FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

HERRAMIENTA DE RECONOCIMIENTO FACIAL CON TÉCNICA DE VISIÓN COMPUTACIONAL 2D

AUTOR

JEFFERSON DAVID ESCOBAR LOAYZA

AÑO

2019

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

HERRAMIENTA DE RECONOCIMIENTO FACIAL CON TÉCNICA DE VISIÓN COMPUTACIONAL 2D.

Trabajo de Titulación presentado en conformidad con los requisitos establecidos para optar por el título de Ingeniero en Electrónica y Redes de la Información.

Profesor Guía

PhD. Wilmar Hernández Perdomo

Autor

Jefferson David Escobar Loayza

Año

2019

DECLARACIÓN DEL PROFESOR GUÍA

“Declaro haber dirigido el trabajo, Herramienta de reconocimiento facial con técnica de visión computacional 2D, a través de reuniones periódicas con el estudiante Jefferson David Escobar Loayza, en el semestre 201910, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación”

_____

Wilmar Hernández Perdomo

Doctor Ingeniero en Electrónica

CI: 0151721016

DECLARACIÓN DEL PROFESOR CORRECTOR.

“Declaro haber revisado este trabajo, Herramienta de reconocimiento facial con técnica de visión computacional 2D, de Jefferson David Escobar Loayza en el semestre 201910, dando cumplimiento a todas las disposiciones vigentes que regalan los Trabajos de Titulación”

_____

Héctor Fernando Chinchero Villacis

Máster en Domótica y Hogar Digital

CI: 1715451330

DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE

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

_____

Jefferson David Escobar Loayza

CI: 2100623475

AGRADECIMIENTOS

Agradezco al Creador Divino por la fortaleza dada y por haberme permitido alcanzar uno de mis grandes sueños como es, mi formación profesional.

A mi familia por su compresión y apoyo incondicional ofrecido en todo momento.

A mis padres y hermanos, quienes fueron mi soporte, en estos cinco años de preparación.

DEDICATORIA

Dedico este trabajo en primera, instancia a todos los docentes que contribuyeron en mi formación académica, quienes con su amplio bagaje de conocimientos han hecho que se replique y fortalezca cada aprendizaje en mí.

En segunda instancia, a mis abnegados padres Julio César y Lillian Axuliadora, guías e impulsadores en mi formación profesional.

En tercera instancia, a mis hermanos Jonathan Adrián y Kevin Santiago por estar conmigo estos cinco años.

RESUMEN

En esta tesis se ha diseñado un sistema de reconocimiento de imágenes por visión artificial, centrado en rostros de personas. Aquí, se ha estudiado la complejidad de dos algoritmos que han demostrado su eficiencia y robustez en aplicaciones de reconocimiento de rostros usando visión por computador. Además, se ha implementado un sistema con tal propósito basado en dichos algoritmos y la propuesta de mejoras en los mismos.

Los algoritmos mencionados anteriormente son: 1) El método Autorostros (Eigenfaces) y 2) El método Análisis Discriminante Lineal (Fisherfaces). Ambos métodos son capaces de detectar rostros por comparación con imágenes guardadas en bases de datos, y entre las características fundamentales de dichos métodos podemos destacar lo siguiente: el método de Autorostros se encarga de comparar con una única imagen de la persona que sirve de referencia y que se encuentra guardados en la base de datos; mientras que el método Análisis Discriminante Lineal realiza la comparación contra un promedio de imágenes de referencia guardadas en la base de datos.

En esta tesis se ha podido demostrar experimentalmente que el primer método necesita menos tiempo de procesamiento, pero es menos robusto; mientras que el segundo método requiere un tiempo de procesamiento más alto, pero es más robusto. Por lo que, este último es el apropiado para trabajar en entornos afectados por la contaminación y señales no deseadas.

Con los antecedentes expuestos, con el objeto de mejorar la robustez del método Autorostros, se propone crear una base de datos con un conjunto de fotos de referencia de cada uno de los rostros a identificar, buscar una medida de la tendencia central de dichas fotos para cada rostro y tomar esta medida como la nueva referencia. Sin embargo, incluso haciendo esto, los resultados siguen siendo inferiores al método Análisis Discriminante Lineal. Por lo tanto, si se desea un sistema de identificación rápido y se tolera un determinado margen de error, el método recomendado es el Autorostros. Por otra parte, si se desea implementar un sistema de procesamiento robusto, el método apropiado es el Análisis Discriminante Lineal.

ABSTRACT

In this thesis an image recognition system has been designed by artificial vision, centered on people's faces. Here, we have studied the complexity of two algorithms that have demonstrated their efficiency and robustness in face recognition applications using computer vision. In addition, a system has been implemented for such purpose based on said algorithms and the proposal of improvements in them.

The algorithms mentioned above are: 1) The method Autorostros (Eigenfaces) and 2) The method of Linear Discriminant Analysis (Fisherfaces). Both methods are able to detect faces by comparison with images stored in databases, and among the fundamental characteristics of these methods we can highlight the following: the method of Autorostros is responsible for comparing with a single image of the person who serves as a reference and that is stored in the database; while the Linear Discriminant Analysis method performs the comparison against an average of reference images stored in the database.

In this thesis it has been possible to demonstrate experimentally that the first method needs less processing time, but is less robust; while the second method requires a longer processing time, but is more robust. Therefore, the latter is appropriate for working in environments affected by pollution and unwanted signals.

With the above background, in order to improve the robustness of the method Autorostros, it is proposed to create a database with a set of reference photos of each of the faces to identify, find a measure of the central tendency of these photos to each face and take this measure as the new reference. However, even doing this, the results are still inferior to the Linear Discriminant Analysis method. Therefore, if a rapid identification system is desired and a certain margin of error is tolerated, the recommended method is the Autorostros. On the other hand, if you want to implement a robust processing system, the appropriate method is Linear Discriminant Analysis.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 INTRODUCCIÓN

La especie humana tiene aproximadamente (entre sus 260.000 y 350.000 años de existencia) (Brink, 1987) se han empleado las características físicas para identificar individuos de su entorno social, estas características físicas o rasgos particulares de cada uno, son las que permiten la identificación o diferenciación entre unos y otros.

En este sentido, la ciencia ha evolucionado a tal grado que actualmente mediante el computador se pueden desarrollar herramientas que permiten el reconocimiento biométrico con elevado nivel de precisión y confianza. El presente proyecto está orientado al estudio, puesto en práctica y comprobación experimental de métodos de reconocimiento facial que gozan de reconocido prestigio en el campo del conocimiento. En concreto, después de haber desarrollado un estudio previo, se tomaron como referencia dos métodos de reconocimiento facial, uno basado en características propias (valores propios) de los rostros bajo estudio y el otro en una técnica que, a nuestro entender, es una mejora del primero, cuyo nombre es Análisis Discriminante Lineal. En concreto, se realizará una comparación de los algoritmos denominados Autorostros (Eigenfaces) y Análisis Discriminante Lineal (Fisherfaces)

En los sistemas biométricos constan varias derivaciones del reconocimiento tomadas de la voz, la firma, el iris, el rostro, etc. (Jain, Flynn, & Ross, 2008); sin embargo, se estima que en el futuro estas técnicas sean más utilizadas en los procesos de reconocimiento facial, y en todos los ámbitos, estas herramientas de identificación incorporada con otras permiten el reconocimiento más preciso de individuos u objetos mediante la integración sensorial.

No obstante, en base a la tecnología actual hay que reconocer que todavía el progreso de los procesadores no garantiza una forma rápida de implementar sistemas de identificación basada en algoritmos de cierta complejidad. Es por esto que, en ocasiones el mismo algoritmo siendo ejecutado en distintos procesadores, demora más tiempo en dar la respuesta deseada; por lo tanto,

sigue jugando un rol fundamental el hecho de que el tipo del procesador en donde se ejecutará la acción.

Esto ya era conocido desde el invento de las técnicas algebraicas y estadísticas que sustentan los métodos aquí analizados tal y como dicen Hernández & Mendez, 2018. Person inventó el análisis de componentes principales como parte del análisis factorial en 1901, el primer desarrollo teórico apareció en 1933 en un artículo escrito por N. Hotelling y no fue hasta pasada la década de los años 70 que se introdujo el uso de la técnica en aplicaciones prácticas en distintas ramas de la ciencia e ingeniería.

Entre los años 1970 hasta los 1990 se puede encontrar muchos trabajos de investigación basados en el análisis de componentes principales, debido a ciertas limitaciones de esta técnica “Análisis de Componentes Principales” ha sido sustituido su uso por otras herramientas; sin embargo, el trabajo científico realizado sobre el microprocesador se ha incrementado significativamente, para eliminar las limitaciones mencionadas anteriormente, en la última década se vuelve a retomar el análisis de componentes principales como herramienta fundamental, por ejemplo en reconocimiento facial.

Las técnicas utilizadas en esta tesis, se fundamentan en el Análisis de Componentes Principales o se basan en herramientas del Análisis de Componentes Principales.

Los resultados conseguidos de la comparación entre el método de Autorostros y el método ADL, demuestran que el primero es más rápido que el segundo, pero menos robusto, sin embargo, el segundo necesita más tiempo de procesamiento, pero no le afecta el ruido y perturbaciones, finalmente, se proponen algunas ideas que pudieran mejorar el desempeño del primer método, aunque también se demuestra que debido a las características de este método de Autorostros Mejorado, este no tiene mejor desempeño que el método ADL.

1.1 **Antecedentes**

En las primitivas civilizaciones las personas vivían en comunidades pequeñas, donde se reconocían sin dificultad por su número limitado de habitantes; sin embargo, debido al rápido aumento poblacional y la movilidad humana, la identificación se hizo un proceso complicado, de tal manera que las sociedades innovadoras han considerado necesario implementar sofisticadas técnicas de registro de identificación. La identificación es un conjunto de informaciones relacionadas a una persona, tales como: nombres, apellidos, lugar y fecha de nacimiento, etc. (Andrés, 2016)

En el año 1882, el policía Bertillon Alphonse presentó un algoritmo, el que años más tarde se constituiría en el primer sistema biométrico para la identificación de personas, fundamentado en rasgos físicos, al que llamó antropometría (Serratosa, 2013, p. 7). Este sistema se lo considera como un sistema biométrico científico, mediante el cual la policía podría registrar a todos los criminales.

Bertillon Alphonse se encargó de clasificar a los culpables por la altura y el rostro. (Serratosa, 2013, p. 7).

Los métodos de control de identidad son usados en varias aplicaciones que agilizan distintos procesos, tales como; transacciones financieras, abordajes de vuelos y el acceso a instalaciones restringidas, por indicar algunas de ellas. Los sistemas de reconocimiento se fundamentan en el análisis de los rasgos característicos del ser humano (Saeed & Nagashima, 2012).

1.2 Alcance

Este proyecto de titulación tiene como propósito diseñar y evaluar un software para solventar la necesidad de la identificación de rostros, el cual contará con dos algoritmos, para comparar su funcionamiento y medir su efectividad.

El software mencionado anteriormente contará con dos técnicas biométricas, las cuales consisten en dos fases:

1. **Fase de recaudación de información**: El sistema se programará sobre un software desarrollado en Matlab, para realizar la adquisición de rasgos biométricos se emplea una cámara webcam situada a una distancia determinada que sirve para el ingreso de información de los rasgos faciales de la persona, los que serán almacenados en una base de datos, para procesar la información se utilizarán los algoritmos Autorostros y ADL.

2. **Fase de reconocimiento**: El sistema obtendrá los rasgos característicos de las personas y los comparará con las imágenes previamente preparadas, las cuales se encuentran guardadas en la base de datos. El sistema determina si los rostros se encuentran en los registros y si existe una similitud entre los patrones almacenados, se establece o concluye que pertenecen a un usuario registrado.

1.3 Justificación del uso del reconocimiento facial

Cuando interactuamos cotidianamente, el principal foco de atención es nuestro rostro. Los seres humanos observamos las expresiones y las características faciales de quienes forman parte del entorno en donde nos encontramos. A través de la visualización y el almacenamiento en la memoria a largo plazo, somos capaces de recrear imágenes de cientos de rostros en unos cuantos segundos (Sternheim & Kane, 1989).

Tradicionalmente se utilizan muchos métodos para identificar personas, por ejemplo, para acceder al internet necesitamos usuario y contraseña, muchas veces también se necesita; la cédula de identidad, el pasaporte, licencias y otra información; sin embargo, se ha determinado que lo anterior no es una técnica

100% segura, debido al robo de la contraseña y el usuario, clonación de tarjetas, suplantación de identidad, entre otras, es mucho más difícil copiar el rostro; por tanto, el reconocimiento facial supera las técnicas anteriores, justo en su propia esencia. (Saeed & Nagashima, 2012).

El reconocimiento facial es una solución ideal, dado que posee un grado alto de aplicabilidad, se requiere identificar a una o varias personas y clasificarla de acuerdo a sus rasgos faciales. La fisonomía del ser humano está basada en el análisis y comparación de rasgos faciales y se determina que el algoritmo a utilizar en este proyecto podrá ser incorporado en diferentes casos donde se requiera hacer autenticación.

La identificación personal es fundamental para los seres humanos, cada ser es único, es decir, posee sus propios rasgos que lo diferencia de las demás personas, con la identidad se puede acceder a las obligaciones y derechos sociales. En la actualidad el mayor delito y que tiene un crecimiento acelerado es el robo o suplantación de identidad, esto como resultado de que la información no es resguardada de forma responsable y objetiva. Existe un gran crecimiento en el mercado informático y electrónico informático que vulnera dicho resguardo de la identidad (Cernándes Gómez & Zapata Ramírez, 2006).

Si se analiza el auge que han generado las redes sociales, es importante mencionar la infinidad de formas o maneras de suplantar o falsear identidades, una persona puede registrarse con nombres, alias o datos que no le pertenecen sin ser rechazada u objetada (Carlini, 2018).

En Ecuador existen registros de suplantación o robo de identidad, en el año 2018, sólo en la ciudad de Quito, la policía judicial registró un total de 383 denuncias por este delito (Pùblica, 2018) (UNIVERSO, 2018).

Los sistemas de reconcomiendo son fundamentados en la biometría, brindan resultados de identificación robusta, se usan los rasgos característicos del

cuerpo humano que son intransferibles y permanentes. Los rasgos característicos físicos se extraen del iris, del rostro y de las manos, también se puede usar rasgos conductuales, tales como; la escritura, la firma, entre otros (Jain, Flynn, & Ross, 2008).

2 OBJETIVOS

2.1 Objetivo general

Implementar y evaluar un software de reconocimiento de rostros, para facilitar el proceso de identificación de personas.

2.2 Objetivos específicos

1. Estudiar y poner en práctica las técnicas de reconocimiento facial conocidas como: Autorostros y Análisis Discriminante Lineal
2. Diseñar un software para reconocimiento facial.
3. Construir una base de datos con todos los rostros patrones.
4. Comparar algoritmos de reconocimiento facial recaudando información y probando su efectividad.

3 MARCO TEORICO.

3.1 Biometría

En el leguaje griego **Bios** significa vida y **metría** medición, por lo tanto, la biometría analiza la posición de rasgos característicos, estudia las distancias entre los rasgos físicos del cuerpo y clasifica e identifica a las personas basándose en sus rasgos característicos únicos (Wayman, 2011)

3.1.1 Historia de la biometría

La biométrica automatizada ha estado disponible en las últimas décadas, esto debido a los adelantos en el campo de procesamiento de la información; sin

embargo, estas técnicas no son nuevas, las ideas fueron concebidas originalmente hace muchos años. (Wayman, 2011)

En una cueva se visualizó pruebas del uso de la biometría por parte de los seres humanos; se estima que tengan una antigüedad de 31.000 años. La caverna estaba con pintura en las paredes, se pretende que fue realizada por los hombres prehistóricos, se encontraron en ella huellas de las manos, se piensa que pudo ser una firma del dueño de la caverna. (Digital, 2013)

3.1.2 Rasgos biométricos.

Los rasgos son únicos y se consideran como rasgos biométricos únicos de cada persona, algunos de estos rasgos biométricos del ser humano son; el iris, la voz, el rostro y el lenguaje corporal, se dividen en dos tipos fisiológicos y de comportamiento, los cuales se registrarán y son calculados por medio de computadoras. (Vázquez, 2014)

Los rasgos fisiológicos son características que poseen los seres humanos y se extraen de las manos, del rostro y del ADN, por mencionar algunos; los rasgos de conducta son el comportamiento y los gestos o reacciones frente a ciertas situaciones, tales como: la forma de escribir, de caminar, de hablar y la firma, entre otros. (Vázquez, 2014)

3.1.3 Variabilidad de los rasgos característicos del rostro

Por lo general, cuando se habla de rasgos únicos del rostro, se refiere aquellos que no se pueden repetir en los seres humanos y que se ven perjudicados por la herencia genética; se tiene la posibilidad de cierta semejanza en los rasgos biométricos entre familiares, los gemelos tienen un alto porcentaje de similitud, pero en su forma de actuar o reaccionar son diferentes; por lo tanto, se necesitaría de la ayuda de un sistema biométrico que mida varios parámetros del individuo, se cita un ejemplo, las huellas dactilares. (Wayman, 2011)

Otro elemento que influye es la permanencia de los rasgos biométricos, son los definidos como atributos que pueden ser constantes, sin embargo, esta condición no siempre se cumple por diferentes factores como; cirugías estéticas y envejecimiento.

La variación del conjunto de rasgos característicos de un ser humano se denomina “variabilidad intra-usuario” (Wayman, 2011)

3.2 Introducción a la visión computacional

Intenta emular la visión humana con la apreciación de las imágenes, de modo que se extraen características del entorno y el computador recibe los datos por medio de cámaras. Se deben considerar diversas situaciones del entorno, tales como la oscuridad y la luminosidad. (Baker & Nayar, 1996)

3.2.1 Visión humana.

El ser humano tiene la cualidad de receptar información mediante imágenes por medio del ojo. Este proceso se simplifica de la siguiente manera: El ojo capta la luz y la convierte en un impulso neuronal; como dicho impulso se enlaza con la corteza cerebral para procesar la información, la luz pasa a través del lente (llamado cristalino), traspasa una capa que se llama retina, la cual está compuesta en el segmento inferior del ojo por células receptoras, las cuales denotan la presencia de luz. Los impulsos neuronales son enviados al cerebro y se procesa dicha información, generando diversas sensaciones conocidas como percepción visual. (Marín, 2014)

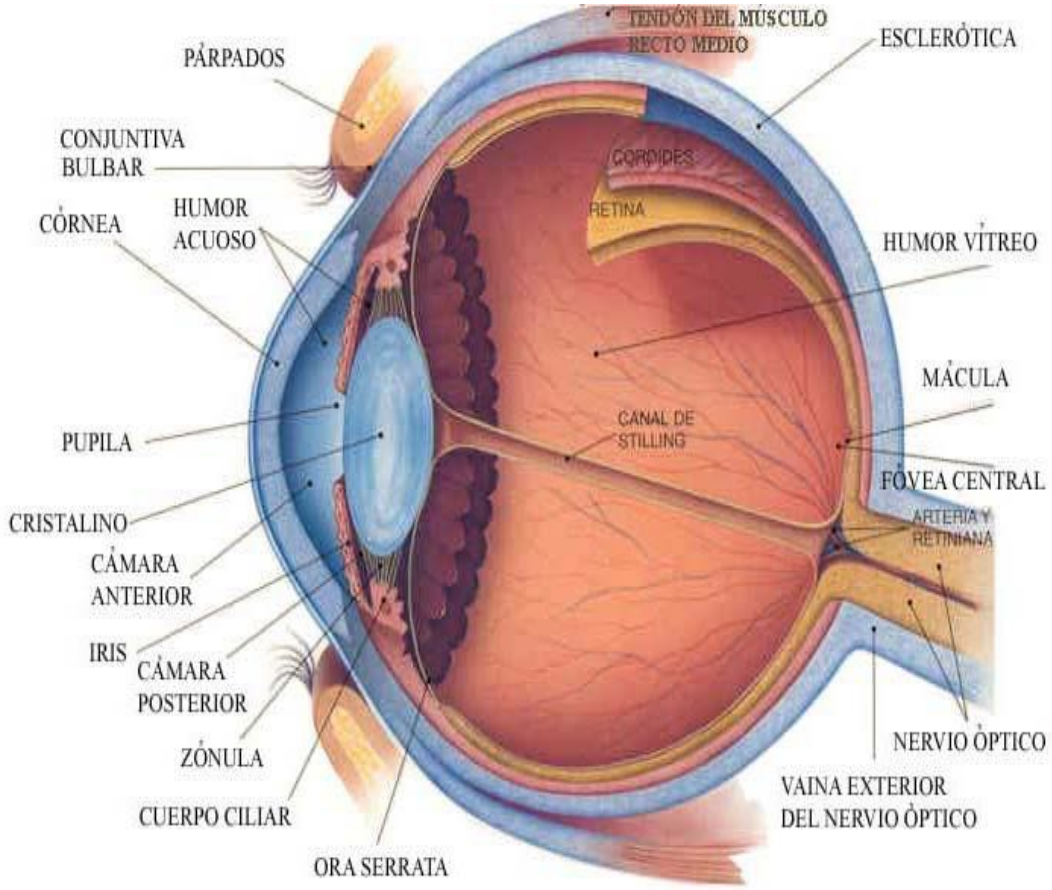

Figura 1: Ojo Humano

Tomado de *(Arcasoptica, 2016)*

3.2.2 Inicio de la visión computacional.

Ésta tuvo sus inicios cuando se realizó la primera toma fotográfica en 1825 año y el autor fue Joseph Nicephore. En aquel entonces, se utilizaron materiales fotosensibles en la parte interna de la cámara fotográfica oscura para centrar la imagen. (Torres, 2016)

En el año 1920, en los periódicos se aumentó la calidad de impresión, por este medio se enviaba la información a diferentes lugares del mundo, este fue un adelanto importante para llegar al proceso de visión computacional, años después, la resolución de las cámaras de video aumentó con los microprocesadores, que ayudaron directamente al progreso de desarrollo de las

computadoras, las mismas que efectúan el proceso de visión computacional, extraen las características más relevantes del entorno y desarrollan procesos automatizados. (Haralick & Shapiro, 2005)

3.2.3 Herramientas para la Visión computacional

Para la utilización de visión por computadora existen dispositivos como cámaras.

3.2.3.1 Cámaras

Primeramente, se necesita un dispositivo el cual sea sensible a la luz perceptible, el cual nos permitirá guardar las fotografías en un formato digitalizado cuyo proceso es que la luz atraviesa el lente hasta llegar al diafragma donde tiene materiales fotosensibles, el diafragma se puede abrir o cerrar dependiendo del monto de luz que necesite y el enfoque sirve de igual manera que el cristalino en el ojo del ser humano y al finalizar se encuentra el obturador permite que pasar la cantidad de luz en un tiempo determinado para proceder a ser capturada. (Pajares, 2002)

Es decir, para la obtención de una fotografía se debe utilizar un terminal como lo son: Webcam o cámara de video y las cuales se pueden acoplar mediante un puerto USB o Thunderbolt, para realizar la obtención de la fotografía en tiempo real. (Pajares, 2002)

3.2.3.1.1 Imágenes y Matrices

Las imágenes digitales están compuestas por pixeles, donde son los componentes de una imagen, donde representan la definición del brillo y el color de la imagen para la cual son almacenados en una computadora. (Klette & Reinhard, 2001).

Cada pixel se define como el color y la cantidad de datos que sujeta una imagen

(Bovik, 2005), donde la función de la intensidad de luz es:

$f(x, y)$ (Ecuación 1)

Donde la variable f se define como el brillo y (x, y) son las coordenadas espaciales de una imagen o un pixel. (Jain A.K, 1986)

3.2.4 Algoritmo Viola-Jones para detectar rostros.

En el año 2001, este algoritmo fue presentado por primera vez en una conferencia por Pablo Viola y Michael Jones, al algoritmo se lo conoce como una técnica para la localización del rostro, debido a su bajo requerimiento computacional, dado que se realiza quince veces más rápido que los demás algoritmos, esto empleado en tiempo real. Este algoritmo sigue mejorando, se puede emplear en diversas funciones como detectar objetos, además usar patrones de iluminación. Su funcionamiento es adecuado para rostros frontales, se basa en varias plantillas de características tipo HAAR. (Viola & Jones, 2001)

3.2.4.1 Características tipo HAAR

Este clasificador busca características comunes en los rostros humanos, que pueden estar basadas en las diferentes intensidades entre regiones de rectángulos continuos que son negros y blancos. Estos rectángulos adquieren un valor para realizar una comparación, se realiza la suma de la intensidad del valor de los pixeles por cada región y se revisa la desigualdad entre los valores obtenidos de las sumas realizadas para cada rectángulo. (Viola & Jones, 2001).

Existen tres características tipo HAAR. Estas están indicadas en la figura 2 y en la figura 4 se aprecia ejemplos de características comunes en la detección del rostro. (Viola & Jones, 2001).

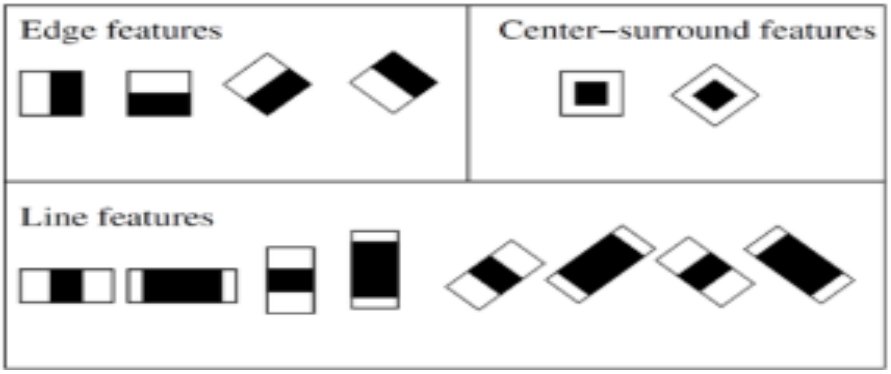

Figura 2: Características tipo HAAR

Tomado de *(Pavón, 2017)*

Figura 3: Ejemplos de características comunes para detectar el rostro

Tomado de *(Pavón, 2017)*

3.2.4.2 Imagen integral

Las cantidades de cada rectángulo que aparece en la figura 2 pueden ser calculadas de una manera efectiva y rápida, empleando fácilmente un barrido de la fotografía. Este barrido contendrá la suma del conjunto de valores del rectángulo en cualquier área de la imagen. (Viola & Jones, 2001)

3.2.4.3 Adaboost

Es un método de aprendizaje automatizado, se puede utilizar con otros

algoritmos para mejorar los resultados. Su función es realizar un proceso supervisado y concatenar las salidas de clasificadores no tan eficaces, creando así un detector robusto. (Viola & Jones, 2001)

3.2.4.4 Filtro en cascada

El algoritmo Viola-Jones utiliza las técnicas anteriormente planteadas para organizarlas en cascada; un usuario que consiga pasar todos los filtros será considerado como un rostro. De esta forma los requisitos computacionales disminuyen, las áreas que no contengan el rostro serán rechazadas en alguna de las fases de la cascada. (Viola & Jones, 2001)

3.2.4.5 Pseudocódigo de los clasificadores

1. Dando los pares $(x1, y1), (x2, y2), \dots (xi, yi), \dots (x_n, y_n), donde\ yi = \{\pm 1\}$

2. Iniciar los pesos:

$$W_{1,i} = \frac{1}{2m}\ for\ y_i = -1 \quad \text{(Ecuación 2)}$$

$$W_{1,i} = \frac{1}{2p}\ for\ y_i = -1 \quad \text{(Ecuación 3)}$$

Donde m es la cantidad de los números negativos y la variable p es la cantidad de números positivos.

3. Donde $t = 1, \dots T$

3.1 Normalización de los pesos;

$$W_{t,i} = \frac{W_{t,i}}{\sum_{j=1}^{n} W_{t,j}} \quad \text{(Ecuación 4)}$$

3.2 Se escoge el clasificador h_t con menor error épsilon:

$$\varepsilon_t = \min \sum_i W_i * (h_j(x_i))! = y_i \quad \text{(Ecuación 5)}$$

3.3 Sistema Biométrico

Su función es reconocer automáticamente a una persona utilizando rasgos característicos o de comportamiento, las personas registradas deberán tener los rasgos que solicite el sistema para realizar la comparación. Los sistemas biométricos se usan para varias aplicaciones, civiles, forenses y comerciales. (Cortés, Medina, & Escobar, 2010)

3.3.1 Propiedades del sistema de reconocimiento

Debe tener las siguientes propiedades para utilizar los rasgos característicos del rostro y probar de esta manera su eficacia, en este proceso es importante toma en cuenta los parámetros siguientes:

- Los usuarios deberán tener los rasgos característicos que solicite en el sistema.
- Los rasgos biométricos son irrepetibles y exclusivos, serán almacenados en el sistema.
- Todos los rasgos característicos deberán ser elementos que se puedan cuantificar, registrar y procesar.

3.3.2 Sistemas biométricos según su uso.

En la actualidad, los sistemas biométricos son muy usados por varias entidades o instituciones, realizan la función de identificar personas. Los sistemas biométricos se clasifican por el grado de confiabilidad; en la figura 4 se detallan estos aspectos. *(Informática, 2010)*

Figura 4: Sistema biométrico según su uso

Tomado de *(Informática, 2010)*

3.4 Reconocimiento facial

El reconocimiento facial es una parte de la biometría que utiliza técnicas de sistemas fundamentados en apariencia, estos sistemas son conocidos como métodos holísticos.

3.4.1 Sistemas holísticos

Realizan el análisis de los rasgos del rostro como un grupo, de un todo y hacen un análisis usando herramientas estadísticas, Estos sistemas buscan un subespacio con menor dimensión sobre los cuales proyectar los rostros y que incluyan un gran monto de información de los mismos contenida en menos vectores. Entre los métodos más relevantes se mencionan: Autorostros y Análisis Discriminante Lineal (ADL) (Hernández G. , 2010).

3.4.2 Sistema de reconocimiento facial

Determina el reconocimiento del ser humano examinando su rostro, el procedimiento que realiza es la extracción de los rasgos característicos del rostro de la persona, mediante una fotografía y comparar con los rasgos característicos de los rostros de las demás personas almacenadas en el sistema. (Carrasco, Portugal, & Peralta, 2011), provee de ventajas la utilización del reconocimiento facial y sus desventajas; en la siguiente tabla 2 se las puede apreciar.

Tabla 1
Identificación facial sus Ventajas.

Ventajas	Desventajas
• Alto grado de eficiencia • Dificultad al vulnerar el sistema, Los rasgos característicos son más difíciles de suplantar • Facilidad de uso • No requiere contacto físico por el usuario. • No se requiere: tarjetas de control, contraseñas	• Sensibilidad al cambio de iluminación • Ubicación del rostro • Cambios de rasgos faciales (Cirugías) • Utilización de objetos que obstaculizan la extracción de características tales como gorras, gafas.

Tomado de (Suárez & Duque, 2013)

3.4.2.1 Etapas del reconocimiento facial

Al momento de desarrollar un sistema de identificación facial se consideran las siguientes etapas:

3.4.2.2 Captura.

Es la obtención de la fotografía del usuario utilizando la visión computacional para extraer los rasgos característicos. (Perez, 2011)

3.4.2.3 Procesamiento.

Se ajusta la información de las características biométricas y se realiza el tratamiento del rostro adquirido, para que en la siguiente fase se realice la extracción de las características. (Perez, 2011)

3.4.2.4 Comparación

Se calcula la mayor similitud de los rasgos característicos del rostro a identificar con los que se encuentran almacenados previamente en el sistema. (Perez, 2011)

3.4.2.5 Identificación.

En esta última fase, se realiza la identificación del rostro de una persona específica, se toma el rostro con mayor semejanza y se lo considera un usuario registrado. (Perez, 2011)

4 ALGORITMOS DE RECONOCIMIENTO FACIAL

Estos algoritmos se emplean para obtener la información relevante del rostro, que permita la identificación por ordenador mediante los rasgos característicos específicos, con la mayor independencia posible; en el instante de la identificación la imagen a comparar puede ser que se encuentre contaminada, en este sentido gracias al avance tecnológico experimentado por el desarrollo de los microprocesadores fundamentalmente desde 1971, en los últimos años se ha podido implementar métodos de análisis de datos basados en técnicas algebraicas y de estadística, que permiten llevar a cabo los procesos de análisis de datos en entornos afectados severamente por ruido.

En consecuencia, algoritmos tales como el Análisis de Componentes Principales (PCA), basado en el método de reconocimiento facial Autorostros, a pesar de haber sido inventado por Pearson Karl en el año 1901 (Hernández & Mendez,

2018, pág. 1), logra dar sus primeros pasos en el mundo de procesado y compresión de imágenes en la década de los años 70, así mismo el algoritmo de Análisis discriminante lineal (LDA), se basa en el método de reconocimiento facial Fisherfaces que logra en su incursión en el mundo en los años 1936 (Feldesman, 2002).

En el presente trabajo de grado se ubica los dos métodos mencionados anteriormente, ya que dan los mejores resultados en la actualidad, finalizando con una comparación entre ambos y haciendo patente mi contribución al tema.

4.1 Método de Análisis de Compontes Principales

EL Análisis de Componentes Principales (PCA) es un método donde se refiere a la estructura de la varianza y covarianzas con un número reducido de combinaciones lineales de valores originales con la menor cantidad posible de pérdida de la información. Encuentra un conjunto nuevo de ejes ortogonales donde la varianza sea maximizada. Cuyo objetivo se usa para la disminución de las dimensiones (comprimir imágenes), es decir, se tiene un conjunto de variables, cuyo propósito es disminuir a un número inferior de variables. (Hernández & Mendez, 2018, pág. 2)

Si se tiene p variables las cuales se realiza el análisis, en ocasiones la variabilidad de la información se pueden encontrar en unos pequeños números k de componentes principales cuyo origen radica en la redundancia de información de las variables originales. Donde con tan solo los k de componentes principales se pueden representar a la variable p, de tal manera que la información original costa de la variable p que tiene n medidas y se disminuye a n medidas con k componentes principales. (Hernández & Mendez, 2018, pág. 2)

El objetivo del PCA es representar las variables numéricas en subespacios de menor dimensión, las representaciones mencionadas anteriormente pueden determinar que al suprimir dimensiones la perdida de información es mínima. Por

ejemplo; si se tiene un rectángulo en bidimensional realizando el proceso, se pierde información de los lados que se encuentran opuestos, sin embargo, la información que desaparece es recompensada por la simplificación de la cantidad de variables. (Hernández & Mendez, 2018, pág. 2)

4.1.1 Método matemático

4.1.1.1 Los vectores de la covarianza

Según Hernández & Mendez, (2018, pág. 3) $\boldsymbol{X} = [x_1 \dots x_p]^{\boldsymbol{t}}$ es un vector columna de dimensiones p , donde cada componente x_1 es una variable aleatoria con su promedio:

$$E[x_1] = \mu_1 \quad \text{(Ecuación 6)}$$

La varianza es:

$$V[x_1] = E[(x_1 - \mu_1)^2] = \sigma_{ii} \quad \text{(Ecuación 7)}$$

Obteniendo dos x_1 y x_j, donde se define la covarianza entre ellos como:

$$COV[x_1, x_j] = E[(x_1 - \mu_1)(x_1 - \mu_1)] = \sigma_{ii} \quad \text{(Ecuación 8)}$$

La covarianza se puede agrupar en vectores:

$$\mu = E[x] = \begin{pmatrix} \mu 1 \\ \dots \\ up \end{pmatrix}, \sum = Cov[x] = E[(x - \mu)(x - \mu)^t] \quad \text{(Ecuación 9)}$$

$$\mu = \begin{bmatrix} \sigma 11 & \dots & \sigma 1p \\ \dots & \dots & \dots \\ \sigma p1 & \dots & \sigma pp \end{bmatrix} \quad \text{(Ecuación 10)}$$

La matriz de correlación es dada por:

$$p = (p_{ij}), where\ p_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}}\sqrt{\sigma_{jj}}} \quad \text{(Ecuación 11)}$$

Se define como un ejemplo aleatorio:

$$\boldsymbol{X} = \begin{bmatrix} x11 & \dots & x1p \\ \dots & \dots & \dots \\ xp1 & \dots & xpp \end{bmatrix} \quad \text{(Ecuación 12)}$$

Donde se tiene p dimensión ordenando los valores de cada columna, dicha columna es la media de la muestra de p dimensión el vector es $\bar{X} = [\bar{X}]$, donde $\bar{X} = \frac{1}{p}\sum_{m=1}^{p} xim$, la matriz de covarianza es:

$$\boldsymbol{S} = [s_{ij}] = \frac{n}{n-1}\,\boldsymbol{Sn} = \frac{n}{n-1}(X - \bar{X})(X - \bar{X})^t \quad \text{(Ecuación 13)}$$

La varianza de la muestra se determina de $\boldsymbol{S}$ y donde la matriz de correlación es $\boldsymbol{R} = r_{ij}$, donde $r_{ij} = \frac{s_{ij}}{\sqrt{s_{ij}}\sqrt{s_{ij}}}$ con $i, j = 1 \dots p$.

4.1.1.2 Eigenvectors y Eigenvalues

Según Hernández & Mendez, (2018, pág. 3) la algebra alberga unos problemas en la simplificación de matrices, se ha utilizado métodos de triangular o diagonizar, que se aplican en sistemas de resolución: $\boldsymbol{Ax} = \boldsymbol{b}$**.** Definiendo A como una matriz cuadrada, $If\ \boldsymbol{v^t Av} \geq \boldsymbol{0}$ para cualquier vector v.

Una matriz con valores positivos, $If\ \boldsymbol{Av} \geq \boldsymbol{\gamma v}$, como $\boldsymbol{v} \neq \boldsymbol{0}$, γ es un eigenvalues asociado con los eigenvectors $\boldsymbol{v}$ (Hernández & Mendez, 2018, pág. 4)

4.1.1.3 Distancia Euclidiana

Según Hernández & Mendez, (2018, pág. 4) Es la medida más básica para el cálculo de distancias, se especifica como distancia ordinaria entre dos puntos en un plano, la que se deduce del teorema de Pitágoras. Por ejemplo, para calcular la distancias entre dos puntos distantes que están en un plano, es decir, dos dimensiones con coordenadas x e y. Con dos puntos considerándolos como $X1$

y $X2$ con unas coordenadas $(x1, y1)$, $(x2, y2)$ por lo tanto, el cálculo de la distancia se desarrollará con la siguiente ecuación: (Hernández & Plasencia, 2016)

$$D(X1, X2) = \sqrt{(x2 - x1)^2 + (y2 - y1)^2} \quad \text{(Ecuación 14)}$$

La distancia euclidiana con dos puntos en un espacio:

$$X = (x1, x2, \dots, xn)\ y\ Q = (q1, q2, \dots, qn) \quad \text{(Ecuación 15)}$$

En un espacio de n-dimensión se definiría como la distancia euclidiana:

$$D(P, Q) = \sqrt{\sum_{i=1}^{n} (pi - qi)^2} \quad \text{(Ecuación 16)}$$

Glosario de Análisis de Compontes Principales

$\mu_1 = promedio$

$\sigma_{ii} = varianza$

$Cov[x] = covarianza$

$(p_{ij}) = \ matriz\ de\ correlación$

$S\ = matriz\ de\ covarianza$

$R\ = matriz\ de\ correlación$

$\gamma = valores\ propios$

$v = vectores\ propios$

$D(P, Q) = distancia\ euclidiana$

4.1.2 Algoritmo de Autorostros.

Para estudio de los métodos multivariables es una gran ayuda la algebra matricial (Strang G, 2006), Además las expresiones matriciales se pueden programar con un grado de facilidad en los software.

Para una mejor comprensión al desarrollo del proceso del algoritmo de Eigenfaces se estima conveniente analizar y desarrollar el siguiente proceso.

1. Para almacenar todos los rostros de las personas en una base de datos (BD) se debe considerar el proceso que a continuación se detalla:
   1.1 Se realiza la **detección del rostro**
   1.2 **Recorte** del rostro
   1.3 Transformación a **escala de grises**
   1R.4 **Convertir** a una dimensión de nxm
2. Transformar los rostros en un vector columna, los cuales son asignados en una matriz **imágenes**, donde **cada columna** es **un rostro**.
3. Realizar el **promedio** de la matriz **imágenes**,
4. **Restar** el **promedio** a la matriz **imágenes.** Dichos resultados se guardan en la matriz que se denomina **Sin media**.
5. Calcular la matriz de covarianza de **imágenes**, para proceder a calcular los **componentes principales,** que dan como resultado los **Eigenvectors y Eigenvalues**.
6. Se coloca el número de **Eigenvectors** que se va a utilizar.
7. Multiplicar los **Eigenvectors** por la matriz **Sin media** los resultados se guardan en un vector columna **características patrón.**
8. **Escoger la imagen** a ser sometida al **proceso de comparación;** repetir las fases del **numeral 1.**
9. Convertir la **imagen escogida** en **vector columna** y **restar** con la matriz **promedio.**
10. Los resultados se guardan en una matriz denominada **características,** los mismos que son **multiplicados** por los **Eigenvectors**
11. Se calcula el inverso de la **norma L2 (euclidea)** de la matriz **características patrón** y de la matriz **características.**
12. Se **elige la imagen** con **mayor similitud,** la misma que queda establecida como la **imagen ganadora.**

El proceso técnico precedente y la verificación de las fórmulas empleadas se registran a continuación:

Según Ottado (2010, págs.2) para la realización del algoritmo se necesita que los rostros tengan las mismas dimensiones. Supongamos que se tiene una M cantidad de rostros de las personas de una tamaño NxN, que pertenecen a distintas personas; se requiere imágenes de preparadas $\{x1, x2, \dots xM\}$ en versión vector columna, donde $xi \in \mathbb{R}^d$ y $d = N\ x\ N$, Se espera que la cara sea tratada como un punto representada por $\mathbb{R}^d$, en nuestro caso es $N = 128$, todas las caras tendrán la misma dimensión y con un método de clasificación para poder desviar los demás rostros.

Se define la matriz **X** $\in d\ x\ M$ donde se agrupan los rostros de las personas en columna para las preparadas.

$$\mu = \frac{1}{M}\sum_{i=1}^{M} xi \qquad \text{(Ecuación 17)}$$

Se realiza la diferencia de cada rostro i con la media μ que está dada como el vector $\varphi i = xi - \mu$ El objetivo es logar obtener un grupo de vectores que describan mejor la distribución de la información; estos vectores son denominados Eigenvectors que están asociados a los Eigenvalues. (Ottado, 2010, pág. 3). La covarianza de los rostros se define como:

$$\boldsymbol{COV} = \frac{1}{M}\sum_{i=1}^{M} \varphi i \varphi i^T = \Phi\Phi^T \qquad \text{(Ecuación 18)}$$

Esta técnica se denomina Autorostros, su nombre se debe a la visualización de los vectores característicos o eigenfaces que tienen semejanza de un rostro; en la fase de preparación, se calculan los Eigenvectors del grupo de rostros de preparación. Uno de los inconvenientes intratable es el tamaño de las matrices; de igual manera, se pueden calcular los Eigenvalues y Eigenvectors de la matriz **COV** al hallar primero los Eigenvalues y Eigenvectors de un rostro $N\mathrm{x}\mathrm{N}$, donde

sea $\boldsymbol{L} = \boldsymbol{\Phi}^T \boldsymbol{\Phi}$, Tomar en cuenta que (Ottado, 2010, pág. 3)

$$\boldsymbol{\Phi}^T \boldsymbol{\Phi}\mathbf{v}\mathrm{i} = \gamma\mathrm{i}\ \mathbf{v}\mathrm{i} \qquad \text{(Ecuación 19)}$$

Donde **v**i son los Eigenvectors de **L,** ambos lados multiplicados por $\boldsymbol{\Phi}$ y el resultado es:

$$\boldsymbol{\Phi}\boldsymbol{\Phi}^T \boldsymbol{\Phi}\mathbf{v}\mathrm{i} = \gamma\mathrm{i}\ \boldsymbol{\Phi}\ \mathbf{v}\mathrm{i} \qquad \text{(Ecuación 20)}$$

$$\mathbf{COV}(\boldsymbol{\Phi}\mathbf{v}\mathrm{i}) = \gamma\mathrm{i}\ (\boldsymbol{\Phi}\ \mathbf{v}\mathrm{i}) \qquad \text{(Ecuación 21)}$$

Los Eigenvectors de **COV** son $\boldsymbol{W}\mathrm{i} = \boldsymbol{\Phi}\ \mathbf{v}\mathrm{i}$, Estos son los Eigenfaces con los cuales, mediante una combinación lineal, se representa a todos las caras de preparadas.

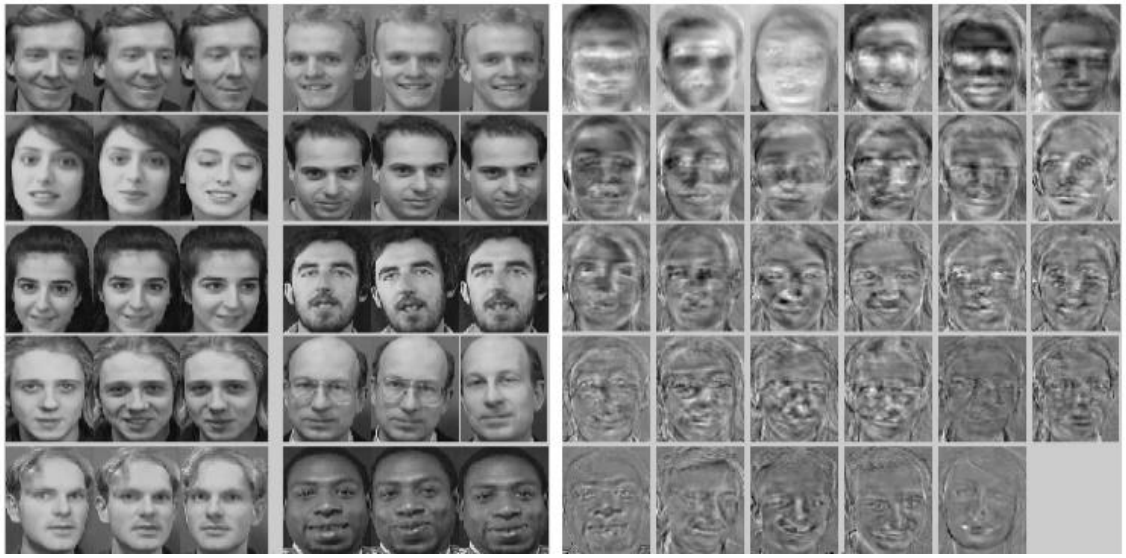

Figura 5: Eigenfaces conjunto de imágenes preparadas

Tomado de (Ottado, 2010)

Se escogen los mejores Eigenvectors, donde $\boldsymbol{K}$ corresponde a los Eigenvalues. Se procede a ordenarlos y se elige a los que tengan mayor información de los rasgos característicos; de esta forma se consigue una matriz de proyección $W = \{w1, w2, \dots, wk\}$ con las que se representa a los rostros en un menor subespacio de dimensión.

Se realiza una proyección de cada uno de los rostros en el espacio Eigenfaces. Cada matriz se aproxima usando a los mejores **K** Eigenfaces; la proyección se logra así:

$$\mathrm{Yi} = \mathrm{W}^T \varphi i \qquad \text{(Ecuación 22)}$$

En la fase de reconocimiento se escoge una imagen, se la proyecta en el espacio de Eigenfaces, se obtiene una serie de datos y se resta con el promedio del grupo de rostros preparados. Para hacer la clasificación se realiza una comparación con cada uno de los rostros de preparación, se calcula con la norma L2, definida por:

$$dL2(x,y) = \sum_i (xi - yi)^2 \qquad \text{(Ecuación 23)}$$

Glosario de Autorostros:

$\mu_1 = promedio$

$M = rostro$

$xi = numero\ de\ rostro$

$\sigma_{ii} = varianza$

$\Phi^T \Phi vi = covarianza$

$K = Eigenfaces$

$W = matriz\ de\ proyección$

$dL2(x,y) = distancia\ euclidiana$

4.2 Método de Análisis Discriminante Lineal

Es una técnica multivariables cuya función es clasificar en grupos la información, por lo tanto, se debe conocer previamente en qué grupos se encuentran o pertenecen, a esta técnica se la considera como un análisis de retroceso, la variable dependiente está sujeta a pertenecer a un grupo con etiqueta y las variables autónomas son continuas y determinan a que conjunto pertenecen. Se intenta hallar las relaciones lineales con los valores continuos, los cuales su discriminación sea la mejor con los demás grupos. (Belhumeur, Hespanha, &

Kriegman, 1996)

Una función es la construcción de una norma que designe un nuevo objeto, que no se sabe de forma previamente a que grupo pertenece y para prefijarlo a un grupo con un umbral de riesgo, es necesario tener en cuenta unos supuestos o restricciones:

- Se requiere al menos de dos grupos o clases.
- La cantidad de variables discriminantes deberá ser menor a la cantidad de objetos donde $p < (n-2)$ y *n* es la cantidad de objetos
- Los valores deben tener un patrón de distribución continuos

4.2.1 Método matemático

Según Fernández (2011, págs.3) Con un grupo q los cuales se asignan una cantidad de objetos y con una variable p de medidas sobre los cuales $(x1,..,xp)$, obteniendo una serie de puntos donde indique a que grupo pertenece $(y1,..,ym)$, cuya función lineal es de $x1,..,xp$

$$y1 = a_{11}x_1 + \cdots + a_{1p}x_p + a_p$$
$$\cdots\cdots.$$
$$y_m = a_{m1}x_1 + \cdots + a_{mp}x_p + a_{m0} \qquad \text{(Ecuación 24)}$$

Para $m = \min(1-1, p)$, de tal manera que desvíen lo mejor posible a los q grupos. La combinación lineal debe maximizar la variabilidad entre las clases y minimizar la variabilidad dentro de la clase, se realiza una descomposición de la varianza.

4.2.1.1 Descomposición de la varianza

Al descomponer la varianza de la muestra entre los grupos y dentro de un grupo, se parte de:

$$Cov(x_j, x_{j'}) = \frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \overline{x_j})(x_{ij} - \overline{x_{J'}}) \quad \text{(Ecuación 25)}$$

Se considera el promedio de la variable $(\boldsymbol{x_j})$ donde cada grupo es $(I_1, \dots, I_q)$, se deduce que:

$$\overline{x_{kJ}}\ \frac{1}{n_k}\sum_{i\in I_k}^{n} x_{ij} \quad \text{(Ecuación 26)}$$

Donde $k = 1, \dots, q$

La media de cada grupo se denomina así;

$$\sum_{i\in I_k} x_{ij} = n_k\, x_{kj} \quad \text{(Ecuación 27)}$$

Entonces tenemos:

$$\bar{x}_j \sum_{k=!}^{q} x_{ij} = \frac{n_k}{n}\ x_{kj} \quad \text{(Ecuación 28)}$$

Se deduce:

$$Cov(x_j, x_{j'}) = \frac{1}{n}\sum_{k=1}^{q}\sum_{k=!}^{q}(x_{ij} - \overline{x_j})(x_{ij} - \overline{x_{J'}}) \quad \text{(Ecuación 29)}$$

Se determina lo siguiente:

$$Cov(x_j, x_{j'}) = \frac{1}{n}\sum_{k=1}^{q}\sum_{k=1}^{q}(x_{ij} - \overline{x_j})(x_{ij} - \overline{x_{j'}}) + \sum_{k=1}^{q}\frac{n_k}{n}\left(x_{ij} - \overline{x_j}\right)\left(x_{ij} - \overline{x_{J'}}\right) \quad \text{(Ecuación 30)}$$

$$= d(x_{j,}x_{j'}) + e(\,x_{j,}x_{j'})$$

El resultado de la covarianza es similar a la covarianza entre grupos y dentro del grupo, se denomina como $t(\,x_{j,}x_{j'})$, entonces se expresa lo anteriormente dicho de la siguiente manera.

$$t(x_{j,}x_{j'}) = d(x_{j,}x_{j'}) + e(x_{j,}x_{j'}) \quad \text{(Ecuación 31)}$$

En términos matriciales es igual a:

$$\boldsymbol{T} = E + D \quad \text{(Ecuación 32)}$$

4.2.1.2 Extracción de las funciones discriminantes

El fundamento primordial consta en extraer de $(x1,..,xp)$ valores de un grupo k, m funciones y $(y_i,..,y_m)$ de tal manera que:

$$y_i = a_{i1}x_1 + \cdots + a_{i0} \quad \text{(Ecuación 33)}$$

Para una función $m = min(q-1, p)$, donde la correlación $(y_i, y_j) = 0$ para todos los $i \neq j$.

Las variables $(x1,..,xp)$ están estandarizadas, la función es:

$$y_i = a_{i1}x_1 + \cdots + a_{i0}x_p \quad \text{(Ecuación 34)}$$

Para todos los $i = 1, \dots, m$, es una función canónica.

Las funciones $(y_1,..,y_m)$, se realizan las extracciones donde:

(i) y_1 Es la combinación lineal de $(x1,..,xp)$ la cual proporciona la mejor discriminación entre los grupos.

(ii) y_2 Es una combinación lineal de $(x1,..,xp)$ proporcionando la mayor discriminación entre grupos posteriormente a y_1, donde $(y_1, y_2) = 0$

Por lo tanto y_1 es una combinación lineal la cual proporciona una mejor discriminación y de tal manera es correlacionada para $j = 1, \dots, (i-1)$

4.2.1.3 Procedimiento matricial

De igual manera que el análisis factorial, se busca una función de $(x1, .., xp)$: $y = a'x$, de tal forma que:

$$Var(y) = a'Ta = aEa + A'Da \quad \text{(Ecuación 35)}$$

La varianza entre los grupos, más la varianza dentro del grupo.

Se realiza la maximización de la varianza entre grupos para aumentar la efectividad lo que es igual a:

$$f(a) = \frac{a'Ea}{a'Ta} \quad \text{(Ecuación 36)}$$

Observar que la función f sea homogénea, por lo tanto:

$$f(a) = f(\mu\, a) \quad \text{(Ecuación 37)}$$

Donde $\mu \in \mathbb{R}$

La función homogénea involucra calcular el máximo de $\frac{a'Ea}{a'Ta}$ siendo igual a:

$$max(a'Ea) \qquad tal\ que \qquad a'Ta = 1 \quad \text{(Ecuación 38)}$$

Con el esquema se realiza la multiplicación de LaGrange:

$$L = a'Ea - \lambda(\mathrm{a'Ta} - 1) \quad \text{(Ecuación 39)}$$

Se procede a derivar:

$$\frac{\partial L}{\partial a} = 0 \quad \text{(Ecuación 40)}$$

$$\frac{\partial L}{\partial a} = 2Ea - 2\,\lambda\mathrm{Ta} = 0 \Rightarrow$$

$$Ea = \lambda\mathrm{Ta} \Rightarrow$$

$$(T^{-1}E)a = \lambda\mathrm{a} \quad \text{(Ecuación 41)}$$

Los vectores propios que se encuentran en la función inicial de la matriz $T^{-1}E$, lo general no es simétrica.

$a'Ea = \lambda a'Ta = \lambda$ (Ecuación 42)

Se realiza la objeción del vector propio contienen una gran cantidad del poder discriminante.

El vector propio de la función discriminante muestra la mayor parte de la variabilidad de la función m que almacena los valores de y_i.
Para la obtención de funciones discriminantes se realiza los cálculos de valores propios de la matriz $T^{-1}E$, se ordena en forma descendiente:

$$\begin{array}{ccc} \boldsymbol{a'_2} & \Rightarrow & \boldsymbol{a'_2 x = y_2} \\ \ldots & \ldots & \ldots \\ \boldsymbol{a'_m} & \Rightarrow & \boldsymbol{a'_m x = y_m} \end{array}$$ (Ecuación 43)

Para el caso de $m = \min(q-1, p)$
Son vectores independientes y funciones sin correlación, la suma de los valores propios proporciona la variabilidad total.

Glosario de Análisis Discriminante Lineal:

$q = grupo$
$y_m = función\ lineal$
$Cov(x_j, x_{j'}) = Covarianza$
$e(x_j, x_{j'}) = Covarianza\ entre\ grupos$
$d(x_j, x_{j'}) = Covarianza\ dentro\ del\ grupo$
$t(x_j, x_{j'}) = Covarianza\ total$
$T = Covarianza\ total$
$E = Covarianza\ entre\ grupos$
$D = Covarianza\ dentro\ del\ grupo$
$Var(y) = Varianza$
$dL2(x, y) = distancia\ euclidiana$

4.2.2 Algoritmo de Fisherfaces

1. Para almacenar todos los rostros de las personas en una base de datos (BD) se debe considerar el proceso que a continuación se detalla:
   1.1 Se realiza la **detección del rostro**
   1.2 **Recorte** del rostro
   1.3 **Transformación** a escala de grises
   1.4 **Convertir** a una dimensión de nxm
2. Transformar los rostros en un vector columna, los cuales son asignados en una matriz **imágenes**, donde **cada columna** es **un rostro**.
3. Realizar el **promedio total** de la matriz **imágenes**,
4. **Restar** el **promedio** a la matriz **imágenes.** Dichos valores se guardan en la matriz **Sin media**.
5. Se calcula la covarianza de matriz **Sin media**
6. Se calcula los **Eigenvectors** y **Eigenvalues** de la matriz de **covarianza sin media**
7. Verificar si es una matriz singular, si lo es así, **multiplicar** la matriz **Sin media** con los **Eigenvectors** la cual será mi nueva matriz **Sin media** y se repetirá nuevamente desde **numeral 6.**
8. Ordenar los **Eigenvectors** según su valor
9. Realizar el **promedio de cada clase** de los BD
10. Restar dicho **promedio** a **cada clase** para ser almacenado en una matriz denominada **Sin media clases.**
11. **Restar** el **promedio de cada clase** al **promedio general**
12. Se calcula la **matriz S_b,** denominada **entre clases,** la cual se obtiene de la matriz covarianza del resultado anteriormente; los resultados son guardados en la **matriz S_b.**
13. Se calcula la **matriz S_w,** denominada **con clase,** la misma que se obtiene de la matriz covarianza **Sin media clases;** los resultados son guardados en la **matriz S_w.**
14. **Escoger la imagen** a ser sometida al **proceso de comparación,** repetir las fases del **numeral 1.**

15. Convertir la **imagen escogida** en **vector columna** y **restar** con la matriz **promedio general y** el resultado de los mismos son **multiplicados** por los **Eigenvectors,** se guardan en una matriz denominada **características.**
16. Se calcula la **norma L2** (**euclidea**) de la matriz **características patrón** y de la matriz **características.**
17. Se **elige la imagen** con **menor distancia,** la misma que queda establecida como la **imagen ganadora.**

El proceso técnico precedente y la verificación de las fórmulas empleadas se encuentran registradas a continuación:

Según Lu & Plataniotis, (2003, pág. 3) este método lleva a los rostros a un subespacio de menor dimensión donde incrementa la separación entre clases, el principal objetivo de esta técnica es hallar un subespacio que contenga los discriminantes de todas las clases, la matriz de dispersión se calcula con las clases que son diferentes y con la matriz de igual clase.

El Análisis discriminante lineal maximiza la variedad entre las clases en un grupo de datos y disminuye la variedad entre la misma clase, se utiliza principalmente para la problemática de clasificación. (Belhumeur, Hespanha, & Kriegman, 1996)

Este método necesita una base de datos que debe tener un gran grupo de individuos con diferentes expresiones, iluminación y condiciones de fondo, por ejemplo; distintas posiciones del rostro, una fotografía con gafas y otra sin ellas. La selección de un conjunto de expresiones certifica los resultados finales. (Belhumeur, Hespanha, & Kriegman, 1996)

Los rostros normalizados en matrices $M\,x\,N$, sea el rostro: $I(x,y)$ de dos dimensiones con un tamaño de N^2, representa a el rostro por un vector, el grupo que contiene todos los rostros es un espacio vectorial altamente dimensional. Definiendo a cada persona que pertenece a una clase y el resto de las personas

pertenecientes a distintas clases, se realiza el análisis de alejamiento de clúster y se etiqueta a todo el grupo de rostros preparados. Una vez ya teniendo todas las muestras de las clases se calcula la matriz de dispersión entre clases.

Se calcula similar a PCA, $\{x1, x2, \dots xM\}$ el grupo de imágenes de los rostros que se encuentran preparados en un vector columna, estos rostros deben pertenecer a una clase c donde $\{X1, X2, \dots Xc\}$, este método realiza los cálculos de tres matrices, S_b la matriz de esparcimiento entre clases, S_w la matriz de esparcimiento intra-class y S_T cuya matriz es el resultado de la suma de esparcimiento de S_B y S_w, se detallan como:

$$S_{B=} \sum_{i=1}^{c} |Xi|(\mu - \mu i)(\mu - \mu i)^T \quad \text{(Ecuación 44)}$$

$$S_B = \sum_{i=1}^{L} \sum_{i=1}^{c} (xk - \mu i)(xk - \mu i)^T \quad \text{(Ecuación 45)}$$

Donde $|Xi|$ es el número de los rostros, μi es el promedio de clases Xi y μ el promedio de todos los grupos de rostros preparados. El discriminante lineal de Fisher proyecta una maximización de la razón del esparcimiento de las clases y las intra-clases.

$$W_{FLD} = \max \frac{|W^T S_B W|}{|W^T S_W W|} = \{w1, w2, \dots wK\} \quad \text{(Ecuación 46)}$$

Este problema se resuelve calculando los eigenvectors de S_B y S_w, esto quiere decir $S_B wi = \gamma S_w wi$ o también ${S_B}^{-1} S_B wi = \gamma\, wi, i = 1,2, \dots K$. Ahora la proyección es optimizada siempre y cuando S_w no sea singular, esto quiere decir que tenga inversa, cuando se trabaja sobre un grupo de rostros se tiene que S_w casi la mayoría del tiempo es singular, su rango es mínimo a: $M - c$, por lo general, cada vector tiene mayor dimensión o es más grande que el número de rostros preparados. Para solucionar el inconveniente de singularidad, el método de Fisherfaces realiza una proyección en un espacio que sea de menor dimensión, donde S_w no sea singular usando el método PCA y se proyecta en un espacio

$M - c$ de dimensiones y después se utiliza la proyección donde se disminuye el espacio conseguido a dimensiones $c - 1$. La matriz FLD se detalla:

$$W_{FLD} = \max \frac{|W^T W_{PCA} S_B W_{PCA} W|}{|W^T W_{PCA} S_W W_{PCA} W|} \quad \text{(Ecuación 47)}$$

Para terminar, el método Fisherfaces se utiliza la proyección que está compuesta como W_{PCA} y la proyecta W_{FDL}

$$W^T{}_{Fisher} = W^T{}_{FLD} W^T{}_{PCA} \quad \text{(Ecuación 48)}$$

Es similar como el método Eigenfaces, los rostros se proyectan como en la figura 6, y almacenan los resultados:

$$y_i = W^T{}_{Fisher} \varphi i \quad \text{(Ecuación 49)}$$

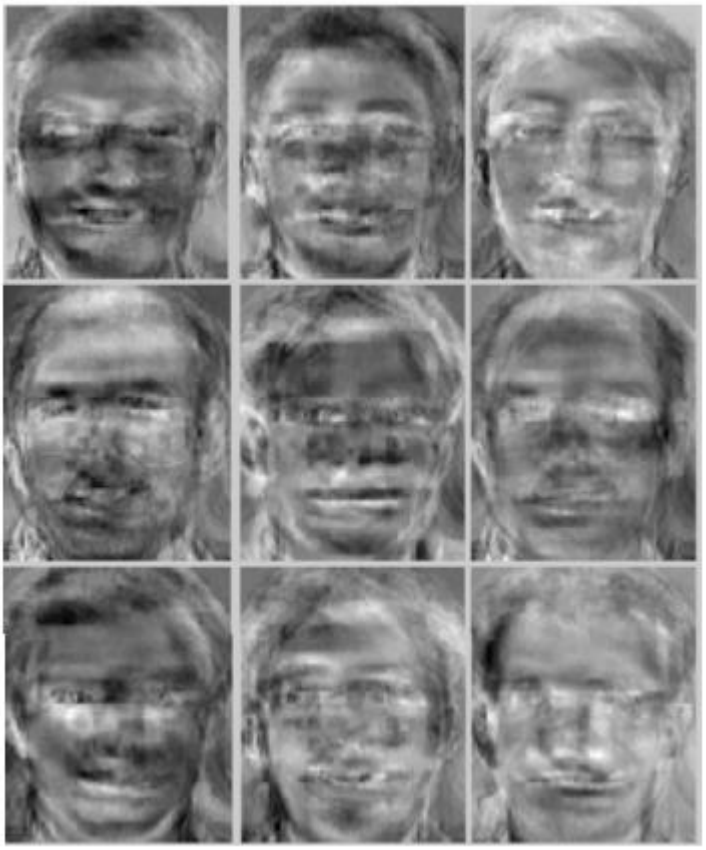

Figura 6: Proyección de Fisherfaces

Tomado de (Massieu, 2013)

Un rostro a reconocer se proyecta en el espacio Fisherfaces, se comparan con

los rostros preparados y los resultados obtenidos se calculan con norma L2 o distancia euclidiana, igual que el método anterior. Fisherfaces tiene la ventaja de tolerar ciertas variaciones en expresiones faciales y de iluminación.

Glosario de Fisherfaces:

$c = clases$

$\mu = promedio$

$S_w =$ entre clases

$S_b = con\ clases$

$W_{FLD} = matriz$ esparcimiento

$W_{PCA} = matriz\ de\ PCA$

$W = matriz\ de\ proyeccion$

$dL2(x, y) = distancia\ euclidiana$

5 DISEÑO

En este proyecto se realizó tres programas para el reconocimiento de rostros, dos son los anteriormente estudiados (Autorostros, Fisherfaces) y el otro es la mejora del algoritmo (Autorostros con clases) al cual se le agrego clases para mejorar su funcionamiento, para programar el sistema se utilizó el software denominado Matlab que tiene un entorno de desarrollo integrado y brinda la facilidad de trabajar con matrices.

Para su correcta compresión se ha separado por módulos los sistemas de identificación.

5.1 Diagrama de los módulos

En la figura 7 se especifica el diagrama de módulos del sistema de autenticación de rostros al que se accede mediante un aplicativo que consta de; detección, procesamiento de la imagen, extracción de características, comparación de características y decisión.

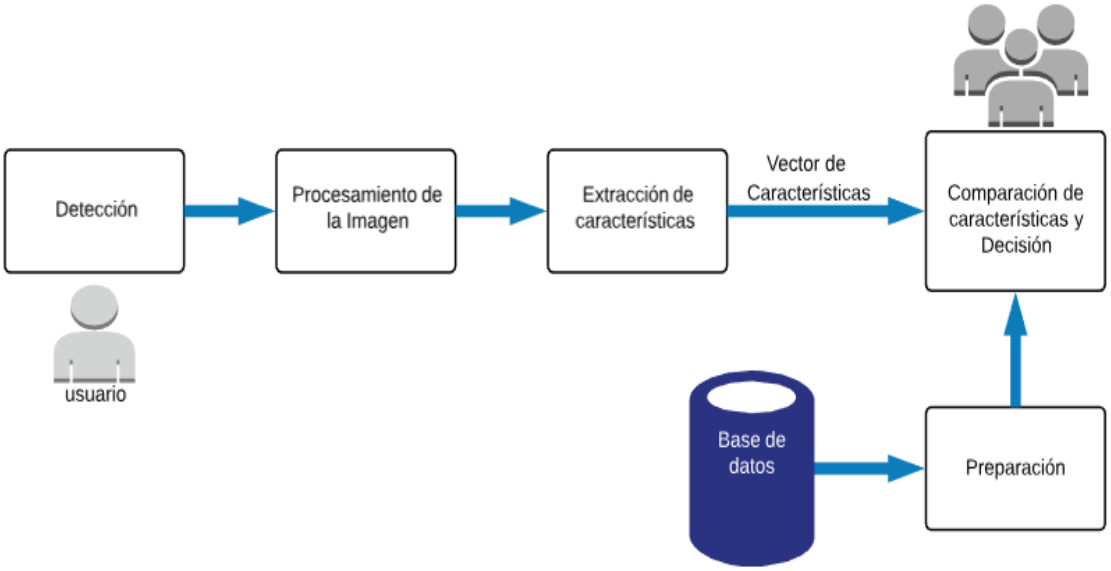

Figura 7: Diagrama de los módulos

Se detalla la lógica empleada para realizar la programación del reconocimiento facial utilizando diagramas de flujo, los que ayudan a la compresión de su configuración y funcionamiento.

5.1.1 Diagramas de flujo Autorostros

En la figura 8 se detalla el proceso que realiza el programa para la identificación.

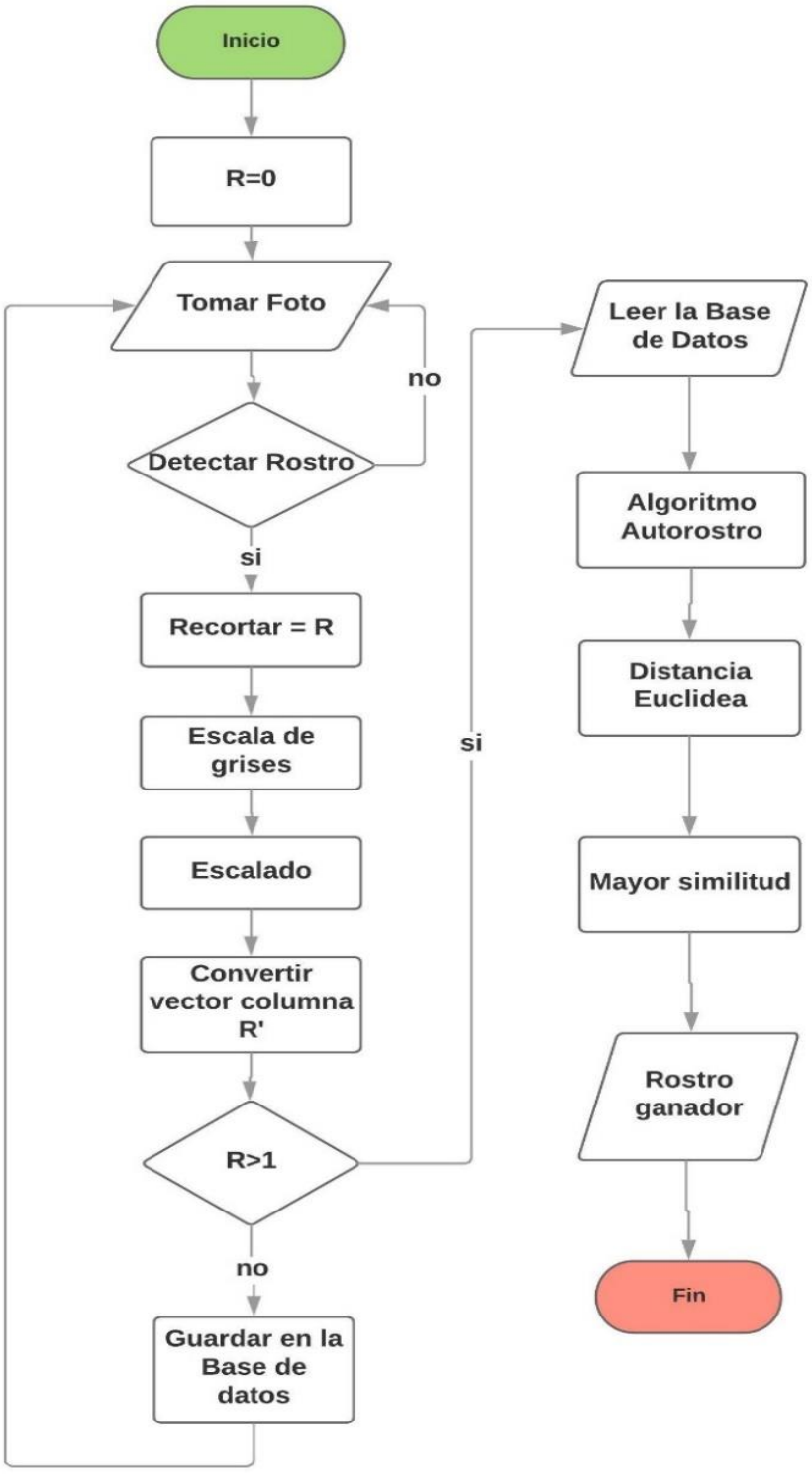

Figura 8: Diagrama de flujo de Autorostros.

El código del sistema de reconocimiento se encuentra registrado en el anexo 1.

5.1.2 Autorostros mejorado

En la figura 9 se detalla el proceso del programa para la identificación con el que se realizó una mejora al algoritmo anterior, añadiendo clases.

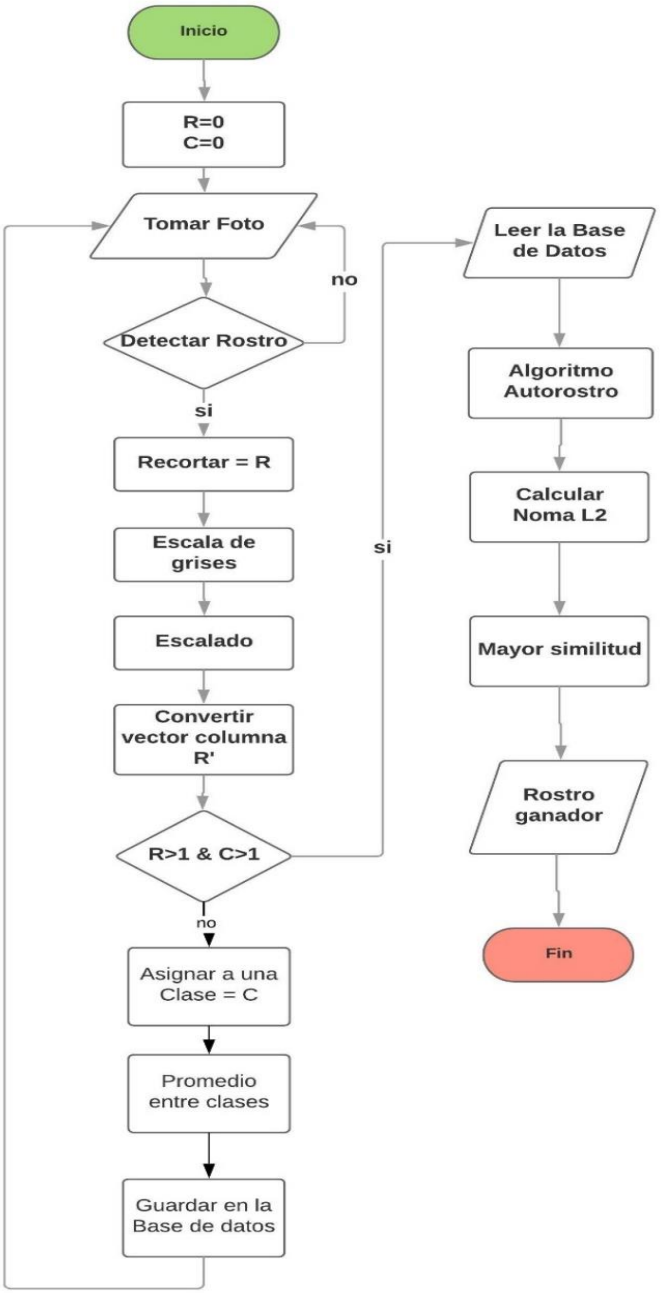

Figura 9: Diagrama de flujo de Autorostros mejorado

El código del sistema de reconocimiento se encuentra registrados en el anexo 2.

5.1.3 Fisherfaces

En la figura 10 se detalla el proceso del programa para el reconocimiento.

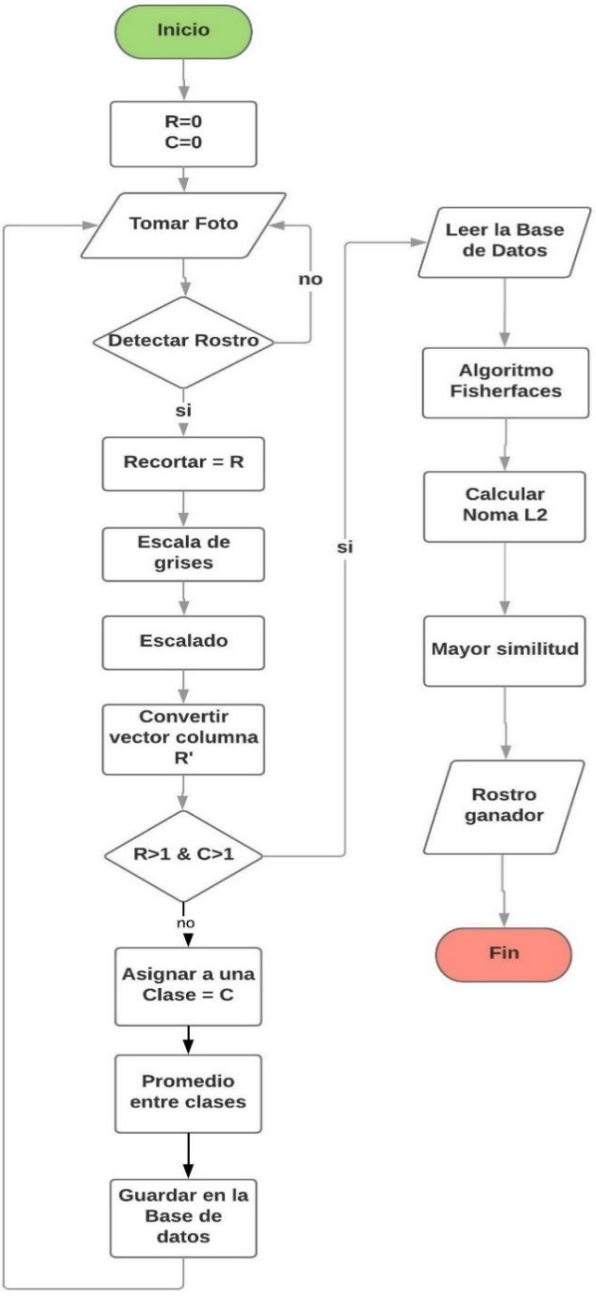

Figura 10: Diagrama de flujo de Fisherfaces

El código del sistema de reconocimiento se encuentra registrados en el anexo 3. Detalle de cada módulo del diagrama.

5.1.4 Detección

En este módulo, considerado el más importante del sistema, se estima que una mala localización del rostro daría como resultado un error en los siguientes módulos. Para la captura de una imagen se debe instalar un adaptador que se llama “winvideo”, el cual permite acceder a la cámara web mediante el software.

Esta herramienta permite comunicarse con el terminal; es recomendable en equipos Toshiba y HP debido a que son los equipos que arrojan altos niveles de compatibilidad con la versión del software Matlab 2015Ra. (The MathWorks, 2018)

Para la instalación del adaptador se debe crear una cuenta en la página de MathWorks, el siguiente paso es seleccionar el paquete el cual contendrá el controlador para ser instalado.

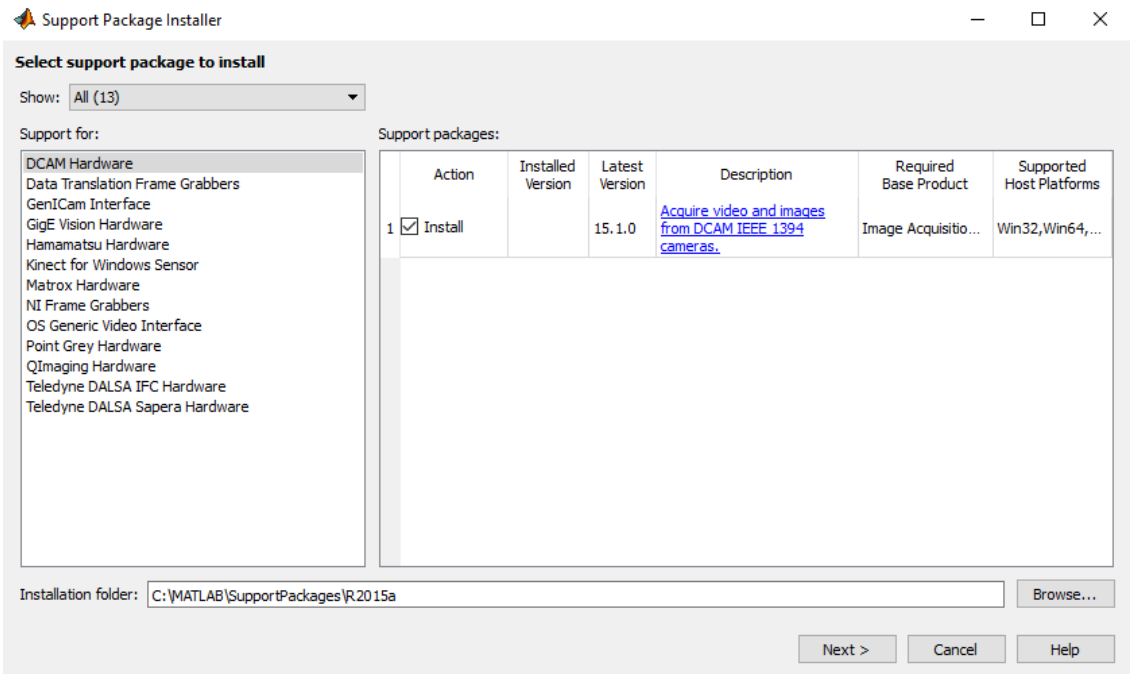

Figura 11: Sección de package.

Una vez seleccionado el paquete que contiene el controlador, se iniciar sesión en MathWorks y se procede a instalarlo.

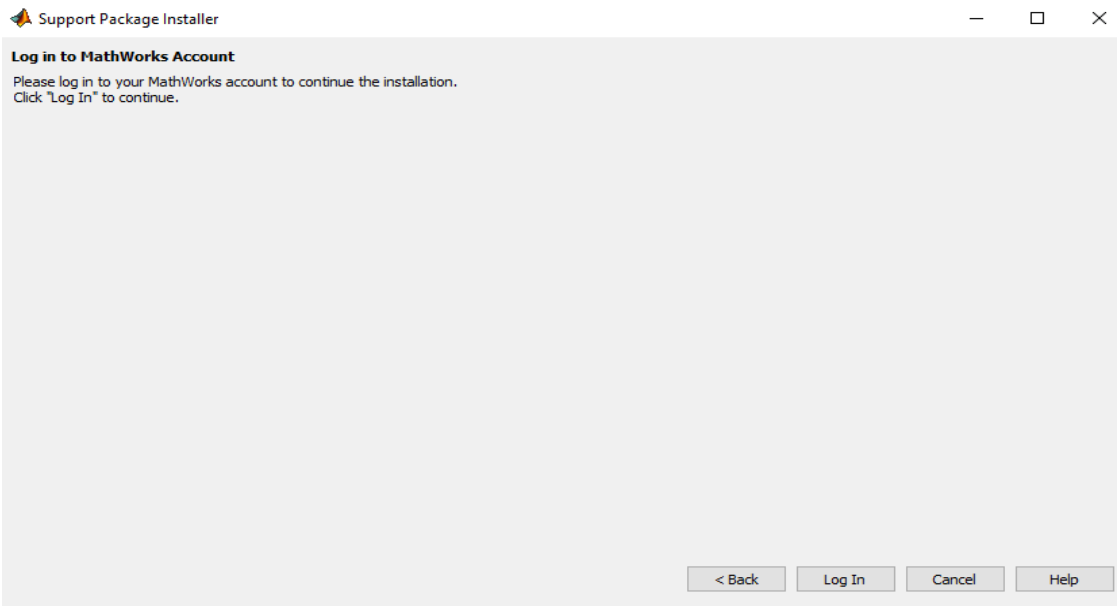

Figura 12: Crear cuenta en MarthWorks

El ultimo paso para la instalación es la verificación del paquete, se debe colocar el comado dato=imaqhwinfo en la seccion de comandos de Windows, el controlador se instalo correctamente se visualiza en la figura 13:

```
InstalledAdaptors: {'winvideo'}
    MATLABVersion: '8.5 (R2015a)'
      ToolboxName: 'Image Acquisition Toolbox'
   ToolboxVersion: '4.9 (R2015a)'
```

Figura 13: Adaptador instalado Winvideo

Para el desarrollo del sistema de reconocimiento facial; una cámara web con las siguientes características, que se detalla en la tabla 2:

Tabla 2

Características de la Webcam

Características	
Nombre	HP Truevision HD
Resolución	1280x720

Formato	MJPG

Antes de realizar la toma de imágenes se debe crear un canal o ventana, donde se indica de forma permanente el video y una vez creado el canal se realiza la configuración para la adquisición de las imágenes, en el sistema se incorpora el método Viola-Jones el que es robusto y ayuda en la detectarían de rostros en tiempo real.

5.1.4.1 Detección de rostro

Aplicando el algoritmo Viola-Jones, realizada la detección del rostro se debe verificar que su funcionamiento es correcto al visualizar un recuadro de color azul indicando el contorno del rostro, de esta manera, se puede validar que el proceso es el correcto, como se aprecia en la figura 14. (MathWorks T. , 2018)

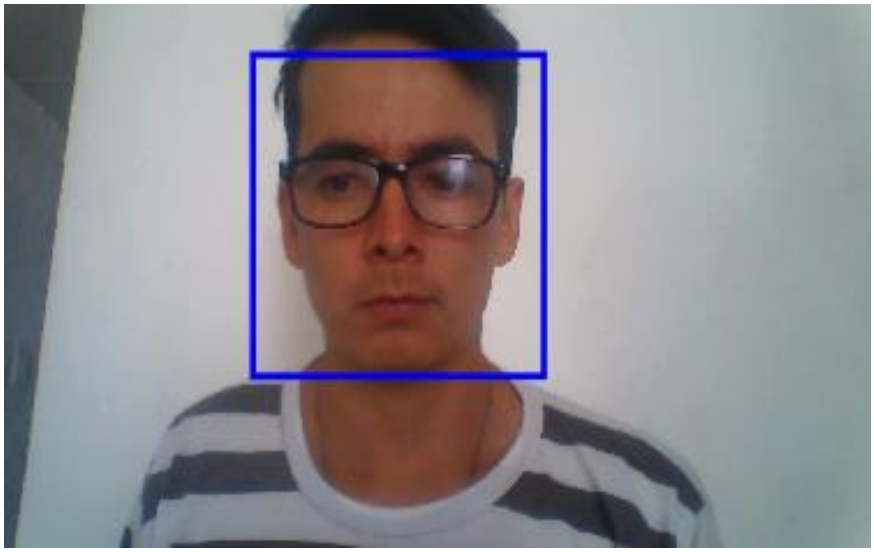

Figura 14: Detección de rostro

Las evaluaciones realizadas para dictaminar que el algoritmo es robusto en ciertos ambientes y que existen aspectos que influyen significativamente en su funcionamiento como; la iluminación, la distancia de la cámara. Debido a que únicamente puede detectar rostros frontales; cabe destacar que, pueden existir objetos que contribuyen en la alteración del funcionamiento del algoritmo.

En la figura 15 se observa el recuadro de color azul mal ubicado, esto sucede por la utilización de una gorra que cubre el rostro lo que afecta el desempeño del algoritmo.

Figura 15: Evaluación del algoritmo Viola-Jones

Al realizar una prueba con poca iluminación en el entorno, el resultado es notorio, en la figura 16 se puede verificar lo mencionado, el algoritmo detecta el rostro correctamente y sigue funcionando con total normalidad, es importante recomendar que el algoritmo debe ser empelado en un ambiente controlado.

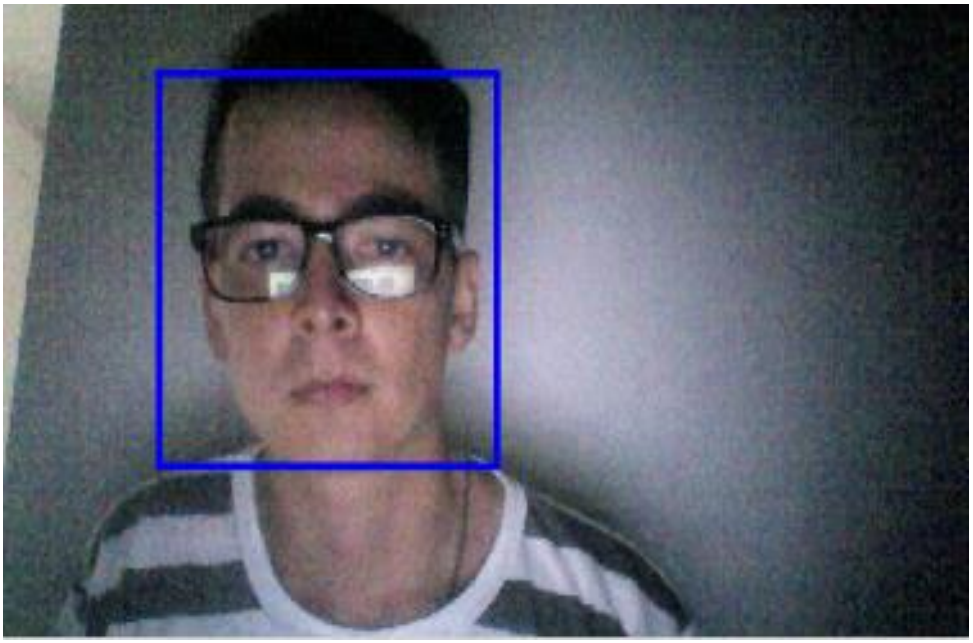

Figura 16: Detectar el rostro en condiciones de poca iluminación.

5.1.5 Procesamiento de la Imagen

Esta etapa se lleva a cabo después de la detección del rostro, una vez que se ha determinado la imagen idónea a utilizar, se realiza una serie de pasos, los cuales se detallan en la figura 17.

Figura 17: Procesamiento del rostro.

5.1.5.1 Recorte

Una vez identificado el rosto, se realiza el recorte en la fotografía para obtener el área de interés (rostro). (MathWorks, 2018)

5.1.5.2 Escalado

En esta fase se amplía o reduce las dimensiones del rostro y el empleo de los algoritmos estadísticos de reconocimiento facial así lo requieren dado que las fotografías deben tener las mismas dimensiones para todos rostros. Se emplean las coordenadas $k\ y\ k+n$, las que pueden ir ampliándose hasta utilizar el espacio que se desee. (Pajares, 2002)

$$fa = \left[\frac{tam}{(k+1)-k+1}\right] = [\frac{tam}{n+1}] \qquad \text{(Ecuación 50)}$$

Se calcula la coordenada de origen de cada uno de los puntos, de tal forma como:

$$Xorigen = k + \left[\frac{n}{tam-1}\right] Xaumentada \qquad \text{(Ecuación 51)}$$

La variable $Xorigen$ y $Xaumentada$ pertenecen a las coordenadas de los pixeles de origen y aumentada.

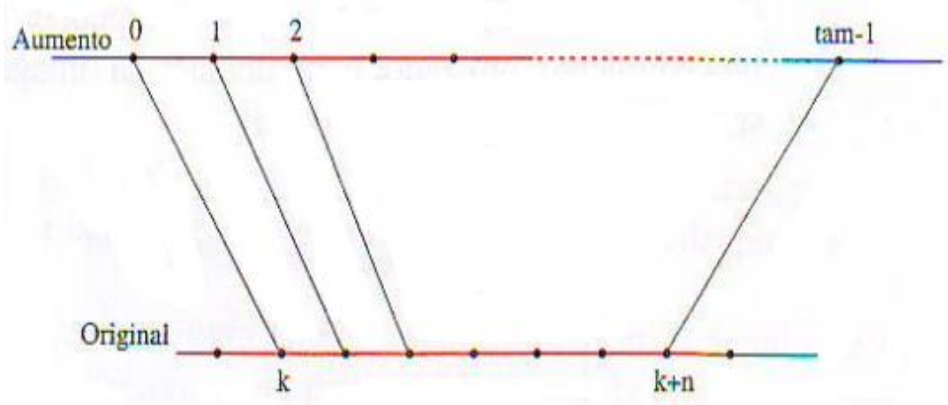

Figura 18: Acción de ampliar

Tomando de: (Pajares, 2002)

5.1.5.3 Escala de grises

El último paso del procesamiento es convertir la cara de RBG (rojo, azul y verde) de acuerdo a las siglas en inglés; para su transformación a escala de grises,

cuya conversión a imágenes de escala de grises no es directa se realiza un ajuste multiplicando a los componentes por tres constantes denominadas: Alfa α , Beta β y Gamma γ. Con este proceso se realiza la sustracción del color por cada pixel dejando un nivel de 0 hasta 255 entre negro y blanco.

Tenemos:

$$G = \begin{pmatrix} (g11, g11, g11 & \dots & (g1n, g1n, g1n) \\ \dots & \dots & \dots \\ (gm1, gm1, gm1) & \dots & (gmn, gmn, gmn \end{pmatrix} \quad \text{(Ecuación 52)}$$

$$g_{ij} = \alpha r_{ij} * \beta g_{ij} * \gamma b_{ij} \quad \text{(Ecuación 53)}$$

$$i = 1,2,3, \dots , \qquad m\ y\ j = 1,2,3, \dots , n. \quad \text{(Ecuación 54)}$$

Donde se obtiene que los valores α, β, γ son contantes:

$$\alpha = 0{,}2999\ , \beta = 0{,}599, \gamma = 0{,}11 \quad \text{(Ecuación 55)}$$

Obteniendo los porcentajes tales como: Rojo: 30%, verde: 59%, azul: 11%, los expertos deducen que es lo más parecido a la intensidad de la luz para cada color. (Norman & Zarate, 2007)

Cuya ecuación de la iluminación se expresa de la siguiente manera:

$$Y = (R * 30\%) + (G * 59\%) + (B * 11\%) \quad \text{(Ecuación 56)}$$

Para la representación de una imagen en escala de grises se utiliza la ecuación anterior, se debe crear una nueva matriz donde cada byte por pixel dará la información correcta de la iluminación.

5.1.6 Preparación

En esta fase se requiere un conjunto de rostros los cuales fueron escogidos aleatoriamente y luego se aplican todos los procedimientos anteriores que se sintetizan en detección de rostro y procesamiento. Los rostros de preparación se

utilizan de imágenes patrón. Una vez finalizada esta fase, los rostros de preparación quedarán, como se indica en la figura 19.

Figura 19: Rostros preparados

5.1.7 Extracción de características

En esta fase se utiliza los algoritmos anteriormente explicados, es necesario, que se analicen las técnicas a profundidad para adaptarlas a la necesidad, cada método necesita requerimientos mínimos; por lo tanto, es recomendable que se utilice el algoritmo Autorostros para el desarrollo de procedimientos rápidos, porque no consume mucho coste computacional, por otra parte si lo que se necesita es un sistema robusto Fisherfaces es un algoritmo al que no le afectan los cambios de iluminación, pero el coste computacional es elevado. En esta fase se empleó un equipo con las siguientes características:

Tabla 3

Características del Equipo.

Características	
Sistema operativo	Windows 10 Pro 64 bits
Modelo del sistema	HP 1000
Procesador	Inter(R) Core(TM) i5-3230M 2.60GHz(4 CPUS)
Memoria	8192 Mb RAM
Disco Duro	500 Gb

5.1.8 Comparación de características y decisión

En esta última fase se realiza la comparación de la información, que se extrae del rostro seleccionado, la misma que es sometida a un proceso de similitud con las imágenes preparadas; dichos resultados son dictaminados mediante el cálculo de la distancia euclidea que sirve para la toma de decisión.

Finalizado el cálculo de las distancias de las proyecciones del rostro a comparar y con el conjunto de imágenes preparadas, se busca en las proyecciones la imagen que esté más cerca, de la imagen de comparación, dicha proyección dará como resultado la mayor similitud con el rostro analizado.

6 FUNCIONAMIENTO DEL SISTEMA

El software Matlab da la facilidad de crear una interface gráfica, gracias a la ayuda de un Guide, que contribuye a una mayor operatividad del manejo del programa, se ha establecido una interface gráfica amigable para el usuario.

En la primera interface se visualiza la portada, en la que se presenta información como: nombre de la universidad, facultad, título de la investigación y creador; en la parte inferior izquierda se encuentra un botón, el cual tiene como función

desplegar la siguiente interface gráfica, que corresponde a un menú del cual se debe elegir el algoritmo deseado. En la figura 20 se observa lo antepuesto.

Figura 20: Portada

Cuando se presiona el botón de inicio, se despliega el siguiente menú, el que tiene por objeto presentar las diferentes opciones de los algoritmos de reconocimiento facial. Este detalle es observado en la figura 21.

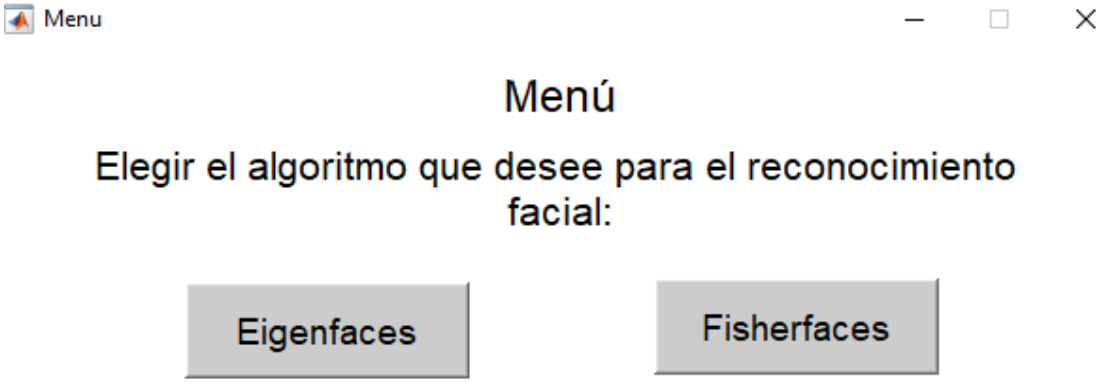

Figura 21: Menú para la selección de algoritmo.

6.1 Interface gráfica

El sistema de reconocimiento facial consta de varias funciones, las mismas que

serán explicadas a continuación.

6.1.1 Sistema de reconocimiento facial Autorostros.

El primer paso a realizar es presionar el botón **encender cámara** para visualizar el rostro de la persona que va a ser registrada en la base de datos o que va a servir para el reconocimiento facial; se enciende el adaptador que hemos acoplado de forma previa a la instalación y la comprobación se visualizar en el recuadro de la izquierda. Un ejemplo de este proceso se observar en la figura 22.

Figura 22: Interface gráfica del programa Eigenfaces

Con la cámara encendida, se procede a la captura de la imagen oprimiendo el botón **tomar foto,** después se visualiza en el recuadro de la derecha (cuadro de fotografía), la imagen. Realizada la captura de la foto el programa procederá a la localización del rostro y hace el enmarcarlo en el recuadro de color azul, es recomendable apagar la cámara si ya estamos conformes con la fotografía. En la figura 23 se observa la evidencia de esta fase.

Figura 23: Captura de fotografía.

Para **añadir un nuevo usuario** a la base de datos, se debe oprimir el botón **guardar usuario**, después se podrá visualizar un mensaje que indicará que nombre le queremos colocar a la imagen para guardarla, como se puede apreciar en la figura 24.

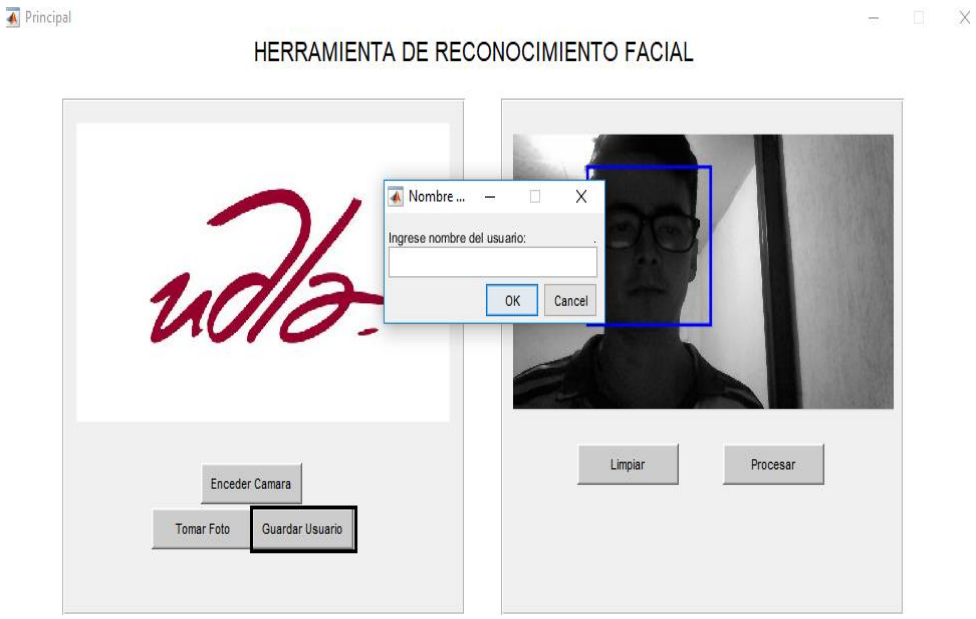

Figura 24: Colocar nombre de usuario al rostro

Una vez colocado el nombre del usuario, se guarda en la base de datos con su respectivo ID; se visualiza un mensaje de terminación del proceso, de esta manera se sabe que se ha cumplido el requisito satisfactoriamente. Como se puede observar en la figura 25.

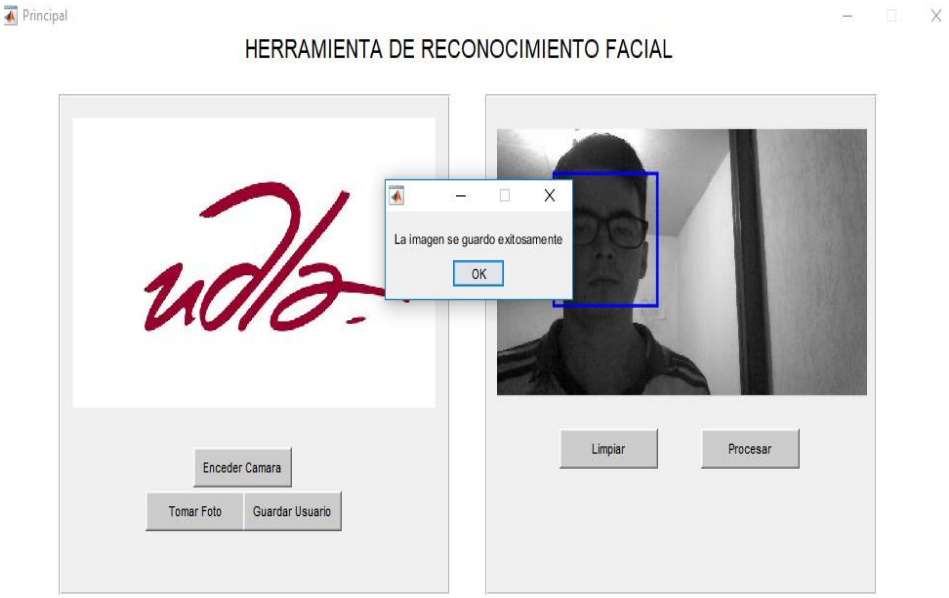

Figura 25: Guardar fotografía correctamente

Guardados los usuarios en la base de datos para la realización del reconocimiento facial se procede a plasmar una nueva captura (fotografía), se oprime el botón **procesar** el cual ejecuta el algoritmo de Autorostros. Este algoritmo está explicado anteriormente, acto seguido se visualiza en un **Text** (recuadro inferior blanco) el nombre con el cual se ha guardado la persona en la base de datos y su distancia euclidea; la fotografía patrón es visualizada en el recuadro de la izquierda y con la foto de comparación visualizada en el recuadro de la derecha se concluye esta fase.

Figura 26: Reconocimiento facial

Para limpiar la base de datos se debe oprimir el botón **Limpiar,** a continuación, se visualiza un mensaje de comprobación. En la figura 27 se detalla este proceso.

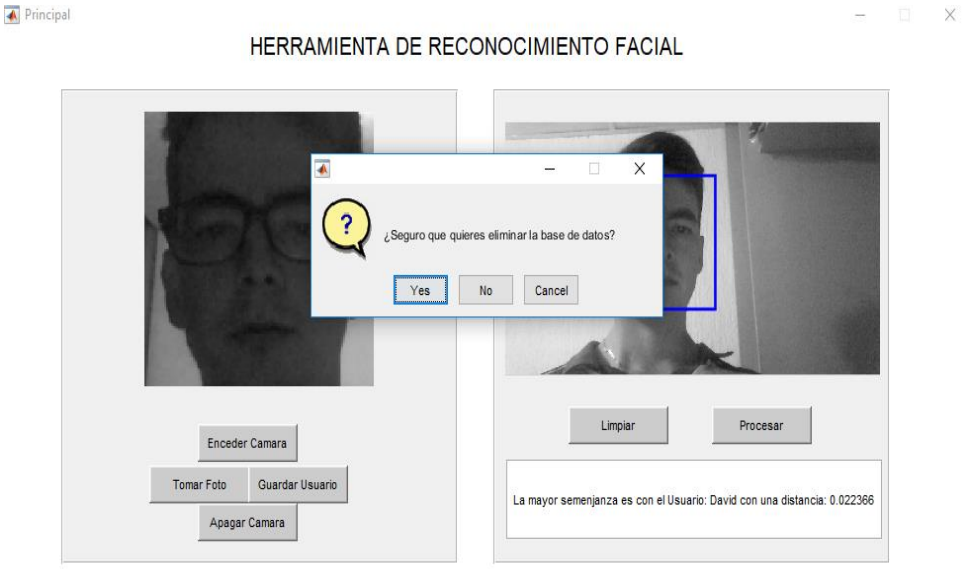

Figura 27: Confirmar limpiar de la base de datos

En la culminación de la limpiada de la base de datos se oprime **Yes,** luego se visualizar el mensaje que el proceso ha sido exitoso. Se comprueba lo ejecutado

en figura 28; si por el contrario no se desea limpiar, se oprime **No.**

Figura 28: Base de datos borrada

6.1.2 Sistema de reconocimiento facial Fisherfaces.

El sistema de identificación de rostros Fisherfaces tiene su interface principal, la cual tiene funciones que a continuación se detallarán; obsérvese en la figura 29 la interface gráfica de inicio del sistema.

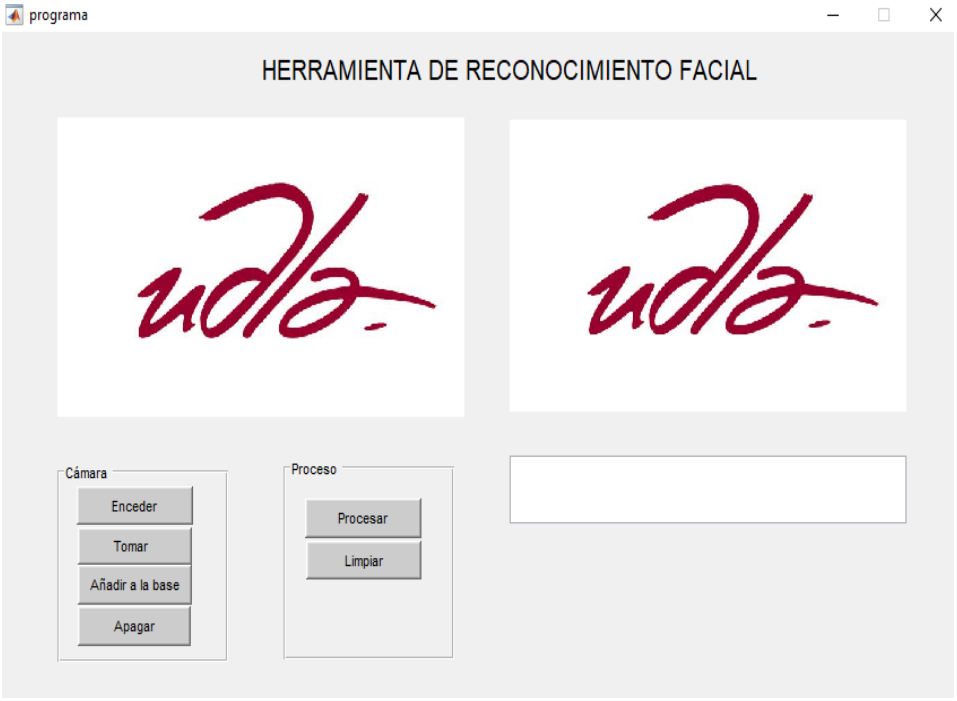

Figura 29: Pantalla principal

El sistema se ha desarrollado con la misma lógica que el programa anterior sistema Autorostros; en primera instancia se debe presionar el botón **encender cámara**, donde se podrá visualizar el video para después realizar la adquisición de la foto que se desee; en la figura 30 se observa la interface.

Figura 30: Encender cámara

Para la captura de una fotografía se debe oprimir el botón **Tomar,** cuya función es la **captura de la imagen**; una vez capturada la imagen se puede visualizar en el recuadro de la derecha, la cual contendrá un cuadrado de color azul para indicar el contorno del rostro a guardar; el detalle de lo realizado se observa en la figura 31.

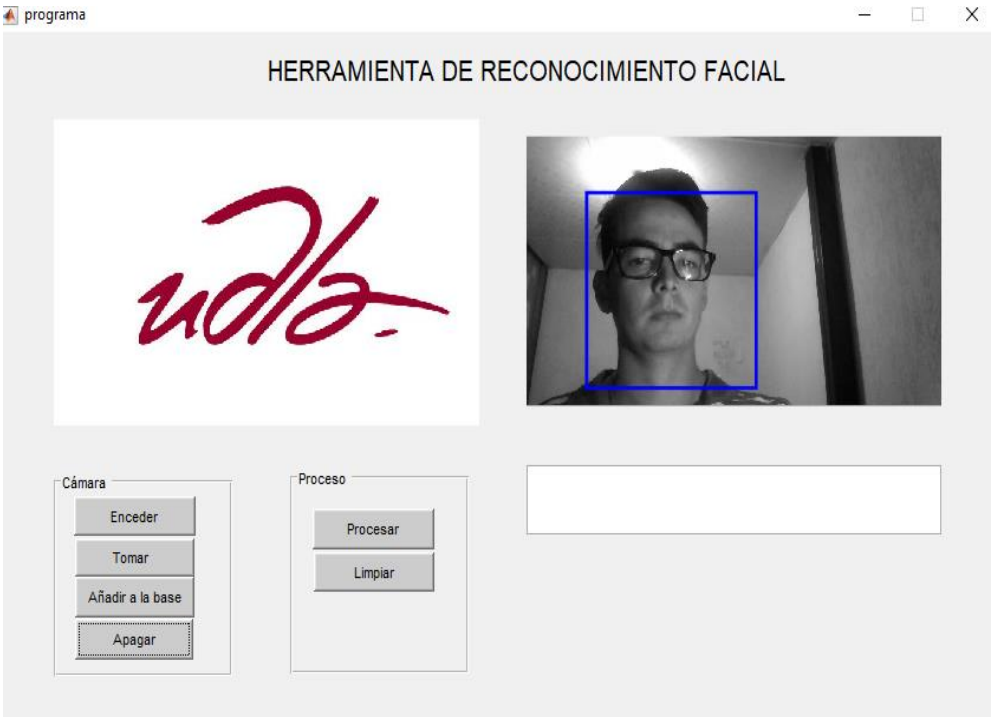

Figura 31: Captura de rostro

Analizados los dos sistemas se establecen diferencias, como:

- En este caso se va a utilizar clases donde van guardando diferentes fotos de la misma persona que corresponde a dicha clase.
- Cada vez que se ingresa un nuevo usuario se incrementa en uno cada clase.

Para añadir el nuevo usuario a la base de datos se debe presionar el botón **Añadir** a la base, donde en la imagen siguiente se visualiza un mensaje que pide registrar a que clase pertenece, para guardar en la base de datos; apréciese lo antes dicho, figura 32.

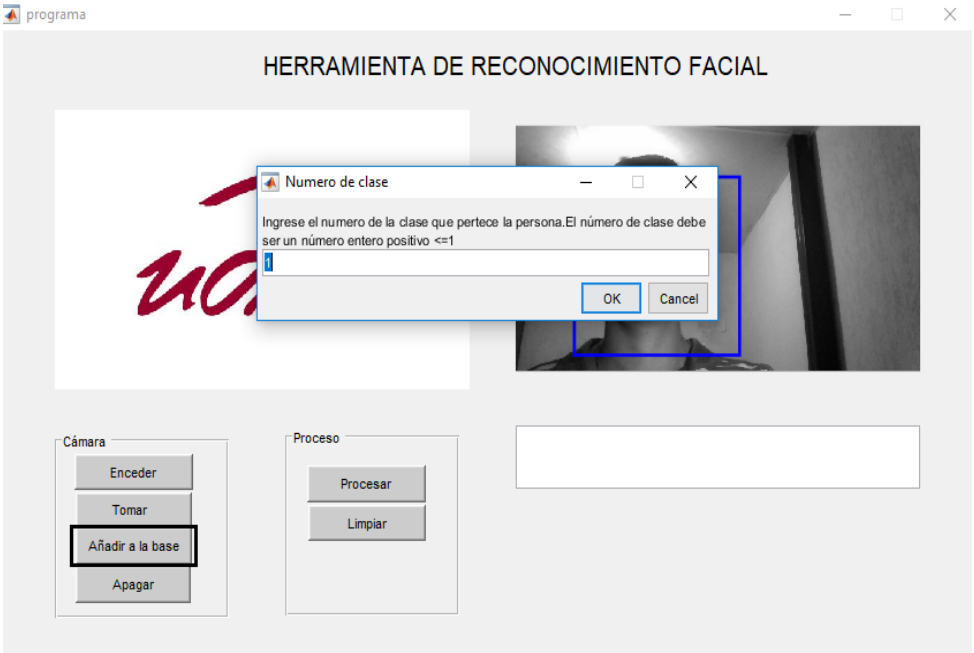

Figura 32: Asignar clase

El mensaje que se observa **es el número de clase,** que se coloca en función del orden de la clase, si el programa no realiza **el proceso de guardado**; por lo que, debe ser la clase un **número positivo y entero.** En caso que no se coloque **el rango de la clase,** se proyecta el aviso de alerta. Un ejemplo de lo anterior dicho se observa en la figura 33.

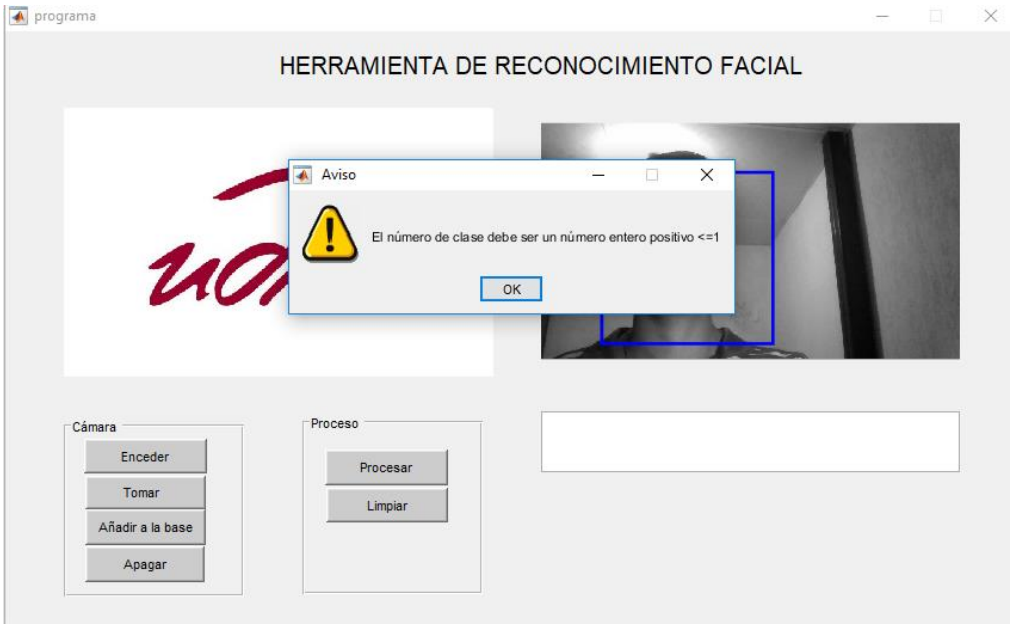

Figura 33: Aviso

El **usuario** que vamos a guardar corresponde a la **primera clase,** al que se considera primer usuario en el sistema**;** por lo tanto**,** si se desea ingresar otro usuario pertenecerá a la **clase dos** y de esta manera los registros serán en forma sucesiva.

Colocada **la clase correctamente** se podrá visualizar el mensaje de comprobación de que la fotografía se ha guardado correctamente en la base de datos; este proceso esta detallado en la figura 34.

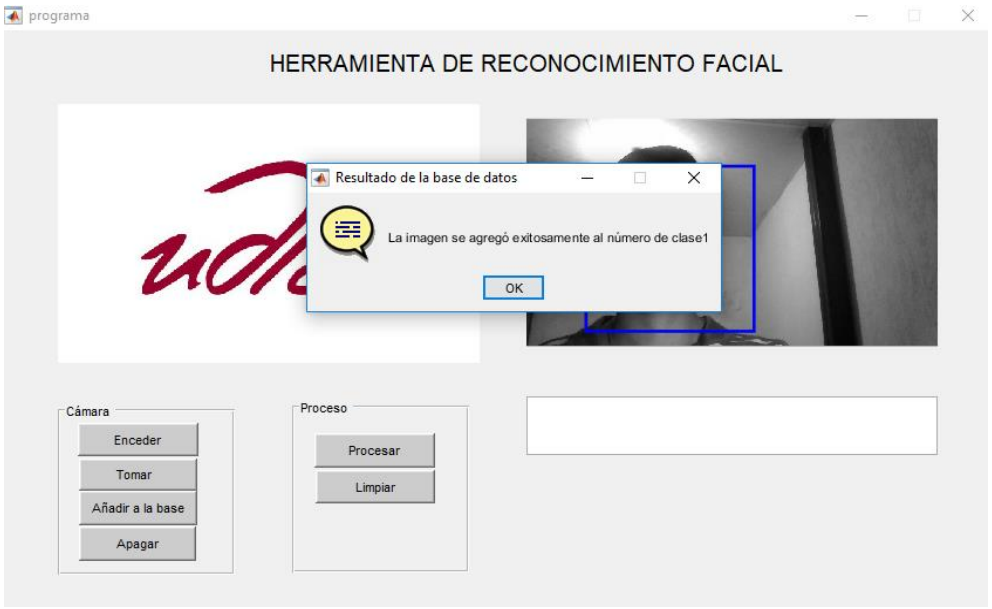

Figura 34: Guardar clase exitosamente

Una vez llena la base de datos con fotografías patrones, se procede a realizar la captura de la fotografía que va a ser identificada como se puede apreciar en la figura 35.

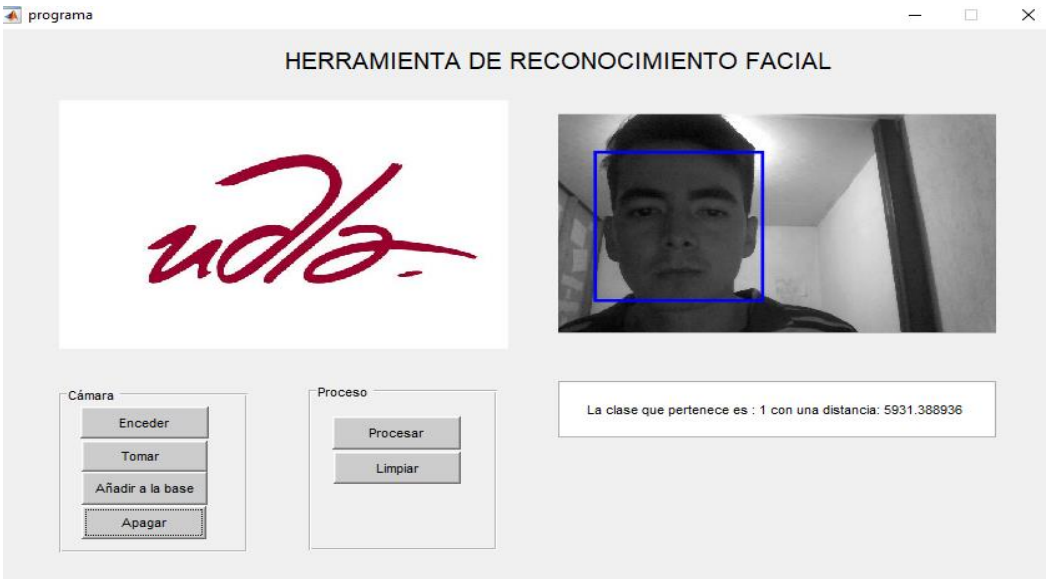

Figura 35: Fotografía para comparación

Se procede a presionar el botón **Procesar** y se aplica el algoritmo Fisherfaces, explicada la funcionalidad de este algoritmo anteriormente; se indica cual es la mayor semejanza y obsérvese el detalle de lo ejecutado en la figura 36.

Del proceso anteriormente descrito se puede determinar que los resultados obtenidos se aprecian en un **Text** (recuadro inferior blanco), donde se observa a la clase que pertenece la foto de la persona, donde el recuadro de la derecha es la persona a identificar y en el recuadro de la izquierda es la persona guardada en la base de datos.

Figura 36: Resultado de la comparación del sistema Fisherfaces.

6.2 Evidencia del funcionamiento de los sistemas.

En esta sección se realiza la comprobación de los algoritmos para diferentes tipos de rostros con los algoritmos planteados

6.2.1 Autorostros

Figura 37: Resultado del algoritmo Autorostros

Figura 38: Resultado del algoritmo Autorostros

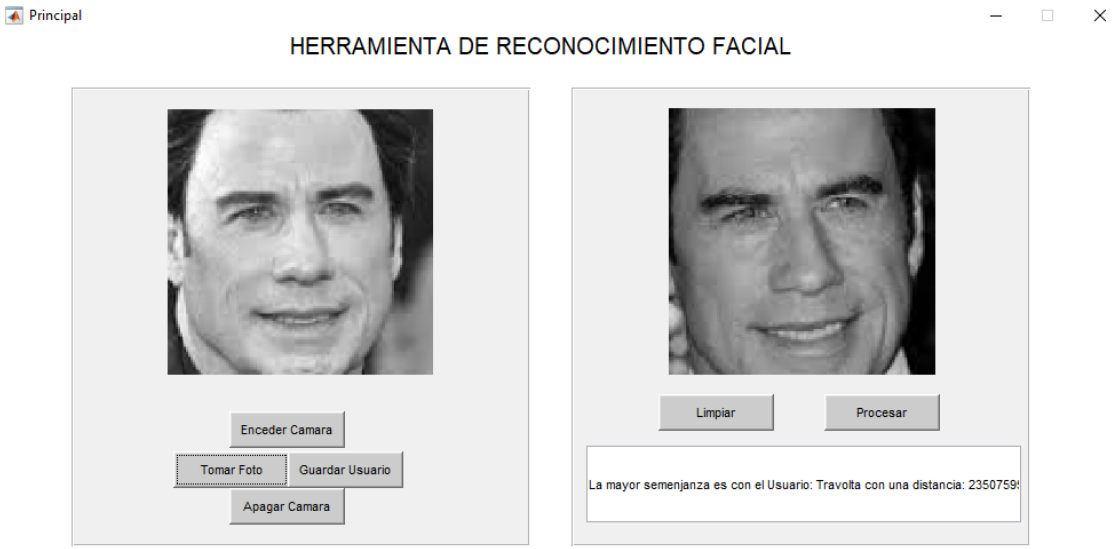

Figura 39: Resultado del algoritmo Autorostros

Figura 40: Resultado del algoritmo Autorostros

Para visualizar más evidencias del proceso de identificación dirigirse al anexo 4 del proyecto

6.2.2 Fisherfaces

Figura 41: Resultado del algoritmo Fisherfaces

Figura 42: Resultado del algoritmo Fisherfaces

Figura 43: Resultado del algoritmo Fisherfaces

Figura 44: Resultado del algoritmo Fisherfaces

Para visualizar más evidencias del proceso de identificación dirigirse al anexo 5 del proyecto

6.2.3 Autorostros con clase

Figura 45: Resultado del algoritmo Autorostros con clases

Figura 46: Resultado del algoritmo Autorostros con clases

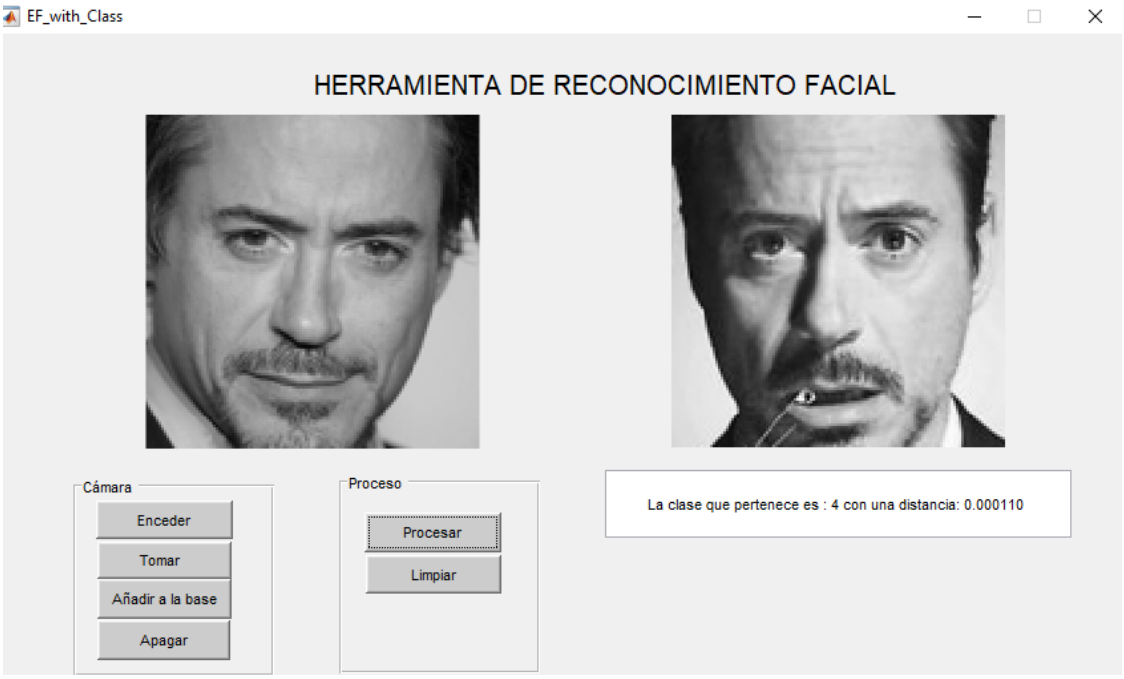

Figura 47: Resultado del algoritmo Autorostros con clases

Figura 48: Resultado del algoritmo Autorostros con clases

Para visualizar más evidencias del proceso de identificación dirigirse al anexo 6 del proyecto

7 EVALUACIÓN DEL SISTEMA

En esta sección se evalúa el rendimiento de los sistemas planteados; los experimentos se llevan a cabo para examinar la efectividad y eficacia de los dos algoritmos, se realiza la experimentación en dos secciones, una con imágenes previamente preparadas e ingresándole ruido para comprobar su robustez; esto quiere decir, imágenes seleccionadas y en la otra sección según el número de vectores característicos de los algoritmos; adicionalmente se prevé realizar una comparación entre ellas.

7.1 Imágenes preparadas.

7.1.1 Autorostros.

En esta fase de la experimentación se observa el rendimiento del algoritmo Autorostros. En la verificación de su efectividad según el número de usuarios

establecidos en la base de datos, se realiza unas pruebas de sensibilidad colocando ruido a la foto de comparación y verificar así su eficacia. El primer experimento se realizó con 20 rostros como se puede apreciar en la figura 49.

{} 20x3 cell

	1	2	3
4	16384x1 uint8	4	'Individuo4'
5	16384x1 uint8	5	'Individuo5'
6	16384x1 uint8	6	'Individuo6'
7	16384x1 uint8	7	'Individuo7'
8	16384x1 uint8	8	'Individuo8'
9	16384x1 uint8	9	'Individuo9'
10	16384x1 uint8	10	'Individuo10'
11	16384x1 uint8	11	'Individuo11'
12	16384x1 uint8	12	'Individuo12'
13	16384x1 uint8	13	'Individuo13'
14	16384x1 uint8	14	'Individuo14'
15	16384x1 uint8	15	'Individuo15'
16	16384x1 uint8	16	'Individuo16'
17	16384x1 uint8	17	'Individuo17'
18	16384x1 uint8	18	'Individuo18'
19	16384x1 uint8	19	'Individuo19'
20	16384x1 uint8	20	'Individuo20'

Figura 49: 20 rostros en la base de datos

A continuación, se establece aspectos relacionados con el funcionamiento operativo como se indica en la figura 50. Es decir, se determina que el algoritmo tiene un correcto desempeño, pero cabe recalcar que tiene una tasa de error de un 2% aproximadamente.

Figura 50: Comprobación con 20 imágenes

En la siguiente prueba con los mismos 20 rostros se ha introducido ruido gaussiano, para comprobar hasta cuanto ruido no infiere en el resultado; por lo tanto, se ha introducido por porcentajes; se utiliza la misma foto de comparación para que no se encuentren alteraciones en los resultados. En la prueba de sensibilidad con ruido gaussiano se le introdujo un 13% de ruido a la foto de comparación, como se aprecia en la figura 51 en el recuadro de la derecha. El sistema funciona correctamente.

Figura 51: Prueba de 13% de ruido gaussiano en 20 rostros

Se puede apreciar en la figura 52 que se ha ingresado un ruido gaussiano superior a 13% y el sistema de reconocimiento ha fallado en la detección facial; cabe recalcar que el programa estaría trabajando con una sensibilidad de un 87% de ruido máximo en 20 rostros.

Figura 52: Falla del sistema

La siguiente prueba se realizará con 60 rostros como se puede observaren la figura 53, la cual tiene por finalidad probar la eficacia del sistema.

60x3 cell

	1	2	3
43	16384x1 uint8	43	'Individuo42'
44	16384x1 uint8	44	'Individuo44'
45	16384x1 uint8	45	'Individuo45'
46	16384x1 uint8	46	'Individuo46'
47	16384x1 uint8	47	'Individuo47'
48	16384x1 uint8	48	'Individuo48'
49	16384x1 uint8	49	'Individuo49'
50	16384x1 uint8	50	'Individuo50'
51	16384x1 uint8	51	'Individuo51'
52	16384x1 uint8	52	'Individuo52'
53	16384x1 uint8	53	'Individuo53'
54	16384x1 uint8	54	'Individuo54'
55	16384x1 uint8	55	'Individuo56'
56	16384x1 uint8	56	'Individuo57'
57	16384x1 uint8	57	'Individuo58'
58	16384x1 uint8	58	'Individuo59'
59	16384x1 uint8	59	'Individuo60'
60	16384x1 uint8	60	'Individuo61'

Figura 53. 60 individuos en la base de datos

Se puede acotar que el algoritmo Autorostros en el proceso de identificación con los 60 individuos es correcto. Su funcionamiento se puede comprobar en la figura 54 y con una tasa de error de un 5% aproximadamente.

Figura 54: Comprobación en 60 individuos.

En cuestión de la sensibilidad tenemos el mismo rango, con tan solo un 13% de ruido gaussiano; en caso de colocar un mayor porcentaje, el programa fallará en el reconocimiento facial.

Para el último caso de estudio del algoritmo se realizará un muestreo con 100 rostros como se observa la figura 55

100x3 cell

	1	2	3
83	16384x1 uint8	83	'Individuo84'
84	16384x1 uint8	84	'Individuo85'
85	16384x1 uint8	85	'Individuo86'
86	16384x1 uint8	86	'Individuo87'
87	16384x1 uint8	87	'Individuo88'
88	16384x1 uint8	88	'Individuo89'
89	16384x1 uint8	89	'Individuo90'
90	16384x1 uint8	90	'Individuo91'
91	16384x1 uint8	91	'Individuo92'
92	16384x1 uint8	92	'Individuo93'
93	16384x1 uint8	93	'Individuo94'
94	16384x1 uint8	94	'Individuo95'
95	16384x1 uint8	95	'Individuo96'
96	16384x1 uint8	96	'Individuo97'
97	16384x1 uint8	97	'Individuo98'
98	16384x1 uint8	98	'Individuo99'
99	16384x1 uint8	99	'Individuo100'
100	16384x1 uint8	100	'Individuo101'

Figura 55: 100 individuos en la base de datos

El sistema tiene una demora en el procesamiento de los rostros, pero no es significativo. El algoritmo con 100 individuos sigue funcionando con total normalidad en la figura 56 y tiene una tasa de error de un 8% aproximadamente.

Figura 56: 100 individuos comprobación

Se realizó la prueba de sensibilidad y dio como resultado el mismo porcentaje, de las anteriores pruebas, se concluye que el sistema tiene una tolerancia al ruido de un 13%.

7.1.2 Fisherfaces

Se desarrollan las pruebas para comprobar el rendimiento del algoritmo; por lo tanto, se utilizan los mismos parámetros que el algoritmo anteriormente comprobado; este proceso se desarrolla con el mismo número de individuos y el mismo porcentaje de ruido gaussiano.

Se emplean los mismos 20 individuos para la experimentación; los datos están registrados en la figura 57.

20x2 cell

	1	2
3	16384x1 uint8	3
4	16384x1 uint8	4
5	16384x1 uint8	5
6	16384x1 uint8	6
7	16384x1 uint8	7
8	16384x1 uint8	8
9	16384x1 uint8	9
10	16384x1 uint8	10
11	16384x1 uint8	11
12	16384x1 uint8	12
13	16384x1 uint8	13
14	16384x1 uint8	14
15	16384x1 uint8	15
16	16384x1 uint8	16
17	16384x1 uint8	17
18	16384x1 uint8	18
19	16384x1 uint8	19
20	16384x1 uint8	20

Figura 57: 20 individuos en la base de datos.

Después de la prueba se determina que el algoritmo sigue funcionando correctamente; esto se puede constatar en la figura 58. Cabe destacar que tiene una mejora significativa y solo el 1% de error aproximadamente en los 20 rostros.

Figura 58: Funcionamiento del algoritmo

Para la realización de la prueba de sensibilidad con un ruido gaussiano del 14% el algoritmo sigue funcionando correctamente; es importante recalcar que el anterior algoritmo tenía un límite de ruido de dicha cantidad; así que, el algoritmo Fisherfaces se pondrá a prueba para determinar hasta cuanto ruido gaussiano no afecta o altera su resultado.

Los resultados de las pruebas son contundentes y se determina que el algoritmo soporta hasta un 30% de ruido gaussiano; de los datos obtenidos se deduce que este algoritmo es mucho más efectivo que el método Autorostros. Por otro lado, el procesamiento del algoritmo es más demoroso que el algoritmo anterior. En la figura 59 se puede apreciar los resultados de la eficacia de este sistema.

Figura 59: Reconocimiento con 30% de ruido

En la siguiente prueba se emplean 60 rostros los cuales se encuentran almacenados en la base de datos, se puede apreciar en la figura 60.

60x2 cell

	1	2
43	16384x1 uint8	43
44	16384x1 uint8	44
45	16384x1 uint8	45
46	16384x1 uint8	46
47	16384x1 uint8	47
48	16384x1 uint8	48
49	16384x1 uint8	49
50	16384x1 uint8	50
51	16384x1 uint8	51
52	16384x1 uint8	52
53	16384x1 uint8	53
54	16384x1 uint8	54
55	16384x1 uint8	55
56	16384x1 uint8	56
57	16384x1 uint8	57
58	16384x1 uint8	58
59	16384x1 uint8	59
60	16384x1 uint8	60

Figura 60: 60 rostros en la base de datos

Para la comprobación de su efectividad del algoritmo con los 60 rostros, que como resultado podemos observar, tiene un buen funcionamiento en la figura 61. Se realizaron pruebas para determinar su eficacia y consta de un 2% de error.

Figura 61: Reconocimiento con 60 individuos

Al realizar la prueba para incluir ruido en la foto de comparación se nota que es el mismo porcentaje, es decir el 30%. Se constata que no existe variación en los resultados.

Para la prueba final se realiza la comparación con 100 rostros como se aprecia en la figura 62, en la base de datos y con la misma foto de comparación, se obtiene resultados.

101x2 cell

	1	2
86	16384x1 uint8	86
87	16384x1 uint8	87
88	16384x1 uint8	88
89	16384x1 uint8	89
90	16384x1 uint8	90
91	16384x1 uint8	91
92	16384x1 uint8	92
93	16384x1 uint8	93
94	16384x1 uint8	94
95	16384x1 uint8	95
96	16384x1 uint8	96
97	16384x1 uint8	97
98	16384x1 uint8	98
99	16384x1 uint8	99
100	16384x1 uint8	100

Figura 62: Cien individuos en la base de datos

En la prueba con 100 individuos se puede apreciar en la figura 63 su funcionamiento correcto y la tasa de error se mantiene con tan solo un 2%; se coloca la misma tasa de ruido y su funcionamiento de identificación es idóneo. Esto quiere decir que el algoritmo soporta una tasa de ruido del 30%.

Figura 63: 100 individuos de comprobación

7.1.3 Autorostros con clases

En este trabajo de investigación se ha realizado una mejora al algoritmo de Autorostros; tal mejora consiste en implementar clases, las cuales tienen la función de agrupar una variedad de fotos de la misma persona y obtener un promedio de dichas fotografías.

Para realizar esta evaluación se contará con 20 rostros de personas las cual tendrán en cada clase 2 fotografías. En la figura 64 observa lo dicho.

{} 40x2 cell

	1	2
25	16384x1 uint8	13
26	16384x1 uint8	13
27	16384x1 uint8	14
28	16384x1 uint8	14
29	16384x1 uint8	15
30	16384x1 uint8	16
31	16384x1 uint8	16
32	16384x1 uint8	16
33	16384x1 uint8	17
34	16384x1 uint8	17
35	16384x1 uint8	18
36	16384x1 uint8	18
37	16384x1 uint8	19
38	16384x1 uint8	19
39	16384x1 uint8	20
40	16384x1 uint8	20

Figura 64: 20 Individuos con 2 fotos

Se aprecia que el algoritmo mejoró, introduciéndole la misma cantidad de ruido que al algoritmo original funciona correctamente. Como resultado de varias pruebas se dictamina cuanto sería el grado de sensibilidad máxima que podría ingresar sin afectar la respuesta; después de varias evaluaciones al sistema, se establece el resultado en un 20% de ruido con tan solo ingresar 2 fotos a cada clase; existe una gran mejora en el sistema de acuerdo a los resultados establecidos en la figura 65.

Figura 65: Algoritmo Eigenfaces mejorado con un 20% de ruido.

7.2 Numero de rostro por clases

En la siguiente sección se realizó pruebas para verificar la efectividad de los algoritmos Fisherfaces y Autorostros con clases el cual es una modificación al algoritmo original, en los cuales se introdujo el mismo número de rostros en cada clase; como se puede apreciar en la siguiente tabla se identifica los parámetros:

Tabla 4

Numero de rostros por clase

Algoritmo	Clases	Rostros por clase	Total de rostros
Fisherfaces y Eigenfaces con clases	10	5	50
	10	10	100

7.2.1 Fisherfaces

Al hacer la prueba con cinco rostros por cada clase en 10 individuos, se evidencia que el sistema mejora contundentemente, después de haber realizado varias pruebas se llega a la conclusión, que el sistema soporta un 50 % de ruido, se aprecia en la figura 66 la evidencia.

Figura 66 : 50% de ruido

Al realizar la última prueba del sistema de reconocimiento en el algoritmo Fisherfaces, la cual consta con 10 rostros y en cada clase después de realizar las evaluaciones de sensibilidad, los resultados arrojan un 60% de ruido en la foto de comparación. Como se puede notar en la figura 67.

Figura 67: 60% de ruido en 100 individuos

7.2.2 Autorostros con clases

Se realiza las pruebas para dictaminar la sensibilidad y se puede apreciar que tiene un buen rendimiento porque con 5 fotos por cada clase, llega a un 35% de ruido en la foto de comparación. Se puede notar en la figura 68.

Figura 68: 35% de ruido en la foto de comparación

En la última prueba del algoritmo se puede notar mejoría, que con 10 fotos en cada clase tiene una tasa de ruido del 45% aproximadamente.

Figura 69:45% de ruido en la foto de comparación.

7.3 Por número de vectores característicos.

Se realizan las pruebas para observar el funcionamiento del sistema según los Eigenvectors; en dichas pruebas se escogió 20 rostros y a los dos algoritmos se colocan el mismo número de vectores característicos para lograr determinar cuál es más eficaz, para después realizar una tabla comparativa con su respectivo análisis.

7.3.1 Autorostros.

Para la prueba se inicia con un vector característico para determinar su eficacia y se observó que su rendimiento disminuyó; se irá incrementando para buscar su correcto funcionamiento; después de realizar pruebas se evidencia que con 30 vectores característicos el programa funciona correctamente; apréciese en la figura 70.

Figura 70: 30 vectores característicos

7.3.2 Autorostros con clases.

Se realizaron pruebas con el algoritmo y los mismos parámetros anteriormente planteados y se determina que, con 25 vectores característicos, arroja un óptimo funcionamiento del sistema; esto determina que es más robusto que el algoritmo anterior.

Figura 71: 25 vectores característicos.

8 RESULTADOS

Realizadas varias pruebas y obtenidos resultados sobre el funcionamiento de los algoritmos, se establece que el nivel de eficiencia es muy alto y su funcionalidad es recomendada en el reconocimiento de rostros de personas; a continuación, se tabulan los datos obtenidos durante el proceso investigativo.

8.1 Confiabilidad del sistema.

Efectuada la etapa de evaluación de los algoritmos y se procede a realizar los cálculos de la efectividad que estos proporcionan; para el efecto, se aplica la siguiente formula.

$$X = \frac{número\ de\ casos\ favorables * 100\%}{total\ de\ intentos\ realizados} \qquad \text{(Ecuación 57)}$$

La comparación se realiza con los aciertos del programa como está indicado en la formula anteriormente mencionada, la cual da los siguientes resultados:

Tabla 5

Resultados de la tasa de error en 20 rostros.

Algoritmo	Casos	Error	Aciertos
Autorostros	20	2%	98%
Fisherfaces	20	1%	99%
Autorostros con clases	20	1%	99%

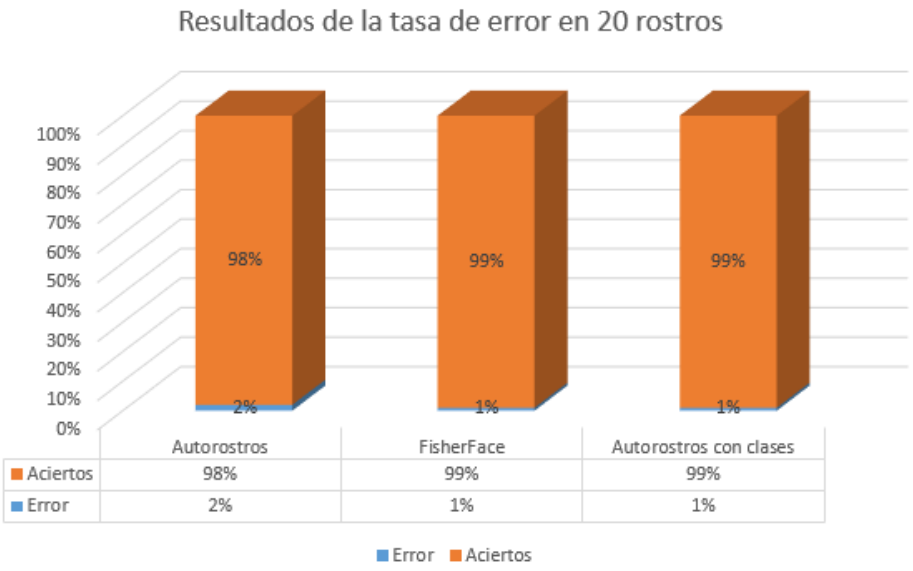

Figura 72: Grafico de 20 rostros con su tasa de error

Se realizó esta prueba para los tres algoritmos planteados con 20 rostros a cada uno, el resultado es contundente que el algoritmo Fisherfaces, Autorostros y el mejorado tienen un buen rendimiento con una tasa de error mínima.

Tabla 6

Resultados de la tasa de error en 60 rostros

Algoritmo	Casos	Error	Aciertos
Autorostros	60	5%	95%
Fisherfaces	60	2%	98%
Autorostros con clases	60	4%	96%

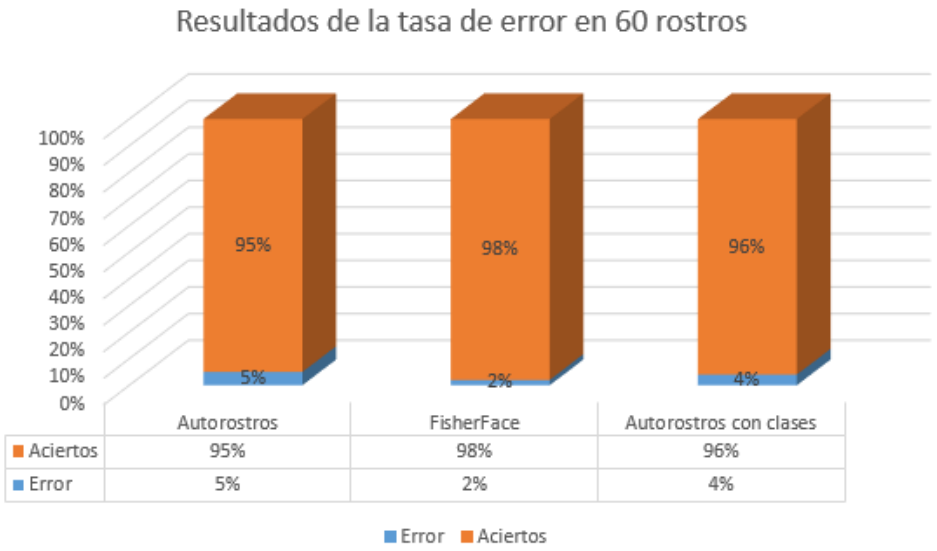

Figura 73: Grafico de 60 rostros con su tasa de error

Se realizó esta prueba para los tres algoritmos planteados con 60 rostros a cada uno, el resultado es contundente que el algoritmo Fisherfaces, Autorostros con clases y sin clases tiene un buen rendimiento con una tasa de error mínima.

Tabla 7

Resultados de la tasa de error en 100 rostros

Algoritmo	Casos	Error	Aciertos
Autorostros	100	8%	92%
Fisherfaces	100	2%	98%
Autorostros con clases	100	6%	94%

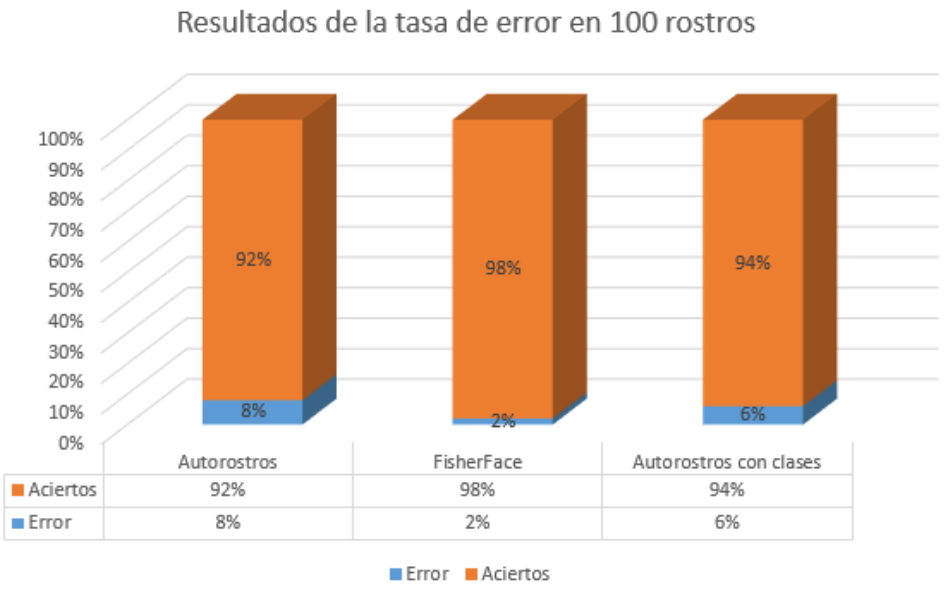

Figura 74: Tasa de error

La comparación se realizó a los aciertos a los sistemas, los resultados determinan un margen de error del 2% al algoritmo Análisis Discriminante Lineal, por lo que se establece que es más robusto.

Tabla 8

Tabla comparativa de sensibilidad

	Autorostros			Fisherfaces	
Casos	Ruido	Sensibilidad	Casos	Ruido	Sensibilidad
20	13%	87%	20	30%	70%
60	13%	87%	60	30%	70%
100	13%	87%	100	30%	70%

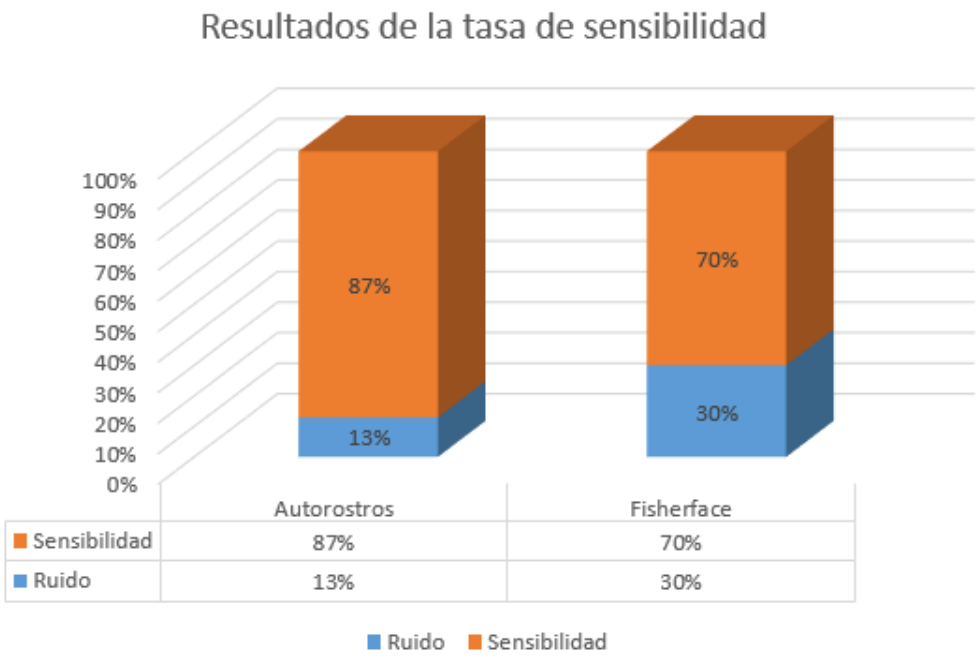

Figura 75: Resultados de tasa de sensibilidad

El resultado de sensibilidad se puede observar en la tabla número 9; el algoritmo Fisherfaces es mucho más robusto y su sistema tolera un 30% de ruido con 100 individuos.

Tabla 9

Niveles de sensibilidad de acuerdo a Autorostros con y sin clases

Autorostros			Autorostros con clases		
Casos	Ruido	Sensibilidad	Casos	Ruido	Sensibilidad
20	13%	87%	20	20%	80%

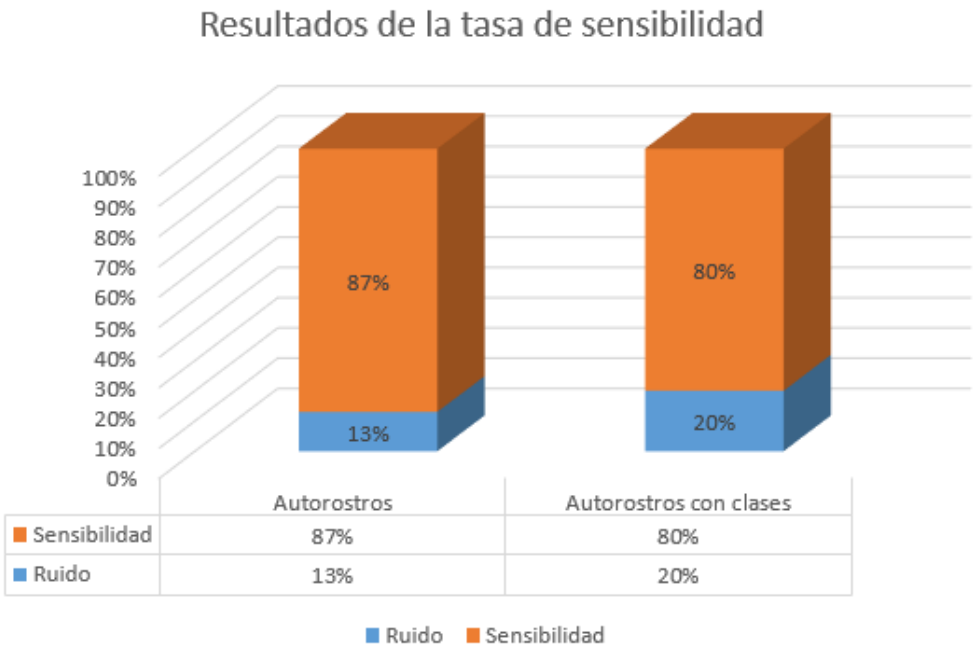

Figura 76: Prueba de sensibilidad

En el algoritmo Autorostros con clases es mucho más robusto que el algoritmo original, se puede apreciar que, ingresando 2 rostros por cada clase, el algoritmo aumenta su sensibilidad notablemente como se observa en la tabla 10

Tabla 10

Por número de clases

Algoritmo	Rostro por clases	Numero de clases	Ruido
Fisherfaces	5	10	50%
	10	10	60%
Autorostros con clases	5	10	35%
	10	10	45%

Se realizó una prueba con el algoritmo Fisherfaces y Autorostros con clases la cual consiste en ingresar una cantidad de rostros en todas las clases para que exista equidad, la cual arroja los resultados, que el algoritmo Fisherfaces es mucho más robusto; el único inconveniente es el coste elevado del procesamiento.

Tabla 11

Número de Eigenvectors

Números	Autorostros	Autorostros con clases
Eigenvectors	Aciertos %	Aciertos %
1	9	11
5	61	74
10	75	80
15	86	91
25	92	100
30	100	100

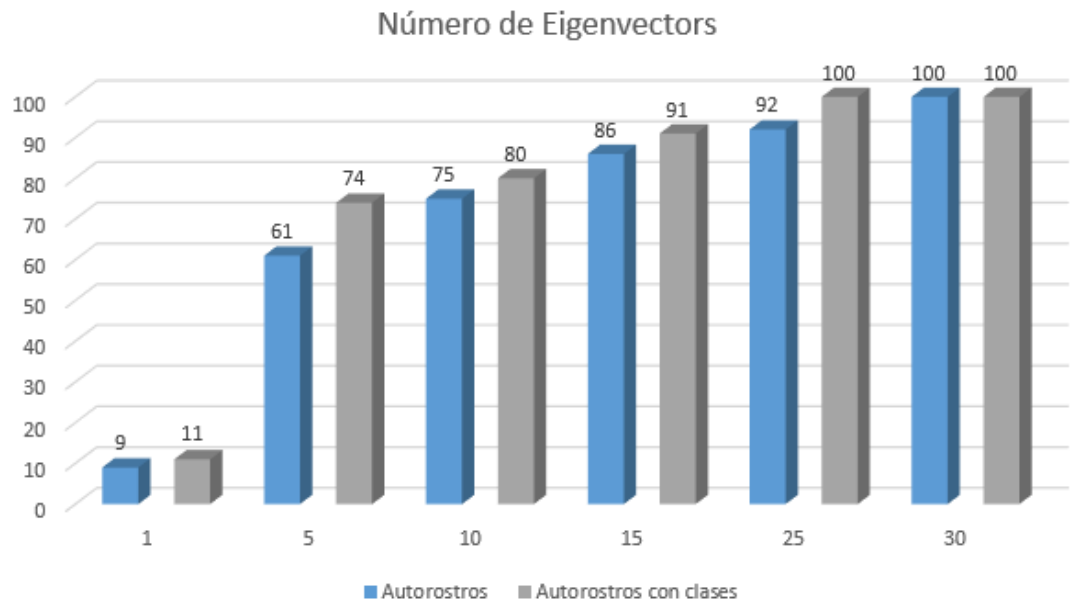

Figura 77: Vectores característicos

Se puede evidenciar que al trabajar con 30 vectores característicos se nota que el sistema tiene efectividad del 100%. Se aprecia que el algoritmo Autorostros con clases tiene una mayor robustez trabajando con tan solo 25 vectores característicos.

9 CONCLUSIONES Y RECOMENDACIONES

9.1 Conclusiones

Los objetivos planteados se cumplieron a cabalidad, se desarrolló un sistema de reconocimiento facial que tiene un alto nivel de efectividad del 99% al momento de comprobar la identificación; esto es, porque realiza la extracción de rasgos característicos por medio de los eigenvectors $\boldsymbol{v}$ y eigenvalues γ.

Al estudiar los algoritmos de reconocimiento facial, se pudo apreciar su complejidad por las fórmulas matemáticas que fueron aplicadas en matrices las cual contienen el rostro. Cada algoritmo tiene sus ventajas y desventajas, con su respectiva tasa de sensibilidad de contaminación de ruido, se realizó una comparación con imágenes previamente preparadas en una base de datos que permitió la obtención de resultados transparentes y exactos

La desventaja del algoritmo de detección de rostros, es la presencia de objetos que obstaculizan la adquisición de rasgos característicos como; sombreros, gorras, gafas y anteojos; estos objetos restan eficacia al sistema y originan errores en la extracción de rasgos característicos.

Al aplicar los algoritmos, los requerimientos de cada método son distintos: Autorostros es una buena opción para un bajo procesamiento con un margen de 92% de aciertos en un grupo de 100 casos; por otra parte, se puede notar que el método Fisherfaces es mucho más compacto debido a su nivel de sensibilidad y por número de usuario con un margen de 98% de aciertos en un grupo de 100 casos, el inconveniente es el uso elevado del coste computacional.

Al comparar los dos algoritmos planteados para el reconocimiento facial que fueron implementados (Autorostros y Fisherfaces), se determina que el algoritmo Fisherfaces, obtuvo un óptimo rendimiento por su mayor porcentaje de aciertos, con una tasa de error del 2% en los 100 casos estudiados.

Los resultados arrojan una variación en el porcentaje de efectividad; esto se debe a diferentes elementos como; iluminación y posición del rostro al instante de realizar la captura. El sistema se ve perjudicado por la mala ubicación de la cámara y la consecuencia es fotografías anacrónicas.

Se realizó una mejora en el algoritmo Autorostros, donde se agregó clases al método y los resultados se denota aumento en la efectividad con un margen de 6% de error en los 100 casos. Este método incrementó su efectividad en un 2% al compararlo con el método original.

Los resultados de efectividad obtenidos de los algoritmos (Autorostros sin clase y con clases), se evidencia que cuando se realiza el incremento de casos de 20 hasta 100 su tasa de error aumenta, en cambio el algoritmo Fisherfaces, es más robusto porque toma en cuenta diversas variables como: S_b y S_w, se maximiza la distribución entre clase y la separación con las clases, en consecuencia se considera esta técnica como más exacta con bajo nivel de error con un 2% y con una tasa de sensibilidad del 30%.

En la prueba de sensibilidad del algoritmo (Autorostros sin clases y con clases), da como resultado que el algoritmo Autorostros sin clases tiene una tolerancia al ruido del 13%, y una sensibilidad del 87%; por otro lado, el algoritmo Autorostros con clases colocado solo 2 fotos en cada clase, su tolerancia al ruido aumenta a un 30%, lo que determina una mejora del 17% en 20 casos de estudio.

Cabe recalcar que los algoritmos Fisherfaces y Autorostros con clases necesitan para su correcto funcionamiento, un gran número de rostros por clases, con diferentes expresiones y diferentes niveles de iluminaciones, para que su tolerancia al ruido se incremente; en las pruebas de sensibilidad con 10 rostros por clases arroga que el algoritmo Autorostros con clases tiene una tolerancia al ruido del 45%, por otra parte, el algoritmo Fisherfaces logra alcanzar una tolerancia al ruido del 60%.

Las pruebas de los dos algoritmos, tanto al Autorostros y Autorostros con clases, se determina que la modificación que se realizó al segundo algoritmo arrojó como resultado un excelente funcionamiento con tan solo 25 vectores característicos, con el consiguiente aumento de la sensibilidad.

9.2 Recomendaciones

En las pruebas se comprobó el funcionamiento de los algoritmos de reconocimiento facial y para tener parámetros equitativos una de las fases prioritarias que influyó en el resultado es: la fase de detección de rostro por lo que se recomienda omitir el uso de gafas, gorras o cualquier otro tipo de objeto que obstaculice la obtención del rostro, por lo tanto, se debe realizar una captura óptima para que las siguientes fases no existan errores en el proceso de identificación.

Se debe tener presente las variaciones lumínicas excesivas, por lo que esto produce sombras en las fotografías al momento de la captura de la imagen, o por el contrario escases de iluminación, lo que afecta a la detección del rostro para realizar la identificación. Lo recomendable es utilizar los algoritmos en un ambiente controlado.

Es fundamental conservar una distancia determinada de la cámara a la persona a fotografiar, para que no exista mucha variación con las fotos patrones almacenadas en la base de datos, esto asegura el correcto funcionamiento del sistema.

Antes de utilizar alguno de los algoritmos para un sistema de reconocimiento de rostros, es recomendable analizar y estudiar las técnicas a profundidad. En la seleccionar la mejor técnica que se adapte según las necesidades y requisitos que se desea.

En el algoritmo Autorostros con clases con 25 eigenvectors, se obtuvo un funcionamiento idóneo su tiempo de respuesta se reduce, el uso del procesador

se reduce en un 40%.

Para el correcto funcionamiento de los algoritmos: Fisherfaces y Autorostros con clases, se recomienda la utilización de una variedad de expresiones faciales y diferentes condiciones de iluminación del rostro en cada clase, se demostró que mientras más rostros de dicha persona se encuentren en su clase, la respuesta es más contundente con una tasa de acierto del 98%.

REFERENCIAS

Albesa, L. A. (2015). *Matlab para matemáticas en ingenierías.* Valencia: Universitar Politécnica de Valencia. Recuperado el 01 de octubre de 2018 de https://riunet.upv.es/bitstream/handle/10251/67728/IPP-Agud%3BPla%20-%20MATLAB%20PARA%20MATEM%C3%81TICAS%20EN%20INGENIERIAS.pdf?sequence=2

Andrés, M. G. (2016). *La movilidad humana, realidad social y jurídica de los migrantes.* Quito. Recuperado el 4 de octubre de 2018 de http://www.dspace.uce.edu.ec/bitstream/25000/7953/1/T-UCE-0013-Ab-386.pdf

Ángel, S. C. (2005). *Aplicaciones de la visión Artificial y la Bioometría informática.* Universidad Rey Juan Carlos. Recuperado el 2 de octubre de 2018 de https://www.urjc.es/estudios/grado/834-vision-artificial

Arcasoptica. (2016). *El ojo humano*. Recuperado de http://www.arcasoptica.com/ojhu.html

Armengot, M. (2006). *Análisis comparativo de métodos basados en subespacios aplicados al reconocimiento de caras.* España: Universidad de Valencia. Recuperado el 5 de octubre de 2018 de https://www.uv.es/marjoari/pdf/definitivo.pdf

Baker, S., & Nayar, S. (1996). *Pattern Rejection.* San Francisco: In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Recuperado el 10 de octubre de 2018 de http://www1.cs.columbia.edu/CAVE/publications/pdfs/Baker_CVPR96.pdf

Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. (1996). *Eigenfaces vs. Fisherfaces: recognition using class specific linear projection.* European Conf. Computer Vision. Recuperado el 10 de octubre de 2018 de https://cseweb.ucsd.edu/classes/wi14/cse152-a/fisherface-pami97.pdf

Bovik, A. (2005). *Handbook Image and Video Processing.* All Bovik, 2ª Edición. Recuperado el 8 de octubre de 2018 de https://preetikale.files.wordpress.com/2018/07/handbook-of-image-and-video-processing-al-bovik1.pdf

Brink, J. S. (1987). *The archaeo-zoology of Florisbad.* Orange Free State. Memoirs van die Nasionale Museum 24: 1–151. Recuperado el 18 de octubre de 2018 de https://www.researchgate.net/publication/249869220_Pollen_indicati

ons_of_Holocene_palaeoenvironments_at_Florisbad_spring_in_the_central_Free_State_South_Africa

Carlini, A. (2018). *Las redes sociales como factor de desestabilización.* España: Instituto Español de Estudios Estratégicos. Recuperado el 10 de octubre de 2018 de http://www.ieee.es/Galerias/fichero/docs_opinion/2018/DIEEEO79-2018_RRSS_FactorDesestabilizacion_ACarlini.pdf

Carrasco, M., Portugal, R., & Peralta, B. (2011). *Reconocimiento biométrico de audio y rostro: un sistema viable de identificación.* Santiago de Chile: Departamento de Ciencia de la Computación Pontificia Universidad Católica de Chile. Recuperado el 11 de octubre de 2018 de https://docplayer.es/22668117-Reconocimiento-biometrico-de-audio-y-rostro-un-sistema-viable-de-identificacion-miguel-a-carrasco-roberto-portugal-billy-peralta.html

De la Fuente Fernández, S. (2011). *Análisis discriminante.* Universidad Autonoma. Recuperado el 12 de octubre de 2018 de http://www.fuenterrebollo.com/Economicas/ECONOMETRIA/SEGMENTACION/DISCRIMINANTE/analisis-discriminante.pdf

Digital, L. (28 de 06 de 2013). *Altamira fue pintada por los primeros 'homo sapiens' europeos*. Recuperado el 1 de octubre de 2018 de https://www.libertaddigital.com/ciencia-tecnologia/ciencia/2013-06-28/altamira-fue-pintada-por-los-primeros-homo-sapiens-europeos-1276494106/

Feldesman, M. R. (2002). *Classification trees as an alternative to linear discriminant analysis.* Oregon: Wiley-Liss, Inc. Recuperado el 1 de octubre de 2018 de https://pdfs.semanticscholar.org/3c5b/ff7d54400add7a7149691a9e2f06eedc7842.pdf

Gámez, C. (2009). *Diseño y Desarrollo de un Sistema de Reconocimiento de caras.* Madrid: Universidad Carlos III de Madrid. Recuperado el 2 de octubre de 2018 de https://e-archivo.uc3m.es/bitstream/handle/10016/5831/PFC_CarmenVirginia_Gamez_Jimenez.pdf

Giménez, L. L. (1998). *Representación de caras mediante Eigenfaces.* Buran Nº11. Recuperado el 1 de octubre de 2018 de https://upcommons.upc.edu/bitstream/handle/2099/9853/Article005.pdf

González, R., & Woods R. (2008). *Digital Imagen Processing.* Prentice Hall, 3ª Edisión. Recuperado el 30 de de noviembre de 2018 de http://web.ipac.caltech.edu/staff/fmasci/home/astro_refs/Digital_Imag

e_Processing_2ndEd.pdf

Gualdrón, O. E., Suárez, O. M., & Rojas, M. A. (2013). *Diseño De Un Sistema De Reconocimiento De Rostros Mediantela Hibridación De Técnicas De Reconocimiento De Patrones, Visión Artificial E Ia, Enfocado A La Seguridad E Interacción Robótica Social.* Revista Mundo FESC, 3(6), 16-28. Recuperado el 9 de octubre de 2018 de http://www.fesc.edu.co/Revistas/OJS/index.php/mundofesc/article/view/3

Haralick, & Shapiro. (2005). *Computer and robot vision.* Addison-Wesley Longman Publishing Co., Inc. Recuperado el 11 de octubre de 2018 de https://dl.acm.org/citation.cfm?id=573190

Hernández, M., & Plasencia, Y. (2016). *Aprendizaje de métrica para el reconocimiento de rostros a partir de imágenes de baja resolución.* Cuba: Revista Cubana de Ciencias Informáticas Vol. 10, num 1. Recuperado el 15 de de octubre de 2018 de https://upcommons.upc.edu/bitstream/handle/2099.1/9782/PFC_RogerGimeno.pdf

Hernández, W., & Mendez, A. (2018). *Application of Principal Component Analysis to Image.* IntechOpen. Recuperado el 9 de de octubre de 2018 de https://cdn.intechopen.com/pdfs/59936.pdf

Informática, S. (2010). *Reconocimiento biométrico a través de la palma de la mano*. Recuperado de https://seguinfo.wordpress.com/category/biometria/page/5/

Jain A.K. (1986). *Fundamentals of digital image processing.* California: Thomas Kailath. Recuperado el 30 de octubre de 2018 de https://www.springer.com/gp/book/9780387710402

Jain, A. K., Flynn, P., & Ross, A. (2008). *Handbook of biometrics.* New York: Springer. Recuperado el 15 de octubre de 2018 de http://ultra.sdk.free.fr/docs/DxO/Fundamentals%20of%20Digital%20Image%20Processing.pdf

Marín, C. P. (2014). *Óptica Fisiolófica.* Madrid: Universidad Complutense Madrid. Recuperado el 30 de septiembre de 2018 de http://eprints.sim.ucm.es/14823/1/Puell_%C3%93ptica_Fisiol%C3%B3gica.pdf

Mark S., & Nixon, A. (2002). *Feacture Extration and Image Processing.* Aguado Newnes. Recuperado el 17 de octubre de 2018, de http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.375.6848&rep=rep1&type=pdf

Martínez, R. C. (2015). *Software para la detección y el reconocimiento de rostros.* Universidad autónoma de Barcelona (UAB). Recuperado el 11 de octubre de 2018 de https://ddd.uab.cat/pub/tfg/2016/tfg_49339/Software_para_la_detecci o_n_y_el_reconocimiento_de_caras.pdf

MathWorks. (2018). *imcrop*. Recuperado el 18 de octubre de 2018 de https://la.mathworks.com/help/images/ref/imcrop.html

MathWorks. (2018). *imresize*. Recuperado el 21 de octubre de 2018 de https://la.mathworks.com/help/images/ref/imresize.html

MathWorks. (2018). *rgb2gray*. Recuperado el 17 de octubre de 2018 de https://la.mathworks.com/help/matlab/ref/rgb2gray.html

MathWorks, T. (2018). *vision.CascadeObjectDetector System object*. Recuperado el 13 de octubre de 2018 de https://la.mathworks.com/help/vision/ref/vision.cascadeobjectdetector -system-object.html

Moreno, A., & Aguilar, J. (2003). *Modelo Óptico de Estéreo Visión usando Visión por Computadora.* México: ULSA. Recuperado el 17 de octubre de 2018 de https://www.researchgate.net/publication/242168746_Modelo_Optico _de_Estereo_Vision_usando_Vision_por_Computadora

MyHPSupport. (2019). Notebook HP 1000-1220LA: Especificaciones del producto. *HP Development Company*, 2-4. Recuperado el 19 de octubre de 2018 de https://support.hp.com/es-es/document/c03509072

Ottado, G. (2010). *Reconocimiento de caras: Eigenfaces y Fisherfaces.* Uruguay: Guillermo Ottado. Recuperado el 20 de octubre de 2018, de https://eva.fing.edu.uy/file.php/514/ARCHIVO/2010/TrabajosFinales2 010/informe_final_ottado.pdf

Pavón, S. D. (2017). *Reconocimiento facial mediante el Análisis de.* Sevilla: Dep. Teoría de la Señal y Comunicaciones. Recuperado el 29 de octubre de 2018 de http://bibing.us.es/proyectos/abreproy/91426/fichero/TFG_SARA_DO MINGUEZ_PAVON.pdf

Perez, S. (2011). *Guía sobre las tecnologías biométricas aplicadas a la seguridad.* España. Recuperado el 18 de octubre de 2018 de http://nidoapp.com/files/guia_tecnologias_biometricas_web.pdf

Pùblica. (27 de 07 de 2018). *Falsificación de cédulas y suplantación de identidad en Ecuador, un problemática aún latente.* Recuperado el 2 de octubre

de 2018 de https://www.publicafm.ec/noticias/actualidad/1/falsificacion-cedulas-suplantacion-identidad-ecuador

Serrano, J. F., Díaz, A. B., & Calle, Á. S. (2015). *Visión por computador.* Universidad Rey Juan Carlos, 6ª Edición. Recuperado el 29 de octubre de 2018, de https://www.urjc.es/estudios/grado/834-vision-artificial

Serratosa, F. (2013). *La biometría para la identificación de las personas.* España: España de Creative Commons. Recuperado el 19 de octubre de 2018, de https://docplayer.es/11145275-La-biometria-para-la-identificacion-de-las-personas.html

Technology, N. S. (2012). *History, Biometrics.* Recuperado de http://www.biometrics.gov/documents/biohistory.pdf

The MathWorks, I. (2018). *Installing the Support Packages for Image Acquisition Toolbox Adaptors*. Recuperado el 1 de octubre de 2018 de https://la.mathworks.com/help/imaq/installing-the-support-packages-for-image-acquisition-toolbox-adaptors.html

Torres, T. (2016). *Biografía de Joseph Nicéphore Niépce.* Laurate. Recuperado el 29 de octubre de 2018 de https://www.academia.edu/22020270/Biograf%C3%ADa_de_Joseph_Nic%C3%A9phore_Ni%C3%A9pce

Turk, M. A., & Pentland, A. P. (1991). *Face recognition using eigenfaces. In Computer Vision and Pattern.* IEEE Computer Society Conference. Recuperado el 10 de octubre de 2018 de https://www.cs.ucsb.edu/~mturk/Papers/mturk-CVPR91.pdf

Turk, M., & Petland, A. (1991). *Eigenfaces for Recognition.* Journal of Cognitive Neurosience, vol 3, num 1. Recuperado el 20 de octubre de 2018 de http://www.vision.jhu.edu/teaching/vision08/Handouts/case_study_pca1.pdf

UNIVERSO, E. (2018). *Suplantación de identidad.* Recuperado el 1 de octubre de 2018 de https://www.eluniverso.com/opinion/2018/07/17/nota/6863126/suplantacion-identidad

Vázquez, M. Á. (2014). *Sistema de Reconocimiento Facial Mediante.* Guanajuato: Centro de investigacion de optica, A.C. Recuperado el 23 de octubre de 2018, de https://bibliotecas.cio.mx/tesis/15950.pdf

Viola, P., & Jones, M. (2001). *Rapid object detection using a boosted cascade of simple features, In Computer.* Proceedings of the 2001 IEEE

Computer Society Conference. Recuperado el 1 de octubre de 2018, de https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf

ANEXOS

Anexo 1: Código del sistema Autorostros

- Encender cámara

```
function btn_encender_Callback (hObject, eventdata, handles)

%Encender cámara%
%Instalar el winwineo previamente
global win
set (handles.btn_apagar,'visible','on');
set (handles.btn_encender,'visible','off');
set (handles.btn_foto,'visible','on');
set (handles.btn_crear,'visible','on');

%prender el axes1
set (handles.axes1,'visible','on');
axes(handles.axes1);

%para poder utilizarlo esto establece el mismo Matlab el formato
win=wineoinput('winwineo',1);

%para abrir en nuestro GUIDE la camaraweb
%color de la foto
win.ReturnedColorSpace='rgb';

%resolución por defecto
winRes= get(win,'wineoResolution');
nBands=get(win,'NumberOfBands');
hImage=image (zeros (winRes (2), winRes (2), nBands));
preview (win, hImage);
```

- Tomar fotografía

```
function btn_foto_Callback (hObject, handles)

set (handles.btn_apagar,'visible','on');
set (handles.btn_foto,'visible','on');
set (handles.btn_procesar,'visible','on');
set (handles.btn_limpiar,'visible','on');

%--------------Tomar foto%
global win
foto=getsnapshot(win)
imagen_gray=rgb2gray(foto);

%detectar el rostro
facedetector=vision.CascadeObjectDetector();
BBOX=step (facedetector, foto);
```

```
%recortar el rostro
recorte=imcrop (imagen_gray, BBOX (1, :));

%convertir la dimensión de la imagen recortada en una de dimensión
%128x128
%rostro=imresize (recorte,[128 128]);
%rostro = imnoise(rostro,'salt & pepper',0.10);
%rostro=imnoise(rostro,'gaussian',0.30);
%rostro=imnoise(rostro,'speckle',0.35);
axes(handles.axes2);
imshow(recorte)

%Se coloca un rectángulo en la imagen de color azul de tamaño dé la
%línea de 2
rectangle ('position', BBOX (1, :),'edgecolor','b','linewidt',2);

%esto es para enviar la imagen del botón tomar foto a el botón
%procesar para que se aplique el algoritmo
handles.img=rostro;
guidata (hObject, handles);
```

- Crear usuario

```
function btn_crear_Callback (hObject, eventdata, handles)
%lee la imagen que tome
%img es una variable matriz que contiene la foto que se va a meter a
%la base de datos. Por ejemplo, en una foto determinada voy a tomar
una %foto de Alfredo y se va a guardar en la variable img. Luego se
%realiza el proceso correspondiente y se vuelve a guardar una foto en
%img.
img=handles.img;

%en esta variable guardara el total de rostros
numb_rostros=0;

%el id del rostro para guardar en la base de datos
ID=0;

%se visualizará un mensaje donde se debe ingresar el nombre
mensaje='Ingrese el nombre del usuario:                    .';

%si la imagen existe entra en la función
if exist('img')

%carga la base de datos
load('face_db_EigenFace.dat','-mat');

%-------------------------------------------------------------
%número de fotos aumento 1
numb_rostros=numb_rostros+1;
```

```
% el id de los usuarios es incrementado
ID=ID+1;

%guardo la foto en vector columna en mi variable data con su
respectivo ID
data{numb_rostros,1} =img (:);
data{numb_rostros,2} =ID;

%nombre=get (handles. edit2,'String');
prompt= {strcat (mensaje,'`, num2str(nombre))};

%título de la ventana
title='Nombre        ';

%número de líneas
lines=1;

%default se coloque el 1
def={''};

%recibo mi respuesta
respuest=inputdlg(prompt,title,lines,def);
respuest =char(respuest);

if isempty(respuest)
mensaje=questdlg('Ingrese el nombre del usuario
','Mensaje','OK','OK');

if strcmp(mensaje,'OK')
prompt={strcat(mensaje,'',num2str(nombre))};
%título de la ventana
title='Nombre                ';

%número de líneas
lines=1;
%default se coloque el 1
def={''};

%recibo mi respuesta
respuest=inputdlg(prompt,title,lines,def);
respuest =char(respuest);
end
end

data{numb_rostros,3}=respuest;
%save ayuda a guardar en la base de datos
save('face_db_EigenFace.dat','data','numb_rostros','ID','nombre');

%mensaje
msgbox(strcat('La imagen se guardó exitosamente' ))
```

```
%limpiar mi variable
clear('img')
else
warndlg(strcat('Tome nuevamente la foto ','Aviso'))
end
```

- Algoritmo de Autorostros

```
function btn_procesar_Callback(hObject, eventdata, handles)

set (handles.editext ,'visible','on');
set (handles.btn_encender ,'visible','on');

%se envía la foto que se encuentra captura en la variable img.
%-------------------------------------------------------------
img=handles.img;
%-------------------------------------------------------------
%cargar la base de datos

load('face_db_EigenFace.dat','-mat');
%se crea la matriz que contendrá todos los rostros
matriz=zeros(size(data{1,1},1),numb_rostros);

for ii=1:numb_rostros
matriz(:,ii)=im2double(data{ii,1});
end

mean_face=mean(matriz,2);

%Le quito el promedio a todas mis imágenes de mi base de datos
for ii=1:numb_rostros
matrizT(:,ii)=matriz(:,ii)-mean_face;
end

[evectors,score,evalues]=princomp(matriz');
%el número de vectores característicos

num_eigevectors=30;
evectors=evectors(:,1:num_eigevectors);
features=evectors'*matrizT;
rostro=imresize(img,[128,128]);

%se le da una doble escala
rostro=im2double(rostro);

feacture_vec=evectors'*(rostro(:)-mean_face);
similarity_score=arrayfun(@(n)1/(norm(features(:,n)-
feacture_vec)),1:numb_rostros);
```

```
%se calcula la imagen mayor similitud
%match_1x es el índice de la foto con mayor similitud
[match_score,match_1x]=max(similarity_score);

for n=1:numb_rostros

if(match_1x==data{n,2})
rostro=data{n,1};
break;
end

end

rostro=reshape(rostro,[128 128]);
axes(handles.axes1);
imshow(rostro);
editext=sprintf('La mayor semejanza es con el Usuario: %s con una
distancia: %f',data{n,3}, match_score );
set(handles.editext,'String',editext);
%end
```

- Apagar la cámara

```
function btn_apagar_Callback(hObject, ~, handles)

%set (handles.axes1,'visible','off');
set (handles.btn_encender ,'visible','on');
set (handles.btn_apagar ,'visible','off');
set (handles.btn_crear ,'visible','on');
set (handles.btn_foto ,'visible','on');

fondo=imread('C:\Users\David J\Documents\MATLAB\ejemplo\foto.jpg');
closepreview

axes(handles.axes1);
imshow(fondo);
```

- Limpiar base de datos

```
function btn_limpiar_Callback(hObject, eventdata, handles)

button = questdlg('¿Seguro que quieres eliminar la base de datos?');

if strcmp(button,'Yes')
data=[];

numb_rostros=0;
ID=0;
nombre='';
```

```
save('face_db_EigenFace.dat','data','numb_rostros','ID','nombre');
fondo=imread('C:\Users\David J\Documents\MATLAB\ejemplo\foto.jpg');
closepreview

axes(handles.axes1);
imshow(fondo);

axes(handles.axes2);
imshow(fondo);
msgbox ('La base de datos está vacía', 'Eliminar base de
datos','help');
end

set (handles.editext ,'visible','off');
set (handles.btn_encender,'visible','on');
```

Anexo 2: Código del sistema Autorostros mejorado

- Encender cámara.

```
function pushbutton3_Callback(hObject, eventdata, handles)

global vid
axes(handles.axes1);

vid=videoinput('winvideo',1);
%para abrir en nuestro GUIDE la camaraweb
%color de la foto

vid.ReturnedColorSpace='rgb';

%resolución por defecto
vidRes= get(vid,'videoResolution');
nBands=get(vid,'NumberOfBands');
hImage=image(zeros(vidRes(2),vidRes(2),nBands));

preview(vid,hImage);
```

- Tomar fotografía

```
function pushbutton2_Callback(hObject, eventdata, handles)

%-------
foto=getsnapshot(vid)

imagen_gray=rgb2gray(foto);

%detectar el rostro
facedetector=vision.CascadeObjectDetector();
BBOX=step(facedetector,foto);

%recortar el rostro
recorte=imcrop(imagen_gray,BBOX(1,:));

%convertir la dimensión de la imagen recortada en una de dimensión
%128x128
rostro=imresize(recorte,[128 128]);

rostro=imnoise(rostro,'gaussian',0.35);
axes(handles.axes2);
imshow(rostro)

rectangle('position',BBOX(1,:),'edgecolor','b','linewidt',2);
```

```
handles.img=rostro;
guidata(hObject,handles);


end
```

- Crear usuario

```
function pushbutton1_Callback(hObject, eventdata, handles)

messaggio='Ingrese el número de la clase que perece la persona.';

%img es una variable matriz que contiene la foto que se va a meter a
la base
%de datos. Por ejemplo, en una foto determinada voy a tomar una foto
de
%Alfredo y se va a guardar en la variable img. Luego se realiza el
proceso
%correspondiente y se vuelve a guardar una foto en img.
img=handles.img;
numb_rostro=0;

%Inicializamos nuestra variable numb_rostro en 1 la cual contendrá la
%cantidad de fotos. en otras palabras, la cantidad de
objetos(personas)
%diferentes que habrá en la base de datos.

clase=1; % se inicializa la variable clase con 1 y se meten tantas
%fotos de la misma persona como una desea dentro de cada clase
%y, por lo tanto, cuando cambiemos de persona la variable se aumentará
%en 1
%y se repite el proceso. Y cada clase contendrá n fotos
%de la persona que representa.

%si la imagen que hemos elegido existe o no
if exist('img')

%cargamos la base de datos que contendrá nuestras fotos.
load('eigen_class_db.dat','-mat');
%número de fotos se aumentará en 1 cada vez que carguemos en la base
%de datos
numb_rostro=numb_rostro+1;

%primero convertimos en vector columna a nuestra imagen que está en la
variable img
%y después guardamos en la variable data el vector img. La variable
data{m,n}
%representa un array muldimensional, el cual tendrá en el elemento m,n
(es decir la
```

```
%columna n y fila m) a un vector o una matriz de la dimensión que sea.

data{numb_rostro,1}=img(:); %Aquí estoy poniendo en la fila
numb_rostro y columna 1 de data estoy metiendo
%la imagen guarda en el vector img

%mensaje donde ingresar la clase que pertenece la imagen y
%mi variable clase me indica el número de la clase actual
prompt={strcat(messaggio,'El número de clase debe ser un número entero
positivo <= ',num2str(clase))};

%título de la ventana
title='Numero de clase';

%número de líneas
lines=1;

%default se coloque el 1
def={'1'};

%recibo mi respuesta
answer=inputdlg(prompt,title,lines,def);

%transformó mi variable a double
parametro=double(str2num(char(answer)));

%comprobación si parametro es diferente de 0
if size(parametro,1)~=0

%numb_class es igual a mi variable parametro
numb_class=parametro;

%comprobar que la clase sea positiva
if
(numb_class<=0)||(numb_class>clase)||(floor(numb_class)~=numb_class)||
(~isa(numb_class,'double'))||(any(any(imag(numb_class))))
warndlg(strcat('El número de clase debe ser un número entero positivo
<= ',num2str(clase)),' Aviso ')
else

%compruebo que mi clase este se vaya aumentando en 1
if numb_class==clase;

%mi máxima clase aumente en 1
clase=numb_class+1;
end

%graba la base de datos mi clase
data{numb_rostro,2}=numb_class;

%save ayuda a guardar en la base de datos
```

```
save('eigen_class_db.dat','data','numb_rostro','clase');

%mensaje
msgbox(strcat('La imagen se agregó exitosamente al número de clase
',num2str(numb_class)),'Resultado de la base de datos','help');

%limpia mi variable img que contiene la imagen actual
clear('img')
end
else
warndlg(strcat('El número de clase debe ser un número entero positivo
<= ',num2str(clase)),' Aviso ')
end
end
```

- Algoritmo de Autorostros mejorado

```
function pushbutton5_Callback(hObject, eventdata, handles)

img=handles.img;
%comprobación si existe

if exist('img')
%-------------------------------------------------------------
%comprobar que existe la base de datos y cargar
load('eigen_class_db.dat','-mat');

%-------------------------------------------------------------
L = length(img(:));
media_clases = zeros(L,clase-1);

persona = zeros(clase-1,1);
for ii=1:numb_rostro
%saco la clase de mi variable data y la guardo en mi variable ID que
%contendrá mis clases
ID = data{ii,2};
%se va sumando las clases según el id de la clase
media_clases(:,ID) = media_clases(:,ID)+double(data{ii,1});
persona(ID) = persona(ID)+1;
end

%se calcula la media por clases
for ii=1:(clase-1)
media_clases(:,ii)=media_clases(:,ii)/persona(ii);
end

%-------------------------------------------------------------
matriz=zeros(size(data{1,1},1),(clase-1));

mean_face=mean(media_clases,2);
```

```
%le quito la media a todas mis imágenes de todas las clases
for ii=1:(clase-1)
matriz(:,ii)=media_clases(:,ii)-mean_face;
end

[evectors,score,evalues]=princomp(media_clases');

num_eigevectors=20;
evectors=evectors(:,1:num_eigevectors);

features=evectors'*matriz;
rostro=imresize(img,[128,128]);

%se le da una doble escala
rostro=double(rostro);

feacture_vec=evectors'*(rostro(:)-mean_face);
similarity_score=arrayfun(@(n)1/(norm(features(:,n)-
feacture_vec)),1:(clase-1));
%se calcula la imagen mayor similitud
%match_1x es el índice de la foto con mayor similitud

[match_score,match_1x]=max(similarity_score);
for n=1:numb_rostro

if(match_1x==data{n,2})
rostro=data{n,1};
break;
end

end

rostro=reshape(rostro,[128 128]);
axes(handles.axes1);
imshow(rostro);
edit1=sprintf('La clase que pertenece es : %i con una distancia: %f
',match_1x, match_score);
set(handles.edit1,'String',edit1);
end
```

- Limpiar la base de datos

```
function pushbutton6_Callback(hObject, eventdata, handles)

button = questdlg('¿Seguro que quieres limpiar la base de datos?');
if strcmp(button,'Yes')
data=[];
numb_rostro=0;
max_class=0;
```

```
save('eigen_class_db.dat','data','numb_rostro','max_class');
fondo=imread('C:\Users\David J\Documents\MATLAB\ejemplo\foto.jpg');
closepreview

axes(handles.axes1);
imshow(fondo);
axes(handles.axes2);
imshow(fondo);
msgbox ('La base de datos está vacía', 'limpiar base de
datos','help');
end
```

- Apagar cámara.

```
function pushbutton4_Callback(hObject, eventdata, handles)

fondo=imread('C:\Users\David J\Documents\MATLAB\ejemplo\foto.jpg');
closepreview
axes(handles.axes1);

imshow(fondo);
```

Anexo 3: Código del sistema Fisherfaces.

- Encender cámara

```
function btn_encender_Callback(hObject, eventdata, handles)

global vid
axes(handles.axes1);

vid=videoinput('winvideo',1);
%para abrir en nuestro GUIDE la camaraweb
%color de la foto
vid.ReturnedColorSpace='rgb';

%resolución por defecto
vidRes= get(vid,'videoResolution');
nBands=get(vid,'NumberOfBands');
hImage=image(zeros(vidRes(2),vidRes(2),nBands));
preview(vid,hImage);
```

- Tomar fotografía

```
function btn_cargar_Callback(hObject, eventdata, handles)

clc;

foto=getsnapshot(win)
%Si la selección es diferente de cero entra en el bucle para mostrar
la %imagen
if archivo~=0

%la función strcat con concatena la dirección con la imagen para
%visualizarla con la función imshow.
axes(handles.axes2);
imshow(img);

%envió la imagen a el siguiente con la función handles
handles.img=img;
guidata(hObject,handles);
else

%caso contrario si la imagen no es señalada saldrá el mensaje de
%seleccione una imagen
warndlg('Seleccione una imagen.',' Aviso ')
end
```

- Crear usuario

```
function btn_base_Callback(hObject, eventdata, handles)
```

```
mensaje='Ingrese el número de la clase que pertenece la persona.';

%img es una variable matriz que contiene la foto que se va a meter a
la base
%de datos. Por ejemplo, en una foto determinada voy a tomar una foto
de
%Alfredo y se va a guardar en la variable img. Luego se realiza el
proceso
%correspondiente y se vuelve a guardar una foto en img.
img=handles.img;
num_rostro=0;

%Inicializamos nuestra variable num_rostro en 1 la cual contendrá la
%cantidad de fotos. en otras palabras, la cantidad de
objetos(personas)
%diferentes que habrá en la base de datos.

clase=1; % se inicializa la variable clase con 1 y se meten tantas
fotos de
%la misma persona como una desea dentro de cada clase
%y, por lo tanto, cuando cambiemos de persona la variable se aumentará
en 1
%y se repite el proceso. Y cada clase contendrá n fotos
%de la persona que representa.

%si la imagen que hemos elegido existe o no
if exist('img')

%cargamos la base de datos que contendrá nuestras fotos.
load('db_Fisher.dat','-mat');

%número de fotos se aumentará en 1 cada vez que carguemos en la base
de datos
num_rostro=num_rostro+1;

%primero convertimos en vector columna a nuestra imagen que está en la
variable img
%y después guardamos en la variable data el vector img. La variable
data{m,n}
%representa un array muldimensional, el cual tendrá en el elemento m,n
(es decir la
%columna n y fila m) a un vector o una matriz de la dimensión que sea.
data{num_rostro,1}=img(:); %Aquí estoy poniendo en la fila num_rostro
y columna 1 de data estoy metiendo
%la imagen guarda en el vector img

%%%%%%%%
%mensaje donde ingresar la clase que pertenece la imagen y
%mi variable clase me indica el numero de la clase actual
```

```
prompt={strcat(mensaje,'El número de clase debe ser un número entero
positivo <= ',num2str(clase))};

%título de la ventana
title='Numero de clase';

%número de líneas
lines=1;

%default se coloque el 1
def={'1'};

%recibo mi respuesta
answer=inputdlg (prompt, title, lines, def);

%transformó mi variable a double
parameter=double(str2num(char(answer)));

%comprobación si parameter es diferente de 0
if size(parameter,1) ~=0

%numb_class es igual a mi variable parameter
numb_class=parameter;

%comprobar que la clase sea positiva
if (numb_class<=0) ||(numb_class>clase) ||(floor(numb_class)
~=numb_class) ||(~isa(numb_class,'double'))
||(any(any(imag(numb_class))))
warndlg (strcat ('El número de clase debe ser un número entero
positivo <= ', num2str(clase)),' Aviso ')

else
%compruebo que mi clase este se vaya aumentando en 1

if numb_class==clase;
%mi máxima clase aumente en 1
clase=numb_class+1;
end

%graba la base de datos mi clase
data{num_rostro,2} =numb_class;

%save ayuda a guardar en la base de datos
save('db_Fisher.dat','data','num_rostro','clase');

%mensaje
msgbox (strcat ('La imagen se agregó exitosamente al número de clase
', num2str(numb_class)),'Resultado de la base de datos','help');

%limpiar mi variable img que contiene la imagen actual
clear('img')
```

```
end

else
warndlg (strcat ('El número de clase debe ser un número entero
positivo <= ', num2str(clase)),' Aviso ')
end
end
```

- Algoritmo Fisherfaces

```
function pushbutton4_Callback (hObject, eventdata, handles)

%envió mi imagen
img=handles.img;

%comprobación si existe la imagen
if exist('img')

%------------------------------------------------------------
%convierto mi imagen en im2double y vector columna
ingreso=im2double (img (:));

%cargo la base de datos que contiene mis fotos
load('db_Fisher.dat','-mat');

%------------------------------------------------------------
% EIGENFACES reducción
%crear una matriz de ceros donde num_rostro es la cantidad de fotos
que van a estar dentro de la base de
%datos. Por ejemplo: la base de datos contiene 10 fotos de David, 10
de Estaban, 10 fotos de Alfredo y
%10 fotos de Fabricio y en este caso
%num_rostro sería igual a 40.
matriz=zeros (size (data {1,1},1), num_rostro);
%Transformar todas mis imágenes
% dadas en la variable matriz multidimensional data {} en im2double
%y luego guardar cada una de mis imágenes transformadas en im2double
como vectores columna en mi variable matriz
% por lo tanto en cada columna de mi variable matriz tengo las
imágenes de mi base de
% datos en im2double.
for ii=1:num_rostro
matriz (: ii) =im2double(data{ii,1});
end

%Sumo todos los valores de mis matriz imágenes de mi base de datos
mediante la %función sum
suma=sum(matriz,2);
```

```
%Le saco el promedio a todas las imágenes
media=suma/num_rostro;

%Le quito el promedio a todas mis imágenes de mi base de datos
for ii=1:num_rostro
matriz (: ii) =matriz (: ii)-media;
end

matriz=matriz/sqrt(num_rostro);
%La variable matriz que contiene todas mis imágenes menos la media,
%le multiplico por su transpuesta. Por lo tanto, lo que estoy
obteniendo la %varianza de la matriz
matrizT=matriz'*matriz;

%se calculan los egivectors y valores propios de mi variable matrizT
[V, D] = eig(matrizT);

%calculo el determinante si es una de los autovalores (D)
if det(D)~=0

%la función abs devuelve el valor absoluto.
Vtrue=matriz*V*(abs(D)) ^-0.5;

else
%si la matriz es singular multiplico por los eigenvectors
Vtrue=matriz*V;
end
%Guardó la diagonal de mi variable D que contiene mi autovalores
Dtrue=diag(D);

%ordena los eigenvalues
[Dtrue, ordine]=sort(Dtrue);
%cambia el orden de la fila
%nuestra fila se ordena de abajo así arriba como se muestra en el
ejemplo: %a= [1; 2; 3] ... flipud[a]= [3;2;1]
Dtrue=flipud(Dtrue);

%doy la vuela a mi matriz
ordine=flipud(ordine);
Vtrue(:1:num_rostro) =Vtrue (: ordine);
Vtrue=Vtrue (:1: clase-1);
Dtrue=Dtrue (1: clase-1);

Vtrue0 = Vtrue;
Dtrue0 = Dtrue;

%-------------------------------------------------------------
% método FISHERFACES
%length me retorna la longitud de mi vector columna
L = length(media);
```

```
Sb = zeros (L, L); % entre-class
Sw = zeros (L, L); % con-class
%creamos mi variable mean_classes con ceros la cual tendrá en el
numero
%de columnas el número de clases creadas y en las filas tendrá nuestro
vector L
%cual tiene una longitud de 16384.
%La variable mean_classes va a contener el promedio de todas las
%clases
mean_classes = zeros (L, clase-1);

%creo mi variable persona que contendrá el total de imágenes por cada
%clase
persona = zeros(clase-1,1);

for ii=1:num_rostro
%saco la clase de mi variable data y la guardo en mi variable ID que
%contendrá mis clases
ID = data{ii,2};

%Primero transformó mis valores de la base de datos a im2double, luego
se van %sumando cada clase por su ID cuya variable tiene la clase a
que pertenece cada imagen.
mean_classes(:ID) = mean_classes(:ID) +im2double(data{ii,1});

%se sumarán el total de fotos que exista en cada clase
persona(ID) = persona(ID)+1;
end
%se calcula el promedio por clase

for ii=1:(clase-1)
mean_classes (: ii) = mean_classes (: ii) /persona(ii);
end
% Sb computación

for ii=1:(clase-1)
v = mean_classes (: ii)-media;
Sb = Sb + v*v';
end

% Sw computación
%se le quita la media a cada imagen según su clase
for ii=1:num_rostro
ID = data{ii,2};
v = im2double(data{ii,1})-mean_classes(:ID);
Sw = Sw + v*v';
end

Sbr = Vtrue0'*Sb*Vtrue0;
Swr = Vtrue0'*Sw*Vtrue0;
```

```
[V, D] = eig (Sbr, Swr);

Vtrue = Vtrue0*V;
Dtrue = diag(D);

Dtrue, ordine] = sort(Dtrue);
Dtrue = flipud(Dtrue);

ordine = flipud(ordine);
nordine = length(ordine);
Vtrue (:1: nordine) = Vtrue (: ordine);

Vtrue=Vtrue (:1: clase-1);
Dtrue=Dtrue (1: clase-1);

lengthV = size(Vtrue,2);
for ii=1: lengthV

if norm (Vtrue (: ii)) ~=0 && norm (Vtrue (: ii)) ~=Inf
Vtrue (: ii) =Vtrue (: ii) /norm (Vtrue (: ii));
end
end

pesi=Vtrue'*(ingreso-media);

pesi_database = zeros(clase-1,clase-1);
pesi_database_media = zeros (clase-1, clase-1);

numero_elemento_clase =zeros(clase-1,1);
for ii=1:num_rostro
ingresar_database=im2double(data{ii,1});
clase_database=data{ii,2};
posion_corrent=Vtrue'*(ingresar_database-media);

pesi_database(:,clase_database)=pesi_database(:,clase_database)+posion
_corrent;

numero_elemento_clase(clase_database)=numero_elemento_clase(clase_data
base)+1;
end
for ii=1:(clase-1)

pesi_database_media(:,ii)=pesi_database(:,ii)/numero_elemento_clase(ii
);
end

distanze_pesi=zeros(clase-1,1);
for ii=1:(clase-1)
distanze_pesi(ii)=norm (pesi-pesi_database_media (: ii));
end
```

```
[minima_pesiion, posicion]=min(distanze_pesi);

for n=1:num_rostro

if (posicion ==data{n,2})
rostro=data{n,1};
break;
end

end

rostro=reshape (rostro, [128 128]);
axes (handles. axes1);
imshow(rostro);

edit1=sprintf ('La clase que pertenece es: %i con una distancia: %f ', posicion, minima_pesion);
set (handles.edit1,'String', edit1);

else
warndlg ('No es posible el procesamiento de imágenes. La base de datos está vacía.',' Aviso ')
end
```

- Limpiar base de datos

```
function btn_limpiar_Callback (hObject, eventdata, handles)

button = questdlg ('¿Seguro que quieres eliminar la base de datos?');
if strcmp(button,'Yes')
data= [];
num_rostro=0;
max_class=0;
save('db_Fisher.dat','data','num_rostro','max_class');

fondo=imread ('C:\Users\David J\Documents\MATLAB\ejemplo\foto.jpg');
closepreview
axes(handles.axes1);

imshow(fondo);
axes(handles.axes2);
imshow(fondo);

msgbox ('La base de datos está vacía', 'Eliminar base de datos','help');
end
```

- Apagar cámara

```
function pushbutton7_Callback (hObject, eventdata, handles)

fondo=imread ('C:\Users\David J\Documents\MATLAB\ejemplo\foto.jpg');
closepreview
axes(handles.axes1);
imshow(fondo);
```

Anexo 4: Evidencia del funcionamiento del algoritmo Autorostros.

Figura 78: Resultado del algoritmo Autorostros

Figura 79: Resultado del algoritmo Autorostros

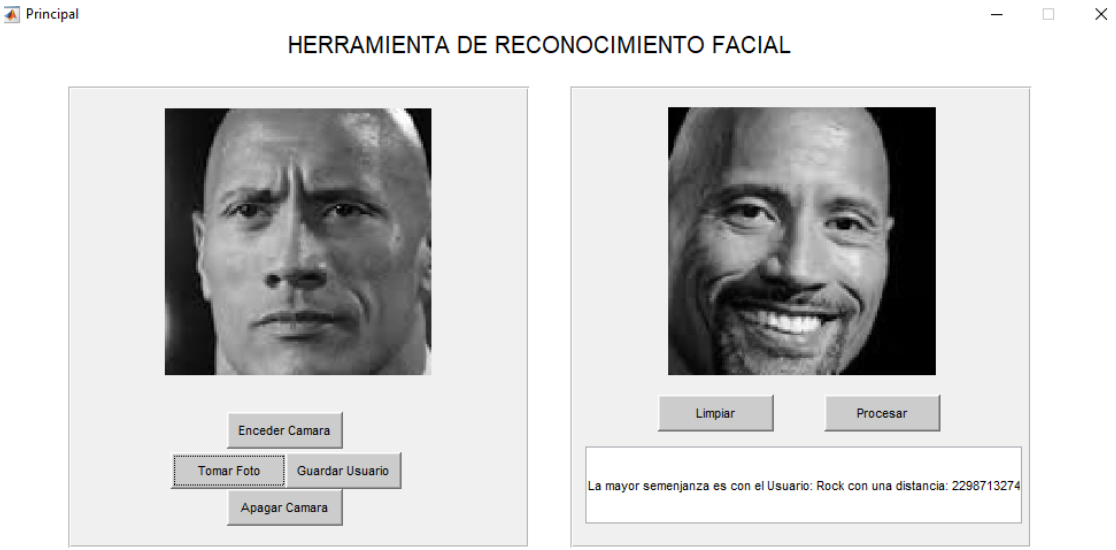

Figura 80: Resultado del algoritmo Autorostros

Figura 81: Resultado del algoritmo Autorostros

Anexo 5: Evidencia del funcionamiento del algoritmo Fisherfaces.

Figura 82: Resultado del algoritmo Fisherfaces

Figura 83: Resultado del algoritmo Fisherfaces

Figura 84: Resultado del algoritmo Fisherfaces

Anexo 6: Evidencia del funcionamiento del algoritmo Autorostros con clases.

Figura 85: Resultado del algoritmo Autorostros con clases

Figura 86: Resultado del algoritmo Autorostros con clases

Figura 65: Resultado del algoritmo Autorostros con clases

Figura 87: Resultado del algoritmo Autorostros con clases