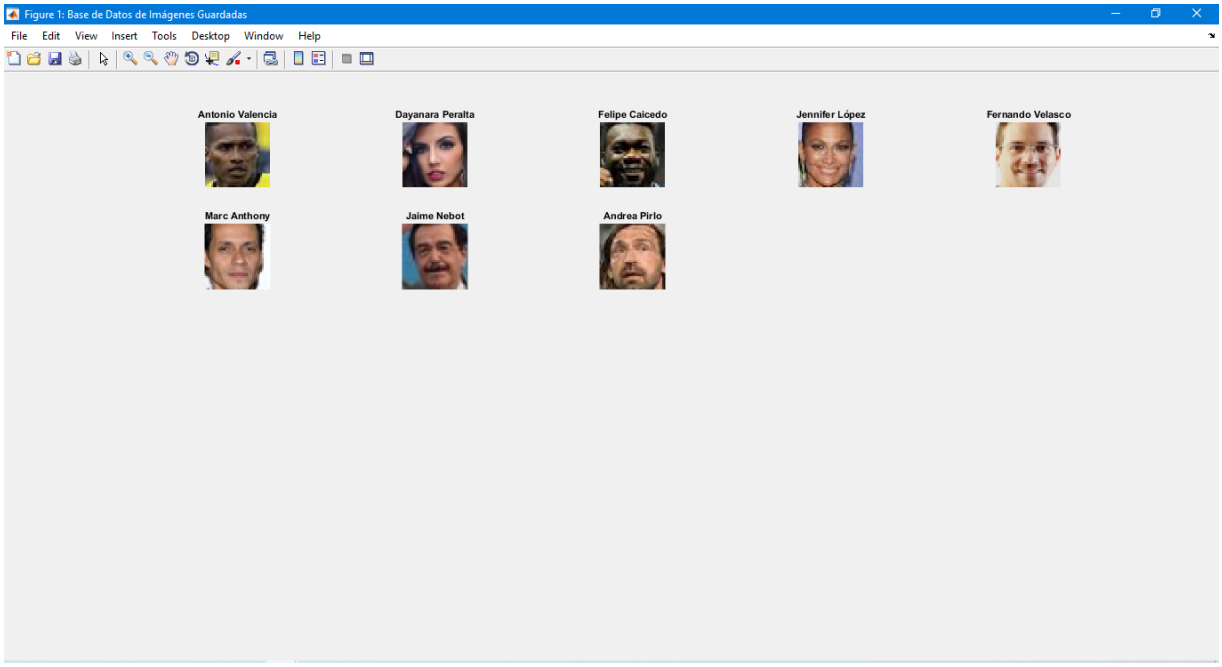


Ilustración 29: Ver Rostro.

En este caso se mostrarán los rostros de las personas que se han ingresado previamente,

4.2.9. Reconocer

La aplicación tiene la posibilidad de reconocer personas. El usuario debe leer una imagen con el rostro de una persona y a través de comparaciones con los todos los rostros almacenados en la base de datos, se obtendrá si el rostro fue detectado o no.

```
function btnreconocer_Callback(hObject, eventdata, handles)
global g imgr;
arch=exist(strcat(g,'bd.mat'));  %Comprueba si existe o no la base de
datos
if arch==0
    msgbox('No existe imágenes en la base de datos','Mensaje');
else
     imagen=imresize(im2bw(imgr),[40 40]);
     load(strcat(g,'bd.mat'));
     tam=length(dper);
     comp=[];
     for i=1:tam
      sem=corr2(imgperbn(:,:,i),imagen);
      comp=[comp sem];
     end
     comp=abs(comp);
     disp(comp);
     vd=find(comp==max(comp));
     if comp(vd)<0.30
         msgbox('No se ha reconocido la persona','Detección');
     else
        msgbox(strcat('La persona es:  ',dper(vd)),'Detección');
     end
end
```

La imagen que se desea reconocer es redimensionada a un tamaño de 40 x 40 y posteriormente transformada en imagen binaria. Luego se cargan todos los datos almacenados en la base de datos bd.mat. Después se verifica cuántas personas se han ingresado en la base de datos. Para esto se utiliza el comando length. Se inicializa la variable comp que es un vector donde se guardará el índice de semejanza de la imagen.

Se procede a comparar la imagen leída transformada en binario con las imágenes binarias almacenadas en la base de datos a través de una estructura para. La comparación se la realiza con el comando corr2. El índice de semejanza se almacena en la variable sem. Luego todos los valores obtenidos en sem se almacenan en el vector comp.

Después que se ha hecho el recorrido en todas las imágenes de la base de datos, se procede a sacar el valor absoluto del vector comp, es decir, transformar todos los valores obtenidos a positivos. Con el comando find localizo el índice de valor más alto en comp. Si el valor más alto es menor a 0.30 (valor que es muy bajo), se muestra un mensaje que no se ha reconocido a la persona, caso contrario se visualiza el nombre de la persona reconocida.

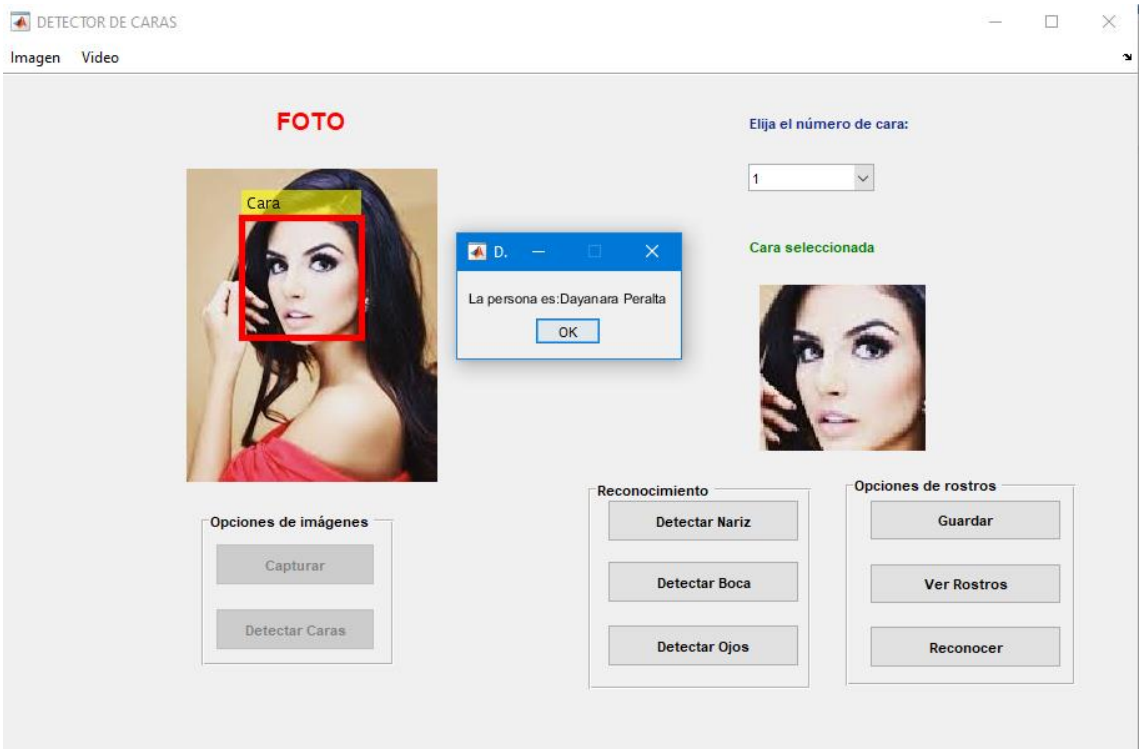


Ilustración 30: Detección.

CAPÍTULO 5

PRUEBAS Y RESULTADOS

5.1. Evaluación de la detección

La detección es una de las fases más relevantes de la aplicación, ya que una mala detección conlleva un error en el resto de las fases.

La aplicación implementa para la detección el algoritmo de Viola – Jones. Un requisito muy importante para que este algoritmo funcione es que los individuos deben aparecer em las imágenes de frente y con el rostro limpio, libre de elementos o accesorios para que la detección sea lo más exacta posible.

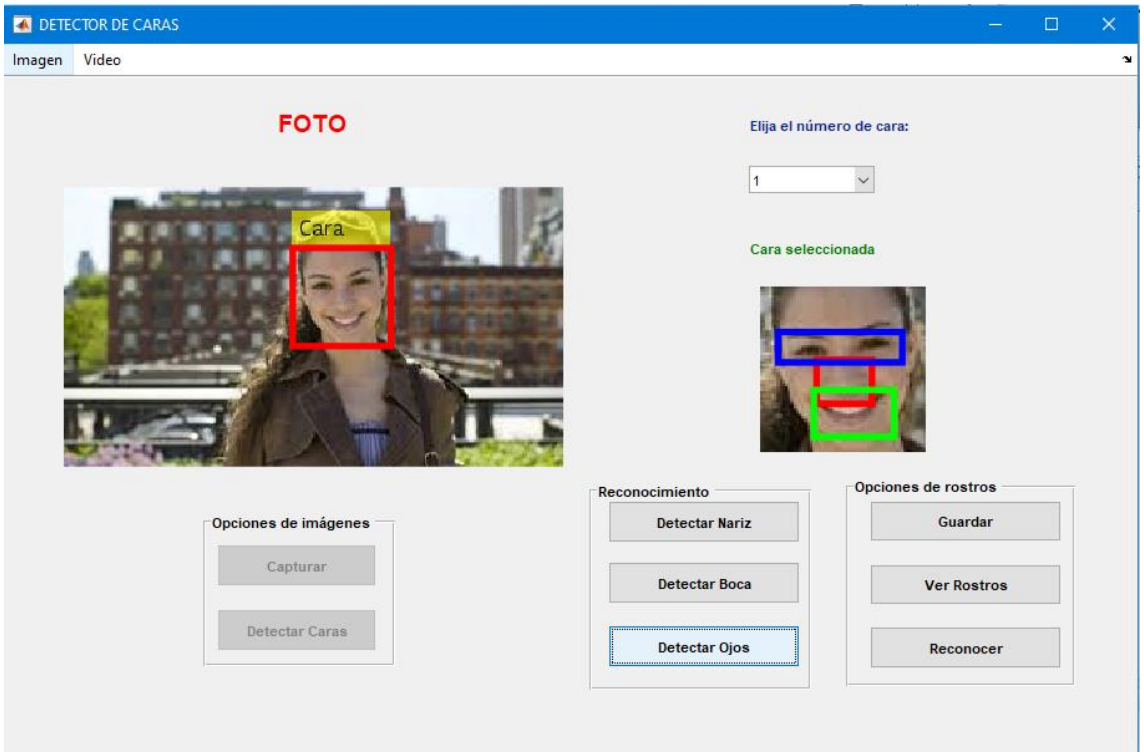


Ilustración 31: Rostro Detectado.

Existen casos especiales en los que el algoritmo no funciona. A continuación, se muestra algunos ejemplos de imágenes que no servirán ya que no es posible la detección de rostros, ojos, nariz y/o boca.

Uno de los inconvenientes que se obtiene es cuando el individuo no está de perfil. En este caso no se consigue detectar el rostro y por lo tanto no se pueden detectar las partes del rostro.

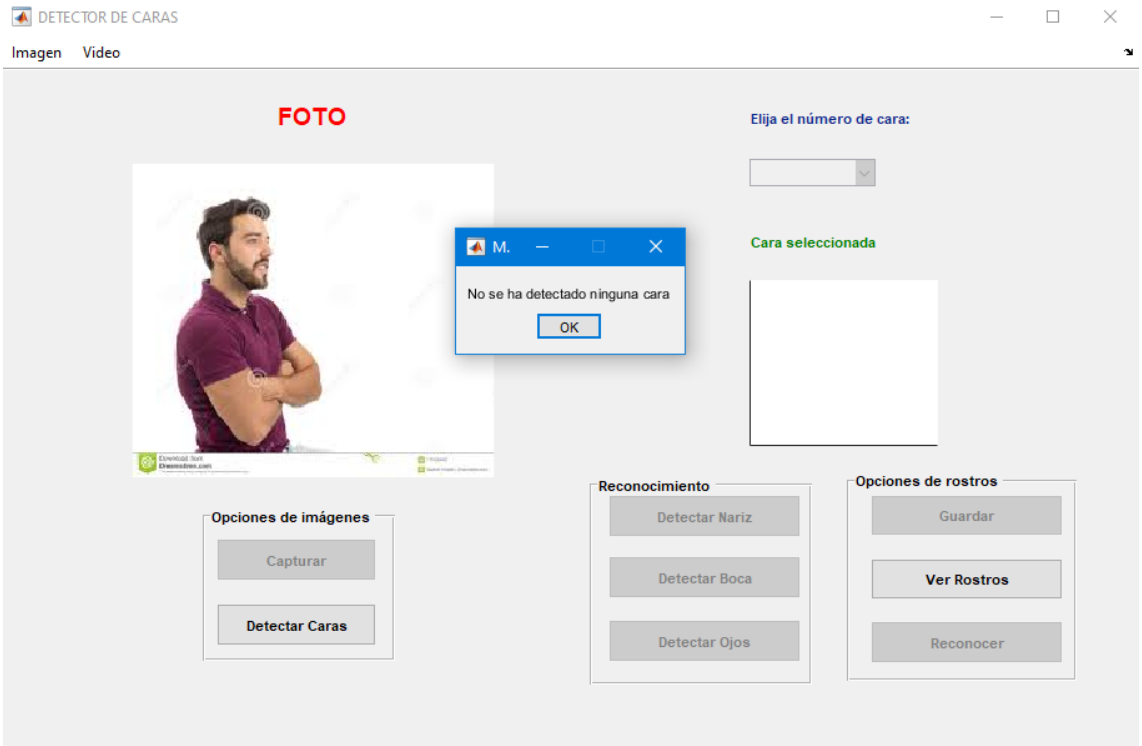


Ilustración 32: Rostro no detectado.

En la ilustración anterior, se muestra claramente un ejemplo donde el rostro de la persona no es reconocido. Esto se debe a que en la foto la persona no está de perfil o de frente, por lo cual no es detectado.

Otro problema es cuando el individuo posee algún elemento o accesorio tapando la parte de rostro. Es decir, cuando la persona aparece con lentes/gafas haciendo imposible detectar los ojos. En estos casos el algoritmo no funciona correctamente.

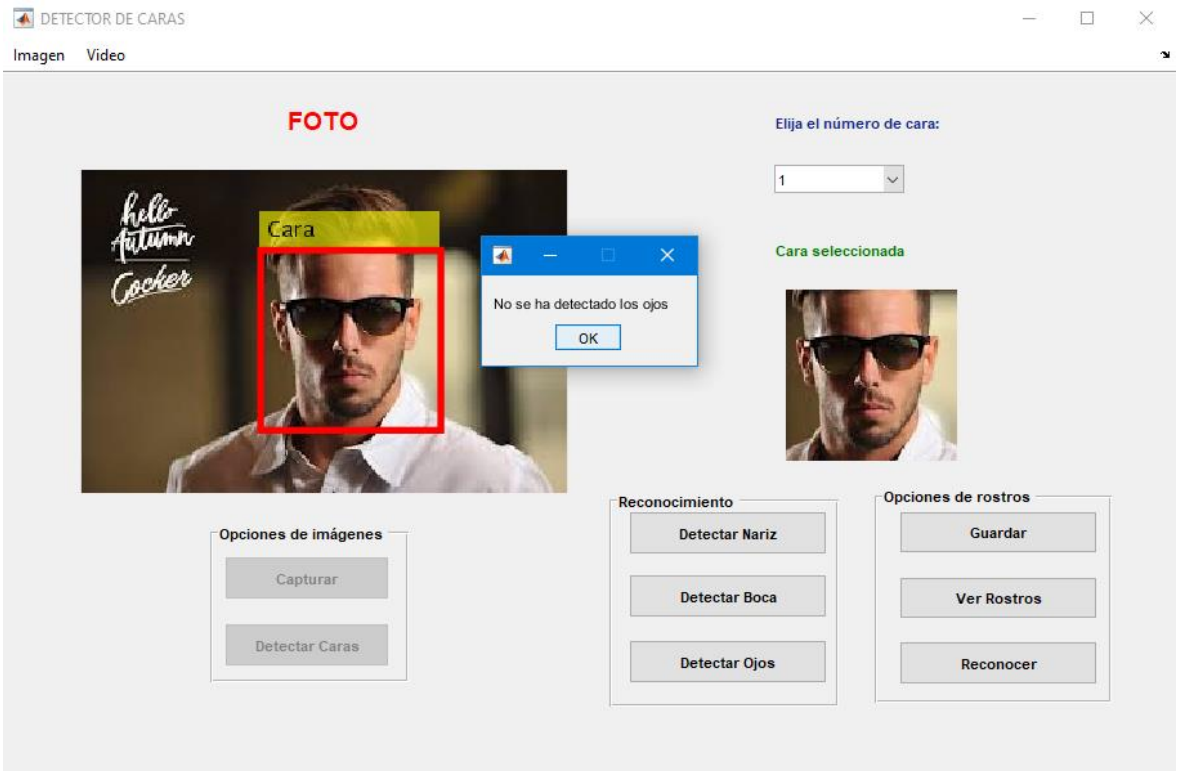


Ilustración 33: Ojos no reconocidos.

En la ilustración anterior, el rostro de la persona si es reconocido, pero al momento de extraer las características los ojos no son reconocidos. Esto se debe a que la persona está con gafas.

Otro inconveniente más para la detección de rostros es la iluminación. En el siguiente ejemplo se muestra la foto de una persona con poca iluminación en donde el rostro de la persona se hará imposible ser reconocida por la aplicación.

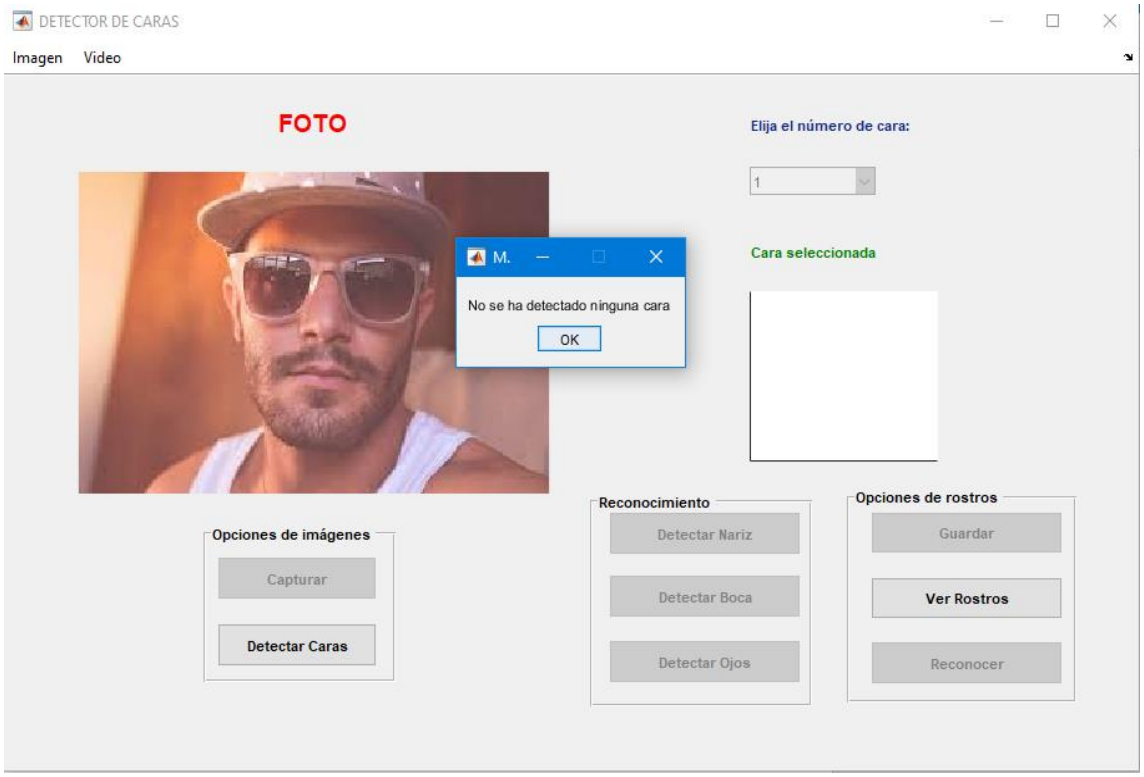


Ilustración 34: Rostro no reconocido por problema de iluminación.

Por eso antes de procesar las imágenes sería mejor evitar estos tipos de casos como los antes mencionados y mejorar los resultados.

5.2. Ejecución de pruebas y resultados obtenidos

Se realizaron pruebas de reconocimiento con tres bases de datos de fotografías. Para ello se han realizado diferentes pruebas tomando una muestra de 10 fotografías para cada caso, de manera aleatoria. Las imágenes fueron buscadas desde Google, descargadas y guardas en el computador.

En cada caso se va a evaluar cada una de las imágenes y se detallará el porcentaje de efectividad en cada caso de detección (cara, ojos y nariz).

5.2.1. Fotos con una sola persona

A continuación, se muestra una tabla con los resultados obtenidos en cada una de las pruebas.

Archivo	Detección de caras		Detección de nariz	Detección de boca	Detección de ojos
	Caras Detectadas	% Efectividad	% Efectividad	% Efectividad	% Efectividad
imagen1.jpg	1	100	0	0	0
imagen2.jpg	1	100	0	0	0
imagen3.jpg	1	100	100	100	100
imagen4.jpg	1	100	100	100	0
imagen5.jpg	1	100	100	100	0
imagen6.jpg	1	100	100	100	100
imagen7.jpg	1	100	100	100	100
imagen8.jpg	1	100	0	100	0
imagen9.jpg	1	100	100	0	100
imagen10.jpg	1	100	100	0	100
TOTAL EFECTIVIDAD:	**100%**		**70%**	**60%**	**50%**

Tabla 1: Resultados - Fotos con una sola persona.

5.2.2. Fotos con dos personas

A continuación, se muestra una tabla con los resultados obtenidos en cada una de las pruebas.

Archivo	Detección de caras		Detección de nariz	Detección de boca	Detección de ojos
	Caras Detectadas	% Efectividad	% Efectividad	% Efectividad	% Efectividad
imagen1.jpg	2	100	50	50	50

imagen2.jpg	2	100	0	50	0
imagen3.jpg	2	100	100	0	50
imagen4.jpg	2	100	100	50	0
imagen5.jpg	1	50	100	100	0
imagen6.jpg	2	100	50	50	0
imagen7.jpg	2	100	100	0	100
imagen8.jpg	2	100	0	0	0
imagen9.jpg	2	100	50	50	0
imagen10.jpg	2	100	50	50	0
TOTAL EFECTIVIDAD:	**95%**		**60%**	**40%**	**20%**

Tabla 2: Resultados - Fotos con dos personas.

5.2.3. Tres o más personas

A continuación, se muestra una tabla con los resultados obtenidos en cada una de las pruebas.

El número entre cada foto oscila entre mínimo 3 y un máximo de 10 personas.

Archivo	**Detección de caras**		**Detección de nariz**	**Detección de boca**	**Detección de ojos**
	Caras Detectadas	**% Efectividad**	**% Efectividad**	**% Efectividad**	**% Efectividad**
imagen1.jpg	3/4	75	0	0	0
imagen2.jpg	10/10	100	50	90	0
imagen3.jpg	3/8	37,50	0	0	0
imagen4.jpg	3/3	100	100	100	66,67

imagen5.jpg	3/6	50	0	0	0
imagen6.jpg	5/5	100	100	80	0
imagen7.jpg	3/6	50	0	0	0
imagen8.jpg	4/4	100	0	0	0
imagen9.jpg	3/5	60	0	66,67	0
imagen10.jpg	3/4	75	0	33,33	0
TOTAL EFECTIVIDAD:	**74,75%**		**25%**	**37%**	**6,67%**

Tabla 3: Resultados - Fotos con tres o más personas.

5.2.3.1. Resultados de la detección de ojos, nariz y boca.

Los resultados de la detección de las partes de la cara de una persona en las imágenes que contienen más de dos individuos se expresan a continuación, en donde 1 es el valor si se detectó la nariz, boca, etc. y 0 si es que no se la detectó:

Imagen1.jpg

	Detección de nariz	**Detección de boca**	**Detección de ojos**
Persona 1	0	0	0
Persona 2	0	0	0
Persona 3	0	0	0
Persona 4	No se reconoció la cara		
% Efectividad	0/3	0/3	0/3

Tabla 4: Resultados imagen1.jpg.

Imagen2.jpg

	Detección de nariz	**Detección de boca**	**Detección de ojos**
Persona 1	1	1	0
Persona 2	1	1	0
Persona 3	0	1	0
Persona 4	1	1	0
Persona 5	0	0	0

Persona 6	1	1	0
Persona 7	0	1	0
Persona 8	1	1	0
Persona 9	0	1	0
Persona 10	0	1	0
% Efectividad	5/10	9/10	0/10

Tabla 5: Resultados imagen2.jpg.

Imagen3.jpg

	Detección de nariz	**Detección de boca**	**Detección de ojos**
Persona 1	0	0	0
Persona 2	0	0	0
Persona 3	0	0	0
Persona 4	No se reconoció la cara		
Persona 5	No se reconoció la cara		
Persona 6	No se reconoció la cara		
Persona 7	No se reconoció la cara		
Persona 8	No se reconoció la cara		
% Efectividad	0/3	0/3	0/3

Tabla 6: Resultados imagen3.jpg.

Imagen4.jpg

	Detección de nariz	**Detección de boca**	**Detección de ojos**
Persona 1	1	1	1
Persona 2	1	1	0
Persona 3	1	1	1
% Efectividad	3/3	3/3	2/3

Tabla 7: Resultados imagen4.jpg.

Imagen5.jpg

	Detección de nariz	**Detección de boca**	**Detección de ojos**
Persona 1	0	0	0
Persona 2	0	0	0
Persona 3	0	0	0
Persona 4	No se reconoció la cara		
Persona 5	No se reconoció la cara		
Persona 6	No se reconoció la cara		
% Efectividad	0/3	0/3	0/3

Tabla 8: Resultados imagen5.jpg.

Imagen6.jpg

	Detección de nariz	Detección de boca	Detección de ojos
Persona 1	100	100	0
Persona 2	100	0	0
Persona 3	100	100	0
Persona 4	100	100	0
Persona 5	100	100	0
% Efectividad	5/5	4/5	0/5

Tabla 9: Resultados imagen6.jpg.

Imagen7.jpg

	Detección de nariz	Detección de boca	Detección de ojos
Persona 1	0	0	0
Persona 2	0	0	0
Persona 3	0	0	0
Persona 4	No se reconoció la cara		
Persona 5	No se reconoció la cara		
Persona 6	No se reconoció la cara		
% Efectividad	0/3	0/3	0/3

Tabla 10: Resultados imagen7.jpg

Imagen8.jpg

	Detección de nariz	Detección de boca	Detección de ojos
Persona 1	0	0	0
Persona 2	0	0	0
Persona 3	0	0	0
Persona 4	0	0	0
% Efectividad	0/4	0/4	0/4

Tabla 11: Resultados imagen8.jpg

Imagen9.jpg

	Detección de nariz	Detección de boca	Detección de ojos
Persona 1	0	0	0
Persona 2	0	1	0
Persona 3	0	1	0
Persona 4	No se reconoció la cara		

Persona 5	No se reconoció la cara		
% Efectividad	0/3	2/3	0/3

Tabla 12: Resultados imagen9.jpg

Imagen10.jpg

	Detección de nariz	**Detección de boca**	**Detección de ojos**
Persona 1	0	1	0
Persona 2	0	0	0
Persona 3	0	0	0
Persona 4	No se reconoció la cara		
% Efectividad	0/3	1/3	0/3

Tabla 13: Resultados imagen10.jpg.

5.2.4. Resultados de las pruebas de reconocimiento de rostros

En los siguientes gráficos se muestra en forma resumida el porcentaje de acierto entre los diferentes métodos para una misma cantidad de fotos en la base, y como se modifica según se incrementa la cantidad de personas en la imagen:

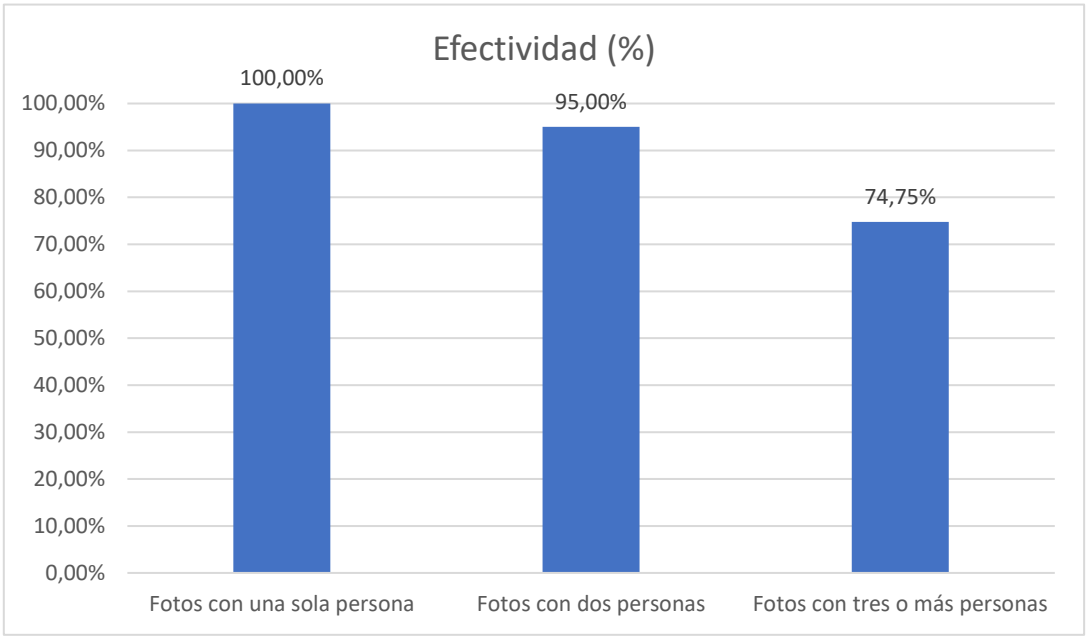


Ilustración 35: Resultados reconocimiento de rostros.

En el gráfico se muestra que mientras van aumentando el número de personas en la foto la efectividad de reconocimiento va disminuyendo. Cabe recalcar que en cada uno de los casos se utilizó personas que estaban con la mirada de frente.

En el primer caso, en las fotos donde aparece una persona, el porcentaje de efectividad de reconocimiento es del 100%, es decir, en todas las imágenes cargadas fue detectado el rostro de la persona.

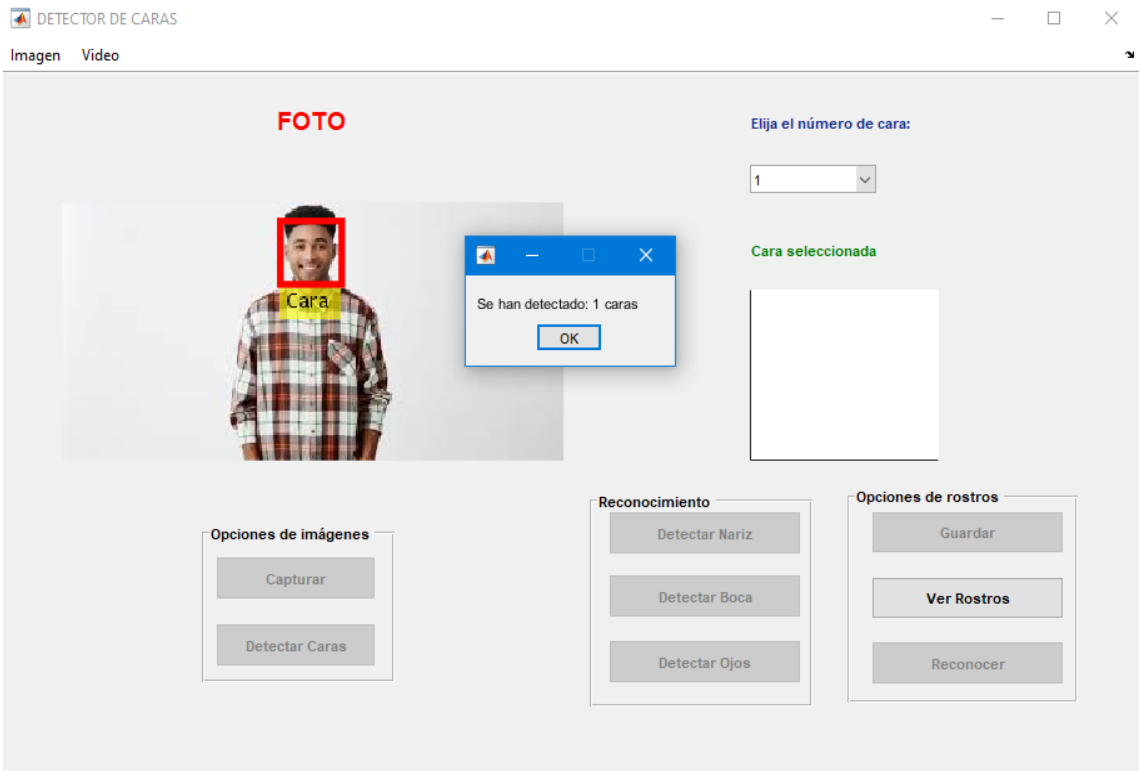


Ilustración 36: Ejemplo de detección de rostro.

En el segundo caso, en donde aparecen dos personas, hubo una efectividad de reconocimiento del 95%, es decir de 10 fotografías cargadas, en una no detectó el rostro de una persona. De acuerdo con las pruebas realizadas este inconveniente se suscitó en la imagen cuyo nombre de archivo es imagen5.jpg. La siguiente figura muestra el resultado que devuelve la aplicación al momento de realizar la detección de caras:

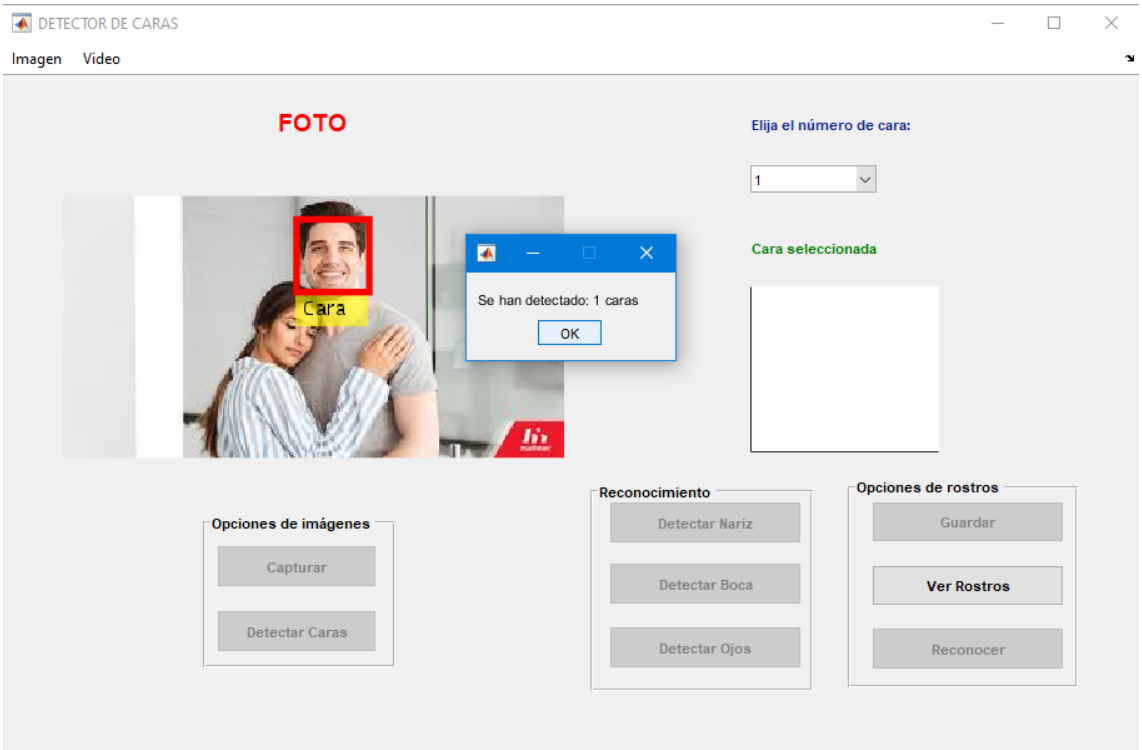


Ilustración 37: Detección imagen5.jpg.

En este caso la aplicación nos indica que existe una cara cuando en realidad en la imagen existen dos personas. En este ejemplo se puede notar muy claramente que la persona detectada está con la mirada al frente y la cara de la mujer, la cual no es reconocida por la aplicación, su rostro está apegado al cuerpo del hombre, motivo por lo que hace imposible su reconocimiento.

En el tercer caso donde la imagen tiene más de dos personas, se observa que el porcentaje de reconocimiento de rostro baja notoriamente. El porcentaje de efectividad en este caso es del 74,75%

Ilustración 38: Reconocimiento fotos más de dos personas.

En el caso de la ilustración anterior se puede notar que sólo se detectan el 50% de las caras es decir 3/6. El motivo por el cuál las caras no se detectan es porque las personas no están de perfil o mirada al frente. En otros casos se pudo notar que el reconocimiento de rostros depende de la resolución de la foto, ya que, a mayor resolución, mayor porcentaje de efectividad de reconocimiento.

5.2.5. Raza

La aplicación es capaz de detectar personas de razas afro, mestizas y blancas. Se realizaron pruebas a 20 imágenes y los resultados que se tuvieron fueron los siguientes:

IMAGEN	RAZA RESULTANTE	RAZA REAL	% ACIERTO
Imagen 1	Afro	Afro	100
Imagen 2	Blanca	Blanca	100
Imagen 3	Mestiza	Blanca	0

Imagen 4	Mestiza	Mestiza	100
Imagen 5	Blanca	Blanca	100
Imagen 6	Mestiza	Afro	0
Imagen 7	Blanca	Blanca	100
Imagen 8	Blanca	Blanca	100
Imagen 9	Afro	Afro	100
Imagen 10	Mestiza	Blanca	0
Imagen 11	Mestiza	Afro	0
Imagen 12	Mestiza	Mestiza	100
Imagen 13	Mestiza	Mestiza	100
Imagen 14	Mestiza	Blanca	0
Imagen 15	Mestiza	Mestiza	100
Imagen 16	Afro	Blanca	0
Imagen 17	Blanca	Blanca	100
Imagen 18	Mestiza	Blanca	0
Imagen 19	Mestiza	Blanca	0
Imagen 20	Blanca	Blanca	100

Tabla 14: Prueba de las razas.

		VALOR RESULTANTE		
		BLANCO	**MESTIZO**	**AFRO**
VALOR REAL	**BLANCO**	6	5	1
	MESTIZO	0	4	0

	AFRO	0	2	2

Tabla 15: Resultados de las pruebas.

Como se puede observar en la tabla anterior, hubo un 50% de acierto en la detección de personas de raza blanca, mientras que el otro 50% fue de detecciones erróneas. De hecho, una persona de raza blanca fue detectada como afro. Esto se debe a que como sólo se tomaba una parte al azar de la piel de la persona para la comparación, en el caso erróneo la parte que se tomó pudo ser parte del cabello negro de la persona, y por eso muestra esos resultados.

En el caso de las personas mestizas fue en cambio hubo un 100% de acierto y en la detección de personas afros un 50% de acierto y un 50% de desacierto.

5.2.6. Resultados de las pruebas de reconocimiento de ojos, nariz y boca

Después de haber realizado las pruebas con diferentes números de rostros, en cuanto al reconocimiento de nariz, boca y ojos se pudo obtener los siguientes resultados:

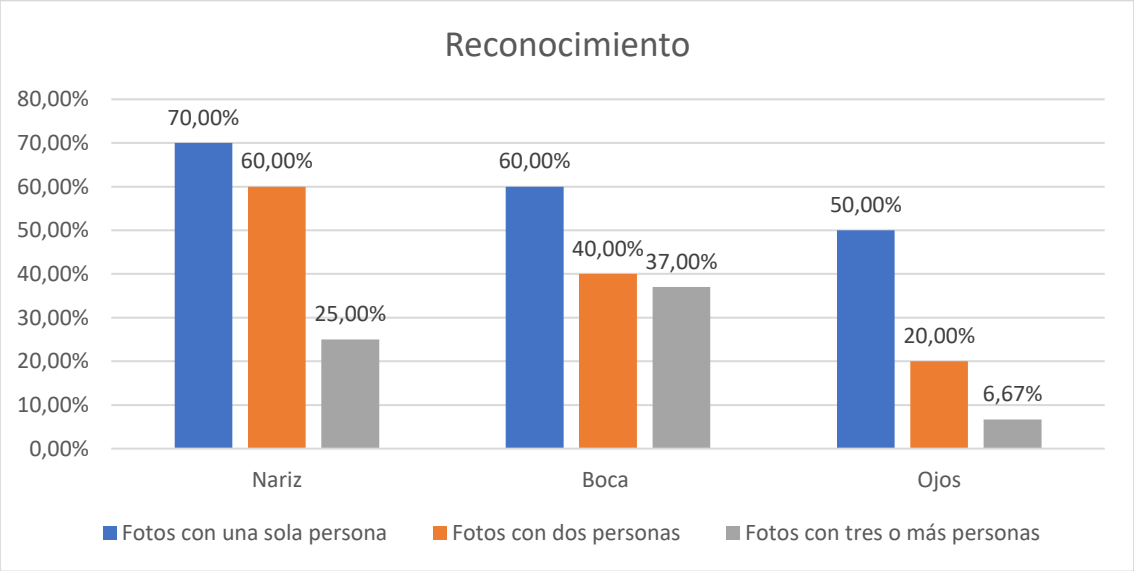


Ilustración 39: Resultados de las pruebas de reconocimiento.

El gráfico detalla la efectividad en cada uno de los casos. Se puede observar que la característica del rostro que tiene mayor efectividad de reconocimiento es la nariz. A medida que va aumentando el número de personas en las imágenes, este porcentaje va disminuyendo. Lo mismo se pudo notar en la detección de boca. En donde se pudo notar

problemas fue en la detección de los ojos en donde los valores de efectividad dieron demasiado bajo en cada uno de los casos, incluso cuando se aumenta el número de personas este valor no llega ni al 10%. El problema que se pudo observar es que el porcentaje según el número de personas va disminuyendo. Esto se origina porque al extraer el rostro de la persona, la imagen del rostro se pixelea, haciendo imposible que se detecte, en algunos casos las partes del rostro de la persona. En los ojos en cambio ocurre que hay más problema porque la detección de los ojos depende notablemente de la alineación de los ojos. Si uno de los ojos está más arriba o más abajo que el otro no se detectará.

5.2.7. Resultados de las pruebas de reconocimiento de personas

Para la realización de esta prueba se almacenó un total de 8 imágenes de cantantes y futbolistas en la base de datos. Las imágenes fueron elegidas al azar. Después de haber realizado las pruebas necesarias, se obtuvieron los siguientes resultados:

	Pers. 1	**Pers. 2**	**Pers. 3**	**Pers. 4**	**Pers. 5**	**Pers. 6**	**Pers. 7**	**Pers. 8**	**Reconoc.**
Pers.1	0.1390	0.0247	0.0455	0.0120	0.0693	0.0226	0.0409	0.0283	SI (P1)
Pers.2	0.1745	0.5322	0.1829	0.1738	0.0155	0.1600	0.2935	0.3549	SI (P2)
Pers.3	0.1829	0.3842	0.0360	0.0623	0.1904	0.0036	0.2853	0.2924	NO
Pers.4	0.1055	0.3855	0.1138	0.2573	0.1135	0.2596	0.4226	0.3648	NO
Pers.5	0.0590	0.0416	0.3510	0.1107	0.1638	0.1740	0.1102	0.0676	NO
Pers.6	0.0277	0.0705	0.1879	0.2176	0.2101	0.4120	0.0141	0.0169	SI (P6)
Pers.7	0.1087	0.2144	0.2294	0.0207	0.2410	0.0705	0.1467	0.1639	NO
Pers. 8	0.1697	0.4819	0.2804	0.2662	0.0306	0.2818	0.3787	0.4425	NO

Tabla 16: Índice de reconocimiento.

En la tabla anterior se pueden notar las pruebas y los resultados obtenidos con la evaluación de 8 personas. De acuerdo con los resultados en tres casos se pudo reconocer

a la persona, mientras que en 5 casos no se pudo obtener el reconocimiento. Cabe recalcar que en todos los casos la aplicación toma el índice más alto de comparación. Por ejemplo, en el primer caso el índice más alto es 0.1390 correspondiente a la persona 1, es decir que el programa visualizará el nombre de la persona que esté en el lugar número uno de la base de datos. En cambio, en el lado negativo se tiene en la evaluación de la persona 4 puesto que en este caso el índice más elevado es 0.4226 correspondiente a la persona 7 lo cuál no es aceptable. En este caso el índice de la persona correcta es 0.2573 que está muy por debajo del índice más alto.

Cabe recalcar que la evaluación se hizo con un mínimo de fotos almacenados en la base de datos, una foto por cada persona. Por esto se obtienen estos resultados. Para mejor la evaluación se debería tener más fotos de la misma persona almacenada en la base de datos para que la aplicación tenga una mayor posibilidad de reconocer a la persona.

El siguiente gráfico muestra el porcentaje de efectividad de reconocimiento de personas:

Ilustración 40: Reconocimiento de personas.

CAPÍTULO 6

CONCLUSIONES Y TRABAJOS FUTUROS

6.1. Conclusiones

El objetivo de este proyecto fue profundizar en el estudio de las técnicas que permiten el reconocimiento de caras y sus partes, así como también la identificación de las personas que estén en la imagen.

Se ha desarrollado una aplicación de reconocimiento de caras que solucione la detección de la cara dentro de una imagen, la extracción de las características de la cara y finalmente el reconocimiento de la persona.

Para la realización de este proyecto se han usado imágenes estáticas e imágenes en movimiento a través de la cámara web. Se ha trabajo con imágenes a color en modelo RGB (Red, Green y Blue) y en imágenes binarias para el proceso de comparación de rostros. Debido a la imposibilidad de mostrar las pruebas de videos en el informe, sólo se han expuesto los resultados que tienen que ver con imágenes.

En este proyecto se ha concluido que el tipo de imágenes que se utilizan en la aplicación del sistema de reconocimiento, así como la posición de las personas, iluminación, entre otras cosas, influye de manera directa en los resultados que se obtienen. Imágenes donde la persona no esté de perfil no son detectadas, así como cuando la persona posee lentes o gafas en algunos casos los ojos no son detectados y también el ángulo de posición de los ojos también influye mucho en la detección.

6.2. Trabajos futuros

Para mejorar lo investigado en este proyecto, se proponen posibles trabajos a futuros. Este trabajo al tratarse de una aplicación que trabaja con funciones, se puede cambiar fácilmente las técnicas empleadas, ya sea con el propósito de comparar los resultados o buscar alguna mejoría.

Uno de los puntos que se podría mejorar es la detección en el aspecto de que la persona no aparezca de frente y completamente con el rostro limpio, puesto que como se observa en los resultados este sistema tiene limitaciones. De esta manera se propondría a que se creen sistemas con la posibilidad de que reconozcan a las personas en otras poses, por ejemplo, que no esté de perfil o con elementos que obstaculicen la cara.

Otra opción que se propone para trabajos futuros es el de desarrollar una aplicación de rostros que permita la identificación de usuarios, como por ejemplo docentes universitarios con la finalidad que tengan acceso al sistema (notas estudiantes, registro de faltas, entre otros).

Finalmente, algo que se puede añadir en trabajos que se realicen a futuro, es llevar a cabo una interfaz gráfica en donde se pueda trabajar en tiempo real, una aplicación que sirva para video vigilancia y reconocimiento facial. La aplicación actual también permite para trabajar con una cámara web de una computadora, lo que permitiría modificar muy fácilmente para trabajar con una cámara ya instalada en una casa o en un trabajo, permitiendo evaluar el sistema de manera dinámica.

Referencias

[1] B. Moghaddam, W. Wahid y A. Pentland, «Beyond eiggenface: probabilistic matching for face recognition.,» *Proceedings of the second international conference on automatic face and gesture recognition,* pp. 30-35, 1998.

[2] F. Cambell y Robson, «Application of Fourier analysis to the visibility of grating. Journal the Psicology.,» pp. 551-566, 1998.

[3] A. John, Hartigan y A. Manchek, «Algorithm AS 136: A k-means clustering algorithm,» *Journal of the Royal Statistical Society.,* pp. 100-108, 1979.

[4] F. Suárez y J. Sayago, «Mapas auto organizativo para la asistencia médica oftalmológica.,» *Tecnología informática y comunicaciones.,* 2018.

[5] J. Oropeza, M. Nakano y H. Pérez, «Reconocimiento de rostros.,» 2004.

[6] L. Sirovich y M. Kirby, «Low-dimensional procedure for the characterization of human faces,» *Journal of the optical society of America a-optics image,* 1987.

[7] M. A. Turk y A. P. Pentland, «Face recognition using eigenfaces. In Computer Vision and Pattern Recognition, Proceedings,» *Computer Society Conference,* 1991.

[8] B. Moghaddam y A. Pentland, «Probabilistic Visual Learning for Object Representation.,» *IEEE Transactions on Pattern Analysis and Machine.,* 1997.

[9] J. Calli Olvea, «Reconocimiento facial basado en el algoritmo eigenface,» 2015.

[10] «Descripción general de los conceptos de la detección de rostros | Firebase,» [En línea]. Available: https://firebase.google.com/docs/ml-kit/face-detection-concepts?hl=es-419.

[11] R. Trapiella Pino, «Desarrollo y evaluación de técnicas de visión artificial aplicadas a un problema de reconocimiento facial,» 2017.

[12] J. R. Valvert Gamboa, «Métodos y técnicas de reconocimiento de rostros en imágenes digitales bidimensionales,» Guatemala, 2006.

[13] D. Calvo, «Red Neuronal Recurrente – RNN,» [En línea]. Available: http://www.diegocalvo.es/red-neuronal-recurrente/.

[14] M. Kirby y L. Sirovich, «Application of the karhunen-loeve procedure for the characterization of human faces.,» 1990.

[15] A. Turk y A. Pentland, Eigenfaces for recognition. Vision and modeling group., Massachusetts Institute of technology..

[16] F. De la Torre y M. Black, «Robust principal component analysis for computer vision.,» de *Computer Vision*, 2001.

[17] Yal, «Base de Datos Facial.,» [En línea]. Available: http://cvc.yale.edu/projects/yalefaces/yalefaces.html.

[18] E. Parra Barrero, «Aceleración del algoritmo de Viola - Jones mediante rejillas de procesamiento masivamente paralelo en el plano focal.,» Sevilla, 2015.

[19] MATLAB, «MATLAB,» [En línea]. Available: https://www.mathworks.com/help/vision/ref/vision.cascadeobjectdetector-system-object.html.

[20] MATLAB, «YOLO MATLAB,» [En línea]. Available: https://la.mathworks.com/help/vision/ug/yolo-v2-basics.html.

[21] R. Trapiella Pino y C. Cerrada Somolinos, «Desarrollo y evaluación de técnicas de visión artificial aplicadas a un problema de reconocimiento facial.».

[22] C. J. Pardo Herrera, «HOG Histograma de Gradientes Orientados,» [En línea]. Available: https://carlosjuliopardoblog.wordpress.com/contact/.

[23] MATLAB, «KLT Algorithm,» [En línea]. Available: https://la.mathworks.com/help/vision/examples/face-detection-and-tracking-using-the-klt-algorithm.html.

[24] «https://osl.ull.es/software-libre/opencv-libreria-vision-computador/,» [En línea].