UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE CUENCA

CARRERA DE INGENIERÍA ELECTRÓNICA

Trabajo de titulación previo
a la obtención del título
de Ingeniero Electrónico

PROYECTO TÉCNICO

DESARROLLO DE UN SOFTWARE DE SEGURIDAD PARA DETECCIÓN Y RECONOCIMIENTO FACIAL BASADO EN LOS ALGORITMOS DE VIOLA-JONES Y PCA EIGENFACE

Autores:

Christian Xavier Niola Quito
Wiliam Armando Sanango Zhinin

Tutor:

Ing. Walter Humberto Orozco Tupacyupanqui, PhD.

CUENCA - ECUADOR

2019

CESIÓN DE DERECHOS DE AUTOR

Nosotros, Christian Xavier Niola Quito con documento de identificación N° 0104886817 y Wiliam Armando Sanango Zhinin con documento de identificación N° 0302298831, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación**: DESARROLLO DE UN SOFTWARE DE SEGURIDAD PARA DETECCIÓN Y RECONOCIMIENTO FACIAL BASADO EN LOS ALGORITMOS DE VIOLA-JONES Y PCA EIGENFACE**, mismo que ha sido desarrollado para optar por el título de: *Ingeniero Electrónico*, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia, suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, marzo del 2019

Christian Xavier Niola Quito
CI: 0104886817
AUTOR

Wiliam Armado Sanango Zhinin
CI: 0302298831
AUTOR

CERTIFICACIÓN

Yo declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: **DESARROLLO DE UN SOFTWARE DE SEGURIDAD PARA DETECCIÓN Y RECONOCIMIENTO FACIAL BASADO EN LOS ALGORITMOS DE VIOLA-JONES Y PCA EIGENFACE**, realizado por Christian Xavier Niola Quito y Wiliam Armando Sanango Zhinin, obteniendo el *Proyecto Técnico* que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana.

Cuenca, marzo del 2019

Ing. Walter Humberto Orozco Tupacyupanqui PhD.

CI: 0102523974

TUTOR DEL TRABAJO DE TITULACIÓN

DECLARATORIA DE RESPONSABILIDAD

Nosotros, Christian Xavier Niola Quito con número de cédula 0104886817 y Wiliam Armando Sanango Zhinin con número de cédula 0302298831, autores del trabajo de titulación: **DESARROLLO DE UN SOFTWARE DE SEGURIDAD PARA DETECCIÓN Y RECONOCIMIENTO FACIAL BASADO EN LOS ALGORITMOS DE VIOLA-JONES Y PCA EIGENFACE**, certificamos que el total contenido del *Proyecto Técnico*, es de nuestra exclusiva responsabilidad y autoría.

Cuenca, marzo del 2019

Christian Xavier Niola Quito
CI: 0104886817
AUTOR

Wiliam Armado Sanango Zhinin
CI: 0302298831
AUTOR

AGRADECIMIENTOS

Quiero dar gracias a Dios quien me ha acompañado y guiado en momentos de debilidad, por brindarme fortaleza espiritual y así no rendirme a lo largo de mi carrera.

A mis padres, quienes en todo momento creyeron en mí y con su esfuerzo me dieron la oportunidad de tener una excelente formación académica, a mi hermano que con sus palabras me alentaba a seguir adelante y a mi tía por sus inmensas ayudas.

A la Universidad Politécnica Salesiana, que a través de sus docentes obtuve los conocimientos necesarios de tan maravillosa carrera.

También quiero agradecer de manera especial al PhD. Walter Orozco tutor de nuestro trabajo de titulación y a todas las personas que me brindaron su apoyo.

Christian Niola

Agradezco a mis padres por ser los principales promotores de este logro alcanzado, por su apoyo incondicional, por los consejos y principos que me supieron dar a lo largo de mi formacion academica. A los docentes de la Universidad Politécnica Salesiana Sede Cuenca por haber compartido sus conocimientos, y en especial al PhD. Walter Orozco por el apoyo brindado durante todo este proyecto. Y a todas las personas que, de forma directa o indirecta fueron parte fundamental en mi formacion académica y personal.

Wiliam Sanango

DEDICATORIAS

Con orgullo y amor dedico a mi madre Blanca Esthela Quito, quien siempre estuvo a mi lado apoyándome de manera incondicional sin pedir nada a cambio y por ser el pilar fundamental de mi vida y formación académica. De igual manera a mi padre Julio Cesar Niola, que supo apoyarme y estará muy orgulloso de este gran paso. También a mi hermano Julio Fernando que siempre creyó en mí.

Christian Niola

Dedico este trabajo a Dios por darme las fuerzas necesarias para seguir adelante. A cada miembro de mi familia, en especial a mis padres Rafael y María por ser un ejemplo y pilar fundamental en mi vida, ya que gracias a sus consejos, amor, trabajo y sacrificio diario ayudaron en mi formacion profesional. A mis hermanos Danilo y Alexandra que siempre me apoyaron y creyeron en mí. Y a todas las personas que confiaron en mi capacidad y me apoyaron hasta el final.

Wiliam Sanango

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

En este proyecto se desarrolla un software de seguridad para detección y reconocimiento facial bajo condiciones controladas, como una solución biométrica, el cual permite comprobar la identidad de una persona en base a sus características faciales. La detección permite localizar rostros dentro de una imagen de prueba y, el reconocimiento se basa en comparar el rostro detectado, con otras que han sido guardadas en una base de datos, para lograr esto utilizamos los algoritmos de Viola-Jones y PCA Eigenface respectivamente. Estas dos técnicas son dependientes entre sí, ya que un reconocimiento facial exitoso necesita una buena detección.

Primero, se realizó un estudio del estado del arte sobre sistemas de seguridad basados en técnicas biométricas (huella digital, geometría de la mano, iris, facial, firma, voz), dando énfasis a los sistemas de reconocimiento facial. Adicional a esto se analizó las características más importantes de los algoritmos de Viola-Jones y PCA Eigenface para su implementación.

Segundo, se elaboró una base de datos empleando el algoritmo de Viola-Jones (el cual detecta y enmarca el área del rostro) de 32 personas con 18 fotografías cada uno en un ambiente controlado, dando un total de 576 imágenes faciales, cada una con 120×120 pixeles. Todas las fotografías fueron preprocesadas para aumentar el número de aciertos y mejorar el rendimiento del software.

Tercero, se realizó el reconocimiento facial utilizando el algoritmo PCA-Eigenfaces, y se comprobó su validez mediante pruebas desarrolladas en distintos escenarios. El primer escenario fue; reconocer fotografías y usuarios que están en la base de datos; el segundo, reconocer fotografías que no están en la base de datos, pero si el usuario, el tercer, verificar que un usuario y sus fotografías no constan en la base de datos, y en el cuarto escenario, se agregaron nuevos usuarios a la base de datos para luego reconocerlos.

Para interactuar de una manera sencilla con el usuario final, se realizó una interfaz gráfica que contiene todos los procesos de detección y reconocimiento facial combinados, incluyendo otros procedimientos que mediante interfaces secundarias permiten ingresar nuevos usuarios a la base de datos existente.

INTRODUCCIÓN

Las técnicas de identificación biométrica ofrecen una solución de seguridad más exacta, debido a que utiliza rasgos físicos de la persona que son propios y exclusivos. Estos rasgos pueden ser extraídos del ojo (iris), de las manos (huellas, geometría) o de las características de la cara, entre otros. Mediante sensores específicos se obtiene las características de dichos rasgos, para luego digitalizarlos y crear sistemas automatizados de reconocimiento humano basados en hardware y software. Este proceso es idéntico al que utiliza el cerebro para reconocer un objeto o persona.

De los principales sistemas biométricos, el reconocimiento facial, permite identificar a una persona con solo capturar su rostro. De manera general se puede decir que su funcionamiento es el siguiente: se toma fotografías del usuario, se las procesa y guarda en una base de datos, para cuando el sistema reciba fotos nuevas, este pueda identificar y decidir si está o no dentro de la base de datos conocidas [1].

Por todo lo dicho anteriormente se realizó un software para detección y reconocimiento facial de personas, el cual está basado en dos algoritmos reconocidos por personas ligadas al campo de la seguridad biométrica: PCA Eigenfaces y Viola-Jones. Actualmente el reconocimiento facial es una de las aplicaciones más usadas como por ejemplo en teléfonos móviles, oficinas de migración, control de personas en establecimientos, sistemas de seguridad domiciliaria, robótica, etc. Con las características de estos algoritmos y su fusión empezamos a desarrollar nuestro proyecto de seguridad.

ANTECEDENTES DEL PROBLEMA DE ESTUDIO

No es difícil saber que desde hace tiempo atrás el ser humano ha identificado un individuo gracias a cualquier rasgo que este muestre, por ejemplo: un corte sobre la piel, su color, su estatura, su voz, etc., y que en su momento estas características serían suficientes para reconocer una persona, con tan solo verlas.

Con el pasar de los años, identificar a una persona requería mayores exigencias; por lo que, fueron aparecieron nuevas técnicas para identificar personas, como por ejemplo: usuario y claves de acceso, número de identificación personal (NIP), tarjeta de identificación por radio frecuencia (RFID), pasaporte, licencia de conducir, entre otros; pero estas técnicas utilizan atributos que se pueden extraviar, olvidar, manipular o ser robados; creando percances económicos y problemas de seguridad [1]. Es así como empieza la necesidad de crear otros sistemas mucho más confiables para la identificación de personas; y digitalizar las características morfológicas del ser humano sería la solución (uso de la biometría).

Entre los tipos de sistemas biométricos (huella, iris, mano, firma, voz, rostro), el reconocimiento facial es el que más ha evolucionado, y poco a poco deja de verse solo en ciencia ficción sino aplicados en la vida real. Este sistema es muy usado para identificar personas y crear métodos de seguridad y protección sofisticados. De ahí que nos ha llevado a pensar en este tema de trabajo, por lo que es conveniente saber algo de su historia, la cual se describe a continuación.

Mediciones manuales

En la década de 1960, Wilson Bledsoe trabajó en un sistema donde se podía clasificar imágenes o fotografías de rostros de forma manual utilizando una tableta RAND (dispositivo donde se podía ingresar coordenadas verticales y horizontales sobre una cuadricula usando un lápiz óptico que generaba pulsos electromagnéticos). Este sistema era usado para registrar de forma manual los puntos o coordenadas de las características faciales (incluyendo ojos, nariz, línea de boca y cabello). Luego estas métricas se guardaban en una base de datos y cuando se recibía una foto nueva de un usuario, el sistema sería capaz de recuperar la imagen que más se asemejara a esta persona dentro de la base de datos [2].

Marcadores faciales

En los años de 1970, Goldstein, Harmon y Lesk mejorarían el sistema manual de reconocimiento facial utilizando 21 marcadores específicos que incluirían el espesor del labio y el color de pelo. Pero al igual que el anterior sistema, estas métricas aun debían ser registradas manualmente [2].

Eigenfaces

En el año de 1980, Kirby y Sirovich empezarían a solucionar el problema de reconocimiento facial usando algebra lineal. Mediante el enfoque Eigenface buscaban representar imágenes faciales de baja dimensión. Estos dos científicos demostraron que se podía tener un conjunto de características fundamentales, realizando el análisis de las características de un grupo de imágenes faciales. Turk y Pentland en el año de 1991 desarrollarían aún más el enfoque Eigenface ya descubrieron como localizar rostros dentro de una imagen, siendo los primeros pasos para el reconocimiento automático de caras [2].

Programa FERET

En la década de 1990, con el propósito de impulsar el mercado de reconocimiento facial, el Instituto Nacional de Estándares y Tecnología y la Agencia de Proyectos de Investigación DARPA de USA crearon el programa FERET (Tecnología de Reconocimiento Facial). Que implicaba la elaboración de una base de datos con imágenes de rostros para la innovación del reconocimiento facial [2].

Proveedores de reconocimiento facial

A inicios de la década de 2000, el instituto NIST inicia el programa FRVT (Pruebas de Reconocimiento Facial) partiendo de FERET; para evaluaciones gubernamentales mediante sistemas de reconocimiento facial comerciales. Esto para cumplir con las leyes estadounidenses [2].

Medios sociales

Desde el año 2010, la página social Facebook implementa el reconocimiento facial para identificar la cara de una persona que aparezca en la actualización de

fotos que realicen los usuarios. Esto tuvo gran controversia ya que genero temas controversiales sobre la privacidad [2].

Reconocimiento facial en aeropuertos

En 2011, el gobierno de EE.UU. y Panamá lanzaron el programa piloto de reconocimiento facial FaceFirst en el aeropuerto Tocumen para controlar el contrabando y crimen organizado, siendo un éxito [2].

Reconocimiento de la cara móvil

Desde 2014, el ARJIS (Sistema Automatizado de Información de Justicia Regional), distribuye la plataforma móvil FaceFirst a las agencias asociadas. La cual es una red que intercambia datos para el cumplimiento de la ley a personas no identificas y se consideraban sospechosas [2].

Reconocimiento facial en teléfonos móviles

En el año de 2017, Apple lanzó su teléfono IPhone X el cual poseía una característica principal de reconocimiento facial para seguridad del dispositivo. El modelo se vendió rápidamente dando a conocer que el reconocimiento facial sería el nuevo estándar para la seguridad [2].

JUSTIFICACIÓN

La delincuencia es una problemática social de la cual nadie está exento. Los hechos delictivos encierran varias acciones que ocurren a diario y alimentan la percepción de inseguridad en el país. Según el Servicio Integrado de Seguridad ECU 911 [3], existen varios delitos de mayor afectación en la sociedad, siendo estos: muertes violentas, femicidios, robos, abigeato (hurto de ganado). El robo es el delito más común del cual es víctima el ecuatoriano.

Con la aparición de nuevas tecnologías es posible identificar a personas, consultar bases de datos policiales en tiempo real y advertir de la presencia de criminales. Los sistemas de reconocimiento facial permiten a las autoridades policiales rastrear a individuos desde cámaras ubicadas en lugares públicos, pudiendo reconocer entre la multitud, a una persona que haya cometido crímenes.

China, por ejemplo, posee un sistema de reconocimiento facial con Inteligencia Artificial, con el cual se ha detenido a varios delincuentes. Un ejemplo fue la aprensión de un individuo que había cometido infracciones económicas, éste se encontraba en un concierto y pudo ser reconocido entre una multitud de 60 000 asistentes [4]. En Ecuador, el Banco Guayaquil implementó un sistema de reconocimiento facial para su Banca Virtual Móvil, eliminando el uso de claves, esto permite a los usuarios realizar transacciones bancarias de manera rápida y segura [5].

A partir de esto, se desarrolla un software de seguridad para detección y reconocimiento facial, el mismo que permite conocer la identidad de una persona: nombre, edad, domicilio, y si tiene o no antecedentes penales; pudiendo detectar posibles criminales que están dentro de una base de datos e impedir que cometan delitos. El reconocimiento facial es una medida extra de seguridad, que puede evitar un incidente antes de que suceda.

OBJETIVOS

OBJETIVO GENERAL

- Desarrollar un software de seguridad para detección y reconocimiento facial basado en los algoritmos de Viola-Jones y PCA Eigenface.

OBJETIVOS ESPECÍFICOS

- Revisar el estado del arte sobre sistemas de seguridad basados en las técnicas de identificación biométrica.
- Obtener una base de datos con características faciales en un ambiente controlado.
- Diseñar un software que permita la detección y reconocimiento facial basado en el algoritmo de Viola-Jones y PCA Eigenface.
- Realizar pruebas de funcionamiento del software.

CAPÍTULO 1: ESTADO DEL ARTE Y FUNDAMENTACIÓN TEÓRICA

1.1 INTRODUCCIÓN A LA BIOMETRÍA Y SISTEMAS BIOMÉTRICOS

Se entiende por biometría al reconocimiento de una persona de forma automática en base a sus características biológicas o rasgos físicos. Estas características o rasgos son únicas de la persona. Científicamente el concepto de biometría nace de las palabras "bio" que significa vida y "metría" de medida, de ahí que hace referencia a la medición e identificación de características únicas de la persona [1].

Actualmente la biometría es utilizada en casi todas las aplicaciones donde se requiera la identificación de una persona, como por ejemplo ingreso a sistemas de seguridad, centros educativos y de investigación, bases de datos confidenciales, etc. Incluso son utilizados en servicios de inmigración y lugares con alto tráfico de personas, etc. [6] [7].

Un sistema biométrico tiene la capacidad de medir, comparar, almacenar y reconocer características propias de un individuo, de manera precisa y confiable. Existe varios sistemas biométricos que utilizan los atributos como las huellas dactilares, el rostro, la geometría de la mano, la voz, el iris, la forma de caminar, el ADN, la oreja, venas de la mano y la firma; para la identificación y autentificación de las personas [8]. Pero sin embargo las aplicaciones estarán condicionadas por los requisitos y características según su función y personas involucradas. Por ejemplo, existirá problemas con las características biológicas o rasgos, estos sobresaldrán en una persona, pero faltara en otra. Por lo tanto, se deberá tener en cuenta las ventajas y desventajas que se puedan identificar para una aplicación específica. En la Figura 1 se muestra un ejemplo de las características biológicas y rasgos físicos que utiliza la biometría.

Figura 1 *Ejemplo de las características biológicas y rasgos físicos que utiliza la biometría*.
Fuente:[2]

1.2 TIPOS DE SISTEMAS BIOMÉTRICOS

Dependiendo de las características biológicas que se utilicen para la identificación, los sistemas biométricos se pueden clasificar en dos grupos: Sistemas biométricos fisiológicos y sistemas biométricos de comportamiento [8].

Los sistemas biométricos fisiológicos se basan en el estudio de las características inherentes a la fisiología del cuerpo humano (geometría de la mano, retina, cara, iris, piel, venas, huellas dactilares, entre otras); puesto que apenas se modifican con el tiempo, son los sistemas más fiables y poseen una gran unicidad. Los sistemas biométricos de comportamiento se basan en el estudio de las características derivadas de una acción (voz, firma escrita, dinámica del tecleo, gestos, movimientos corporales, entre otras) realizada por una persona; son menos fiables que las estáticas ya que están más sujetas a cambios [9].

Los principales estudios y aplicaciones de sistemas de reconocimiento biométrico están basados en características de la mano, cara, huella digital, iris, voz y firma escrita. A continuación, describimos cada uno de estos sistemas.

1.2.1 SISTEMA DE RECONOCIMIENTO POR HUELLA DIGITAL

Los sistemas de reconocimiento de huellas digitales analizan las minucias de los dedos determinando formas fácilmente identificables; las minucias son los puntos característicos de la huella dactilar, éstas están formadas por las crestas papilares y

surcos interpapilares, en la última falange y hacen la diferencia entre una huella y otra [10]. En la Figura 2 se indica la composición de una huella dactilar.

Figura 2 *Huella dactilar donde se observa los surcos y crestas papilares.*
Fuente:[10]

Debido a que no existen dos dedos que posean huellas digitales exactamente iguales; y, dado que estas características no varían en el transcurso de la vida del individuo, este sistema está catalogado como el más seguro y es altamente utilizado e implementado en aplicaciones para el control de acceso, control del personal dentro de una organización, pago de nóminas, voto electrónico, acceso a computadoras, entre otros. Este sistema es considerado invasivo ya que se requiere de la presencia física de la persona [10].

En [8] se afirma que existen dos métodos para capturar una huella digital, la primera consiste en humedecer la yema del dedo en tinta y luego presionar contra un papel (adquisición off-line) para luego ser escaneada; y en el segundo método se presiona directamente la yema contra una superficie que contenga un sensor (adquisición on-line); en ambos casos se guarda la imagen digitalizada de la huella para realizar la identificación.

Existe un pequeño porcentaje de la población que no es apto para utilizar este sistema de identificación (1% al 2 % de la población); las razones incluyen piel cortada, dedos vendados, callosidades, piel seca o húmeda, etc. [8]

El mercado ofrece una gran variedad de productos para este sistema, en la Figura 3 se presentan ejemplos de sistemas de reconocimiento por huella digital.

	Suprema BioStation A2 – Lector de huella
	Suprema Módulo BioMini Slim 2 – Lector de huella
	Nitgen Hamster II – Lector de huella digital para PC

Figura 3 *Modelos de sistemas para reconocimiento por huella digital.*
Fuente: [11]

1.2.2 SISTEMA DE RECONOCIMIENTO FACIAL

Estos sistemas identifican a una persona a partir de la localización, forma y relaciones espaciales entre sus características faciales (ojos, nariz, barbilla, entre otros). Son sistemas no intrusivos ya que basta con una simple cámara para obtener la imagen digital del rostro, por lo que es considerado, junto con la voz, uno de los sistemas más naturales de reconocimiento biométrico. Se pueden implementar en ambientes inteligentes (casas, coches, oficinas, etc); y, dado que la imagen también puede ser tomada desde grandes distancias, se utiliza en sistemas de vigilancia policial [12].

Los autores de [8] destacan que los sistemas de reconocimiento facial presentan inconvenientes cuando la variabilidad entre imágenes es muy elevada, esto se da cuando las imágenes son adquiridas en condiciones no controladas; es decir, en diferentes condiciones de posición o iluminación, en presencia o ausencia de lentes, barba, cambios en la fisonomía con el tiempo, entre otros; disminuyendo la garantía de fiabilidad del sistema. En condiciones controladas, estos sistemas son rápidos, precisos y económicos.

En el mercado se pueden encontrar distintos dispositivos que realizan algún tipo de reconocimiento facial, en la Figura 4 se presentan varios ejemplos.

	Hanvon FaceID F710 – Terminal de reconocimiento facial 3D
	Logitec HD C920
	Reloj Biométrico Facial uFace 402

Figura 4 *Modelos de sistemas de reconocimiento facial.*
Fuente:[11] [13] [14]

1.2.3 SISTEMA DE RECONOCIMIENTO POR GEOMETRÍA DE LA MANO

Los sistemas de reconocimiento de la geometría de la mano utilizan características como la longitud, anchura, radio, perímetro y área de los dedos; así como también de la palma de la mano, para identificar a los usuarios; de igual manera se puede tener en cuenta los ángulos y la separación entre distintos puntos de referencia de la mano. En la Figura 5 se muestra algunas de las principales características que pueden ser extraídas de una imagen de la geometría de la mano [8].

En [15] se dice que estas características pueden ser obtenidas de imágenes de baja resolución usando una cámara digital ordinaria, o también a través de escáneres; este último es el más fundamental y básico de los sistemas de biometría de la mano. Son sistemas populares utilizados en sistemas de control de acceso, control de horarios y asistencia y otras tareas de verificación. Pueden operar bajo condiciones adversas como suciedad en la mano, y requieren poco almacenamiento de datos.

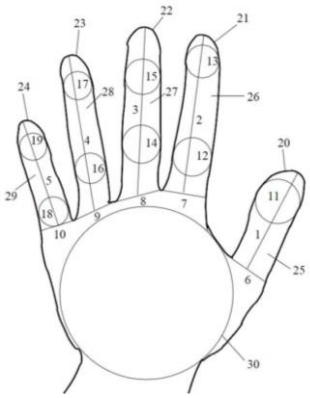


Figura 5 *Principales características que pueden ser extraídas de la geometría de la mano.*
Fuente: [15]

Para calcular el grosor de los dedos se puede utilizar un espejo para obtener una imagen lateral de la mano y así poder medir su grosor. Se utiliza puntos de referencia como las yemas y valles de los dedos para alinear las imágenes, previo a la extracción de características. Los escáneres utilizados no toman en cuenta características como las huellas dactilares, colores, cicatrices, líneas de la palma de la mano, crestas, entre otros [15]. Varios ejemplos de este tipo de sistema se presentan en la Figura 6.

	Reloj Biométrico Allegion Hand Punch HP-2000
	Lector de manos HandKey® II

Figura 6 *Modelos de sistemas para reconocimiento por geometría de la mano.*
Fuente:[16] [17]

1.2.4 SISTEMA DE RECONOCIMIENTO DE IRIS

El iris forma parte del sistema óptico del ser humano. Este se encuentra en medio de la córnea y el cristalino, la cual tiene como forma una membrana circular y coloreada. Donde en su centro se ubica la pupila, que se hace más grande o pequeña controlando la cantidad de luz que pasa al interior del globo ocular [18]. Cuando existe demasiada luminosidad los músculos del iris y la pupila se contraen para dejar

pasar menor cantidad de luz hacia la retina, mientras que al existir menor luminosidad los músculos del iris se relajan y la pupila se dilata, dejando entrar mayor cantidad de luz a la retina [19].

A partir de un concepto general del iris, y que según los autores de [6], es como los investigadores Leonard Flom y Aran Safir patentan en el año de 1987 el primer diagrama de adquisición de datos para la biometría basada en iris. El cual consistía en un sistema que controlaba la luz mediante 4 puntos de enfoque, en donde 3 puntos formaban 120° alrededor del ojo y lo demás frontal al mismo. Con la reflexión que producía el ojo sobre una lente se generaba la imagen buscada.

Con el trabajo de Flom y Safir sería más fácil realizar un sistema electrónico de este tipo. Así en el documento [7] indican dos principales algoritmos utilizados en los sistemas de identificación y reconocimiento basado en el iris. El primero creado por J. Daugman y el segundo por R. Wildes.

Basados en cualquiera de estos dos algoritmos, existen compañías como LG, Panasonic, Oki, IrisGuard, Sagem, Sarnoff, IRIS, CHILD Project, Privium, Clear (RT) y CanPass han creado dispositivos de reconocimiento de iris. Algunos ejemplos se muestran en la Figura 7.

	Ultramach s1000 de Anviz
	IRIS SCANNER Model 2100
	SecuriMetrics Pier 2.3

Figura 7 *Modelos de sistemas de reconocimiento por iris.*
Fuente: [20] [21] [22]

1.2.5 SISTEMA DE RECONOCIMIENTO POR VOZ

La voz del ser humano se produce gracias al aparato fonatorio, que está formado por la laringe (posee las cuerdas vocales), faringe, la cavidad bucal y nasal, los pulmones (flujo de aire que es la fuente de energía) y varios elementos de articulación como: lengua, dientes, labios, alvéolo, paladar y velo paladar [23].

Según [24] gracias a la abertura entre cuerdas vocales que se denomina glotis, se puede generar el sonido (las cuerdas bucales son dos membranas ubicadas adentro de la laringe). Cuando la glotis empieza a cerrarse dichas cuerdas empiezan a vibrar en forma de lengüetas, generándose un sonido periódico. Este sonido tiene una frecuencia que depende del tamaño y masa de las cuerdas además de otros factores como la tensión y velocidad del aire proveniente de los pulmones. Mientras mayor tamaño de glotis sea, menor es la frecuencia de vibración (sonidos graves; varones), en cambio a mayor tensión, mayor es la frecuencia (sonidos agudos; mujeres). También aumentara la frecuencia del sonido cuando el flujo de aire tiene mayor velocidad. De igual manera los autores indican tres propiedades fundamentales que tiene la voz como: timbre, tono e intensidad.

Los métodos para la adquisición de datos de voz se lo realizan mediante un sistema que capta las ondas sonoras, luego son filtradas y se extraen los patrones característicos para finalmente digitalizarlas y guardarlas en una base e ir comparando cuando un usuario haga uso del sistema [23]. A continuación, en la Figura 8 indicamos ejemplos de software y hardware para la adquisición y reconocimiento de voz.

	Kit de dictado y reconocimiento de voz PSE4000
DRAGON NATURALLYSPEAKING	Dragon Naturally Speaking Software para reconocimiento de voz

Figura 8 *Sistemas de reconocimiento de voz.*
Fuente:[25] [26]

1.2.6 SISTEMA DE RECONOCIMIENTO MEDIANTE FIRMA ESCRITA

Una forma de representación gráfica por medio de dibujos o signos que están grabados sobre un soporte. Se puede clasificar la firma según su naturaleza y ámbito de representación; en el primer caso hace referencia a las características estáticas y dinámicas, mientras que para el segundo hace referencia a características locales (obtienen información de puntos específicos en dominio espacial o temporal) y globales (obtienen información como: centro geométrico, tiempo, media y desviación típica) [24].

Con la ayuda sistemas informáticos que permiten usar punteros y lápices con una interfaz de usuario, es posible digitalizar las firmas. Existen dos tipos de datos de firmas según su forma de adquisición, que son: estáticos (trazado en 2D, rasgos equivocados) y dinámicos (ofrece presión, dirección, velocidad de cambio en la firma) [24].

En [1] menciona que para el reconocimiento de firmas generalmente existe dos técnicas que son reconocimiento Off-line y On-line. En el primer reconocimiento se lo hace previo a la información de una imagen que contiene la firma (obtención mediante escáner); y el segundo reconocimiento se lo realiza mientras la persona realiza la firma (obtención mediante sistemas y dispositivos especiales como tablets y lápiz óptico).

Para finalizar, en el documento [27] menciona sistemas de firmas digitales basados en hardware y software. HW son: el Wacom STU-430 para firmas biométricas en PCs de escritorio y Microsoft Surface para firmas móviles; mientras que en SW son Applet o Java Web Start (JWS) y para movilidad sobre un virtual channel Citrix ad hoc. En la Figura 9 se presentan algunos ejemplos de estos dispositivos.

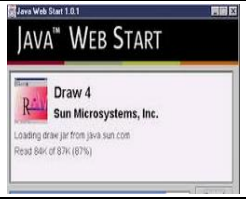
Hardaware		Software	
	Wacom STU-430 para firmas biométricas en PCs de escritorio		Java Web Start (JWS)
	Microsoft Surface para firmas móviles		Virtual channel Citrix ad hoc. para firmas móviles. (Un punto de acceso)

Figura 9 *Hardware y Software para sistemas de firmas digitales.*
Fuente: [28] [29] [30] [31]

1.3 DESCRIPCIÓN DE LOS ALGORITMOS UTILIZADOS

1.3.1 ALGORITMO DE VIOLA-JONES PARA DETECCIÓN FACIAL

El algoritmo de Viola-Jones es un método de detección con bajo coste computacional y gran rapidez, según los autores, en un procesador Pentium III de 700 Mhz, el detector facial puede procesar una imagen de 384 por 288 píxeles en aproximadamente .067 segundos, permitiendo una detección robusta en tiempo real. Este algoritmo efectúa la clasificación mediante características en vez de píxel a píxel y es utilizado para detectar objetos muy específicos teniendo el conjunto de entrenamiento adecuado, pero se aplica principalmente para la detección facial, este procedimiento es el primer paso para el reconocimiento facial [33] [34].

En este método se detallan las siguientes definiciones:

- Características de Haar, son elementos básicos utilizados para realizar la detección.
- La imagen integral, usada para la rápida evaluación de las características.
- Algoritmo Adaboost, es un método de aprendizaje automático que hace referencia a la generación de algoritmos que aprenden y prevén el comportamiento en un conjunto de datos.
- Clasificadores en cascada para combinar características de manera eficiente.

1.3.1.1 Características de Haar

El método de detección de Viola Jones utiliza conjuntos de características simples para conseguir información de una imagen mediante operaciones aritméticas. [34]. Viola y Jones utilizan cuatro tipos de características de Haar: 2 características basados en 2 rectángulos, 1 basado en tres rectángulos y 1 basado en cuatro rectángulos [35]. Como se muestra en la Figura 10.

- Característica de 2 rectángulos: resultado de la diferencia entre la suma de los pixeles de ambas regiones.
- Característica de 3 rectángulos: resultado de la suma de los pixeles de los rectángulos exteriores menos la suma del rectángulo central.
- Característica de 4 rectángulos: resultado de la diferencia entre los grupos diagonales de los rectángulos.

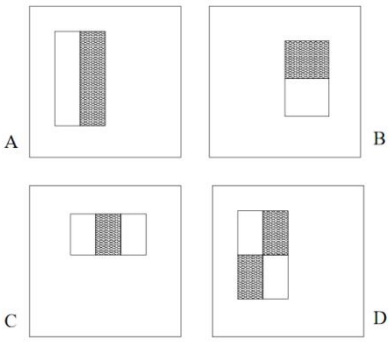


Figura 10 *Características utilizadas en el detector de Viola-Jones.*
Fuente: [36]

Generalmente, las características hacen referencia a valoraciones de la intensidad de conjuntos de píxeles. Cada una de las características es sobrepuesta sobre una imagen en todas las posiciones y tamaños posibles, la suma de los píxeles del área blanca se sustrae de la suma de los píxeles del área oscura. La respuesta obtenida mediante la resta, es el valor de la característica [33].

En el algoritmo de Viola-Jones, estas características se extraen sobre una ventana de búsqueda de 24×24 píxeles. El número de características de cada tipo de una imagen es demasiado alto, por ello se eligen las más apropiadas para la detección. En la Tabla 1 se presenta el número de las características por cada tipo.

Tabla 1 Número de características utilizadas por cada imagen (24×24 píxeles).
Fuente: [34]

Tipo Característica	Número Características
Tipo 1	43200
Tipo 2	43200
Tipo 3	27600
Tipo 4	20736
Total	134736

1.3.1.2 Imagen Integral

Es una representación intermedia de la imagen, permite calcular la suma de los píxeles de una imagen que se ubica por debajo de una área blanca o una área oscura de 1 característica de Haar. Esta se obtiene con la suma de los valores de luminancia de los píxeles que se encuentran arriba y a la izquierda de un punto en la imagen, y da como resultado una matriz de igual tamaño a la matriz de la imagen original [34].

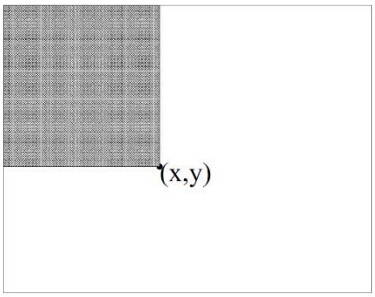


Figura 11 *Valor de la imagen integral en el punto (x,y).*
Fuente: [36]

La imagen integral se define como la doble integral de una imagen, primero a lo largo de las filas y luego al de las columnas. La imagen integral, como se muestra en Figura 11, en la localización (x, y) viene dada por la ecuación (1):

$$ii(x,y) = \sum_{x' \leq x,\, y' \leq y} i(x', y') \qquad (1)$$

Donde:

$\mathrm{ii}(\mathrm{x}, \mathrm{y})$: Imagen integral

$\mathrm{i}(\mathrm{x}', \mathrm{y}')$: Valor de la imagen en un punto específico.

Esto permite un cálculo muy eficiente con solo un recorrido por toda la imagen a partir de la suma acumulada de la fila actual y el valor de la imagen integral en la fila anterior [35].

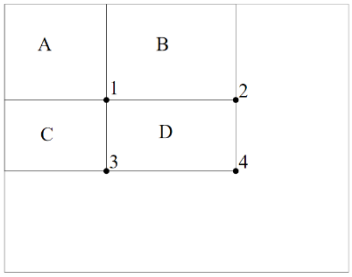


Figura 12 *Ejemplo de cálculo del valor de la imagen integral.*
Fuente: [36]

El valor de una imagen integral en el punto 1 de la Figura 12 es la suma de los valores del rectángulo A dentro su respectiva imagen. El valor en el punto 2 es la suma entre $A + B$, el punto 3 es la suma entre $A + C$, y el del punto 4 es la suma entre $A + B + C + D$. Para un mejor entendimiento, la suma de los píxeles del rectángulo D se calcula como: $4 - 2 - 3 + 1 = 4 + 1 - (2 + 3)$ [34].

1.3.1.3 Clasificadores simples y algoritmo AdaBoost

Una vez encontradas las características de la imagen, se utiliza un clasificador para seleccionar las características que definan de mejor manera un rostro y ayuden a localizarla en la imagen. Para ello se emplea el algoritmo de AdaBoost, creado por Freund y Shaphire, el cual combina clasificadores débiles para crear un clasificador más robusto y exacto. La salida de un clasificador débil es binaria y depende del valor de una característica [34] [35].

El algoritmo AdaBoost entrena clasificadores simples de manera iterativa en base a un grupo de características (features), y suma sus valores para crear un clasificador fuerte. Ya que se tiene un clasificador simple por cada característica de Haar se tendría que analizar un total de $K \times N$ clasificadores simples, donde K representa el número de características por imagen y N es el número de imágenes en la base de datos[35].

Se tiene 2 tipos de respuestas entregadas por los clasificadores: falso positivo (detecta una no-cara) y falso negativo (detecta erróneamente una cara), el

clasificador fuerte mantiene una tasa nula de falsos negativos, aunque no se reduzca la tasa de falsos positivos.

1.3.1.4 Clasificadores en cascada

El algoritmo de Viola-Jones construye clasificadores fuertes mediante el algoritmo AdaBost para luego enlazarlos en forma de cascada como se muestra en la Figura 13 [35].

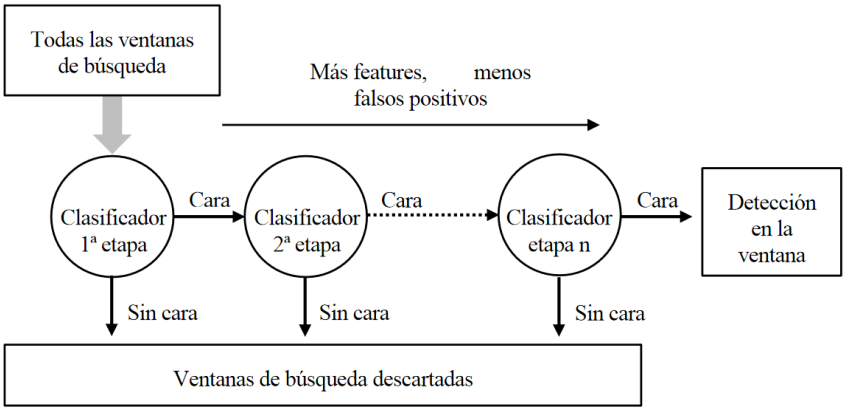


Figura 13 *Clasificador en cascada.*
Fuente: [35]

Utilizando ventanas deslizantes, en cada etapa, se rechazan ventanas negativas (aquellas que no correspondan a una cara) manteniendo casi todas las positivas (aquellas que correspondan a una cara) para seguir a la siguiente etapa. Si la ventana cruza todas las etapas existe una gran posibilidad de que en esa área se encuentre un rostro [34].

Si en una etapa del detector, una ventana es clasificada como no cara, esta es descartada automáticamente; en cambio, si es cara continúa a la siguiente etapa, donde se ejecuta una operación más compleja. De esta manera, se concluye que: si una ventana atraviesa más etapas, tiene mayor probabilidad de contener una cara [33].

Viola y Jones emplearon un detector con 38 etapas, entrenadas con 4.916 imágenes faciales de 24×24 pixeles cada una [33] [34].

1.3.2 TÉCNICAS DE RECONOCIMIENTO FACIAL

Existen tres grupos de técnicas de reconocimiento facial: basadas en apariencia, basadas en características y basadas en modelos, las cuales se detallan a continuación [37] [38]:

- Técnica basada en apariencia: Utiliza la apariencia básica del rostro y generalmente son técnicas algebraicas de reducción del espacio para adquirir componentes principales o información útil. Entre esta técnica tenemos: Eigen Faces y Fisher Faces.
- Técnica basada en características: Extrae las características principales del rostro.
- Técnica basada en modelos: Utiliza modelos o "templates" del rostro para extraer sus características, o alguna propiedad que ayude en el reconocimiento.

El algoritmo PCA-Eigenfaces utiliza la técnica basada en apariencias, la misma que se ve afectada por el entorno en donde se adquieren las imágenes; y como es nuestro tema de estudio es necesario describir sus ventajas y desventajas.

1.3.2.1 Ventajas y Desventajas de la Técnica Basada en Apariencias

Las ventajas y desventajas de la técnica basada en apariencia son:

- **Ventajas**

Se puede representar un grupo de imágenes en una dimensión mucho más pequeña. [37]. Se obtiene excelentes resultados cuando las imágenes faciales a identificar presentan la misma distancia, posición e iluminación que las imágenes que se encuentran en la base de datos

- **Desventajas**

Existen una gran variedad de parámetros que afectan al reconocimiento facial, entre ellos tenemos: cambios en el entorno (posición e iluminación); cambios en la expresión facial (posición, peinados, bello facial) y cambios debido a la edad de las personas. Todos estos parámetros engañan a los sistemas de reconocimiento facial [37] [38].

1.3.3 ALGORITMO PCA-EIGENFACES PARA RECONOCIMIENTO FACIAL

1.3.3.1 Descripción del PCA

El PCA es un método óptimo para disminuir el número de dimensiones y así formar un grupo de vectores que pueden ser por ejemplo imágenes de rostros [39].

Es un algoritmo que reduce dimensiones de un conjunto de variables, coordenadas u otros datos; conteniendo la mayor variación de estos en vectores que se denominan "componentes principales (CP)". Estas componentes son combinaciones lineales que no están correlacionas entre ellas de las variables originarias. Cuando la primera CP haya capturado el valor más grande de varianza de los datos, la segunda CP capturara el segundo valor más grande de varianza, así de manera sucesiva. El número de dimensiones del conjunto de datos representara el número de componentes principales [40].

PCA tiene dos propiedades importantes por así decirlo las cuales son [39] :

- Las CP obtienen de forma secuencial, el valor máximo de variabilidad o varianza de la matriz de datos (entrada), garantizando que sea mínima la información que se pierde (refiriéndose al error de reconstrucción).
- Las CP que se obtienen son ortogonales entre ellas, facilitando su procesamiento, debido a que se las puede tratar de manera independiente.

1.3.3.2 Como trabaja en algoritmo de PCA

El autor del curso de "Reducción de descriptores" [41]. Dice que el objetivo principal del PCA es reducir o disminuir la dimensión del conjunto de datos de entrada manteniendo la mayor cantidad de información para analizarlos. Para esto genera un grupo de variables que pueden estar correlacionadas entre sí, en un grupo menor de variables no correlacionadas, llamadas "componentes principales". Así mismo menciona el concepto personal sobre dimensionalidad el cual dice:

"Un conjunto de datos de alta dimensionalidad tiende a representarse por variables correlacionadas (no representa más información), mientras que un

conjunto de datos con poca dimensionalidad puede representar o contener más información" [41].

De manera general lo que realiza el PCA es:

- Transforma linealmente los datos a un sistema nuevo de coordenadas ortogonales.
- Las direcciones de mayor varianza de los datos de entrada son los vectores de proyección de los datos en el nuevo espacio.
- Las variables resultantes, de proyectar datos entrantes sobre los vectores de proyección, se denominan componentes principales.
- En el sistema nuevo de coordenadas, las CP están clasificadas automáticamente según la varianza de la proyección de datos (es decir según la cantidad de información que contengan).
- Se baja la dimensión de los datos resultantes en el nuevo espacio excluyendo las CP que muestran una menor varianza (aportan menos información) [1].

1.3.3.3 PCA-Eigenfaces para reconocimiento facial

Dada una imagen de $n \times n$ pixeles, esta se representa como una matriz de $N \times N$ píxeles, donde cada píxel tiene un valor de intensidad diferente (0 a 255) que resulta de la transformación de la imagen RGB (rojo, verde y azul) a escala de grises. A continuación el autor de [42] describe los pasos para aplicar el PCA en una base de rostros de M imágenes de 2 dimensión NxN. Cada cara está representada por $\Gamma_1, \Gamma_2, \dots, \Gamma_M$:

1. Cada cara, Γ_i con $i = 1,2, \dots, M$

Es reestablecida como un nuevo vector de dimensión $N^2 \times 1$ cuyo valor es la concatenación de cada una de las filas, de esta manera se obtiene una matriz de $N^2 \times M$. Por ejemplo:

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 1 \end{bmatrix}$$

Cada imagen de la cara está representada por el vector Γ_i

$$\Gamma_{1=}\begin{bmatrix}1\\-2\\1\\-3\end{bmatrix},\Gamma_{2=}\begin{bmatrix}1\\3\\-1\\2\end{bmatrix},\Gamma_{3=}\begin{bmatrix}2\\1\\-2\\3\end{bmatrix},\dots,\Gamma_{M=}\begin{bmatrix}1\\2\\2\\1\end{bmatrix}$$

2. Se obtiene el rostro promedio ψ (vector promedio aritmético):

$$\psi=\frac{1}{M}\sum_{i=1}^{M}\Gamma_i \qquad (2)$$

$$\begin{bmatrix}1\\-2\\1\\-3\end{bmatrix}+\begin{bmatrix}1\\3\\-1\\2\end{bmatrix}+\begin{bmatrix}2\\1\\-2\\3\end{bmatrix}+\cdots+\begin{bmatrix}1\\2\\2\\1\end{bmatrix}\rightarrow\begin{bmatrix}-1\\-1\\2\\-3\end{bmatrix}$$

3. El rostro promedio Ψ que se ha obtenido, es sustraído de cada imagen Γ_i de la base de datos, dando como resultado un nuevo conjunto de vectores. El vector de desviación para cada imagen Φ_i viene dado por:

$$\Phi_i=\Gamma_i-\psi \qquad (3)$$

$$\Phi_1=\begin{bmatrix}2\\-1\\-1\\0\end{bmatrix},\Phi_2=\begin{bmatrix}2\\4\\-3\\5\end{bmatrix},\Phi_3=\begin{bmatrix}3\\2\\-4\\6\end{bmatrix},\dots,\Phi_M=\begin{bmatrix}2\\3\\0\\4\end{bmatrix}$$

Con los vectores diferencia Φ_i obtenidos, se forma la matriz de imagen centrada media A de dimensión N^2xM:

$$A=[\Phi_1,\Phi_2,\Phi_3,\dots,\Phi_M] \qquad (4)$$

La matriz A conserva solo las características distintivas de las imágenes faciales y elimina las características comunes.

4. Se calcula la matriz covarianza C de la matriz A para obtener los vectores propios

$$C=AA^T \qquad (5)$$

$$A=\begin{bmatrix}2&3\\-1&-2\\-1&1\\0&2\end{bmatrix},A^T=\begin{bmatrix}2&-1&-1&0\\3&-2&1&2\end{bmatrix}$$

El tamaño de la matriz de covarianza será $N^2 \times N^2$ (4×4 en este caso). Se debe obtener los vectores y valores propios que corresponden a la matriz de covarianza, se utilizan estos vectores propios para crear la representación de las imágenes. La dimensión de la matriz C no permite obtener los vectores y valores propios debido al gran tamaño de esta matriz. Por lo tanto se asume una manera simplificada para calcular estos vectores y valores, esta se fundamenta en: debido a que el número de imágenes faciales en la base de datos es mucho menor que el número de píxeles de cada imagen, sólo existirán M vectores propios significativos en lugar de N^2, los demás son asociados a valores propios igual a cero [40] [41].

a. La matriz simplificada L tiene un tamaño MxM y es igual a:

$$L = A^T A \qquad (6)$$

Para nuestro ejemplo sería una matriz de 2 x 2.

$$L = \begin{bmatrix} 6 & 7 \\ 7 & 18 \end{bmatrix}$$

b. Se calculan los vectores propios v_i de L, y se los ordena de mayor a menor según sus respectivos valores propios, formando así la matriz v_i $(i = 1,2,..,M)$. Luego se obtienen los vectores propios U_i de la matriz de covarianza C realizando la combinación lineal de vectores diferencia Φ_i y vectores propios de la matriz L [40] [41].

$$U_i = [\Phi_1, \Phi_2, \Phi_3, \dots, \Phi_M][v_1, v_2, v_3, \dots, v_M] = Av_i \qquad (7)$$

Los Eigenfaces son:

$$Eigenface = U = [U_1, U_2, U_3, \dots, U_M] \qquad (8)$$

Para construir las eigenfaces no se utilizan los M vectores propios, sino una matriz menor $M - 1$, con los cuales se pueden generar sin error las M caras del conjunto de entrenamiento.

5. Se calcula el peso ω de cada vector propio. Los eigenvectores corresponden a cada eigenface y descartan para las cuales los valores propios son cero. Las eigenfaces se clasifican según su utilidad para caracterizar la variación entre las imágenes.

$$\omega_i = U^T(\Gamma_i - \psi), \qquad con\ i = 1,2, \dots, M-1 \qquad (9)$$

Donde:

U^T: Matriz transpuesta de los vectores propios de C.

Γ_i: Numero de imágenes de caras.

ψ: Rostro promedio.

Luego se obtiene la matriz de pesos Ω, el cual se obtiene de la matriz transpuesta de los pesos como se observa a continuación:

$$\Omega_i = [\omega_1, \omega_2, \omega_3, \dots, \omega_{M-1}]^T \quad con\ i = 1,2, \dots, M-1 \qquad (10)$$

El peso correspondiente a la imagen de entrada Ω_{ent} es:

$$\Omega_{ent} = U^T\ EntradaNormalizada \qquad (11)$$

6. Se puede reconstruir cualquiera de las imágenes del conjunto de muestras de la siguiente manera:

$$\Gamma_i = \psi + \sum_{i=0}^{M-1} \omega_i U_i \qquad (12)$$

Donde:

U_i: Vectores propios.

ω_i: Pesos de cada vector propio.

ψ: Rostro promedio.

1.3.4 DISTANCIA EUCLIDIANA

El software adquiere la imagen de entrada y la compara con las imágenes de la base de datos para seleccionar la que más se asemeja, para esto se obtiene la matriz de pesos Ω aplicando el procedimiento explicado anteriormente, y se calcula la distancia euclidiana δ. Este valor es una de las medidas más básicas para calcular distancias, y se define como el valor absoluto de la diferencia de matriz de peso de la imagen entrante Ω_{ent} y la matriz de pesos Ω_i del grupo de imágenes de la base de datos [40] [41].

$$\delta = \|\Omega_{ent} - \Omega_i\| \qquad (13)$$

Se considera que la distancia más pequeña es el resultado de coincidencia de rostro.

CAPÍTULO 2: DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA

2.1 METODOLOGÍA

La metodología seguida para el desarrollo e implementación del software se basa directamente en los objetivos planteados. Los cuales se indican a continuación:

El primero, revisar el estado del arte sobre sistemas de seguridad basados en las técnicas de identificación biométrica, implica la realización de una documentación (estado del arte) sobre los sistemas biométricos; en ésta se destacan las características más importantes sobre la biometría y se da una descripción de los rasgos más utilizados: huella, cara, voz, geometría de la mano, iris y firma; dando un mayor énfasis en los sistemas de reconocimiento facial.

Como segundo, obtener una base de datos con características faciales en un ambiente controlado, a través de la adquisición de varias fotografías de distintos individuos. Para esto se tomó 18 fotografías a 32 personas y así obtener la base de datos con 576 imágenes faciales.

El tercer paso se considera lo esencial de este proyecto, que es diseñar un software para la detección y reconocimiento facial basado en el algoritmo de Viola-Jones y PCA-Eigenface, aquí se desarrolla el software capaz de detectar y reconocer el rostro de una persona que se encuentra en la base de datos creada anteriormente. Para el desarrollo de este software se realizaron los siguientes pasos:

1. Creación de la Base de Datos. Se almacena en una carpeta imágenes faciales de 32 personas, con 18 fotografías en diferentes poses de cada individuo, lo que hace un total de 576 imágenes faciales, estas imágenes están convertidas en gris para que sea bidimensional ($N \times N$).

2. Cálculo de la media m y la desviación estándar S de las imágenes: Cada imagen se convierte en un vector de columna de dimensión $1xN^2$, de modo que una imagen típica de tamaño 120×120 pixeles, por ejemplo, se convierte en un vector de dimensión 1×4400. Entonces calculamos estos valores de la siguiente forma:

Media (m): esto se obtiene al dividir la suma de los valores de cada imagen por la cantidad de valores de ese conjunto.

$$m = \frac{valor_{\,1} + \ valor_{\,2} + \cdots + valor_{\,i}}{N^2} \qquad hasta\ i = \ N^2 \qquad (14)$$

Desviación estándar (S): este valor se obtiene de cada una de las imágenes, utilizando la ecuación

$$S = \sqrt{\frac{1}{M}\sum_{i=1}^{M} \Gamma_i^2} \qquad (15)$$

Donde:

M: Número de imágenes de la base de datos

Γ_i: Número de imágenes de caras.

Con los resultados de m y S, se establece un valor que se aproxime a cada uno, denominados $ustd$ (desviación estándar general) y um (media general), para normalizar todas las imágenes.

3. Normalización: La normalización I de las imágenes, se realiza mediante la ecuación (16) para hacer que todas las imágenes tengan una dimensión uniforme y para eliminar los efectos debidos al fondo y la iluminación.

$$I = (\Gamma - \psi) * \frac{ustd}{S} + um \qquad (16)$$

Donde:

Γ: Número de imágenes de caras.

ψ: Rostro promedio.

S: Desviación estándar de cada imagen.

$ustd$: Desviación estándar general.

um: Media general.

4. Cálculo de la matriz de imagen centrada media A: Se resta la media de la matriz de vector de columna de imágenes Γ de entrenamiento para obtener las imágenes centradas A, definida por la ecuación (4).

5. Cálculo de vectores y valores propios (Consulte la Sección 1.3.3.3).

6. Creación de Eigenfaces: Los vectores propios obtenidos en el paso 5 se clasifican en orden descendente y los vectores propios superiores se consideran eigenfaces.

7. Cálculo de los pesos: Los pesos ω se determinan calculando el producto punto de la transpuesta de la matriz eigenfaces U y la matriz de imagen centrada A, o lo que es igual a la ecuación (9). Y posteriormente se genera la matriz de pesos Ω_i, ecuación (10)

8. Clasificador de distancia euclidiana: En primer lugar, todos los pasos de 2 a 7 se aplican en la imagen de prueba para calcular sus pesos de prueba y luego se determina el valor absoluto de la diferencia entre estos pesos y los pesos de las imágenes de la base de datos mediante la siguiente ecuación (13)**:**

9. Reconocimiento facial: Finalmente, Para garantizar el resultado de la identificación facial se establece un umbral de reconocimiento, si el menor valor de la distancia euclidiana sobrepasa el umbral, el software descarta al usuario; caso contrario, el usuario es reconocido e identificado.

Por último, se debe realizar pruebas de funcionamiento del software en diferentes escenarios los cuales permitan poner a prueba el software al ciento por ciento y así determinar las ventajas y desventajas para su funcionamiento.

2.2 MATERIALES Y HERRAMIENTAS

Para desarrollar el proyecto del software de seguridad se utilizaron los materiales y herramientas que se muestran en la Tabla 2.

Tabla 2 Materiales y Herramientas empleados para la creación de la base de datos y prueba
Fuente: Autores

Materiales y Herramientas	
Cámara *SONY CYBER SHOT DSC-W560 14.1MP*	1
PC portátil (rendimiento alto)	2
Tela color azul (para fondo)	2x2 m.
Distancia entre cámara y usuario	1.4 m.

2.3 ETAPAS DEL RECONOCIMIENTO FACIAL

En la Figura 14, se indica el diagrama de bloques para un sistema de reconocimiento facial, en este se puede observar 3 componentes: base de datos, módulo de inscripción y módulo de reconocimiento. Estos componentes ejecutan las funciones necesarias para reconocer a un usuario [8].

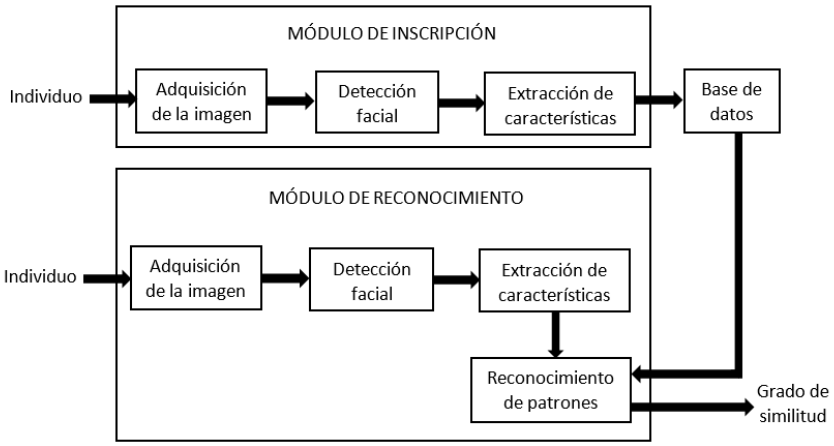


Figura 14 *Etapas del reconocimiento facial*
Fuente: [8]

Módulo de inscripción: Este está formado por un sistema de adquisición que obtiene la imagen del rostro que caracteriza al usuario. Luego de obtener la imagen se realiza la detección facial y se extraen sus características que forman su patrón biométrico. Estas características forman un vector con menor dimensión que la imagen previamente adquirida. Así, se realiza la codificación óptima de la imagen eliminando información irrelevante [8].

Base de datos: Las características de las imágenes obtenidas en el módulo de inscripción son almacenadas en una base de datos del software de reconocimiento. La base de datos contiene todas las imágenes de los usuarios pertenecientes al software [8].

Módulo de reconocimiento: Es la etapa final donde se establece la identidad del usuario que se desea identificar. Para esto, el software compara la imagen adquirida por el módulo de inscripción con las imágenes almacenadas en la base de datos. El resultado de esta comparación es cuantificado y valorado, lo cual permite al software tomar decisiones respecto a la identidad del usuario en función de la similitud obtenida [8]. Si la comparación es exitosa se obtiene el usuario cuyas características faciales están almacenadas en la base de datos [34].

2.4 CREACIÓN DE LA BASE DE DATOS

2.4.1 ADQUISICIÓN DE IMAGENES

Para la adquisición de imágenes que formaran la base de datos, se realizó bajo la condición "ambiente controlado" (entorno cerrado donde parámetros como luz, humedad, temperatura están controlados en su totalidad) con lo cual nos permitirá tener características similares en cada una de las imágenes. Estas características para nuestro caso son iluminación, distancia y fondo. A continuación, se detalla de manera específica los parámetros establecidos se muestra en la Figura 15.

- Fondo de color azul eléctrico
- Distancia entre la cámara y persona de 1.5 metros
- Iluminación basada en la luz natural del día

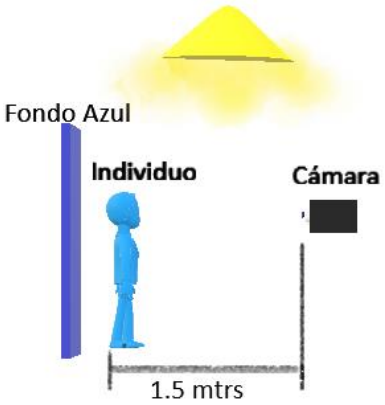


Figura 15 *Adquisición de imágenes para base de datos*
Fuente: [Autores]

Se utilizó una cámara *SONY CYBER SHOT DSC-W560 14.1MP*. El formato de calidad de la imagen fue establecido en ESTANDAR; siendo la dimensión en pixeles de 1920×1080, profundidad en bits 24, representación de color sRGB.

Se entiende por profundidad de imagen al número de bits que se utilizan para codificar y almacenar la información de color por cada pixel de una imagen [43]. Es decir, a mayor profundidad mayor paleta de colores, por ejemplo: imágenes de 1 bit de profundidad correspondería dos colores (0=negro, 1=blanco), para imágenes con 2 bits de profundidad correspondería cuatro colores (00=color negro, 01=color X, 10=color Y, 11=color blanco) y así sucesivamente para $2^{n\ bits\ de\ profundidad}$. Dicho esto, nuestras imágenes de 24 bits de profundidad tendrán 2^{24} colores. En la Figura 16 se muestra el modelo de la cámara utilizada.

Figura 16 *Cámara utilizada, SONY CYBER SHOT DSC-W560 14.1MP*
Fuente: [Autores]

En la Tabla 3 se indica el total de fotografías tomadas a cada usuario.

Tabla 3 Fotografías por cada usuario.
Fuente: [Autores]

ESTADOS A REPRESENTAR POR LA PERSONA				SUBTOTAL
Normal	Alegre	Gestos	Usa lentes*****	
8	5	5	5*	18
NUMERO DE PERSONAS				32
* Si usa lentes se tomarán 5 fotografías reemplazando a cualquiera de los 3 gestos.			TOTAL FOTOGRAFIAS	576

En total se tomaron 18 fotografías a cada persona con 5 posiciones o ángulos diferentes de la cara. Para lograr el número indicado se capturó las 8 primeras en estado normal, las siguientes 5 capturas en estado alegre y las 5 finales realizando cualquier gesto. Existió casos en donde la persona utilizaba lentes con lo que se

capturo 5 imágenes usándolas, pasando a representar cualquiera de los estados (normal, alegre o gestos). En la Figura 17 se puede observar los diferentes gestos.

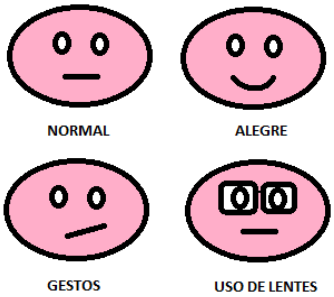


Figura 17 *Estados que a representar por las personas (normal, alegre y gestos; un cuarto estado que representa al caso donde utiliza lentes.*
Fuente: [Autores]

2.4.2 DETECCIÓN FACIAL

Para realizar la detección facial, se utilizó el algoritmo de Viola-Jones el cual identifica y enmarca el área del rostro. Se debe tener en cuenta que a estas imágenes se les aplicó previamente un filtro espacial para mejorar su nitidez, como se indica en la Figura 18. Este procedimiento se realiza a cada una de las fotografías tomadas.

Figura 18 ***Detección facial***
Fuente: [Autores]

2.4.3 BASE DE DATOS

Luego de haber realizado la adquisición de imágenes y la detección facial, se forma el archivo que contendrá las 576 imágenes de los usuarios. En la Figura 19, se presenta las imágenes originales, y en la Figura 20, se muestra la base de datos.

Figura 19 *Base de datos creada de un usuario*
Fuente: [Autores]

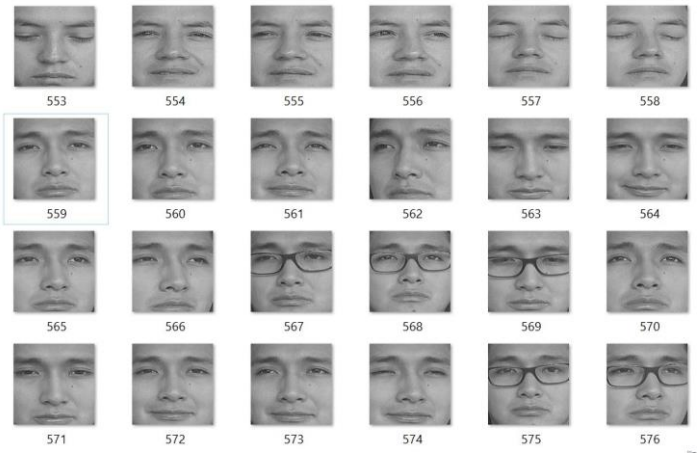


Figura 20 *Base de datos creada de un usuario.*
Fuente: [Autores]

El procedimiento para la creación de la base de datos con las fotografías establecidas se debe realizar de manera secuencial como se indica a continuación:

1. Ingresa la foto normal, se recorta la dimensión de 1920x1080 pixeles a 901x645 pixeles. Esto para eliminar áreas sin interés y mejorar el rendimiento.

2. Aplicamos un filtro espacial para mejorar la nitidez de las imágenes.

3. Detección facial con el algoritmo de Viola-Jones.

4. Recorte del área de rostro identificado y reducción de las dimensiones a 210x210 pixeles.

5. Luego de haber reducido las dimensiones, nuevamente se recorta a 120x160 pixeles; para reducir el error debido a zonas que no contienen mayor información.

6. De igual manera que en el paso 4, se reduce las dimensiones, pero ahora a 120x120 pixeles.

7. Finalmente se convierte a escala de grises.

Este proceso se realiza en las 576 fotografías para formar la base de datos final.

2.4.4 INTERFAZ GRAFICA

Se realiza una interfaz gráfica principal para el reconocimiento facial como se indica en la Figura 21, una secundaria para la base de datos general indicada en la Figura 22, una tercera que permite agregar datos de un nuevo usuario como se muestra en la Figura 23, y una cuarta interfaz que crea un nuevo usuario junto con sus datos, mostrada en la Figura 24. A continuación, se explica el contenido de cada interfaz y su funcionamiento.

2.4.4.1 Interfaz gráfica principal

Para interactuar con el usuario final, se diseñó varias interfaces, las cuales se irán abriendo según la necesidad del personal encargado. A continuación, en la Figura 21, se muestra la interfaz principal.

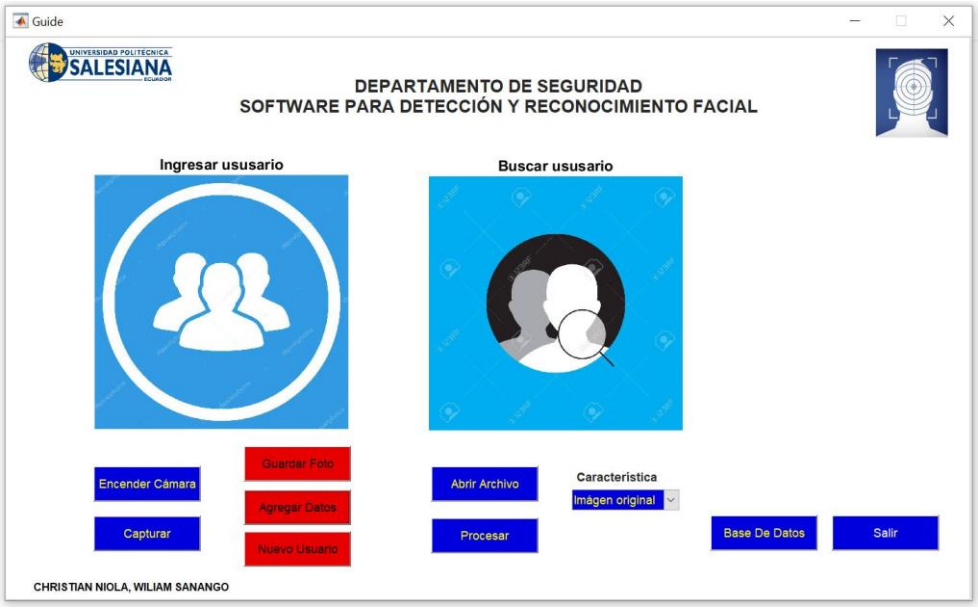


Figura 21 *Interfaz gráfica principal.*
Fuente: [Autores]

Esta interfaz se compone de:

- Botón "Encender Cámara": este permite encender la cámara principal del ordenador.
- Botón "Capturar": toma una nueva fotografía.
- Botón "Guardar Foto": guarda una nueva imagen en la base de datos creada.
- Botón "Agregar Datos": nos redirige a la tercera interfaz. Esta interfaz se muestra en la Figura 23.
- Botón "Nuevo Usuario": este botón nos redirige a la cuarta interfaz. Esta interfaz se muestra en la Figura 24.
- Botón "Abrir Archivo": nos abre un archivo de tipo imagen (.jpg), desde cualquier dirección del ordenador.
- Botón "Procesar": en este botón se encuentra todo el proceso para la detección y reconocimiento facial.
- Botón "Características de imagen": en esta parte se despliega un menú con las siguientes opciones:
  - Imagen Original
  - Detección facial: dibuja un recuadro en todos los rostros detectados
  - Área de trabajo: recorta el área de rostro detectado
  - Escala de grises: transforma la imagen RGB en escala de grises

- Imagen Reconstruida: trata de reconstruir la imagen en base a los eigenfaces.

- Botón “Base De Datos”: redirige a la segunda interfaz. Esta interfaz se presenta en la Figura 22.
- Etiqueta de información: muestra los datos del usuario (nombre, edad, domicilio, antecedentes)
- Etiqueta de tiempo de ejecución: indica el tiempo que tarda el software en realizar la identificación.
- Botón “Salir”: cierra todas las ventanas.

2.4.4.2 Segunda Interfaz

Para interactuar con el usuario final, se muestra la interfaz secundaria para la “Base de Datos Existente”, las cual se irá abriendo según la necesidad del personal encargado. A continuación, en la Figura 22, se muestra la interfaz secundaria.

Figura 22 *Interfaz secundaria “Base de Datos Existente”*
Fuente: [Autores]

Esta interfaz permite visualizar diferentes características de la “Base de Datos Existente”, y está compuesta por:

- Botón “Características”: aquí se despliega un menú con las siguientes opciones:
  - Base de Datos: visualiza las imágenes existentes de la base de datos.

- Normalizar: normaliza las imágenes existentes para reducir errores.
- Media: obtiene la media de las imágenes de la base de datos.
- Eigenfaces: visualiza los vectores propios y valores propios de las imágenes.

En la parte derecha se visualiza las características seleccionadas.

- Etiqueta de tiempo de ejecución: indica el tiempo que tarda el software en realizar las operaciones.
- Etiqueta de información: muestra el número total de imágenes contenidas en la base de datos existente.
- Botón "Terminar": cierra la interfaz actual para regresar a la principal.

2.4.4.3 Tercera Interfaz

Para interactuar con el usuario final, se muestra la tercera interfaz, la cual permite ingresar y guardar datos de un nuevo usuario. Esta se irá abriendo según la necesidad del personal. A continuación, en la Figura 23, se muestra la tercera interfaz.

Figura 23 *Tercera interfaz "Crear Nuevo Usuario". Permite ingresar y guardar datos de un nuevo usuario.*
Fuente: [Autores]

Esta interfaz permite ingresar y guardar datos de un nuevo usuario, y se compone de:

- Botón "Guardar Datos": guarda los datos ingresados en el editor de texto.
- Etiqueta de datos: lugar donde se ingresa información del nuevo usuario (nombre, edad, domicilio, antecedentes)

- Botón “Terminar”: cierra la ventana y regresa a la interfaz principal.

2.4.4.4 Cuarta Interfaz

En la Figura 24, se presenta la cuarta interfaz que permite agregar un nuevo usuario a la base de datos existente y añadir información de este. Como se puede observar en esta interfaz, al igual que la tercera, tiene la misma estructura para ingresar los datos.

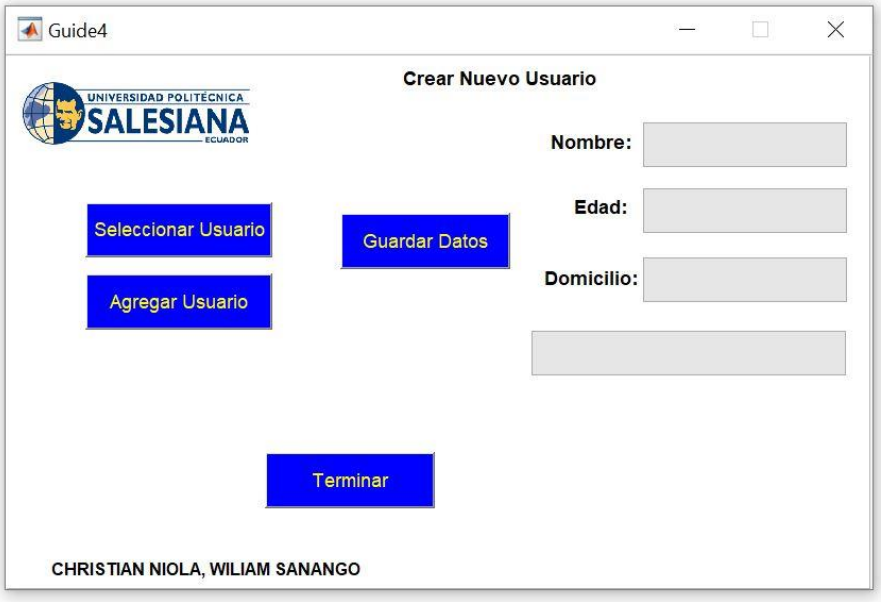


Figura 24 *Cuarta Interfaz gráfica "Crear Nuevo Usuario”. Esta permite agregar un nuevo usuario con toda su información.*
Fuente: [Autores]

Esta interfaz permite agregar un nuevo usuario a la base de datos existente y añadir información del mismo, esta interfaz se compone de:

- Botón “Seleccionar Usuario”: permite seleccionar una carpeta de imágenes desde cualquier dirección del ordenador.
- Botón “Agregar Usuario”: detecta, recorta, cambia a escala de grises y guarda en la base de datos existente las imágenes faciales.
- Etiqueta de información: muestra el número de imágenes contenidas en la carpeta seleccionada.
- Botón “Guardar Datos”: guarda los datos ingresados en el editor de texto.
- Etiqueta de datos: lugar donde se ingresa información del nuevo usuario (nombre, edad, domicilio, antecedentes)
- Botón “Terminar”: cierra la ventana y regresa a la interfaz principal.

CAPÍTULO 3: ANÁLISIS Y RESULTADOS

3.1 PRUEBAS DE DETECCIÓN FACIAL

Estas pruebas se realizaron con 576 fotografías de 32 personas (18 cada una), utilizando dos métodos: con y sin preprocesamiento de las imágenes.

En el primer caso, se utilizó directamente el algoritmo de Viola-Jones para detectar rostros sobre las imágenes, los cuales son enmarcados y se indican en la Figura 25. A continuación, en la Tabla 4, se presenta los resultados obtenidos.

Tabla 4 Resultados obtenidos en las pruebas de detección facial sin preprocesamiento de las imágenes.
Fuente: [Autores]

Usuarios	Fotografías por usuario	Fotografías totales	Detección facial (por fotos)		Porcentaje
32	18	576	Aciertos	513	89.06%
			Desaciertos	63	

Como se observa en la Tabla 4, existe un total de 63 fotografías en las cuales el algoritmo de Viola-Jones detecta zonas que no representan un rostro, dando un porcentaje general del 89.06% de aciertos (10.94% desaciertos). Esto se puede observar en la Figura 13, donde se detectó 2 zonas que no representan un rostro, estas están indicadas con las franjas en verde sobre la vestimenta (camiseta roja).

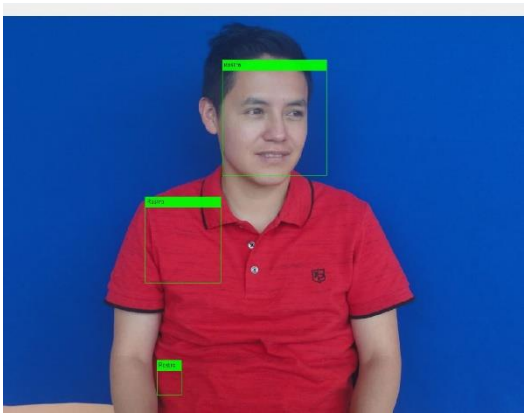


Figura 25 *Detección facial errónea (dos rostros falsos sobre la vestimenta).*
Fuente: [Autores]

Para solucionar dicho problema se realizó el segundo caso donde se aplica un preprocesamiento de la imagen. Aquí se efectúa un recorte para reducir la imagen a 901x645 pixeles, luego; a esta imagen recortada se le aplica un filtro espacial. Estos dos procedimientos permiten eliminar la detección facial errónea.

Luego de haber realizado el preprocesamiento de la imagen, se comprueba la eliminación de la detección errónea de rostros sobre la vestimenta, esto se observa en la Figura 26.

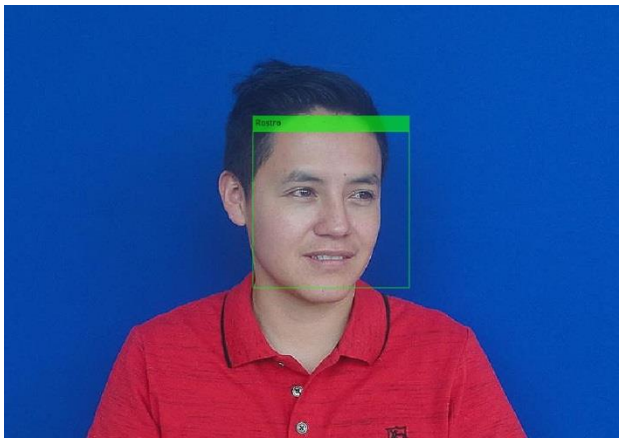


Figura 26 *Detección de facial exitosa.*
Fuente: [Autores]

A continuación, en la Tabla 5, se presenta el resultado luego de aplicar el algoritmo de detección facial sobre las imágenes preprocesadas.

Tabla 5 Resultado obtenido en la prueba de detección facial con preprocesamiento de las imágenes
Fuente: [Autores]

Número de usuarios	Fotografías por usuario	Fotografías totales	Detección facial		Porcentaje
32	18	576	Aciertos	576	100%
			Desaciertos	0	

El resultado indica que, en las nuevas imágenes con preprocesamiento (filtrado y recorte), todas las detecciones faciales erróneas se eliminan.

3.2 PRUEBAS DE RECONOCIMIENTO FACIAL

Las pruebas de reconocimiento facial se realizaron en un ambiente controlado en diferentes escenarios, los cuales se detallan a continuación:

1. Se recurre a las fotografías originales (1920 × 1080 pixeles), utilizadas en la creación de la base de datos, para probar el algoritmo de reconocimiento de imágenes PCA-Eigenfaces, de manera general y de manera específica el reconocimiento facial. Es decir, la foto a procesar se encuentra en la base de datos junto con el rostro del usuario. Al realizar esta prueba se obtuvo los siguientes resultados.

Tabla 6 Resultados obtenidos en las pruebas de reconocimiento de usuarios y sus fotografías
Fuente: [Autores]

Número de usuarios	Fotografías por usuario	Número total de fotografías	Reconocimiento por fotografías	Reconocimiento por usuarios	Porcentaje de aciertos por fotografías	Porcentaje de aciertos por usuario
32	18	576	576	32	100%	100%

Como se puede ver en la Tabla 6, las 576 fotografías que forman parte de la base de datos son identificados consigo mismas, la base de datos consta de 32 usuarios y cada usuario cuenta con 18 fotografías. El software reconoce la misma fotografía y por ende al mismo usuario, dando un resultado general del 100% de éxito en el reconocimiento facial. A continuación, en la Figura 27, se presenta un ejemplo realizado.

Figura 27 *Prueba del escenario 1, de un usuario*.
Fuente: [Autores]

2. Se toma nuevas fotografías (1920×1080 pixeles), a cinco usuarios que constan en la base de datos en un ambiente similar (estas fotografías no se agregan a la base de datos) y se procede a realizar el reconocimiento facial. En la Tabla 7, se muestran los resultados obtenidos en este escenario.

Tabla 7 Resultados obtenidos de las pruebas del escenario 2.
Fuente: [Autores]

Número de usuarios	Número de fotografías con rostro frontal	Número de fotografías con inclinación del rostro	Número total de fotografías	Numero de aciertos		Número total de aciertos por fotografía	Número total de desaciertos por fotografía	Porcentaje de aciertos total
				Rostro frontal	Rostro con inclinación			
1	6	10	16	6	5	11	5	68.75%
2	6	10	16	6	7	13	3	81.25%
3	6	10	16	6	9	15	1	93.75%
4	6	10	16	6	5	11	5	68.75%
5	6	10	16	6	6	12	4	75%

En este escenario se tienen 16 nuevas fotografías de 5 usuarios que constan en la base de datos. Las fotografías fueron tomadas con diferentes gestos y posiciones del rostro (frontal y con movimiento hacia arriba, abajo. Izquierda y derecha). Se tienen 6 fotografías frontales y 10 con movimiento por cada usuario.

Como se observa en la Tabla 7, las 6 fotografías frontales de los 5 usuarios son reconocidas exitosamente. El software falla cuando las fotografías presentan un cambio en la posición del rostro. Para los 5 usuarios, las fotografías con cambios en la posición que son reconocidas presentan un movimiento menor a 45 grados; pasado este ángulo, el software realiza una detección fallida. Se debe tener en cuenta que, a medida que la inclinación aumenta, la distancia euclidiana mínima crece.

Pese a que se tiene una detección exitosa hasta los 45 grados de inclinación como se muestra en la Figura 28, esto provocaría fallos cuando se tengan usuarios que no constan en la base de datos, ya a que, es necesario colocar un límite de aceptación en la distancia euclidiana, y si esta distancia aumenta, un usuario que consta en la base de datos podría quedar fuera del límite fallando en la detección como se muestra en la Figura 29.

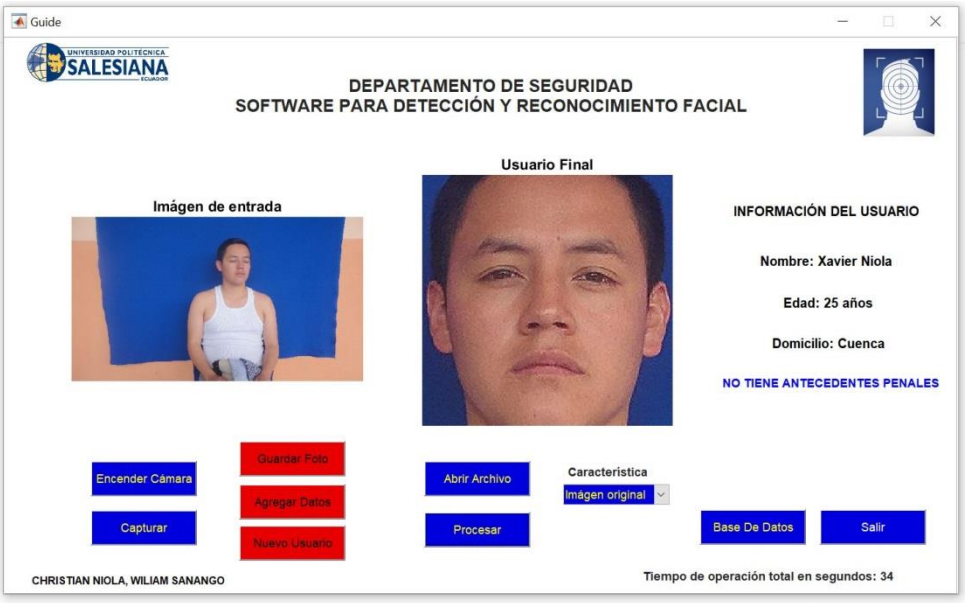


Figura 28 *Ejemplo de la prueba de reconocimiento facial del escenario 2, exitosa.*
Fuente: [Autores]

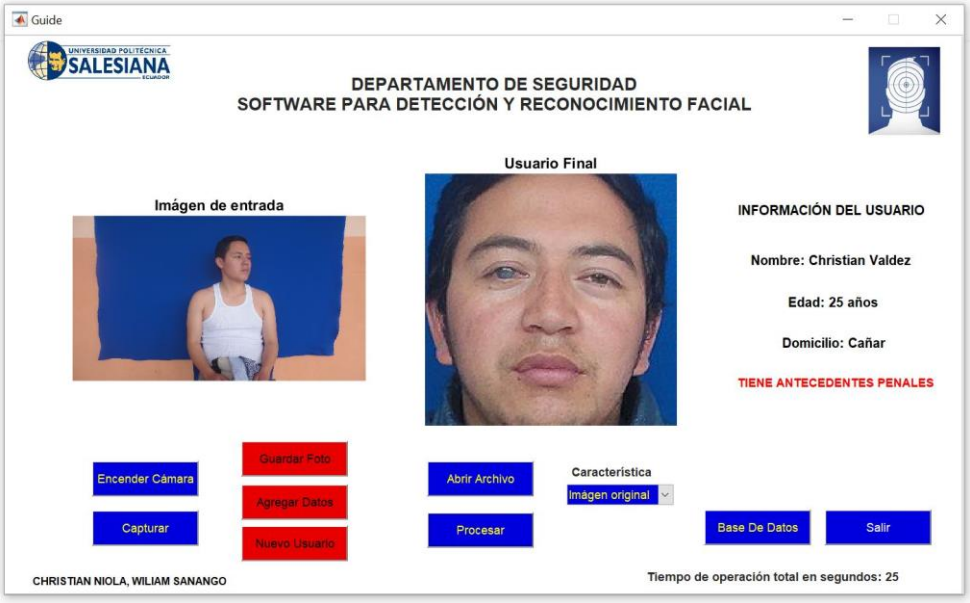


Figura 29 *Ejemplo de detección facial fallida, debido al exceso inclinación.*
Fuente: [Autores]

Finalizadas las pruebas del escenario 1 y 2 se determina un valor de umbral para delimitar el reconocimiento a usuarios que no pertenezcan a la base de datos; para ello utilizamos las distancias euclidianas mínimas que se obtuvieron.

- En el escenario 1, el rango de la distancia euclidiana mínima es de 1.55 a 3.95 unidades.

- En el escenario 2, el rango de la distancia euclidiana mínima es de 2.72 a 6.71 unidades.

Con los rangos obtenidos en las pruebas de reconocimiento facial, se delimita al software para que excluya fotografías con distancias euclidianas menores o iguales a 4.75 unidades (limite). Si se supera este valor se muestra un mensaje de no existencia del usuario en la base de datos.

Una vez delimitando el software realizamos las pruebas en los siguientes escenarios.

3. Se toma fotografías (1920x1080 pixeles) a cinco usuarios que no forman parte de la base de datos en un ambiente similar y realizamos el procedimiento de reconocimiento facial, donde se debe rechazar las fotografías que superen la distancia euclidiana (limite 4.75). En la Tabla 8, se indica el resumen de los resultados obtenidos.

Tabla 8 Numero de aciertos de no existencia de la persona en la base de datos.
Fuente: [Autores]

Número de usuarios	Número de fotografías con rostro frontal	Número de fotógrafas con inclinación del rostro	Número total de fotografías	Número de fotografías descartadas (superan el límite)	Número de usuarios descartados	Porcentaje de usuarios descartados
1	8	10	18	18	5	100%
2	8	10	18	18		
3	8	10	18	18		
4	8	10	18	18		
5	8	10	18	18		

Aquí se tienen 18 fotografías de 5 usuarios que no constan en la base de datos, Las fotografías fueron tomadas con diferentes gestos y posiciones del rostro (frontal y con movimiento hacia arriba, abajo. Izquierda y derecha). Se tienen 8 fotografías frontales y 10 con movimiento por cada usuario.

Como se muestra en la Tabla 8, las 18 fotografías de los 5 usuarios son descartadas por el software debido a que superan el límite establecido, dando un

resultado general del 100% de éxito en el no reconocimiento facial. A continuación, en la Figura 30, se muestra un ejemplo de no reconocimiento facial.

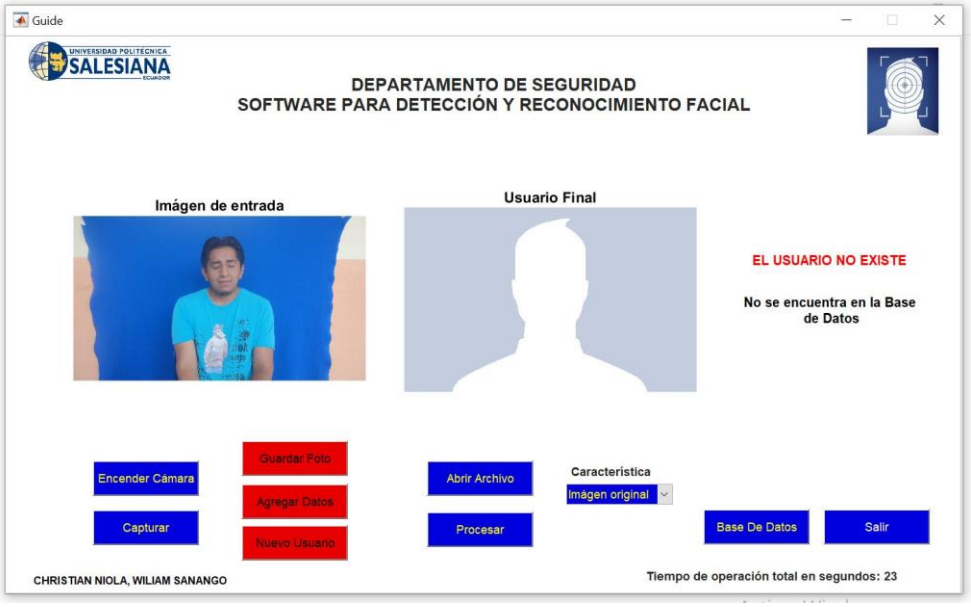


Figura 30 *Rechazo de usuario que no pertenece a la base de datos.*
Fuente: [Autores]

4. Para este escenario, se agrega a la base de datos las 18 fotografías de los cinco usuarios del escenario 3, incluyendo su información personal. Luego, se toma nuevas fotografías (1920x1080 pixeles) de cada usuario y se procede a realizar el reconocimiento facial. En la Figura 31, se muestra el ingreso de los datos de un usuario y el número de fotografías agregadas a la base de datos.

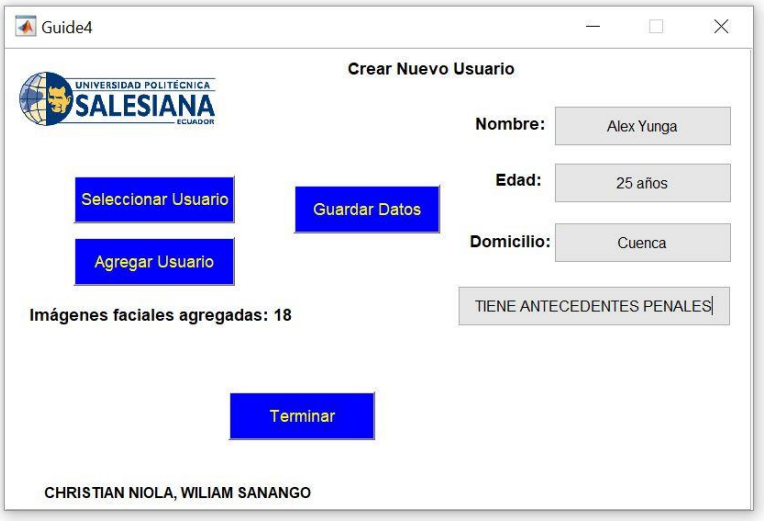


Figura 31 *Interfaz donde se ha realizado la creación de un usuario nuevo.*
Fuente: [Autores]

En la tabla 9 se presentan los resultados obtenidos en este escenario, donde el software reconoció a los usuarios agregados.

Tabla 9 Reconocimiento exitoso de usuarios nuevos que se agregaron a la base de datos. Fuente: [Autores]

Número de usuarios	Número de imágenes con rostro frontal	Número de imágenes con inclinación del rostro	Número total de imágenes	Numero de aciertos dentro del limite		Numero de aciertos por fotografías dentro del limite	Porcentaje de aciertos por fotografías
				Rostro frontal	Rostro con inclinación		
1	6	6	12	6	6	12	100%
2	6	6	12	6	6	12	100%
3	6	6	12	5	2	7	58.33%
4	6	6	12	6	6	11	100%
5	6	6	12	6	6	12	100%

Luego de haber agregado los cinco usuarios a la base de datos, se toman 12 fotografías nuevas con diferentes gestos y posiciones del rostro (frontal y con movimiento hacia arriba, abajo. Izquierda y derecha). Se tienen 6 fotografías frontales y 6 con movimiento leve por cada usuario.

Como se observa en la Tabla 9, el único usuario que presenta una distancia euclidiana mayor al umbral es el usuario 3. Se descarta una fotografía frontal y 4 con movimiento. La fotografía frontal que se descarta presenta un gesto que no está incluido en la base de datos. Para los demás usuarios, las 12 fotografías fueron tomadas con movimientos leves en su posición y con gestos similares a las fotografías de la base de datos. En la Figura 32 se muestra un ejemplo del usuario agregado.

Figura 32 *Ejemplo de reconocimiento de un usuario agregado.*
Fuente: [Autores]

La mayoría de las imágenes con gestos diferentes y movimientos leves son reconocidas exitosamente, pero presentan una distancia euclidiana mínima mayor al umbral; por lo tanto, el reconocimiento de usuarios que constan en la base de datos es exitosa cuando no existe mucha variación entre estos parámetros. Todas las pruebas de los diferentes escenarios fueron realizadas en un ambiente que presenta el mismo nivel de iluminación. Con lo mencionado se puede concluir que el software desarrollado funciona de manera deseada en un ambiente controlado.

CAPÍTULO 4: CONCLUSIONES Y RECOMENDACIONES

Se realizaron pruebas iterativas a medida que se desarrollaba el software, para ir dando solución a los errores que se presentasen. Como, por ejemplo, detección de falsas caras a la hora de crear la base de datos; Para solucionar este problema se aplicó un Filtro Espacial, el cual mejoró la nitidez de la fotografía, sumado a esto se recortó las imágenes tratando de eliminar el área que no contenía el rostro. Luego de haber solucionado los errores, se desarrolló las pruebas finales de detección y reconocimiento facial en distintos escenarios.

Para establecer si una fotografía corresponde o no a un usuario de la base de datos, se aplicó el criterio de distancia euclidiana mínima. En las pruebas realizadas en los escenarios 1, 2 y 3, se utilizó fotografías de usuarios que forman parte de la base de datos para establecer un límite, este límite permite restringir el acceso a usuarios con fotografías que lo superen. En las pruebas del escenario 4 se utilizó fotografías de usuarios que no forman parte de la base de datos para verificar que el límite planteado es el adecuado, obteniendo buenos resultados, en este escenario el software rechazó todas las fotografías

Para el caso uno, los parámetros de iluminación, inclinación del rostro y expresiones faciales no afectan en el reconocimiento. Mientras que para los demás casos estos parámetros son esenciales para un reconocimiento exitoso. Las fotografías con el rostro inclinado de usuarios que no forman parte de la base de datos tienen una distancia euclidiana pequeña, ya que se asemejan en gran medida a las fotografías de usuarios con rostro inclinado que si pertenecen a la base de datos; generando un reconocimiento facial erróneo debido a que no supera el límite establecido. Las fotografías con gestos diferentes de usuarios que se encuentran en la base de datos presentan una distancia euclidiana mayor al límite, lo cual provoca que el programa los descarte.

Para reducir al mínimo la distancia euclidiana de los usuarios a reconocer, se tendría que aumentar el número de fotografías con distintos gestos en la base de datos. Al incrementar el número de imágenes de un usuario en la base de datos,

aumenta el porcentaje de éxito en el reconocimiento facial, pero también aumenta el tiempo necesario para su ejecución, lo que representa una gran desventaja en cuanto a eficiencia computacional.

Con los resultados conseguidos en las diferentes pruebas realizadas, se verifica que el software cumple con el objetivo de detectar y reconocer a un individuo. Pero se debe tener en cuenta ciertos parámetros como: la iluminación, inclinación del rostro y expresiones faciales; ya que la variación de estos factores afecta en gran medida al resultado deseado. El algoritmo de Viola-Jones presenta fallos en la detección cuando el rostro está demasiado inclinado o la foto es tomada a una distancia muy grande; por otro lado, el algoritmo PCA- Eigenfaces está asociado y limitado por todos los parámetros descritos anteriormente, es por esto que el software desarrollado da excelentes resultados cuando las imágenes son adquiridas en un ambiente controlado y las fotos son tomadas de frente. Existen métodos en el cual no afectan estas condiciones como, por ejemplo: el método de la geometría tridimensional del rostro, el cual representa la estructura anatómica interna de la cara, en lugar de su apariencia externa, que es influenciada por las condiciones dichas anteriormente del entorno. Por lo cual, las superficies faciales 3D no son afectadas por la iluminación, postura de la cabeza, pero, siguen teniendo problemas con las expresiones faciales.

Trabajo futuro

Un posible trabajo futuro es la ampliación de la base de datos. Se podría adicionar nuevos usuarios al sistema o agregar imágenes con una variación en los gestos de los usuarios existentes, también se podría incluir imágenes tras el transcurso de intervalos de tiempo; esto permitiría experimentar la robustez del sistema ante cambios de los individuos debido a gestos y al paso del tiempo.

También se podría probar nuevas técnicas de reconocimiento facial que no se vean afectados por el cambio en la posición del rostro; ya que el algoritmo implementado pierde confiabilidad cuando la imagen de prueba no es tomada de frente, una posible solución a esto sería trabajar con imágenes 3D

El sistema desarrollado también trabaja con la webcam de un ordenador, se podría modificar para que funcione con una cámara instalada en un lugar específico que, junto con una interfaz funcional, permita controlar y evaluar el sistema.

REFERENCIAS BIBLIOGRÁFICAS

[1] C. Tolosa Borja and Á. Giz Bueno, "Sistemas Biométricos," vol. 1, p. 39, 2010.

[2] Greer Williams, "BREVE HISTORIA DEL RECONOCIMIENTO FACIAL - FaceFirst Face Recognition Software," 2018. [Online]. Available: https://www.facefirst.com/blog/breve-historia-del-reconocimiento-facial/. [Accessed: 19-Feb-2019].

[3] Servicio Integrado de Seguridad ECU 911, "Rendición de Cuenta 2017 – Servicio Integrado de Seguridad ECU 911," 2018. [Online]. Available: http://www.ecu911.gob.ec/rendicion-de-cuenta-2017/. [Accessed: 20-Feb-2019].

[4] R. G. Elías, "2.300 inocentes marcados como criminales en una noche, el problema del reconocimiento facial - Omicrono," 2018. [Online]. Available: https://omicrono.elespanol.com/2018/05/reconocimiento-facial-policia-falsos-positivos/. [Accessed: 20-Feb-2019].

[5] T. Evelyn, "Banco Guayaquil lanza sistema de reconocimiento facial para transacciones | Revista Líderes," 2015. [Online]. Available: https://www.revistalideres.ec/lideres/banco-guayaquil-lanza-sistema-reconocimiento.html. [Accessed: 20-Feb-2019].

[6] P. Tomé, "Reconocimiento Automático de Patrones de Iris," p. 139, 2008.

[7] M. Gonzalez, "Reconocimiento de iris," *Trab. grado - Ing. Inform.*, pp. 1–12, 2006.

[8] D. E. Villalon de la Vega, "Diseño e implementación de una plataforma de software para reconocimiento facial en video," 2012.

[9] E. D. Parra Gerson, "Biometría," *Univ. Politécnica Madrid*, p. 22, 2012.

[10] G. Pusiol, "Sistema de verificación de huellas digitales," p. 45, 2007.

[11] L. Kimaldi Lusa, "Sistemas biométricos |Kimaldi." [Online]. Available: https://www.kimaldi.com/productos/sistemas_biometricos/. [Accessed: 19-Feb-2019].

[12] K. De Sousa and C. Mora, "Sistema de seguridad basado en reconocimiento facial utilizando una Raspberry Pi," 2016.

[13] S. de I. biométrica ReconoSer, "Equipos Biométricos para el Reconocimiento

Facial - ReconoSer.” [Online]. Available: https://www.reconoser.com.co/equipos-reconocimiento-facial.html. [Accessed: 19-Feb-2019].

[14] Murdotech Sistemas, “Equipos Reconocimiento Facial | Murdoch Sistemas,” 2018. [Online]. Available: https://murdochsistemas.com/Reconocimiento-Facial. [Accessed: 19-Feb-2019].

[15] E. Yörük, E. Konukoğlu, B. Sankur, and J. Darbon, “Shape-based hand recognition,” *IEEE Trans. Image Process.*, vol. 15, no. 7, pp. 1803–1815, 2006.

[16] Schlage, “Control de acceso lectores biométricos de Geometria de mano.” [Online]. Available: https://schlage.com.ar/biometria-de-mano/. [Accessed: 19-Feb-2019].

[17] TECNOSMART, “Reloj Biométrico Allegion Hand Punch HP-2000, 500 usuarios.” [Online]. Available: http://www.tecnosmart.com.ec/v2/productos/reloj-biometrico-allegion-hand-punch-hp-2000.html. [Accessed: 20-Feb-2019].

[18] C. Santiago, M. Jonathan, and G. Montoya, “Biometría: conveniente y segura,” pp. 1–11, 2018.

[19] Clínica Baviera, “Iris del ojo,” 2017. [Online]. Available: https://www.clinicabaviera.com/blog/bye-bye-gafasconoce-tus-ojosiris-del-ojo-que-es-y-que-funcion-tiene/. [Accessed: 19-Feb-2019].

[20] NARA, “U.S National Archives,” *U.S. Natl. Arch.*, Jun. 2004.

[21] Mark Pellegrini, “IriScan model 2100 iris scanner 1.jpg,” 2007. [Online]. Available: https://commons.wikimedia.org/wiki/File:IriScan_model_2100_iris_scanner_1.jpg. [Accessed: 20-Feb-2019].

[22] ECISEC S.A., “Biométrico UltraMatch,” 2015. [Online]. Available: http://www.ecisec.com.ec/detalle-producto/ultramatch/. [Accessed: 20-Feb-2019].

[23] MedlinePlus, “Iris: MedlinePlus enciclopedia médica,” 2018. [Online]. Available: https://medlineplus.gov/spanish/ency/article/002386.htm. [Accessed: 25-Nov-2018].

[24] F. Ochoa, C. San Martin, and R. Carrillo, “Identificacion Biometrica De Locutores Para el ambito forense,” pp. 1–10, 2008.

[25] ITVOZ, “Kit de Dictado y Reconocimiento de Voz PSE4000.” [Online]. Available: https://www.itvoz.com/es/reconocimiento-de-voz/115-speechmike-premium-air-con-reconocimiento-de-voz.html. [Accessed: 20-Feb-2019].

[26] V. Sarnataro, “Dragon Naturally Speaking,” *technologyguide.com*, Nov. 2012.

[27] EcuRed, “Firma biométrica - EcuRed,” 2018. [Online]. Available: https://www.ecured.cu/Firma_biométrica. [Accessed: 26-Nov-2018].

[28] P. P. Nieto, “Firma digitalizada biométrica en las AAPP Experiencia y casos de uso en la AEAT,” 2017.

[29] G. David, “Java WebStart,” 2017. [Online]. Available: https://medium.com/el-acordeon-del-programador/java-webstart-desde-cero-db63a7ba71e8. [Accessed: 20-Feb-2019].

[30] Habeeb Onawole, “Microsoft Surface Studio 2,” 2018. [Online]. Available: https://www.gizmochina.com/2018/09/27/geekbench-listing-reveals-microsoft-surface-studio-2-will-pack-32gb-ram/. [Accessed: 20-Feb-2019].

[31] WACOM, “Pantalla de fácil lectura: Wacom STU-430 / STU-430V.” [Online]. Available: http://signature.wacom.eu/es/productos/hardware/stu-430/. [Accessed: 20-Feb-2019].

[32] S. B. Palacios, “Sistema De Deteccíon Y Reconocimiento Facial De Conductores Mediante Sistemas De Visión Computacional,” 2008.

[33] E. Parra, B. Tutores, and J. Fern, “Aceleración del algoritmo de Viola-Jones mediante rejillas de procesamiento masivamente paralelo en el plano focal,” 2015.

[34] C. Barroso, “IMPLEMENTACIÓN DEL ALGORITMO DE DETECCIÓN DE CARAS DE VIOLA Y JONES SOBRE UNA FPGA,” pp. 1–129, 2014.

[35] C. Modesto Fernando, “Tutorial Viola-Jones,” p. 11, 2009.

[36] P. Viola and M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features,” *Surf. Sci.*, vol. 394, no. 1–3, 1997.

[37] V. B. David, “Empleo de sistemas biométricos faciales aplicados al reconocimiento de personas en aeropuertos,” 2006.

[38] Maya Binetskaya, “RECONOCIMIENTO FACIAL EN EL ÁMBITO FORENSE,” p. 114, 2013.

[39] A. Sánchez, “Análisis de Componentes Principales Versiones Robustas Dispersas y Robustas al Ruido Impulsivo,” p. 130, 2008.

[40] J. Alquicira, “Análisis de componentes principales (PCA),” *Nature*

Biotechnology, 13-Nov-2016. [Online]. Available: http://conogasi.org/articulos/analisis-de-componentes-principales-pca/. [Accessed: 12-Nov-2018].

[41] R. Baldrich, "Reducción de descriptores: PCA - Extracción de características," *Coursera*, 2018. [Online]. Available: https://es.coursera.org/lecture/clasificacion-imagenes/reduccion-de-descriptores-pca-PaTVm. [Accessed: 29-Nov-2018].

[42] J. A. U. Abad, "Reconocimiento Facial," Universidad Politecnica de Madrid, 2014.

[43] Instituto Superior de Formación y Recursos en Red para el Profesorado. and Formación en Red, "Profundidad de color," 2008. [Online]. Available: http://www.ite.educacion.es/formacion/materiales/107/cd/imagen/imagen0103.html. [Accessed: 21-Feb-2019].

APÉNDICES

APÉNDICE A: CÓDIGO IMPLEMENTADO

Guide Principal

```
function varargout = Guide(varargin)
clc
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...

'gui_Singleton', gui_Singleton, ...

'gui_OpeningFcn', @Guide_OpeningFcn, ...

'gui_OutputFcn', @Guide_OutputFcn, ...

'gui_LayoutFcn', [] , ...
                 'gui_Callback', []);

if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Guide is made visible.
function Guide_OpeningFcn(hObject, ~, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Guide (see VARARGIN)
axes(handles.axes8);
Fondo3 = imread(strcat('C:\ProgramaTesis\Fondo\','Fondo1.jpg'));
imshow(Fondo3)
axes(handles.axes10);
Fondo4 = imread(strcat('C:\ProgramaTesis\Fondo\','Fondo4.jpg'));
imshow(Fondo4)
axes(handles.axes1);
Fondo1 = imread(strcat('C:\ProgramaTesis\Fondo\','Fondo5.jpg'));
imshow(Fondo1); title('Ingresar ususario');
axes(handles.axes2);
Fondo2 = imread(strcat('C:\ProgramaTesis\Fondo\','Fondo6.jpg'));
imshow(Fondo2); title('Buscar ususario');

% Choose default command line output for Guide
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Guide wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Guide_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Ejecuta al presionar el botón pushbutton1 (Botón Abrir).
function pushbutton1_Callback(hObject, eventdata, handles)
set(handles.text1,'string','');
%Borra contenido de text.
set(handles.text12,'string','');
set(handles.text14,'string','');
set(handles.text18,'string','');
set(handles.text19,'string','');
set(handles.text20,'string','');
```

```
set(handles.text21,'string','');
global vid % Define una variable
global
global I
[filename          path]          =
uigetfile(('*.jpg'),'Seleccionar
Imágen'); % Abre cuadro de diálogo
de selección de archivos.
vid                                =
imread(strcat(path,filename));    %
Lee la imágen de la carpeta
seleccionada.
axes(handles.axes1); % Selecciona
el eje carteciano para visualizar
la imágen.
imshow(vid);    title('Imágen    de
entrada');
RGB=vid;
I=RGB(50:694,  500:1400,:);%Recorta
imágen de alta resolución 900..

% --- Ejecuta al presionar el
botón pushbutton4 (Botón Encender
Cámara).
function
pushbutton4_Callback(hObject,
eventdata, handles)
set(handles.text1,'string','');
%Borra contenido de text.
set(handles.text12,'string','');
set(handles.text14,'string','');
set(handles.text18,'string','');
set(handles.text19,'string','');
set(handles.text20,'string','');
set(handles.text21,'string','');
global vid % Define una variable
global.
axes(handles.axes1); % Selecciona
el eje carteciano para visualizar
la imágen.
vid=videoinput('winvideo',
1,'MJPG_1280x720'   );   %Crea   un
objeto de entrada de video y
seleccionamos el formato.
vidRes=get(vid,'VideoResolution');
% Muestra las propiedades del
adaptador gráfico 'Resolución'.
nBands=get(vid,'NumberOfBands'); %
Muestra las propiedades del
adaptador gráfico 'Número de
bandas'.
himage=
image(zeros(vidRes(2),vidRes(1),nB
ands)); % Muestra los datos en la
matriz como una imagen. .
rectangle('Position',[300  100  680
300],'EdgeColor','b'); % Dibuja un
rectángulo en la pantalla. izq,
arrib, der,abaj
preview(vid,himage); % Visualizar

% --- Ejecuta al presionar el
botón pushbutton6 (Botón Salir).
function
pushbutton6_Callback(hObject,
eventdata, handles)
opc = questdlg('¿Desea salir del
programa?','SALIR','Si','No','No')
; % Crea un cuadro de diálogo de
pregunta
if strcmp(opc,'No')
    return;
end
delete(handles.figure1); % Cierra
la figura 1 (ventana guide)

% --- Ejecuta al presionar el
botón      pushbutton7      (Botón
Capturar).
function
pushbutton7_Callback(hObject,
eventdata, handles)
global vid
global I
vid = getsnapshot(vid); % Devuelve
inmediatamente el cuadro de una
sola imágen.
axes(handles.axes1);
imshow(vid);    title('Imágen    de
entrada');
closepreview; % Cierra la ventana
de vista previa de vídeo.
RGB=vid;
I=RGB;

% --- Ejecuta al presionar el
botón      pushbutton9      (Botón
Procesar).
function
pushbutton9_Callback(hObject,
eventdata, handles)
tic                        % Toma el
tiempo de ejecución.
global I              % Contiene la
imágen del usuario a identifiar.
global IFaces         % Contiene la
imágen con la detección facial-
global work_area      % Contiene la
imágen facial recortadda.
global img            % Contiene la
imágen de la media.
global ReshapedImage % Contiene la
imágen reconstruida.
global M
tic;
a  ='C:\ProgramaTesis\BaseDeDatos';
% Dirección de la carpeta Base de
Datos
A  =dir( fullfile(a,  '*.jpg')); %
Muestra lista de contenidos de la
carpeta.
M=size(A,1);                       %
Número de imágenes en la base de
datos.

% Para normalizar las imágenes,
definimos     ustd     (desviación
estándar) y um (media) de tal
manera que
% su valor se encuentre cerca de
la media y la desviación estándar
de la mayoría de las imágenes.
ustd=80;um=100;

% Leer imágenes de la base de
datos (.jpg).
```

```
S=[];
for i=1:M
img                                =
imread(strcat('C:\ProgramaTesis\Ba
seDeDatos\',int2str(i),'.jpg')); %
Concatena la dirección de las
imágenes y dos cadenas que forman
el nombre de la imagen.
    [ifil          icol]=size(img);
% Obtenemos el número de filas
(N1) y columnas (N2).

temp=reshape(img',ifil*icol,1);   %
Reordena la matriz, crea una
matriz (N1*N2)x1.
    S=[S                      temp];
% Matriz con todas las imágenes
reordenadas (N1*N2)xM, en una sola
columna los valores de cada foto.
end

   for i=1:2
        axes(handles.axes2);
        Fondo2                     =
imread(strcat('C:\ProgramaTesis\Fo
ndo\','Fondo3.jpg'));
        imshow(Fondo2);
title('Iniciando Búsqueda');
   end

% Normalizar imágenes de la base
de datos para reducir el error
debido a las condiciones de
iluminación.
for i=1:size(S,2)          % Valor
del número de columnas de la
matriz S.
    temp=double(S(:,i));          %
Almacena las variables numéricas
como valores de punto flotante.
    m=mean(temp);         % Saca la
media de toda la columna, un solo
valor.
    st=std(temp);
    S(:,i)=(temp-m)*ustd/st+um;
%(foto-media                    de
foto)*ustd/desviacion estandar de
foto +um
end

% Muestra imágenes normalizadas
for i=1:M
    str=strcat(int2str(i),'.jpg');
    img=reshape(S(:,i),icol,ifil);
    img=img';
end

% Cálculo de la media de las
imagenes de la base de datos
m=mean(S,2);                       %
Obtiene la media de cada fila en
lugar de cada columna.
mn=uint8(m);                  % Se
convierte en un entero de 8 bits
sin signo. Los valores van de 0 a
255.
img=reshape(mn,icol,ifil);  % Toma
el vector N1*N2x1 y crea una
matriz N2xN1
img=img';                     % crea
una matriz N1xN2 mediante la
transposición de la imagen.

%    Cambio    de    imagen    por
manipulación.
X=[]; % X matrix
for i=1:M
    temp=double(S(:,i))-m;
    X=[X  temp];  %Almacena  todos
los temp, todas las restas en una
matriz
end

% Matriz de Covarianza
A=X';
L=A*A';%MxM,   si   fuera   alreves
seria Ncuadrado x Ncuadrado

% vv vector propio para y dd valor
propio para L=X'*X y C=X*X'
[vv     dd]=eig(L);     %evectores
evalores , igual dimencion que
covarianza L MxM, dd evalores son
0 excepto diagonal.

% Ordenar y eliminar aquellos cuyo
valor propio es cero
v=[];
d=[];

for i=1:size(vv,2)
    if(dd(i,i)>1e-4)     % Sólo si
la diagonal de los dd evalores es
>1e-4.
        v=[v vv(:,i)]; % Guarda en
v  columnas  de  vv  cuando  las
columas de dd son >1e-4.
        d=[d dd(i,i)]; % Guarda en
d  columnas  de  dd  solo  >1e-4,
ademas mantiene en una sola fila
la diagonal.
    end
end

% Vectores propios (Eigenvectors)
de la matriz de covarianza L
u=[];
for i=1:size(v,2)
    temp=sqrt(d(i));         % Raiz
cuadrada de cada valor de d
    u=[u   (X*v(:,i))./temp];  %  X
resta de cada imagen menos media,
v   eigen   vectors,   temp   raiz
cuadrada de d eigen valores.
end

% Cálculo del peso de cada rostro
de la base de datos
omega = [];
xx1=10;

for h=1:size(X,2)
    W=[];
    for i=1:size(u,2)
```

```
        t = u(:,i)';
        PesoImagen = t*X(:,h); % Evector normalizado por resta de fotos con media.
        W = [W; PesoImagen];
    end
    omega = [omega W]; % Filas u, columnas

    axes(handles.axes2);
    base = imread(strcat('C:\ProgramaTesis\BaseDeDatos\',int2str(h),'.jpg'));
    imshow(base);
title('Buscando...');
    tocc=ceil(toc);
    texto = sprintf('Tiempo transcurrido en segundos: %d',tocc);

set(handles.text12,'String',texto);
end
axes(handles.axes2);
fin = imread(strcat('C:\ProgramaTesis\Fondo\','Fondo3.jpg'));
imshow(fin);    title('Finalizando busqueda');
pause(1)

% Aplicamos filtro espacial a imágen de entrada
R=I(:,:,1);
G=I(:,:,2);
B=I(:,:,3);
w=[0 -1 0;-1 5 -1;0 -1 0];
Rf=imfilter(R,w);
Gf=imfilter(G,w);
Bf=imfilter(B,w);
I(:,:,1)=Rf;
I(:,:,2)=Gf;
I(:,:,3)=Bf;

%Detección facial aplicando Viola-Jones.
faceDetector = vision.CascadeObjectDetector;
bboxes = step(faceDetector, I);
IFaces = insertObjectAnnotation(I, 'rectangle', bboxes, 'Rostro');
[hi,ni] = size(bboxes);
if(hi>1)
    opc1 = questdlg('Se detectó más de un rostro. Por favor vuelva a intentarlo','DETECCIÓN FALLIDA','Ok','Si'); % Crea un cuadro de diálogo de pregunta
    if strcmp(opc1,'Si')
        return;
    end
end

if hi==1
recorte=imcrop(I,bboxes(1,:));    % Recorta la imágen.
imwrite(recorte, ['C:\ProgramaTesis\RostroNuevoUsuario\','1.jpg']);
work_area=imresize(recorte,[210 210]); % Cambia el tamaño de la imágen a 210x210 pixeles.
recorte2=work_area(28:187, 45:164,:); % Recorta la imágen.

% Area de trabajo
work_area2=imresize(recorte2,[120 120]); % Cambia el tamaño de la imágen a 120x120 pixeles.
img=rgb2gray(work_area2);        % Transforma la imágen a escala de grises.

%  Imágen de entrada 120x120 pixeles (Usuario a identificar).
InImage=reshape(double(img)',ifil*icol,1); % Reordena la matriz, crea una matriz (N1*N2)x1.
me=mean(InImage); % Cálculo de la media.
st=std(InImage);  % Cálculo de la desviación estandar.
NormImage=(InImage-me)*ustd/st+um;
Diferencia = NormImage-m;

% Recontrucción de la imágen
p = [];
aa=size(u,2);
for i = 1:aa
    pare = dot(NormImage,u(:,i)); % Producto punto.
    p = [p; pare];
end
ReshapedImage = m + u(:,1:aa)*p; % m es la media de la imágen, u is el eigenvector
ReshapedImage = reshape(ReshapedImage,icol,ifil);
ReshapedImage = ReshapedImage';

% Cálculo del peso de la imágen de entrada.
PesoInImage = [];
for i=1:size(u,2)
    t = u(:,i)';  % Transpuesto de la matriz eigenfaces.
    PesoInputImage = t*Diferencia;
    PesoInImage = [PesoInImage; PesoInputImage];    % Peso de la imágen de entrada.
end

%  Cálcilo de la distancia Euclidiana.
e=[];
for i=1:size(omega,2)
    q = omega(:,i);   % Matriz de pesos de las imágenes de la base de datos.
    DifPesos = PesoInImage-q;
    mag = norm(DifPesos);     % Distanci euclidiana.
    e = [e mag];   % Vector de distancias euclidianas.
```

```
end

valor = e/1000;
format long     %  Muestra  más
decimales.
[c  d]=min(valor)    %  Indica  la
distncia euclidiana mínima y su
posición en el vector.
 g=num2str(d); % Converte números
a una matriz de caracteres

if  c<=4.75  %Umbral  para  el
reconocimiento
% Usuario 1
    if d<19
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','1.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Álvaro
Vazques'),'ForegroundColor'       ,
'black');

set(handles.text19,'String',sprint
f('Edad: 24 años'));

set(handles.text20,'String',sprint
f('Domicilio: Azogues'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 2
    if d>=19 & d<37
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','2.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                Christian
Valdez'),'ForegroundColor'        ,
'black');

set(handles.text19,'String',sprint
f('Edad: 25 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cañar'));

set(handles.text14,'String',sprint
f('TIENE              ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');
    else
% Usuario 3
    if d>=37 & d<55
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','3.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Joseph
Abarca'),'ForegroundColor'        ,
'black');

set(handles.text19,'String',sprint
f('Edad: 24 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 4
    if d>=55 & d<73
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','4.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Elvis
Zaruma'),'ForegroundColor'        ,
'black');

set(handles.text19,'String',sprint
f('Edad: 24 años'));

set(handles.text20,'String',sprint
f('Domicilio: Gualaceo'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 5
    if d>=73 & d<91
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','5.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));
```

```
set(handles.text1,'String',sprintf
('Nombre:                    José
Vintimilla'),'ForegroundColor'    ,
'black');

set(handles.text19,'String',sprint
f('Edad: 35 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('TIENE              ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');


    else
% Usuario 6
    if d>=91 & d<109
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','6.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                  Eduardo
Ávila'),'ForegroundColor'         ,
'black');

set(handles.text19,'String',sprint
f('Edad: 28 años'));

set(handles.text20,'String',sprint
f('Domicilio: Azogues'));

set(handles.text14,'String',sprint
f('TIENE              ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');

    else
% Usuario 7
    if d>=109 & d<127
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','7.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                     Raúl
Deleg'),'ForegroundColor'         ,
'black');

set(handles.text19,'String',sprint
f('Edad: 27 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cañar'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 8
    if d>=127 & d<145
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','8.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Miguel
Abad'),'ForegroundColor'          ,
'black');

set(handles.text19,'String',sprint
f('Edad: 23 años'));

set(handles.text20,'String',sprint
f('Domicilio: Manabí'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 9
    if d>=145 & d<163
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','9.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final')

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Rafael
Illares'),'ForegroundColor'       ,
'black');

set(handles.text19,'String',sprint
f('Edad: 24 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 10
    if d>=163 & d<181
        axes(handles.axes2);
```

```
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','10.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Klever
Vazquez'),'ForegroundColor'       ,
'black');

set(handles.text19,'String',sprint
f('Edad: 28 años'));

set(handles.text20,'String',sprint
f('Domicilio: Gualaceo'));

set(handles.text14,'String',sprint
f('TIENE               ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');

    else
% Usuario 11
    if d>=181 & d<199
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','11.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Agustin
Yúbi'),'ForegroundColor'          ,
'black');

set(handles.text19,'String',sprint
f('Edad: 25 años'));

set(handles.text20,'String',sprint
f('Domicilio: Azogues'));

set(handles.text14,'String',sprint
f('TIENE               ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');
    else
% Usuario 12
    if d>=199 & d<217
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','12.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                Jheferson
Gualán','ForegroundColor'         ,
'black'));

set(handles.text19,'String',sprint
f('Edad: 27 años'));

set(handles.text20,'String',sprint
f('Domicilio: Tambo'));

set(handles.text14,'String',sprint
f('TIENE               ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');
    else
% Usuario 13
    if d>=217 & d<235
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','13.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Danilo
Vanegas'),'ForegroundColor'       ,
'black');

set(handles.text19,'String',sprint
f('Edad: 26 años'));

set(handles.text20,'String',sprint
f('Domicilio: Deleg'));

set(handles.text14,'String',sprint
f('TIENE               ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');
    else
% Usuario 14
    if d>=235 & d<253
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','14.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Jhanet
Lema'),'ForegroundColor'          ,
'black');

set(handles.text19,'String',sprint
f('Edad: 22 años'));

set(handles.text20,'String',sprint
f('Domicilio: Biblian'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
```

```
PENALES'),'ForegroundColor'      ,
'blue');
    else
% Usuario 15
    if d>=253 & d<271
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','15.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Bruno
Benavidez'),'ForegroundColor'     ,
'black');

set(handles.text19,'String',sprint
f('Edad: 27 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'      ,
'blue');
    else
% Usuario 16
    if d>=271 & d<289
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','16.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    María
Pérez'),'ForegroundColor'        ,
'black');

set(handles.text19,'String',sprint
f('Edad: 22 años'));

set(handles.text20,'String',sprint
f('Domicilio: Loja'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'      ,
'blue');
    else
% Usuario 17
    if d>=289 & d<307
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','17.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                  Carlos
Padilla'),'ForegroundColor'      ,
'black');

set(handles.text19,'String',sprint
f('Edad: 28 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('TIENE              ANTECEDENTES
PENALES'),'ForegroundColor'      ,
'red');
    else
% Usuario 18
    if d>=307 & d<325
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','18.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                  Santiago
Bermeo'),'ForegroundColor'       ,
'black');

set(handles.text19,'String',sprint
f('Edad: 26 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('TIENE              ANTECEDENTES
PENALES'),'ForegroundColor'      ,
'red');
    else
% Usuario 19
    if d>=325 & d<343
        axes(handles.axes2);
        UsuarioFinal             =
imread(strcat('C:\ProgramaTesis\Ro
stros\','19.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN                DEL
USUARIO'),'ForegroundColor'      ,
'black');

set(handles.text1,'String',sprintf
('Nombre: Jhon Lima'));

set(handles.text19,'String',sprint
f('Edad: 26 años'));
```

```
set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('TIENE               ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');
    else
% Usuario 20
    if d>=343 & d<361
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','20.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                      Paola
Paguay'),'ForegroundColor'        ,
'black');

set(handles.text19,'String',sprint
f('Edad: 24 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 21
    if d>=361 & d<379
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','21.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Medardo
Arévalo'),'ForegroundColor'       ,
'black');

set(handles.text19,'String',sprint
f('Edad: 26 años'));

set(handles.text20,'String',sprint
f('Domicilio: Guayaquil'));

set(handles.text14,'String',sprint
f('TIENE               ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');

    else
% Usuario 22
    if d>=379 & d<397
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','22.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                      Manuel
Iglesias'),'ForegroundColor'      ,
'black');

set(handles.text19,'String',sprint
f('Edad: 25 años'));

set(handles.text20,'String',sprint
f('Domicilio: Sigsig'));

set(handles.text14,'String',sprint
f('TIENE               ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');
    else
% Usuario 23
    if d>=397 & d<415
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','23.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                      Pedro
Siguencia'),'ForegroundColor'     ,
'black');

set(handles.text19,'String',sprint
f('Edad: 25 años'));

set(handles.text20,'String',sprint
f('Domicilio: Machala'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 24
    if d>=415 & d<433
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','24.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Abraham
```

```
Renteria'),'ForegroundColor'      ,
'black');

set(handles.text19,'String',sprint
f('Edad: 26 años'));

set(handles.text20,'String',sprint
f('Domicilio: Machala'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 25
    if d>=433 & d<451
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','25.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Elvis
Landi'),'ForegroundColor'         ,
'black');

set(handles.text19,'String',sprint
f('Edad: 25 años'));

set(handles.text20,'String',sprint
f('Domicilio: Yantzaza'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 26
    if d>=451 & d<469
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','26.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                   Isauro
Jaramillo'),'ForegroundColor'     ,
'black');

set(handles.text19,'String',sprint
f('Edad: 25 años'));

set(handles.text20,'String',sprint
f('Domicilio: Machala'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 27
    if d>=469 & d<487
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','27.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                 Alejandro
Abad'),'ForegroundColor'          ,
'black');

set(handles.text19,'String',sprint
f('Edad: 28 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cañar'));

set(handles.text14,'String',sprint
f('TIENE              ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');
    else
% Usuario 28
    if d>=487 & d<505
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','28.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Adrián
Aguilar'),'ForegroundColor'       ,
'black');

set(handles.text19,'String',sprint
f('Edad: 25 años'));

set(handles.text20,'String',sprint
f('Domicilio: Girón'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 29
    if d>=505 & d<523
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','29.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');
```

```
set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                 Christian
Ortiz'),'ForegroundColor'         ,
'black');

set(handles.text19,'String',sprint
f('Edad: 28 años'));

set(handles.text20,'String',sprint
f('Domicilio: Sevilla de Oro'));

set(handles.text14,'String',sprint
f('TIENE               ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');
    else
% Usuario 30
    if d>=523 & d<541
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','30.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Klever
Tepán'),'ForegroundColor'         ,
'black');

set(handles.text19,'String',sprint
f('Edad: 28 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('TIENE               ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'red');
    else
% Usuario 31
    if d>=541 & d<559
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','31.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Xavier
Niola'),'ForegroundColor'         ,
'black');

set(handles.text19,'String',sprint
f('Edad: 25 años'));

set(handles.text20,'String',sprint
f('Domicilio: Cuenca'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario 32
    if d>=559 & d<577
        axes(handles.axes2);
        UsuarioFinal              =
imread(strcat('C:\ProgramaTesis\Ro
stros\','32.jpg'));
        imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));

set(handles.text1,'String',sprintf
('Nombre:                    Wiliam
Sanango'),'ForegroundColor'       ,
'black');

set(handles.text19,'String',sprint
f('Edad: 26 años'));

set(handles.text20,'String',sprint
f('Domicilio: Biblian'));

set(handles.text14,'String',sprint
f('NO      TIENE      ANTECEDENTES
PENALES'),'ForegroundColor'       ,
'blue');
    else
% Usuario Nuevo1
        if d>=577 & d<595
            axes(handles.axes2);
            UsuarioFinal          =
imread(strcat('C:\ProgramaTesis\Ba
seDeDatos\',g,'.jpg'));
            imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));
            load('Nuevo1.mat')

set(handles.text1,'String',sprintf
('Nombre:
%s',x1),'ForegroundColor'         ,
'black');

set(handles.text19,'String',sprint
f('Edad: %s',x2));

set(handles.text20,'String',sprint
f('Domicilio: %s',x3));

set(handles.text14,'String',x4,'Fo
regroundColor' , 'magenta');
        else
% Usuario Nuevo2
            if d>=595 & d<613
            axes(handles.axes2);
```

```
            UsuarioFinal            =
imread(strcat('C:\ProgramaTesis\Ro
stroNuevoUsuario\','1.jpg'));
            imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN DEL USUARIO'));
            load('Nuevo2.mat')

set(handles.text1,'String',sprintf
('Nombre:
%s',x1),'ForegroundColor'         ,
'black');

set(handles.text19,'String',sprint
f('Edad: %s',x2));

set(handles.text20,'String',sprint
f('Domicilio: %s',x3));

set(handles.text14,'String',x4,'Fo
regroundColor' , 'magenta');
            else
% Usuario Nuevo3
            if d>=613 & d<=630
            axes(handles.axes2);
            UsuarioFinal            =
imread(strcat('C:\ProgramaTesis\Ro
stroNuevoUsuario\','1.jpg'));
            imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text18,'String',sprint
f('INFORMACIÓN                 DEL
USUARIO'),'ForegroundColor'       ,
'black');
            load('Nuevo3.mat')

set(handles.text1,'String',sprintf
('Nombre: %s',x1));

set(handles.text19,'String',sprint
f('Edad: %s',x2));

set(handles.text20,'String',sprint
f('Domicilio: %s',x3));

set(handles.text14,'String',x4,'Fo
regroundColor' , 'magenta');
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
```

```
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
    end
else
    axes(handles.axes2);
    UsuarioFinal                  =
imread(strcat('C:\ProgramaTesis\Fo
ndo\','Fondo2.jpg'));
    imshow(UsuarioFinal);
title('Usuario Final');

set(handles.text1,'String',sprintf
('EL         USUARIO            NO
EXISTE'),'ForegroundColor'        ,
'red');

set(handles.text19,'String',sprint
f('No se encuentra en la Base de
Datos'));
end
else
    opc2 = questdlg('No se detectó
ningun rostro. Por favor vuelva a
intentarlo','DETECCIÓN
FALLIDA','Ok','Si');  %  Crea  un
cuadro de diálogo de pregunta
    if strcmp(opc2,'Si')
        return;
    end

end
a2=toc;
tocc=ceil(toc);
texto    =    sprintf('Tiempo    de
operación   total   en   segundos:
%d',tocc);
set(handles.text12,'String',texto)
;

% --- Se ejecuta al cambiiar de
selección en popupmenu1.
function
popupmenu1_Callback(hObject,
eventdata, handles)
global vid
global IFaces
global work_area
global img
global ReshapedImage
value                             =
get(handles.popupmenu1,'Value');
switch value
        case 1
         axes(handles.axes1);
```

```
        imshow(vid),
title('Imágen de entrada');
    case 2
        axes(handles.axes1);
        imshow(IFaces),
title('Detección facial');
    case 3
        axes(handles.axes1);
        imshow(work_area);
title('Área de trabajo');
    case 4
        axes(handles.axes1);
        imshow(img);
title('Escala de grises');
    case 5
        axes(handles.axes1);
        imagesc(ReshapedImage);
colormap('gray');    title('Imágen
Reconstruida');
end

%  ---  Executes  during  object
creation,  after  setting  all
properties.
function
popupmenu1_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to popupmenu1
(see GCBO)
% eventdata   reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all CreateFcns
called

% Hint: popupmenu controls usually
have  a  white  background  on
Windows.
%       See ISPC and COMPUTER.
if            ispc            &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

function   edit2_Callback(hObject,
eventdata, handles)
% hObject    handle to text1 (see
GCBO)
% eventdata   reserved - to be
defined in a future version of
MATLAB
% handles        structure with
handles  and  user  data  (see
GUIDATA)

%   Hints:   get(hObject,'String')
returns contents of text1 as text
%
str2double(get(hObject,'String'))
returns  contents  of  text1  as  a
double

%  ---  Se  ejecuta  durante  la
creación  del  objeto,  después  de
configurar todas las propiedades.
function   text1_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to text1 (see
GCBO)
% eventdata   reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if            ispc            &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

% --- Se ejecuta cuando el usuario
intenta cerrar figure1.
function
figure1_CloseRequestFcn(hObject,
eventdata, handles)
% hObject      handle  to  figure1
(see GCBO)
% eventdata   reserved - to be
defined in a future version of
MATLAB
%  handles        structure  with
handles  and  user  data  (see
GUIDATA)

opc = questdlg('¿Desea salir del
programa?','SALIR','Si','No','No')
;
if strcmp(opc,'No')
    return;
end
delete(hObject);

%  ---  Ejecuta  al  presionar  el
botón  pushbutton11  (Dase  De
Datos).
function
pushbutton11_Callback(hObject,
eventdata, handles)
%   hObject           handle   to
pushbutton11 (see GCBO)
% eventdata   reserved - to be
defined in a future version of
MATLAB
%  handles        structure  with
handles  and  user  data  (see
GUIDATA)
close(Guide);
Guide2
```

```
% --- Ejecuta al presionar el botón pushbutton13 (Guardar Usuario).
function pushbutton13_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton13 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% guardar en formatos
global I
p=1;
ab ='C:\ProgramaTesis\BaseDeDatos'; % Dirección de la carpeta Base de Datos
B =dir( fullfile(ab, '*.jpg')); %Muestra lista de contenidos de la carpeta.
t=size(B,1);
%Número de imágenes en la base de datos.

% Se aplica un filtro espacial a cada imágen.
R=I(:,:,1);
G=I(:,:,2);
B=I(:,:,3);
w=[0 -1 0;-1 5 -1;0 -1 0];
Rf=imfilter(R,w);
Gf=imfilter(G,w);
Bf=imfilter(B,w);
I(:,:,1)=Rf;
I(:,:,2)=Gf;
I(:,:,3)=Bf;

% Detección facial utilizando Viola-Jones
faceDetector = vision.CascadeObjectDetector; % Encuentra rostros en imágenes.
bboxes = step(faceDetector, I); % Detecta rostros en imágenes.
IFaces = insertObjectAnnotation(I, 'rectangle', bboxes, 'Rostro'); % Comenta rostros detectados.
[hi,ni] = size(bboxes);
if(hi>1)
    opc1 = questdlg('Se detectó más de un rostro. Por favor vuelva a intentarlo','DETECCIÓN FALLIDA','Ok','Si'); % Crea un cuadro de diálogo de pregunta
    if strcmp(opc1,'Si')
        return;
    end
end

if hi==1
load('pul.mat','p1')
p1 = p1 + 1
save('pul.mat','p1')

texto = sprintf('Foto: %d',p1);
set(handles.text21,'String',texto);

% Area de trabajo
recorte=imcrop(I,bboxes(1,:)); % Recorta la imágen.
work_area=imresize(recorte,[210 210]); % Cambia el tamaño de la imágen a 210x210 pixeles.
recorte2=work_area(28:187, 45:164,:); % Recorta la imágen.
img=rgb2gray(recorte2); % Transforma la imágen a escala de grises.
work_area2=imresize(img,[120 120]); % Cambia el tamaño de la imágen a 120x120 pixeles.
t = t+1
b = num2str(t);
imwrite(work_area2, ['C:\ProgramaTesis\BaseDeDatos\',b,'.jpg']);
msj = questdlg('Se ha agregado a la Base de Datos correctamente', 'Guardado','Ok','Ok');

if p1>17
   p1 = 0
   msj = questdlg('Límite de fotos permitido por usuario', 'Nuevo Usuario Creado','Ok','Ok');
    save('pul.mat','p1')
end

else
    opc2 = questdlg('No se detectó ningun rostro. Por favor vuelva a intentarlo','DETECCIÓN FALLIDA','Ok','Si'); % Crea un cuadro de diálogo de pregunta
    if strcmp(opc2,'Si')
        return;
    end
end

% --- Ejecuta al presionar el botón pushbutton14 (Agregar Datos).
function pushbutton14_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton14 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
close(Guide);
Guide3

% --- Executes on button press in pushbutton15.
function pushbutton15_Callback(hObject, eventdata, handles)
```

```
%   hObject            handle   to
pushbutton15 (see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)
close(Guide);
Guide4
```

Segunda Interfaz

```
function        varargout        =
Guide2(varargin)

% Begin initialization code - DO
NOT EDIT
gui_Singleton = 1;
gui_State   =   struct('gui_Name',
mfilename, ...

'gui_Singleton',     gui_Singleton,
...

'gui_OpeningFcn',
@Guide2_OpeningFcn, ...

'gui_OutputFcn',
@Guide2_OutputFcn, ...

'gui_LayoutFcn',  [] , ...
                  'gui_Callback',
[]);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback       =
str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}]       =
gui_mainfcn(gui_State,
varargin{:});
else
    gui_mainfcn(gui_State,
varargin{:});
end
% End initialization code - DO NOT
EDIT

% --- Executes just before Guide2
is made visible.
function
Guide2_OpeningFcn(hObject,
eventdata, handles, varargin)
% This function has no output
args, see OutputFcn.
% hObject    handle to figure
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)
%   varargin         command   line
arguments to Guide2 (see VARARGIN)
axes(handles.axes4);
Fondo3                           =
imread(strcat('C:\ProgramaTesis\Fo
ndo\','Fondo1.jpg'));
imshow(Fondo3)

% Choose default command line
output for Guide2
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Guide2 wait for
user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function
are returned to the command line.
function        varargout        =
Guide2_OutputFcn(hObject,
eventdata, handles)
% varargout    cell   array   for
returning   output   args   (see
VARARGOUT);
% hObject    handle to figure
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)

% Get default command line output
from handles structure
varargout{1} = handles.output;

% --- Ejecuta al presionar el
botón      pushbutton2      (Botón
Terminar).
function
pushbutton2_Callback(hObject,
eventdata, handles)
% hObject    handle to pushbutton2
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)
close(Guide2);
Guide

% --- Se ejecuta cuando el usuario
intenta cerrar figura1.
function
figure1_CloseRequestFcn(hObject,
eventdata, handles)
% hObject      handle to figure1
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)
close(Guide2);
Guide
```

```
% --- Executes during object
creation, after setting all
properties.
function edit1_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to edit1 (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if            ispc            &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

% --- Executes during object
creation, after setting all
properties.
function text1_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to text1 (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all CreateFcns
called


% --- Ejecuta en cambio de
selección en popupmenu1 (Botón
Características).
function
popupmenu1_Callback(hObject,
eventdata, handles)
% hObject    handle to popupmenu1
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

%      Hints: contents      =
cellstr(get(hObject,'String'))
returns popupmenu1 contents as
cell array
%
contents{get(hObject,'Value')}
returns selected item from
popupmenu1
clc
a ='C:\ProgramaTesis\BaseDeDatos';
A =dir( fullfile(a, '*.jpg') );
M=size(A,1);
value                          =
get(handles.popupmenu1,'Value');
switch value
        case 1
            tic;
            for i=1:M
            img                =
imread(strcat('C:\ProgramaTesis\Ba
seDeDatos\',int2str(i),'.jpg'));
                if i<=25

axes(handles.uipanel1);

subplot(ceil(sqrt(25)),ceil(sqrt(2
5)),i);
                imshow(img);
                end
            end
        toc;
        tocc=ceil(toc);
        texto = sprintf('Imágenes
faciales: %d',M);

set(handles.text4,'String',texto);
        texto = sprintf('Tiempo de
carga en segundos: %d',tocc);

set(handles.text5,'String',texto);

        case 2
            tic
            ustd=80; um=100;
            S=[];
            for i=1:M
            img                =
imread(strcat('C:\ProgramaTesis\Ba
seDeDatos\',int2str(i),'.jpg'));
            [ifil icol]=size(img);
% Obtenemos el número de filas
(N1) y columnas (N2).

temp=reshape(img',ifil*icol,1);   %
Reordena la matriz, crea una
matriz (N1*N2)x1.
            S=[S temp];
            end

% Normalizar imágenes de la base
de datos para reducir el error
debido a las condiciones de
iluminación.
        for i=1:size(S,2)  %Valor
del número de columnas de la
matriz S
        temp=double(S(:,i));
%almacena todas las variables
numéricas como valores de punto
flotante de doble precisión que
son 8 bytes (64 bits)
        m=mean(temp);   %saca la
media de toda cada (1) columna un
solo valor
        st=std(temp);
        S(:,i)=(temp-
m)*ustd/st+um;   %(foto-media de
foto)*ustd/desviacion estandar de
foto +um
```

```
        end

% muestra imágenes normalizadas
        for i=1:M

str=strcat(int2str(i),'.jpg');

img=reshape(S(:,i),icol,ifil);
        img=img';
            if i<=25

axes(handles.uipanel1);

subplot(ceil(sqrt(25)),ceil(sqrt(2
5)),i);
                imshow(img);
            end
        end
        toc;
        tocc=ceil(toc);
        texto  =  sprintf('Imágenes
Normalizadas: %d',M);

set(handles.text4,'String',texto);
        texto = sprintf('Tiempo de
operación en segundos: %d',tocc);

set(handles.text5,'String',texto);

        case 3
            tic
            S=[];
            ustd=80; um=100;
            for i=1:M
                img                =
imread(strcat('C:\ProgramaTesis\Ba
seDeDatos\',int2str(i),'.jpg'));
                [ifil
icol]=size(img);

temp=reshape(img',ifil*icol,1);
                S=[S temp];
            end

        for  i=1:size(S,2)  %Valor
del número de columnas de la
matriz S
            temp=double(S(:,i));
%almacena todas las variables
numéricas como valores de punto
flotante de doble precisión que
son 8 bytes (64 bits)
            m=mean(temp); %saca la
media de toda cada (1) columna un
solo valor
            st=std(temp);
            S(:,i)=(temp-
m)*ustd/st+um;   %(foto-media   de
foto)*ustd/desviacion estandar de
foto +um
        end

% muestra imágenes normalizadas
        for i=1:M

str=strcat(int2str(i),'.jpg');

img=reshape(S(:,i),icol,ifil);
            img=img';
        end

%Cálculo de la media de las
imagenes;
        m=mean(S,2);     % obtiene
la media de cada fila en lugar de
cada columna
        mn=uint8(m);         %  se
convierte en un entero de 8 bits
sin signo. Los valores van de 0 a
255.
        img=reshape(mn,icol,ifil);
% toma el vector N1*N2x1 y crea
una matriz N2xN1
        img=img';       % crea una
matriz    N1xN2    mediante    la
transposición de la imagen.
        axes(handles.uipanel1);
        imshow(img);
        toc;
        tocc=ceil(toc);
        texto  =  sprintf('Grafica
la media de la base de datos');

set(handles.text4,'String',texto);
        texto = sprintf('Tiempo de
operación en segundos: %d',tocc);

set(handles.text5,'String',texto);

        case 4
            tic
            ustd=80; um=100;
            S=[];
            for i=1:M
            img                    =
imread(strcat('C:\ProgramaTesis\Ba
seDeDatos\',int2str(i),'.jpg'));
            [ifil  icol]=size(img);
%Obtenemos el número de filas (N1)
y columnas (N2)

temp=reshape(img',ifil*icol,1);
%Reordena la matriz, crea una
matriz (N1*N2)x1
        end

        for  i=1:size(S,2)  %Valor
del número de columnas de la
matriz S
            temp=double(S(:,i));
%almacena todas las variables
numéricas como valores de punto
flotante de doble precisión que
son 8 bytes (64 bits)
            m=mean(temp); %saca la
media de toda cada (1) columna un
solo valor
            st=std(temp);
            S(:,i)=(temp-
m)*ustd/st+um;   %(foto-media   de
foto)*ustd/desviacion estandar de
foto +um
        end

% muestra imágenes normalizadas
        for i=1:M
```

```
str=strcat(int2str(i),'.jpg');

img=reshape(S(:,i),icol,ifil);
            img=img';
        end

%Cálculo de la media de las imagenes;
        m=mean(S,2);     % obtiene la media de cada fila en lugar de cada columna
        mn=uint8(m);         % se convierte en un entero de 8 bits sin signo. Los valores van de 0 a 255.
        img=reshape(mn,icol,ifil);
        img=img';

%    Cambio    de    imagen    por manipulación
        X=[];   % A matrix
        for i=1:M
            temp=double(S(:,i))-m;
            X=[X  temp];  %Almacena todos los temp, todas las restas en una matriz
        end

%    Cambio    de    imagen    por manipulación.
        A=X';
        L=A*A';%MxM,    si    fuera alreves    seria    Ncuadrado    x Ncuadrado
        [vv dd]=eig(L); %evectores evalores  ,  igual  dimencion  que covarianza  L  MxM,  dd  es  decir evalores son 0 excepto diagonal

% Ordenar y eliminar aquellos cuyo valor propio es cero
        v=[];
        d=[];
        for i=1:size(vv,2)
            if(dd(i,i)>1e-4)  %solo si la diagonal de los dd evalores es >1e-4
                v=[v vv(:,i)];%guarda en v columnas de vv en donde las columas de dd sean >1e-4
                d=[d dd(i,i)];%guarda en d columnas de dd solo >1e-4, ademas mantiene en una sola fila la diagonal
            end
        end

% Vectores propios (Eigenvectors) de la matriz de covarianza.
        u=[];
        for i=1:size(v,2)
            temp=sqrt(d(i));%raiz cuadrada de cada valor de d.
            u=[u (X*v(:,i))./temp]; %X es la resta de cada imagen menos media.
                                    %    v es eigen vectors.
                                    %    temp raiz cuadrada de d eigen valores
        end

% Muestra Eigenfaces;
        wil=size(u,2);
        for i=1:size(u,2)

img=reshape(u(:,i),icol,ifil);%toma cada columna y reordena, cada columna es una foto de eigenvector
            img=img';
            img=histeq(img,255);
                if i<=25

axes(handles.uipanel1);

subplot(ceil(sqrt(25)),ceil(sqrt(25)),i);
                    imshow(img);
                end
        end
        toc;
        tocc=ceil(toc);
        texto                    = sprintf('Eigenfaces      obtenidas: %d',wil);

set(handles.text4,'String',texto);
        texto = sprintf('Tiempo de operación en segundos; %d',tocc);

set(handles.text5,'String',texto);
        end

%  ---  Executes  during  object creation,  after  setting  all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject     handle to popupmenu1 (see GCBO)
% eventdata    reserved  -  to  be defined  in  a  future  version  of MATLAB
% handles     empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have  a  white  background  on Windows.
%       See ISPC and COMPUTER.
if             ispc             && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes during object creation, after setting all properties.
function text6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

Tercera Interfaz

```
function varargout = Guide3(varargin)

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...

'gui_Singleton', gui_Singleton, ...

'gui_OpeningFcn', @Guide3_OpeningFcn, ...

'gui_OutputFcn', @Guide3_OutputFcn, ...

'gui_LayoutFcn', [] , ...
                 'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Guide3 is made visible.
function Guide3_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Guide3 (see VARARGIN)
    axes(handles.axes2);
    Fondo3 = imread(strcat('C:\ProgramaTesis\Fondo\','Fondo1.jpg'));
    imshow(Fondo3)

    handles.output = hObject;
    guidata(hObject, handles);

function varargout = click_veces_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
% Choose default command line output for Guide3
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Guide3 wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Guide3_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Ejecuta al presionar el botón1 (Guardar Datos).
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
load('count.mat','y1')
count = y1
y1 = count + 1
save('count.mat','y1')
if y1==1

x1=get(handles.edit1,'String');

x2=get(handles.edit2,'String');
```

```
x3=get(handles.edit3,'String');

x4=get(handles.edit4,'String');

save('Nuevo1.mat','x1','x2','x3','
x4')
    msj = questdlg('Los Datos se
han    guardado    correctamente',
'Guardado','Ok','Ok');

else
    if y1==2

x1=get(handles.edit1,'String');

x2=get(handles.edit2,'String');

x3=get(handles.edit3,'String');

x4=get(handles.edit4,'String');

save('Nuevo2.mat','x1','x2','x3','
x4')
    msj = questdlg('Los Datos se
han    guardado    correctamente',
'Guardado','Ok','Ok');
else
    if y1==3

x1=get(handles.edit1,'String');

x2=get(handles.edit2,'String');

x3=get(handles.edit3,'String');

x4=get(handles.edit4,'String');

save('Nuevo3.mat','x1','x2','x3','
x4')
    msj = questdlg('Los Datos se
han    guardado    correctamente',
'Guardado','Ok','Ok');
end
end
end

function    edit1_Callback(hObject,
eventdata, handles)
% hObject     handle to edit1 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)

%   Hints:   get(hObject,'String')
returns contents of edit1 as text
%
str2double(get(hObject,'String'))
returns contents of edit1 as a
double

% --- Executes during object
creation,    after    setting    all
properties.
function   edit1_CreateFcn(hObject,
eventdata, handles)
% hObject     handle to edit1 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles     empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if             ispc             &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

% --- Ejecuta al presionar el
botón      pushbutton2      (Botón
Terminar).
function
pushbutton2_Callback(hObject,
eventdata, handles)
% hObject    handle to pushbutton2
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)
close(Guide3);
Guide

% --- Executes when user attempts
to close figure1.
function
figure1_CloseRequestFcn(hObject,
eventdata, handles)
% hObject      handle to figure1
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)
close(Guide3);
Guide

function   edit2_Callback(hObject,
eventdata, handles)
% hObject     handle to edit2 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
```

```
% handles structure with
handles and user data (see
GUIDATA)

% Hints: get(hObject,'String')
returns contents of edit2 as text
%
str2double(get(hObject,'String'))
returns contents of edit2 as a
double

% --- Executes during object
creation, after setting all
properties.
function edit2_CreateFcn(hObject,
eventdata, handles)
% hObject handle to edit2 (see
GCBO)
% eventdata reserved - to be
defined in a future version of
MATLAB
% handles empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

function edit3_Callback(hObject,
eventdata, handles)
% hObject handle to edit3 (see
GCBO)
% eventdata reserved - to be
defined in a future version of
MATLAB
% handles structure with
handles and user data (see
GUIDATA)

% Hints: get(hObject,'String')
returns contents of edit3 as text
%
str2double(get(hObject,'String'))
returns contents of edit3 as a
double

% --- Executes during object
creation, after setting all
properties.
function edit3_CreateFcn(hObject,
eventdata, handles)
% hObject handle to edit3 (see
GCBO)
% eventdata reserved - to be
defined in a future version of
MATLAB
% handles empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

function edit4_Callback(hObject,
eventdata, handles)
% hObject handle to edit4 (see
GCBO)
% eventdata reserved - to be
defined in a future version of
MATLAB
% handles structure with
handles and user data (see
GUIDATA)

% Hints: get(hObject,'String')
returns contents of edit4 as text
%
str2double(get(hObject,'String'))
returns contents of edit4 as a
double

% --- Executes during object
creation, after setting all
properties.
function edit4_CreateFcn(hObject,
eventdata, handles)
% hObject handle to edit4 (see
GCBO)
% eventdata reserved - to be
defined in a future version of
MATLAB
% handles empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end
```

Cuarta Interfaz

```
function varargout =
Guide4(varargin)
```

```
% Begin initialization code - DO
NOT EDIT
gui_Singleton = 1;
gui_State   =   struct('gui_Name',
mfilename, ...

'gui_Singleton',     gui_Singleton,
...

'gui_OpeningFcn',
@Guide4_OpeningFcn, ...

'gui_OutputFcn',
@Guide4_OutputFcn, ...

'gui_LayoutFcn',  [] , ...
                  'gui_Callback',
[]);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback       =
str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}]       =
gui_mainfcn(gui_State,
varargin{:});
else
    gui_mainfcn(gui_State,
varargin{:});
end
% End initialization code - DO NOT
EDIT

% --- Executes just before Guide4
is made visible.
function
Guide4_OpeningFcn(hObject,
eventdata, handles, varargin)
% This function has no output
args, see OutputFcn.
% hObject    handle to figure
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)
% varargin         command line
arguments to Guide4 (see VARARGIN)
    axes(handles.axes1);
    Fondo3                       =
imread(strcat('C:\ProgramaTesis\Fo
ndo\','Fondo1.jpg'));
    imshow(Fondo3)
    handles.output = hObject;
    guidata(hObject, handles);

function        varargout        =
click_veces_OutputFcn(hObject,
eventdata, handles)
varargout{1} = handles.output;

% Choose default command line
output for Guide4
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Guide4 wait for
user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function
are returned to the command line.
function        varargout        =
Guide4_OutputFcn(hObject,
eventdata, handles)
% varargout     cell array for
returning    output    args    (see
VARARGOUT);
% hObject    handle to figure
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)

% Get default command line output
from handles structure
varargout{1} = handles.output;

% --- Executes on button press in
pushbutton1.          (Seleccionar
Usuario).
function
pushbutton1_Callback(hObject,
eventdata, handles)
% hObject    handle to pushbutton1
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles and user data (see
GUIDATA)
global L
global PathFile
[PathFile]=uigetdir('','Selecciona
r Carpeta'); % Carga los archivos
de la carpeta
A     =dir(     fullfile(PathFile,
'*.jpg') );
L=size(A,1);
for i=1:L
RGB                              =
imread(strcat(PathFile,'\',num2str
(i),'.jpg'));
end
texto = sprintf('Imágenes en la
carpeta: %d',L);
set(handles.text3,'String',texto);

% --- Ejecuta al presionar el
botón     pushbutton2     (Agregar
Usuario).
function
pushbutton2_Callback(hObject,
eventdata, handles)
% hObject    handle to pushbutton2
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
```

```
% handles structure with handles and user data (see GUIDATA)
global L
global PathFile
ab ='C:\ProgramaTesis\BaseDeDatos'; % Dirección de la carpeta Base de Datos
B =dir( fullfile(ab, '*.jpg')); %Muestra lista de contenidos de la carpeta.
t=size(B,1)
%Número de imágenes en la base de datos.
for i=1:L
RGB = imread(strcat(PathFile,'\',num2str(i),'.jpg'));
I=RGB(50:694, 500:1400,:); %Recorta imágen de alta resolución. antes 50:694 Antes con doble recorte este era primero
R=I(:,:,1);
G=I(:,:,2);
B=I(:,:,3);
w=[0 -1 0;-1 5 -1;0 -1 0];
Rf=imfilter(R,w);
Gf=imfilter(G,w);
Bf=imfilter(B,w);
I(:,:,1)=Rf;
I(:,:,2)=Gf;
I(:,:,3)=Bf;
faceDetector = vision.CascadeObjectDetector; % Encuentra rostros en imágenes.
bboxes = step(faceDetector, I); % Detecta rostros en imágenes.
IFaces = insertObjectAnnotation(I, 'rectangle', bboxes, 'Rostro'); % Comenta rostros detectados.
[hi,ni] = size(bboxes);
if(hi>1)
    opc1 = questdlg('Se detectó más de un rostro. Por favor vuelva a intentarlo','DETECCIÓN FALLIDA','Ok','Si'); % Crea un cuadro de diálogo de pregunta
    if strcmp(opc1,'Si')
        return;
    end
end
if hi==1
% Area de trabajo
recorte=imcrop(I,bboxes(1,:)); % Recorta la imágen.
work_area=imresize(recorte,[210 210]); % Cambia el tamaño de la imágen a 210x210 pixeles.
recorte2=work_area(28:187, 45:164,:);
work_area2=imresize(recorte2,[120 120]); % Cambia el tamaño de la imágen a 120x120 pixeles.
img=rgb2gray(work_area2); % Transforma la imágen a escala de grises.
t = t+1;
r = num2str(t);
imwrite(img, ['C:\ProgramaTesis\BaseDeDatos\',r,'.jpg']);
end
texto = sprintf('Imágenes faciales agregadas: %d',L);
set(handles.text3,'String',texto);
end
msj = questdlg('Se han agregado a la Base de Datos', 'Guardado','Ok','Ok');

if hi==0
    opc2 = questdlg('No se detectó ningun rostro. Por favor vuelva a intentarlo','DETECCIÓN FALLIDA','Ok','Si'); % Crea un cuadro de diálogo de pregunta
    if strcmp(opc2,'Si')
        return;
    end
end
% --- Ejecuta al presionar el botón pushbutton3 (Guardar Datos).
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton3 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
load('count.mat','y1')
count = y1
y1 = count + 1
save('count.mat','y1')
if y1==1

x1=get(handles.edit1,'String');

x2=get(handles.edit2,'String');

x3=get(handles.edit3,'String');

x4=get(handles.edit4,'String');

save('Nuevo1.mat','x1','x2','x3','x4')
    msj = questdlg('Los Datos se han guardado correctamente', 'Guardado','Ok','Ok');
else
    if y1==2

x1=get(handles.edit1,'String');

x2=get(handles.edit2,'String');

x3=get(handles.edit3,'String');

x4=get(handles.edit4,'String');

save('Nuevo2.mat','x1','x2','x3','x4')
```

```
    msj = questdlg('Se Datos se
han    guardado    correctamente',
'Guardado','Ok','Ok');
else
    if y1==3

x1=get(handles.edit1,'String');

x2=get(handles.edit2,'String');

x3=get(handles.edit3,'String');

x4=get(handles.edit4,'String');

save('Nuevo3.mat','x1','x2','x3','
x4')
    msj = questdlg('Se Datos se
han    guardado    correctamente',
'Guardado','Ok','Ok');
end
end
end

function    edit1_Callback(hObject,
eventdata, handles)
% hObject     handle to edit1 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles   and   user   data   (see
GUIDATA)

%    Hints:    get(hObject,'String')
returns contents of edit1 as text
%
str2double(get(hObject,'String'))
returns contents of edit1 as a
double

%   ---   Executes   during   object
creation,    after    setting    all
properties.
function   edit1_CreateFcn(hObject,
eventdata, handles)
% hObject     handle to edit1 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles     empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if             ispc             &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

function    edit2_Callback(hObject,
eventdata, handles)
% hObject     handle to edit2 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles   and   user   data   (see
GUIDATA)

%    Hints:    get(hObject,'String')
returns contents of edit2 as text
%
str2double(get(hObject,'String'))
returns contents of edit2 as a
double

%   ---   Executes   during   object
creation,    after    setting    all
properties.
function   edit2_CreateFcn(hObject,
eventdata, handles)
% hObject     handle to edit2 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles     empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if             ispc             &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

function    edit3_Callback(hObject,
eventdata, handles)
% hObject     handle to edit3 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles   and   user   data   (see
GUIDATA)

%    Hints:    get(hObject,'String')
returns contents of edit3 as text
%
str2double(get(hObject,'String'))
returns contents of edit3 as a
double

%   ---   Executes   during   object
creation,    after    setting    all
properties.
function   edit3_CreateFcn(hObject,
eventdata, handles)
```

```
% hObject    handle to edit3 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles     empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if             ispc             &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

function    edit4_Callback(hObject,
eventdata, handles)
% hObject     handle to edit4 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles   and   user   data   (see
GUIDATA)

%   Hints:   get(hObject,'String')
returns contents of edit4 as text
%
str2double(get(hObject,'String'))
returns contents of edit4 as a
double

%   ---   Executes   during   object
creation,   after   setting   all
properties.
function   edit4_CreateFcn(hObject,
eventdata, handles)
% hObject     handle to edit4 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles     empty - handles not
created until after all CreateFcns
called

% Hint: edit controls usually have
a white background on Windows.
%       See ISPC and COMPUTER.
if             ispc             &&
isequal(get(hObject,'BackgroundCol
or'),
get(0,'defaultUicontrolBackgroundC
olor'))

set(hObject,'BackgroundColor','whi
te');
end

% --- Ejecuta al presionar el
botón pushbutton4 (Terminar).
function
pushbutton4_Callback(hObject,
eventdata, handles)
% hObject    handle to pushbutton4
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles   and   user   data   (see
GUIDATA)
close(Guide4);
Guide

% --- Executes when user attempts
to close figure1.
function
figure1_CloseRequestFcn(hObject,
eventdata, handles)
% hObject      handle to figure1
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles         structure with
handles   and   user   data   (see
GUIDATA)
close(Guide4);
Guide
```