UNIVERSIDAD NACIONAL DE LOJA

ÁREA DE LA ENERGÍA, LAS INDUSTRIAS Y LOS RECURSOS NATURALES NO RENOVABLES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

"DISEÑO DE UN SISTEMA DE AUTENTICACIÓN BIOMÉTRICA BASADO EN RECONOCIMIENTO FACIAL"

TESIS DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES.

AUTORA:

SANDRA ELIZABETH GARROCHAMBA SÁNCHEZ

DIRECTOR:

ING. MARCELO FERNANDO VALDIVIEZO CONDOLO

JUNIO DE 2014

LOJA - ECUADOR

CERTIFICACIÓN

Señor Ingeniero
Marcelo Fernando Valdiviezo Condolo
DIRECTOR DEL TRABAJO DE TESIS

CERTIFICA:

Haber dirigido, asesorado, revisado y corregido el presente trabajo de tesis de grado, en su proceso de investigación, cuyo tema versa: **"DISEÑO DE UN SISTEMA DE AUTENTICACIÓN BIOMÉTRICA BASADO EN RECONOCIMIENTO FACIAL"**, previo a la obtención del título de **Ingeniero en Electrónica y Telecomunicaciones**, realizado por la señorita egresada: Sandra Elizabeth Garrochamba Sánchez, la misma que cumple con la reglamentación y políticas de investigación, por lo que autorizo su presentación y posterior sustentación y defensa.

Loja, Junio de 2014.

Ing. Marcelo Fernando Valdiviezo Condolo
DIRECTOR DEL TRABAJO DE TESIS

AUTORÍA

Yo **Sandra Elizabeth Garrochamba Sánchez**, declaro ser la autora del presente trabajo de tesis y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos o acciones legales por el contenido de la misma.

Adicionalmente acepto y autorizo a la Universidad Nacional de Loja, la publicación de mi tesis en el Repositorio Institucional - Biblioteca Virtual.

Autora: Sandra Elizabeth Garrochamba Sánchez.

Firma: 

Cédula: 1900743707

Fecha: Junio de 2014

CARTA DE AUTORIZACIÓN DE TESIS POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO.

Yo **Sandra Elizabeth Garrochamba Sánchez,** declaro ser autora de la tesis titulada: **"DISEÑO DE UN SISTEMA DE AUTENTICACIÓN BIOMÉTRICA BASADO EN RECONOCIMIENTO FACIAL",** como requisito para optar al grado de: **INGENIERA EN ELECTRÓNICA Y TELECOMUNICACIONES**; autorizo al Sistema Bibliotecario de la Universidad Nacional de Loja para que con fines académicos, muestre al mundo la producción, intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Digital Institucional.

Los usuarios pueden consultar el contenido de este trabajo en el RDI, en las redes de información del país y del exterior, con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja no se responsabiliza por el plagio o copia de la tesis que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja, 30 días del mes de Julio del dos mil catorce.

Firma: 

Autor: Sandra Elizabeth Garrochamba Sánchez

Cédula: 1900743707

Dirección: Loja – El Valle – Turupamba

Correo Electrónico: sandra007garro@gmail.com

Teléfono: 254821 **Celular:** 0988051802

DATOS COMPLEMENTARIOS

Director de tesis: Ing. Marcelo Fernando Valdiviezo Condolo

Tribunal de grado: Ing. Diego Vinicio Orellana Villavicencio Mg. Sc

Ing. Juan Pablo Cabrera Samaniego Mg. Sc

Ing. Julio César Guamán Segarra Mg. Sc

DEDICATORIA

A Dios por darme fuerza para cumplir con esta meta, a mis padres, hermana, hermanos y todos mis familiares que me dieron su apoyo para el desarrollo del presente trabajo.

AGRADECIMIENTO

Quiero dejar constancia de mi agradecimiento a mis padres por su apoyo incondicional. Sin sus ánimos, consejos y confianza esto no sería posible.

Un agradecimiento sincero y profundo a la Universidad Nacional de Loja, por concederme la oportunidad de ser estudiante de la carrea de Ingeniería en Electrónica y Telecomunicaciones, así como a mi tutor Ing. Marcelo Valdiviezo por estar presto a ayudarme con la revisión y dirección de tesis.

ÍNDICE

CERTIFICACIÓN i
AUTORÍA ii
CARTA DE AUTORIZACIÓN DE TESIS POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO. iii
DEDICATORIA iv
AGRADECIMIENTO v
ÍNDICE vi
ÍNDICE DE FIGURAS x
ÍNDICE DE TABLAS xi
TEMA: xii
RESUMEN xiii
SUMMARY xiv
OBJETIVOS xv
INTRODUCCIÓN xvi
METODOLOGÍA xviii
CAPÍTULO 1 1
1. MARCO TEÓRICO 1
1.1. CONCEPTOS BÁSICOS DE IMAGEN DIGITAL 1
1.1.1. PIXEL 1
1.1.2. IMAGEN DIGITAL 1
1.1.3. TIPO DE IMÁGENES DIGÍTALES 2
1.1.3.1. MAPA DE BITS 2
1.1.3.2. IMÁGENES VECTORIALES 2
1.1.4. RESOLUCIÓN DE IMAGEN 3
1.1.5. PROFUNDIDAD DE COLOR 3
1.1.6. MODOS DE COLOR 3
1.1.7. FORMATOS DE IMAGEN 6
1.1.7.1. BMP (BITMAP = MAPA DE BITS) 6
1.1.7.2. GIF (GRAPHICS INTERCHANGE FORMAT = FORMATO DE INTERCAMBIO GRÁFICO) 6

1.1.7.3. JPG-JPEG (JOINT PHOTOGRAPHIC EXPERTS GROUP = GRUPO DE EXPERTOS FOTOGRÁFICOS UNIDOS) 7
1.1.7.4. TIF-TIFF (TAGGED IMAGE FILE FORMAT = FORMATO DE ARCHIVO DE IMAGEN ETIQUETADA) 7
1.1.7.5. PNG (PORTABLE NETWORK GRAPHIC = GRÁFICO PORTABLE PARA LA RED) 7
1.2.3. TIPOS DE FILTROS 9
1.2.3.1. FILTROS DE PASO BAJO 9
1.2.3.2. FILTROS DE PASO ALTO 10
1.2.3.3. FILTROS PARA LA DETECCIÓN DE BORDES 10
1.2.3.4. FILTRADO EN FRECUENCIA 11
1.2.3.5. FILTROS MORFOLÓGICOS 11
1.2.3.6. FILTROS DE TEXTURA 12
1.3. LENGUAJE DE PROGRAMACIÓN JAVA 13
1.3.1. QUÉ ES JAVA 13
1.3.2. CUADRO COMPARATIVO DE JAVA CON OTROS LENGUAJES DE PROGRAMACIÓN 13
1.3.3. MOTIVOS PARA PROGRAMAR EN JAVA 14
1.4. BIOMETRÍA 16
1.4.1. DEFINICIÓN DE BIOMETRÍA 16
1.4.2. HISTORIA DE LA DE LA BIOMETRÍA 16
1.4.3. CLASIFICACIÓN DE LA BIOMÉTRIA 17
1.4.4. REQUISITOS BÁSICOS DE LA BIOMETRÍA 17
1.4.5. FUNCIONAMIENTO Y RENDIMIENTO DE LA BIOMETRIA 18
1.4.6. TECNOLOGÍAS BIOMÉTRICAS MÁS USADAS 18
1.4.7. PARÁMETROS BIOMÉTRIOS 20
1.4.8. COMPARACIÓN DE SISTEMAS BIOMÉTRICOS 21
1.5. DETECCIÓN DE ROSTROS 22
1.5.1. DEFINICIÓN 22
1.5.2. MÉTODOS DE DETECCIÓN 22
1.5.3. PROBLEMAS COMUNES EN LA DETECCIÓN 22
1.5.4. ROTACIONES 23
1.5.5. TÉCNICAS BASADAS EN RASGOS 23

1.5.5.1. ANÁLISIS DE BAJO NIVEL 24
1.5.5.2. ANÁLISIS DE RASGOS 24
1.5.5.3. ANÁLISIS DE FORMAS ACTIVAS 25
1.5.6. ALGORITMO DE DETECCIÓN DE ROSTROS VIOLA-JONES 26
1.5.6.1. BASES DEL ALGORITMO 26
1.5.6.2. ENTRENAMIENTO DE LA CASCADA DE CLASIFICADORES 30
1.5.6.3. DETECCIÓN DE ROSTROS CON OPENCV 31
1.6. RECONOCIMIENTO FACIAL 32
1.6.1. DEFINICIÓN DE RECONOCIMIENTO FACIAL 32
1.6.2. MÉTODOS HOLÍSTICOS 33
1.6.2.1. ANÁLISIS DE COMPONENTE PRINCIPAL: PCA (PRINCIPAL COMPONENT ANALYSIS) 33
1.6.2.2. ANÁLISIS DE LA COMPONENTE INDEPENDIENTE: ICA (INDEPENDENT COMPONENT ANALYSIS) 34
1.6.2.3. ANÁLISIS DE LA LÍNEA DISCRIMINANTE: LDA (LINEAR DISCRIMINANT ANALYSIS) 35
1.6.2.4. MÉTODOS BASADAS EN KERNELS 35
1.6.2.5. PERSECUCIÓN EVOLUTIVA: EP(EVOLUTIONARY PURSUIT) 36
1.6.2.6. MÁQUINA DE VECTOR DE SUPORTE: SVM (SUPPORT VECTOR MACHINE) 36
1.6.3. MÉTODOS BASADOS EN CARACTERÍSTICAS LOCALES 36
1.6.3.1. CORRESPONDENCIA ENTRE AGRUPACIONES DE GRAFOS ELÁSTICOS: EBGM (ELASTIC BUNCH GRAPH MATCHING) 36
1.6.3.2. PATRONES BINARIOS LOCALES: LBP (LOCAL BINARY PATTERN) 37
1.6.3.3. MODELO DE APARIENCIA ACTIVA: AAM (ACTIVE APPEARANCE MODELS) 37
1.6.3.4. MODELO OCULTO DE MARKOV: HMM (HIDDEN MARKOV MODELS) 38
1.6.3.5. MÉTODOS 3D 38
1.6.4. EIGENFACES 39
1.6.4.1. RESOLUCIÓN DE LAS IMÁGENES 40
1.6.4.2. CÁLCULO DE EIGENFACES 41

1.6.4.3. CÓMO UTILIZAR LOS EIGENFACES PARA RECONOCIMIENTO DE ROSTROS 42
CAPÍTULO 2 44
2. DISEÑO Y DESARROLLO DEL SISTEMA 44
2.1. ARQUITECTURA DEL SISTEMA 44
2.1.1. PREPROCESADO DE LA IMAGEN 45
2.1.1.1. PREPROCESADO PARA LA DETECCIÓN 45
2.1.1.2. PREPROCESADO PARA EL RECONOCIMIENTO 46
2.1.2. DETECCIÓN DEL ROSTRO 46
2.1.2.1. IMÁGENES DE ENTRADA 48
2.1.2.2. LOCALIZACIÓN FACIAL 48
2.1.3. RECONOCIMIENTO 48
2.2. BASE DE DATOS 52
2.2.1. BASE DE DATOS DE IMÁGENES 52
2.2.2. BASE DE DATOS DE REGISTROS DEL SISTEMA 52
2.3. DIAGRAMAS DE LA METODOLOGÍA DE DESARROLLO XP 53
2.3.1. DIAGRAMA DE DOMINIO 53
2.3.2. DIAGRAMA DE CLASES POR PAQUETES 53
2.3.3. DIAGRAMA DE COMPONENTES 57
2.3.4. MODELAMIENTO DE LA ARQUITECTURA 57
2.4. HERRAMIENTAS UTILIZADAS 58
CAPITULO 3 59
3. PRUEBAS Y RESULTADOS 59
3.1. INICIALIZACIÓN DE LA BASE DE DATOS 59
3.2. FASE DE DETECCIÓN 59
3.3. FASE DE RECONOCIMIENTO 62
3.3.1. TIEMPO DE RECONOCIMIENTO 63
3.4. RENDIMIENTO 63
3.5. PARÁMETROS BIOMÉTRICOS 64
3.6. CONSUMO DEL PROCESADOR 65
3.8. REQUERIMIENTOS DEL SISTEMA 66
CAPÍTULO 4 67

4. ANÁLISIS TÉCNICO Y ECONÓMICO DEL SISTEMA 67
CAPÍTULO 5 70
5. CONCLUSIONES Y RECOMENDACIONES 70
5.1. CONCLUSIONES 70
5.2. RECOMENDACIONES 71
BIBLIOGRAFÍA: 73
ANEXOS 76
ANEXO A. CÓDIGO DE LA APLICACIÓN 77
ANEXO B. IMÁGENES DE LA IMPLEMENTACIÓN 84
ANEXO C. MANUAL DE USUARIO 87
ANEXO D. ANTEPROYECTO 88

ÍNDICE DE FIGURAS

Figura 1. Pixel en una imagen. 1
Figura 2. Imagen de mapa de bits y vectorial. 2
Figura 3. Modo monocromático. 4
Figura 4. Imagen a escala de grises. 4
Figura 5. Imagen de modo de color indexado. 5
Figura 6. Imagen modo RGB. 5
Figura 7. Modo HSB. 6
Figura 8. Formatos de imagen. 8
Figura 9. Histograma de una imagen 8
Figura 10. a) Roberts, b) Sobel, c) Laplaciano. 11
Figura 11. El código Java es Compilado y luego Interpretado 15
Figura 12. Un simple programa escrito en Java. 15
Figura 13. Proceso de un sistema biométrico por medio de la huella dactilar. 18
Figura 14. Proceso de un sistema biométrico por medio del rostro. 19
Figura 15. Proceso de un sistema biométrico por medio de la voz. 20
Figura 16. Rasgos de clasificación de 2, 3, y 4 rectángulos utilizados para la detección de rostros. 27
Figura 17. Imagen de dos rasgos de clasificación y su correspondencia con partes de un rostro. 27
Figura 18. Cálculo de la imagen integral. a) Imagen origina. b) Imagen integral. c) Valor del rectángulo utilizando la imagen integral. 28
Figura 19. Cálculo del valor de un rasgo y del vector W de pesos del rasgo 29
Figura 20. Cascada de clasificadores. 30
Figura 21. Grafo de puntos principales 37
Figura 22. Diagrama de bloques del sistema 44
Figura 23. Diagrama general del sistema de reconocimiento facial. 45

Figura 24. Proceso de detección 47
Figura 25. Proceso de reconocimiento 49
Figura 26. Representación de una imagen matriz como vector..... 49
Figura 27. Diagrama de dominio del sistema..... 53
Figura 28. Dominio de clases.. 54
Figura 29. Vista del sistema 55
Figura 30. Diagrama del controlador del sistema..... 56
Figura 31. Objeto de Acceso a Datos del sistema de reconocimiento facial..... 56
Figura 32. Diagrama de paquetes del sistema. 57
Figura 33. Modelamiento de la arquitectura del sistema..... 57
Figura 34. Rostro detectado por el sistema. 60
Figura 35. Dos rostros detectados por el sistema. 60
Figura 36. Algunas imágenes de la base de datos 62
Figura 37. Consumo del procesador..... 65
Figura 38. Logo de la aplicación.. 84
Figura 39. Pruebas de funcionamiento del sistema. 84
Figura 40. Pruebas de reconocimiento facial. 84
Figura 41. Sistema en funcionamiento 85
Figura 42. Empleados en MySQL. 85
Figura 43. Registros de entrada-salida de un usuario..... 85
Figura 44. Registro de ingreso-salida listo para ser impreso..... 86

ÍNDICE DE TABLAS

Tabla 1. Profundidad del color. 3
Tabla 2. Comparativo de java con otros lenguajes de programación. 13
Tabla 3. Comparación de sistemas biométricos. 21
Tabla 4. Eigenfaces de una imagen. 51
Tabla 5. Eficiencia de la detección del sistema usando una cámara de 1.0 MP. 60
Tabla 6. Eficiencia de la detección del sistema usando una cámara de 2.0 MP. 61
Tabla 7. Eficiencia del sistema en condiciones extremas. 61
Tabla 8. Eficiencia del reconocimiento facial en condiciones ideales. 63
Tabla 9. Eficiencia de reconocimiento facial en condiciones extremas. 63
Tabla 10. Verdaderos positivos en el reconocimiento. 64
Tabla 11. Verdaderos negativos en el reconocimiento facial. 64
Tabla 12. Falsos Positivos en el reconocimiento. 64
Tabla 13. Falsos Negativos en el reconocimiento. 64
Tabla 14. Consumo de recursos del computador. 65
Tabla 15. Recursos Humanos 67
Tabla 16. Recursos Técnicos 68
Tabla 17. Recursos Materiales. 68
Tabla 18. Presupuesto Final. 69

TEMA:

“DISEÑO DE UN SISTEMA DE AUTENTICACIÓN BIOMÉTRICA BASADO EN RECONOCIMIENTO FACIAL”.

RESUMEN

El presente proyecto consiste en el diseño de un sistema biométrico basado en reconocimiento facial. En este sistema se distinguen tres partes principales: detección, reconocimiento y base de datos.

La detección de rostros se hizo con el algoritmo de Viola-Jones, el cual permite realizar la detección facial de una manera rápida y en tiempo real de uno o varios rostros; el reconocimiento se lo realizó con la técnica Eigenfaces, que realiza una comparación de la distancia euclidiana entre las eigenfaces y la imagen de entrada. Se crearon dos bases de datos: una para las imágenes de las personas que van hacer uso del sistema para su posterior reconocimiento y otra para todos los registros del sistema como por ejemplo: datos personales, horarios, registros de entrada-salida.

El diseño y desarrollo del sistema se lo realizó mediante el lenguaje de programación de código abierto Java caracterizado por su flexibilidad y facilidad de codificación; la plataforma de desarrollo que se utilizó fue NetBeans 8.0 por ser amigable e intuitivo para el desarrollador y la metodología de desarrollo de software utilizada fue XP (Programación Extrema), la misma que se basa en un modelo ágil, rápido, flexible y de calidad, diseño simple y trabajo directo con el usuario donde no se necesita mucha documentación.

El presente proyecto permite hacer los registros de horas laboradas de los usuarios del sistema de una manera más rápida, efectiva y confiable; permitiendo tener un control real de las horas laboradas de cada uno de los individuos.

SUMMARY

The present Project talk about the design of a biometric system based on face recognition. In this system, we can recognize three main parts: detection, recognition and data base.

The algorithm of Viola-Jones did the face detection, which permit to perform the face detection in a speedy manner and in real time from one or more faces; the examination was made with the Eigenfaces technique, which makes a comparison of the Euclidean distance between Eigenfaces and the input image. Two databases were created: one for the images of the people who will use the system for subsequent recognition and other for all the record system such as: personal data, scheduling, input-output records.

The design and development of the system was made using the language Java Open Source characterized by its flexibility and ease of coding; the development platform used was NetBeans 8.0, because it is friendly and intuitive for the developer, and for the software development methodology used XP(Extreme Programming), It is based on a flexible, fast, flexible, quality, simple design and direct work model, it can use where the user do not need a lot of documentations.

This present project allow to do the records of hours worked by users of the system in a more quickly, effectively and reliably manner. Also it allows having a real control of the hours worked by each of the individuals.

OBJETIVOS

OBJETIVO GENERAL

- Diseñar un sistema de autenticación biométrica basado en reconocimiento facial.

OBJETIVOS ESPECÍFICOS

- Investigar las tecnologías sobre autenticación biométrica basada en reconocimiento facial.
- Estudiar los diferentes algoritmos de adquisición y comparación que existen para el estudio biométrico facial.
- Definir las herramientas a utilizar para el desarrollo de la aplicación.
- Desarrollar una interfaz hombre máquina de fácil manejo para el usuario para la ejecución del programa.
- Implementar el sistema de autenticación biométrica basado en reconocimiento facial.

INTRODUCCIÓN

A medida que ha evolucionado la tecnología, la búsqueda de un medio de seguridad y acceso a datos personales ha sido uno de los objetivos más importantes de la humanidad.

En la llamada era de la tecnología, cada vez más actividades que hasta entonces se realizaban en papel han pasado a realizarse mediante aplicaciones informáticas, como el comercio electrónico, o el acceso a datos bancarios, operaciones en las que se necesita garantizar la confidencialidad y seguridad.

En la vida cotidiana como en el mundo tecnológico, la importancia de la privacidad y la seguridad de los datos obligan a establecer medidas de identificación y autentificación con el fin de asegurar que ninguna otra persona accede a datos ajenos o servicios privados.

Hasta ahora podíamos observar continuamente métodos sencillos para conseguir resolver estos problemas como, por ejemplo, el uso de las tarjetas plásticas de identificación, en las que incluso se solía incorporar una foto o una firma a modo de identificación entre personas. En el mundo de la tecnología de la información este planteamiento debe ser modificado ya que no existe una persona que te identifique.

El incremento de los requerimientos de seguridad informática y los avances en la tecnología de la información han permitido un rápido desarrollo de sistemas inteligentes de identificación de personas basados en técnicas biométricas. Las técnicas biométricas usan características o comportamientos fisiológicos propios de cada individuo para identificarlo.

Existen muchas tecnologías que usan la biometría para la identificación y verificación de individuos como: Huellas dactilares, reconocimiento del iris del ojo, reconocimiento del rostro, geometría de la mano, etc.

La identificación por medio del reconocimiento facial establece una de las formas más representativas de la utilización de la biometría y ha recibido un gran impulso gracias

al avance en la tecnología de vídeo multimedia propiciándose así un aumento de cámaras en los lugares de trabajo, hogar y dispositivos móviles con un reducido coste.

El rostro humano está formado por facciones morfológicas. Estas formas son características únicas que pueden ser medidas y es posible obtener la identidad de una persona que intenta acceder a un sistema en general.

El reconocimiento facial se puede aplicar en el control de accesos a edificios públicos y privados, cajeros automáticos, laboratorios de investigación, como clave secreta de acceso para el uso de ordenadores personales o terminales móviles de última generación así como para servir de tarjeta de visita de una persona.

Por estos motivos, el propósito de este proyecto de tesis es el desarrollo de un sistema de autenticación biométrica basado en reconocimiento facial que permita llevar un registro de las horas laborables (ingreso y salida) del personal de una empresa.

El proceso de identificación facial se divide básicamente en dos partes: detección y reconocimiento. La primera, la detección comprende la localización de uno o varios rostros dentro de una imagen, ya sea fija o una secuencia de vídeo. La segunda parte, el reconocimiento, consiste en la comparación del rostro detectado con otros almacenados previamente en una base de datos. Los sistemas de reconocimiento facial están fuertemente condicionados por la posición y orientación del rostro del individuo con respecto a la cámara y las condiciones de iluminación en el momento de realizar la detección.

METODOLOGÍA

Para el desarrollo de este sistema se utiliza la Metodología de Desarrollo de Software XP (Programación Extrema), la misma que se basa en un modelo ágil, rápido, flexible y de calidad, diseño simple y trabajo directo con el usuario.

Esta metodología sigue los siguientes pasos:

- Planificación (Análisis) que se fundamenta en lo que se requiere que haga el programa.
- Diseño simple, en esta fase se toma en cuenta todas las etapas que necesita tener el sistema.
- Desarrollo, se basa en la codificación y toma como punto principal la manera de hacer el sistema lo más simple posible para el usuario.
- Como última fase tenemos las Pruebas, en la cual el sistema debe ser probado, para asegurar y garantizar que el proyecto funciona correctamente.

Los métodos utilizados en la presente investigación son:

- **MÉTODO INDUCTIVO.-** Este método se caracteriza porque su desarrollo va de lo particular a lo general, crea leyes a partir de la observación de los hechos, mediante la generalización del comportamiento observado, servirá para la recolección de la información relacionada respecto a los patrones más representativos necesarios para el reconocimiento facial de las personas.

- **MÉTODO DEDUCTIVO.-** Este método se caracteriza porque su desarrollo va de lo general a lo particular y aspira a demostrar, mediante la lógica pura, la conclusión en su totalidad a partir de unas premisas, de manera que se garantiza la veracidad de las conclusiones. El empleo de este método en este proyecto es para buscar y encontrar soluciones adecuadas para el desarrollo del sistema biométrico.

- **MÉTODO CIENTÍFICO.-** Se caracteriza por ser un método sistemático y por el análisis de los problemas que se presentan en una investigación. El empleo de este método sirve para incrementar el conocimiento y así lograr el desarrollo de un buen sistema de reconocimiento facial.

CAPÍTULO 1

1. MARCO TEÓRICO

1.1. CONCEPTOS BÁSICOS DE IMAGEN DIGITAL

1.1.1. PIXEL

Pixel es la abreviatura de Picture Element (Elemento de Imagen). El pixel es la unidad mínima de visualización de una imagen digital. En la figura 1 podemos ver que al aplicar zoom a la imagen se observa que está formada por una rejilla de puntos conocidos como pixeles.

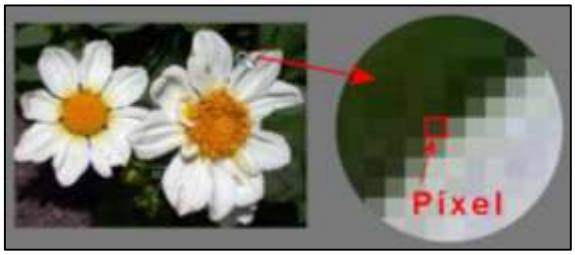

Figura 1. Pixel en una imagen. Fuente: [1]

El número de bits usados para representar cada pixel determina cuántos colores o gamas de gris pueden ser mostrados. Por ejemplo, en modo color de 8-bits, el monitor en color utiliza 8 bits para cada pixel, permitiendo mostrar 2^8 (256) colores diferentes o gamas de gris [2].

1.1.2. IMAGEN DIGITAL

Una imagen digital es una fotografía, un dibujo, un trabajo artístico o cualquier imagen que es almacenada como un archivo de una computadora para luego ser mostrada en la pantalla mediante pixeles.

La imagen digital está formada por una serie de matrices numéricas de ceros y unos en el caso de imágenes binarizadas, que se almacenan en una memoria informática y que definen las características de una fotografía [3].

Se las puede obtener de varias formas:

- Por medio de dispositivos de conversión analógica-digital como los escáneres y las cámaras digitales.

- Directamente mediante programas informáticos, como por ejemplo mediante un programa de renderización 2D.

1.1.3. TIPO DE IMÁGENES DIGÍTALES

1.1.3.1. MAPA DE BITS

Es una imagen creada sobre una cuadrícula, que se guarda como fichero. Cada uno de los cuadros de la cuadrícula se denomina píxel, y este guarda información del color.

1.1.3.2. IMÁGENES VECTORIALES

Están formada por objetos geométricos independientes (segmentos, polígonos, arcos, etc.), cada uno de ellos definido por distintos atributos matemáticos de forma, de posición o de color. Por ejemplo un círculo de color verde quedaría definido por la posición de su centro, su radio, el grosor de línea y su color.

En la figura 2 se ve la diferencia entre una imagen de mapa de bits y una vectorial. La imagen de mapa de bits está formada por un conjunto de puntos de bits, mientras que una imagen vectorial está representada por fórmulas matemáticas.

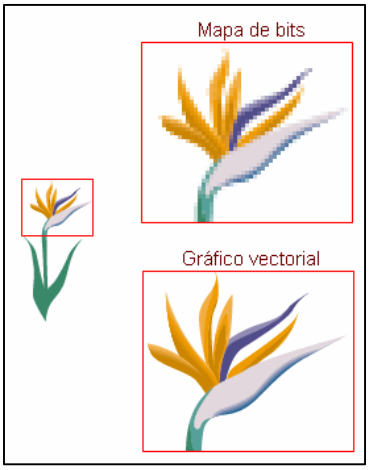

Figura 2. Imagen de mapa de bits y vectorial. Fuente: [24].

1.1.4. RESOLUCIÓN DE IMAGEN

La resolución de una imagen indica cuántos bits se utilizan para representar cada pixel, por lo tanto la calidad de una imagen depende en gran medida de su resolución. El valor de resolución se expresa en ppp (píxeles por pulgada) o en dpi en inglés (dots per inch). Cuantos más pixeles tenga una imagen por pulgada, mayor resolución tendrá.

1.1.5. PROFUNDIDAD DE COLOR

Cuando se habla de profundidad de color se refiere al número de bits necesarios para codificar y guardar la información de color de cada píxel en una imagen. Un bit es una posición de memoria que puede tener el valor 0 ó 1. Cuanto mayor sea la profundidad de color en bits, la imagen dispondrá de una paleta de colores más amplia, como se observa en la tabla 1.

Tabla 1. Profundidad del color.

# DE BITS	# DE COLORES
1-bit	2 colores (Blanco/negro)
2-bits	4 colores
3-bits	8 colores
8-bits	256 colores
24-bits	16.7 millones de colores

Fuente: [4].

1.1.6. MODOS DE COLOR

Se conoce como modos de color al sistema de coordenadas que nos permite describir el color de cada píxel utilizando valores numéricos.

Los modos de color más utilizados son:

- **Modo monocromático.** Se corresponde con una profundidad de color de 1 bit. La imagen está formada por píxeles blancos o píxeles negros puros como se observa en la figura 3.

Figura 3. Modo monocromático. Fuente: [5].

- **Modo Escala de Grises**. También conocida como escala de intensidades maneja el canal negro y permite 256 tonos de gris entre el blanco y negro puros. En la figura 4 se observa una imagen a escala de grises.

Figura 4. Imagen a escala de grises. Fuente: [5].

- **Modo Color indexado**. Es una forma práctica, pero limitada, de representar imágenes en color. Una imagen indexada almacena una imagen como dos matrices. La primera de ellas tiene el mismo tamaño que la imagen y un número para cada pixel. La segunda matriz se denomina mapa de color y su tamaño corresponde al número de colores que se desea que tenga la nueva imagen. Esta técnica es utilizada para ahorrar memoria de la computadora, comprimir imágenes y lograr una mejoría en el tiempo de muestra "display".

  Utiliza un canal de color indexado de 8 bits pudiendo obtener con ello hasta un máximo de 256 colores. En la figura 5 se puede apreciar este modo.

Figura 5. Imagen de modo de color indexado. Fuente: [5].

- **Modo RGB**. Cada color se forma por combinación de tres canales. Cada canal se corresponde con un color primario: Red (rojo), Green (verde), y Blue (azul). Asigna un valor de intensidad a cada color que oscila entre 0 y 255. De la combinación surgen hasta 16,7 millones de colores. La figura 6 muestra una imagen modo RBG.

  El modelo RGB puede verse como una pila de 3 imágenes en escala de intensidades que al ser mostrados por un monitor de color representan una imagen de color tal como la percibe un ser humano. Los colores, rojo, verde y azul son conocidos como los colores primarios, y la combinación de estos en diferentes intensidades produce los colores del espectro humano visible.

Figura 6. Imagen modo RGB. Fuente: [5].

- **Modo HSB**. Está basado en el trabajo de Albert Munsell y sus estudios de la percepción humana del color, definiendo los colores en función de las tres propiedades del color (matiz, luminosidad y saturación).
  - Tono (Hue) es el valor del color: rojo, azul, verde, etc.
  - Saturación (Saturation) se refiere a la pureza del color y va del 0% al 100%.

- Brillo (Brightness) se referencia la intensidad de luz del color, es decir, la cantidad de negro o blanco que contiene estando su valor entre 0 (negro) y 100 (blanco).

En la figura 7 se observa los valores de una imagen modo HSB.

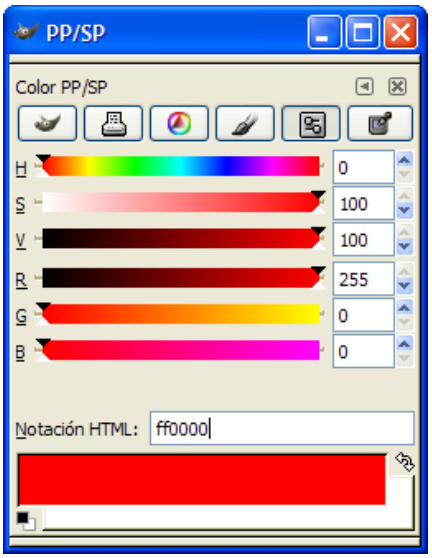

Figura 7. Modo HSB. Fuente: [5].

1.1.7. FORMATOS DE IMAGEN

1.1.7.1. BMP (BITMAP = MAPA DE BITS)

Fue desarrollado para aplicaciones Windows. La imagen se forma mediante una rejilla de píxeles. Este formato no sufre pérdidas de calidad y por tanto resulta adecuado para guardar imágenes que se desean manipular posteriormente.

- **Ventaja:** Guarda gran cantidad de información de la imagen.
- **Inconveniente:** El archivo tiene un tamaño muy grande.

1.1.7.2. GIF (GRAPHICS INTERCHANGE FORMAT = FORMATO DE INTERCAMBIO GRÁFICO)

Diseñado para comprimir imágenes digitales. Este formato durante años ha sido un formato de imagen usado en la web y reduce la paleta de colores a 256 como máximo.

- **Ventaja:** Es un formato apto para publicar imágenes en la web.

- **Inconveniente:** No es recomendable para fotografías de cierta calidad ni originales ya que el color real o verdadero utiliza una paleta de más de 256 colores.

1.1.7.3. JPG-JPEG (JOINT PHOTOGRAPHIC EXPERTS GROUP = GRUPO DE EXPERTOS FOTOGRÁFICOS UNIDOS)

JPEG es el formato de imagen más comúnmente utilizado por las cámaras digitales y otros dispositivos de captura de imágenes fotográficas. También es el formato más común para almacenar y transmitir imágenes fotográficas en Internet.

- **Ventaja:** Es ideal para publicar fotografías en la web siempre y cuando se configuren adecuadamente dimensiones y compresión.
- **Inconveniente:** Si se define un factor de compresión se pierde calidad. Por este motivo no es recomendable para archivar originales.

1.1.7.4. TIF-TIFF (TAGGED IMAGE FILE FORMAT = FORMATO DE ARCHIVO DE IMAGEN ETIQUETADA)

Este formato almacena imágenes de una calidad excelente. Utiliza cualquier profundidad de color de 1 a 32 bits. Es ideal para editar o imprimir una imagen.

- **Ventaja:** Es ideal para archivar archivos originales.
- **Inconveniente:** Produce archivos muy grandes.

1.1.7.5. PNG (PORTABLE NETWORK GRAPHIC = GRÁFICO PORTABLE PARA LA RED)

Es un formato de imagen de mapa de bits que utiliza compresión sin pérdida de datos similar a la de un archivo GIF. Fue creado para mejorar y reemplazar el formato GIF, ya que no es un formato de archivo de imagen que requiere una licencia de patente. Tiene una tasa de compresión superior al formato GIF (+10%). Admite la posibilidad de emplear un número de colores superior a los 256 que impone el GIF.

- **Ventaja:** Funciona muy bien para las imágenes geométricos o líneas, letras, imágenes animados y otras imágenes con colores planos y bordes bien definidos.
- **Desventaja:** No se pueden utilizar para las animaciones.

En la figura 8 se encuentran los formatos de imagen antes mencionados.

Figura 8. Formatos de imagen. Fuente: [1].

1.2. PROCESAMIENTO DE IMAGEN

El objetivo principal del procesamiento digital de imagen es procesar una imagen con el fin de hacerla más adecuada para una determinada aplicación o procesamiento posterior. Depende por tanto del problema específico a resolver el que se emplee una u otra técnica. Los métodos de mejora de imagen se pueden dividir en dos campos diferentes: métodos en el dominio frecuencia y métodos en el dominio espacial. Los primeros se basan en modificar la transformada de Fourier de la imagen, mientras que los segundos se basan en manipulaciones directas sobre los píxeles de la imagen [6].

1.2.1. TÉCNICA DE MODIFICACIÓN DEL HISTOGRAMA

El histograma es la representación de la frecuencia relativa de cada color en una imagen. La modificación del histograma consiste en generalizar las transformaciones de intensidad de los píxeles de la imagen. En general se define una función de transferencia, una función que se aplica a cada pixel de la imagen para transformar su valor. De esta forma se puede definir cualquier transformación de intensidad como la variación del brillo y el contraste, el ecualizado o el negativo.

La ecualización del histograma pretende que para todos los niveles de gris se tenga el mismo número de pixeles, es decir, que sea uniforme. En la figura 9 se muestra el histograma una imagen, los píxeles están distribuidos en todo el rango de 0 – 255.

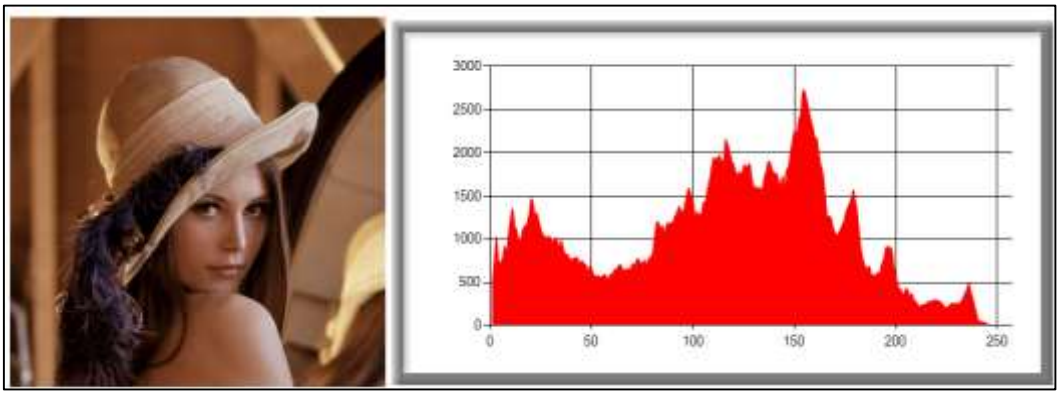

Figura 9. Histograma de una imagen. Fuente: [25].

1.2.2. FILTRADO ESPACIAL

El concepto de filtrado de una imagen está asociado a la representación de una imagen en el dominio de frecuencias. El filtrado espacial es la operación que se aplica a las imágenes para mejorar o suprimir detalles espaciales con el fin de mejorar la interpretación visual. Todo filtro que se diseñe tendrá como objetivo modificar la contribución de determinados rangos de frecuencias a la formación de la imagen. Así por ejemplo si queremos filtrar una imagen de una posible contaminación de ruido aleatorio lo que tendremos que hacer es diseñar un filtro que reduzca lo más posible la contribución de las altas frecuencias en la formación de la imagen.

1.2.3. TIPOS DE FILTROS

1.2.3.1. FILTROS DE PASO BAJO

Estos atenúan o eliminan las frecuencias altas. Tienen como objetivo suavizar la imagen, son útiles cuando la imagen tiene gran cantidad de ruido y se quiere eliminar. Existen varias posibilidades:

- **Filtro de la media:** Asigna al pixel central la media de todos los pixeles incluidos en la ventana. La matriz de filtrado estaría compuesta por unos y el divisor sería el número total de elementos en la matriz.
- **Filtro de media ponderada:** Los elementos de la matriz de filtrado no son todos sino 1 que se da más peso a uno de ellos (generalmente el central) para obtener un resultado más parecido a la imagen original y evitar que aparezca borrosa.
- **Filtro de la mediana:** Tiene la ventaja de que el valor final del pixel es un valor real presente en la imagen y no un promedio, de este modo se reduce el efecto borroso que tienen las imágenes que han sufrido un filtro de media. Además el filtro de la mediana es menos sensible a valores extremos. El inconveniente es que resulta más complejo de calcular ya que hay que ordenar los diferentes valores que aparecen en los pixeles incluidos en la ventana y determinar cuál es el valor central.

- **Filtros adaptativos:** Son considerablemente más complejos ya que los coeficientes de ponderación se recalculan para cada uno de los pixeles en función del histograma de los ND (Nivel Digital) que aparecen en la ventana. Se han utilizado con gran éxito filtros adaptativos para eliminar el speckle (patrón de interferencia que se forma cuando una imagen se obtiene a partir de la iluminación de un medio con una radiación no coherente) de las imágenes de radar y para detectar, con un solo filtro, diferentes elementos.

- **Filtros gaussianos:** Estos filtros se implementan para la eliminación del ruido gaussiano presente en las imágenes. Este ruido tiene un efecto general en toda la imagen, es decir, la intensidad de cada pixel de la imagen se ve alterada en cierta medida con respecto a la intensidad en la imagen original.

1.2.3.2. FILTROS DE PASO ALTO

Tiene como objetivo resaltar las zonas de mayor variabilidad eliminando lo que sería la componente media, precisamente la que detectan los filtros de paso bajo. Existen diversos métodos:

- **Sustracción de la media:** Si se considera que un filtro de paso bajo sirve para resaltar componentes a gran escala eliminando la variabilidad local, si a la imagen original se le resta el resultado de pasarle un filtro de paso bajo se consigue resaltar esa variabilidad local [8].

- **Filtros basados en las derivadas:** La derivada de una función y = f(x) es el incremento de y para cada incremento infinitesimal de x. En el caso de Modelo Digital de Elevaciones la derivada es la pendiente. La segunda derivada es la derivada de la derivada, en el caso de un MDE nos da información acerca de la forma (ladera recta, cóncava o convexa, valle, cresta o cima) del terreno. En el caso de una imagen de satélite nos va a informar de cómo son los cambios, más o menos bruscos, que se producen entre pixeles contiguos [8].

1.2.3.3. FILTROS PARA LA DETECCIÓN DE BORDES

Borde es aquella región donde aparece una fuerte variación del nivel de intensidad en los pixeles vecinos. Sin embargo, estas discontinuidades también aparecen de forma no deseada por la presencia del ruido, lo que hace más complicada la tarea de localizar

los bordes. Estos filtros típicamente crean una imagen con fondo gris y líneas blancas y negras rodeando los bordes de los objetos y características de la imagen.

- **Filtro Roberts:** Emplea la diferenciación como método para calcular el grado de separación entre niveles de grises vecinos.
- **Filtro Sobel:** Es uno de los más utilizados y realza la variación entre filas y columnas.
- **Filtro direccional:** Seleccionando adecuadamente los valores del kernel, podemos obtener el efecto de extraer bordes en una determinada dirección, mientras que los bordes en el resto de direcciones no se ven tan resaltados.

En la figura 10 tenemos imágenes filtradas con distintos filtros detectores de bordes.

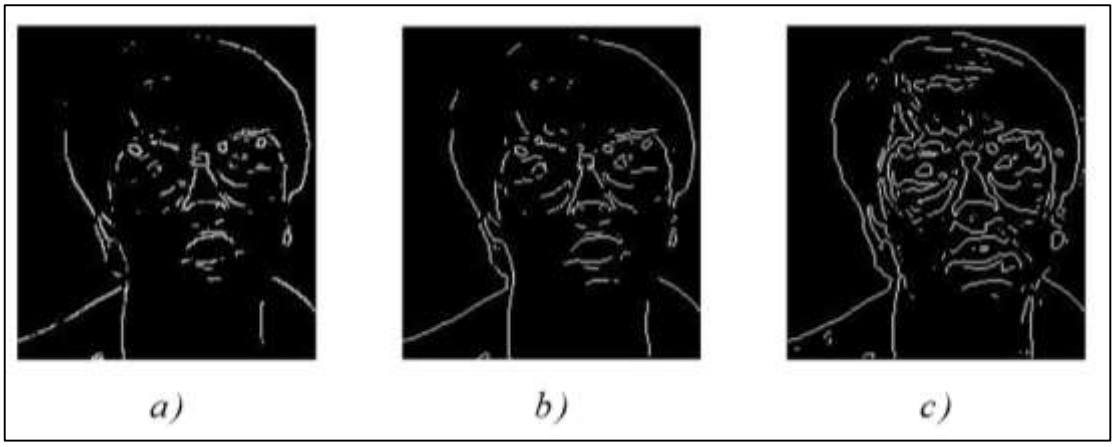

Figura 10. a) Roberts, b) Sobel, c) Laplaciano. Fuente: [6].

1.2.3.4. FILTRADO EN FRECUENCIA

En el dominio de la frecuencia también puede realizarse el proceso de filtrado, con mayor grado de compresión de lo que se está viendo, ya que en una imagen en el dominio frecuencial se sabe dónde se encuentran los distintos rangos de frecuencias. De esta forma, en vez de realizar la convolución, se efectúa su operación correspondiente en el dominio de la frecuencia [7].

1.2.3.5. FILTROS MORFOLÓGICOS

La morfología matemática es un método no lineal de procesar imágenes digitales basando en su forma. El objetivo de las transformaciones morfológicas es la extracción de estructuras geométricas en los conjuntos sobre los que se opera, mediante la utilización

de otro conjunto conocido como elemento estructurante. Estos filtros cumplen las siguientes propiedades:

- Son invariantes ante la traslación.
- La erosión, dilatación, opening y closing son invariantes antes unas traslación, tanto para el conjunto como el elemento estructurante.
- Son monótonamente crecientes.

- **Dilatación**

Tiene como objetivo rellenar agujeros y bahías de tamaño igual o menor que el elemento estructurante.

- **Erosión**

Es lo opuesto a la dilatación. Su objetivo es eliminar grupos de pixeles donde no entra el elemento estructurante, como islas pequeñas o protuberancias.

- **Opening y closing**

La apertura y el cierre es una combinación de erosión más dilatación, y se considera una operación antiextensiva.

1.2.3.6. FILTROS DE TEXTURA

Muchas imágenes contienen regiones caracterizadas por variaciones del nivel de gris, más que por un valor único de grises. La "textura" se refiere precisamente a la variación espacial del nivel de gris de una imagen como función de escala espacial. Para que los píxeles de una determinada área puedan ser definidos como texturalmente diferentes, sus niveles de gris deben ser más homogéneos como unidad que áreas de diferente textura [6].

- **Filtro de recorrido**

Conocido también con el nombre "de rango". Este filtro sustituye el valor central de la ventana de procesamiento por la diferencia entre el valor máximo y mínimo de los píxeles contenidos en esa ventana.

El recorrido será un valor pequeño para zonas "planas" o texturalmente uniformes, y será alto en zonas de alta variabilidad.

- **Filtro RMS (Root-Mean-Square)**

Este filtro calcula primero la varianza de los valores de la ventana, y sustituye el valor central por el RMS de los píxeles de la ventana de proceso [7].

- **Operadores de momento**

El primer y segundo momento son simples medidas de textura, utilizando los "momentos" del histograma de la ventana de proceso. El primer momento es una medida del contraste de la ventana. El segundo momento es una medida de la homogeneidad de la misma. Las imágenes resultantes pueden ser escaladas para crear una imagen que discrimina entre varias texturas [7].

1.3. LENGUAJE DE PROGRAMACIÓN JAVA

1.3.1. QUÉ ES JAVA

Java es un lenguaje de programación orientado a objetos [9] para el desarrollo de aplicaciones. Además, es un lenguaje de plataformas cruzadas, lo que significa que puede ser diseñado para que corra igualmente en Windows de Microsoft, Apple de Macintosh y la mayoría de las versiones UNIX, incluyendo Solaris. Java puede ejecutarse en dispositivos como televisores, relojes de pulso y teléfonos celulares [10].

1.3.2. CUADRO COMPARATIVO DE JAVA CON OTROS LENGUAJES DE PROGRAMACIÓN

Tabla 2. Comparativo de java con otros lenguajes de programación.

Lenguaje de programación	¿Qué es?	Ventajas	Desventajas	Sistemas operativos
C++	Orientada a objetos creado por Bjarne Stroustrup.	• Es potente para la creación de sistemas complejos porque es robusto.	• No es atractivo visualmente. • No soporta la creación de páginas web.	Todos los sistemas operativos pero cada uno con su respectiva versión.

C	Lenguaje de propósito general que ofrece economía sintáctica.	• Simple y flexible que permite múltiples estilos. • Un conjunto reducido de palabras clave.	• Recolección de basura nativa. • Encapsulación	Ligado al sistema operativo UNIX.
Java	Orientado a objetos desarrollado por la SUN Microsystems ahora es propietario ORACLE. Maneja algunas plataformas de desarrollo.	• Puede desarrollar aplicaciones de escritorio que se ejecutan en forma independiente. • Soporta el desarrollo de aplicaciones móviles.	• Esperar la actualización siguiente para que sea más rápido.	Sirve para todos los sistemas operativos:Unix, Linux, Solaris, Windows, Mac.

Fuente: [12].

1.3.3. MOTIVOS PARA PROGRAMAR EN JAVA

A continuación se explican algunos motivos que hacen de Java uno de los mejores lenguajes de programación que se hayan inventado.

- **Independencia de la plataforma**

Esto significa que cuando se programa en Java, no se necesita conocer a priori el tipo de ordenador o el sistema operativo para el que estás programando. Se puede ejecutar el mismo programa en un PC con Windows, otro con Linux, en un Servidor SUN con sistema operativo Solaris, o en un teléfono móvil de última generación.

Cuando se ejecuta un programa escrito en Java, en realidad lo que estamos ejecutando es un simulador de esta máquina virtual, y este simulador es el que realmente ejecuta el código. Por eso se dice que Java es un lenguaje Compilado e Interpretado a la vez (figura 11), porque el código se compila a un lenguaje intermedio de una máquina que no existe (lenguaje *ByteCode*), y cuando queremos ejecutarlo, es un simulador (intérprete) el que lo hace [13].

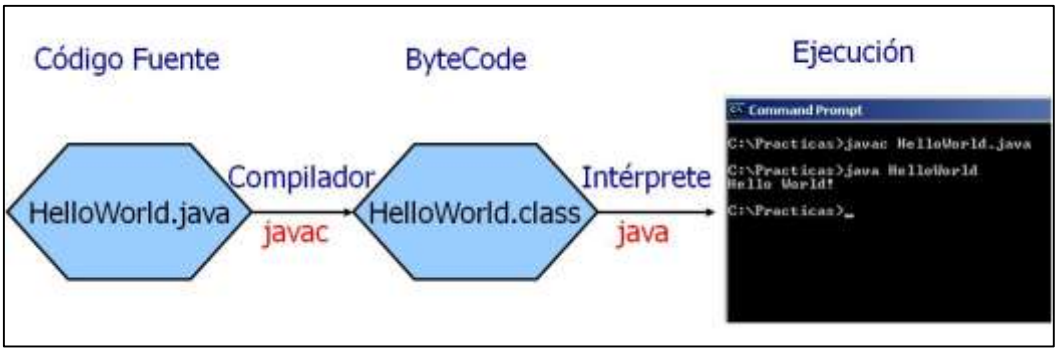

Figura 11. El código Java es Compilado y luego Interpretado. Fuente: [5].

- **Orientado a Objetos**

La programación orientada a objetos permitió un gran avance en el desarrollo de aplicaciones, ya que es capaz de acercar la forma de programar a la forma de pensar del ser humano. En la figura 12 tenemos un programa sencillo realizado en java.

```
package hola;

/**
 * Nombre: Hola.java
 * Descripción: Mi primer programa en java
 * @author Sandra
 */
public class Hola {

    public static void main(String[] args) {
        System.out.println("Hola Sandra");
    }

}
```

Figura 12. Un simple programa escrito en Java. Fuente: [La Autora].

- **Facilidad de Aprendizaje**

Java es relativamente fácil de aprender comparado con otros lenguajes. Tiene algunas ventajas en cuanto a facilidad de aprendizaje, como la no existencia de los denominados punteros, o la ya mencionada gestión automática de memoria gracias al Garbage Collector. Además, el aprendizaje de Java se puede realizar de manera gradual [13].

- **Entornos de Desarrollo**

Antes se solía utilizar editores de texto para programar (como NotePad o cualquier editor común en Windows, o Vi en Linux). El desarrollo con estos editores era bastante lento, porque aportaban muy pocas funcionalidades específicas del lenguaje. Sin embargo, hoy en día existen excelentes editores (IDEs) que aportan multitud de ayudas a la programación, haciendo que el desarrollo sea más fluido y cómodo. Ejemplos de excelentes IDEs son Eclipse, NetBeans o el excelente IntelliJ Idea [13].

1.4. BIOMETRÍA

1.4.1. DEFINICIÓN DE BIOMETRÍA

Biometría proviene de las palabras bio (vida) y metría (medida), es el estudio de métodos automáticos para el reconocimiento único de humanos basados en uno o más rasgos conductuales o rasgos físicos intrínsecos, por lo tanto todo equipo biométrico mide e identifica alguna característica propia de la persona [14].

1.4.2. HISTORIA DE LA DE LA BIOMETRÍA

La biometría se remonta siglos atrás cuando los antiguos Egipcios median a las personas para identificarlas, esto se conoce como antropometría. Esta manera rudimentaria de identificación se basaba en las medidas de algunas partes del cuerpo y sigue siendo utilizada desde entonces. La biometría era utilizada en China desde al menos el siglo XIV.

En Occidente, la identificación confiaba simplemente en la "memoria fotográfica" hasta que Alphonse Bertillon, jefe del departamento fotográfico de la Policía de París, desarrolló el sistema antropométrico (también conocido más tarde como Bertillonage) en 1883. Éste era el primer sistema preciso, ampliamente utilizado científicamente para identificar a criminales y convirtió a la biométrica en un campo de estudio. Funcionaba midiendo de forma precisa ciertas longitudes y anchuras de la cabeza y del cuerpo, así como registrando marcas individuales como tatuajes y cicatrices [15].

La identificación basada en la huella dactilar se viene utilizando en los Estados Unidos y Europa Occidental desde hace más de cien años. Finalizando los años sesenta el FBI comenzó a verificar automáticamente las huellas dactilares.

El primer sistema para la medición de la retina fue introducido en los años ochenta. La identificación basada en la firma y en el rostro es relativamente nueva.

1.4.3. CLASIFICACIÓN DE LA BIOMÉTRIA

La biometría se clasifica en:

- **Biometría física o estática:** Se refiere a la medida de los rasgos físicos de la persona. Las huellas dactilares, la retina, el iris, los patrones faciales, de venas de la mano o la geometría de la palma de la mano son ejemplos.
- **Biometría del comportamiento o dinámicas:** Está relacionada a la conducta de la persona. La firma, el paso y el tecleo son ejemplos.
- **Biometría multimodal:** Realiza la combinación de medidas, tanto físicas como de comportamiento. La voz se considera una mezcla de características físicas y del comportamiento.

1.4.4. REQUISITOS BÁSICOS DE LA BIOMETRÍA

Los requisitos básicos que deben reunir las características biométricas son:

- **Universalidad:** todos los usuarios deben poseer dicha característica.
- **Singularidad o univocidad:** dos personas se deben diferenciar según dicha característica.
- **Permanencia:** en el tiempo y condiciones ambientales diversas.
- **Rendimiento o actuación:** elevado nivel de exactitud.
- **Aceptabilidad:** Grado al que está la gente dispuesta a aceptar el uso de una característica biométrica en su vida diaria. Debe resultar inofensivo [16].
- **Seguridad:** Dificultad para engañar al sistema usando de métodos fraudulentos, Robustez.

1.4.5. FUNCIONAMIENTO Y RENDIMIENTO DE LA BIOMETRIA

En un sistema de Biometría típico, la persona se registra con el sistema cuando una o más de sus características físicas y de conducta son obtenidos, procesada por un algoritmo numérico, e introducida en una base de datos. Idealmente, cuando entra, casi todas sus características concuerdan; entonces cuando alguna otra persona intenta identificarse, no empareja completamente, por lo que el sistema no le permite el acceso. Las tecnologías actuales tienen tasas de error que varían ampliamente (desde valores bajos como el 60%, hasta altos como el 99,9%) [17].

1.4.6. TECNOLOGÍAS BIOMÉTRICAS MÁS USADAS

Las tecnologías biométricas más usadas son: la huella dactilar, reconocimiento facial, análisis de iris y el reconocimiento de voz pues se encuentran en pleno auge y expansión. A continuación se detallan estas técnicas:

- **Reconocimiento biométrico por medio de la huella dactilar**

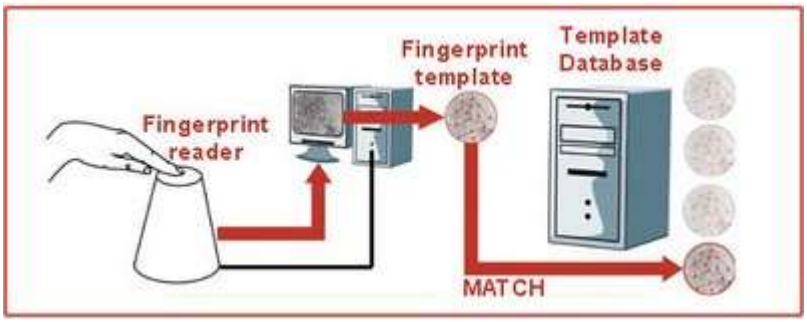

Figura 13. Proceso de un sistema biométrico por medio de la huella dactilar. Fuente: [17].

La huella dactilar es sin duda la técnica más antigua y extendida. Esto se debe a que la gran mayoría de la población tiene huellas dactilares únicas e inalterables. Tiene una alta tasa de precisión y los usuarios la han aceptado sin mayor dificultad. En la figura 13 tenemos el proceso de un sistema biométrico por medio de la huella dactilar.

Sólo una pequeña parte de la población puede no tener huellas adecuadas para la identificación automática, debido o al envejecimiento o profesionales, como obreros con un gran número de cortes [18].

- **Reconocimiento biométrico por medio del rostro**

La técnica más comúnmente usada es el reconocimiento en dos dimensiones (2D), en la que para identificar una persona sólo se necesita la ubicación de sus atributos faciales como ojos, nariz, labios y barbilla y la distancia entre estos [19].

Es un método no intrusivo y las imágenes faciales son, probablemente, la característica biométrica más comúnmente utilizada por los seres humanos para hacer un reconocimiento personal. Tiene buena aceptación por parte de los usuarios y los medios para capturar las imágenes son relativamente económicos.

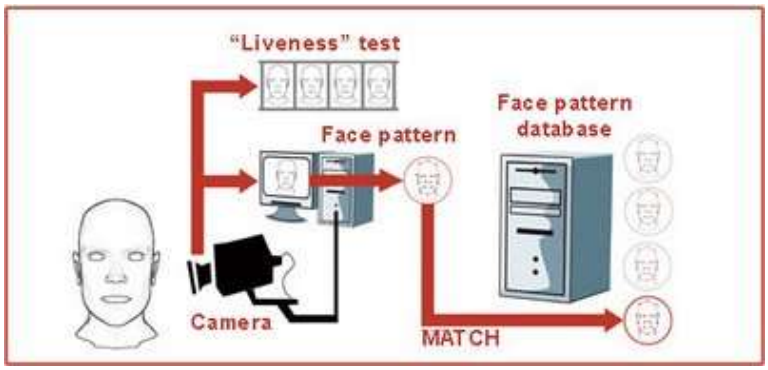

Figura 14. Proceso de un sistema biométrico por medio del rostro. Fuente: [17].

El rendimiento y eficacia de los sistemas de reconocimiento facial disponibles comercialmente es razonable, pero la naturaleza compleja y mutable de los rasgos faciales impone una serie de restricciones tanto en la forma y ángulo de obtención de la imagen, como en el fondo de iluminación. En la figura 14 tenemos un sistema biométrico de reconocimiento facial.

- **Reconocimiento por análisis del iris**

Cada iris es distinto, incluso en gemelos idénticos. Es difícil manipular quirúrgicamente el iris y fácil detectar uno artificial, como, por ejemplo, unas lentes de contacto.

- **Reconocimiento biométrico por medio de la voz**

La voz es una combinación de algunos datos físicos de la persona, como la forma y el tamaño de su boca, fosas nasales o labios, y que se utilizan en la creación del sonido, así como aspectos de comportamiento como son la entonación o el tono.

Aceptada por los usuarios dado que es un método poco invasivo y muy fácil de utilizar. No requiere dispositivos especiales, la calidad de una llamada telefónica es suficiente [20]. La figura 15 muestra que para el proceso de este sistema es necesario un micrófono como entrada al sistema.

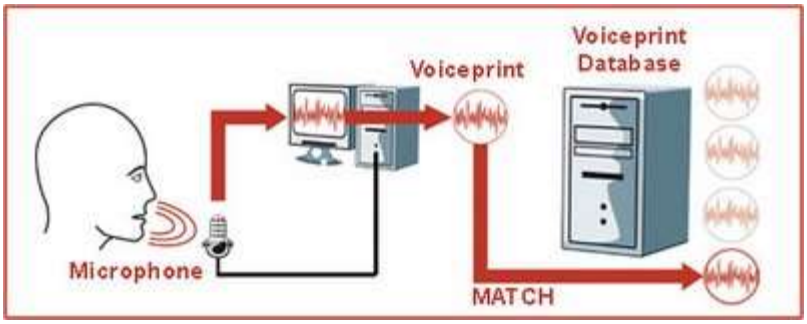

Figura 15. Proceso de un sistema biométrico por medio de la voz. Fuente: [17].

- **Reconocimiento biométrico por medio de ADN**

El ácido desoxirribonucleico (ADN) supone la máxima expresión de unicidad de código para la individualidad, a excepción de los gemelos idénticos que comparten patrones de ADN. Actualmente se utiliza en el ámbito forense, la tecnología ha avanzado lo suficiente para permitir su aplicación en otros ámbitos [21].

Tiene una precisión muy alta y el coste para llevar a cabo todo el proceso no justifica en muchas ocasiones su utilización.

1.4.7. PARÁMETROS BIOMÉTRIOS

Índices para medir la efectividad de un sistema biométrico de identificación y verificación:

- **FAR** (False Acceptance Rate): tasa de falso positivo, hace referencia a la probabilidad de que un usuario no autorizado sea aceptado. Este parámetro deberá ajustarse para evitar el fraude [22].

- **FRR** (False Rejection Rate): tasa de rechazo erróneo, la probabilidad de que un usuario que está autorizado sea rechazado a la hora de intentar acceder al sistema. Si los usuarios son rechazados erróneamente con frecuencia, parecerá que el sistema no funciona correctamente y deberá ser revisado [22].

- **FER** (Failure to Enroll Rate): tasa de fallo de alistamiento, hace referencia a los usuarios que son rechazados cuando van a ser registrados a causa de la mala calidad de su muestra [22].

- **UMBRAL**: es un tipo de referencia, es la puntuación que determina la consistencia de un patrón. Éste se puede ajustar dependiendo del nivel de seguridad [22]**.**

- **FTE** (Failure to Enroll): indica la probabilidad numérica de que alguien no sea registrado a causa de un fallo a la hora de crear un patrón [22].

1.4.8. COMPARACIÓN DE SISTEMAS BIOMÉTRICOS

En la tabla 3 se recogen las diferentes características de los sistemas biométricos:

Tabla 3. Comparación de sistemas biométricos.

	Ojo (Iris)	**Ojo (Retina)**	**Huellas dactilares**	**Geometría de la mano**	**Escritura y firma**	**Voz**	**Cara**
Fiabilidad	Muy alta	Muy alta	Alta	Alta	Media	Alta	Alta
Facilidad de uso	Media	Baja	Alta	Alta	Alta	Alta	Alta
Prevención de ataques	Muy alta	Muy alta	Alta	Alta	Media	Media	Media
Aceptación	Media	Media	Media	Alta	Muy alta	Alta	Muy alta
Estabilidad	Alta	Alta	Alta	Media	Baja	Media	Media

Fuente: [23].

1.5. DETECCIÓN DE ROSTROS

1.5.1. DEFINICIÓN

La detección de rostros es un paso previo al reconocimiento de facial y es una tarea computacional que intenta de manera rápida y robusta, detectar rostros en imágenes o videos. En una imagen cualquiera, la detección de rostros se define como la determinación de la posición y tamaño de todos los rostros que en ella pueda haber.

"La detección consiste en la segmentación de la imagen en regiones conforme a algún criterio de homogeneidad" [7].

1.5.2. MÉTODOS DE DETECCIÓN

Los métodos existentes pueden ser agrupados de la siguiente manera:

- **Métodos Basados en Conocimiento:** Codifican el conocimiento de lo que es un rostro.
- **Métodos Basados en Características Invariantes:** Son algoritmos que tienen como objetivo encontrar características estructurales que permanecen constantes por variación de posición, de punto de vista o condiciones lumínicas. Generalmente utilizan información del color y la textura de la imagen.
- **Métodos Basados en Plantillas:** Es una técnica empleada para detectar objetos de una escena, donde el mismo es representado por una familia de curvas que representan el objeto.
- **Métodos Basados en la Apariencia:** No necesitan conocimiento previo de la característica a ser detectada, generalmente las técnicas que pertenecen a esta clasificación necesitan varias imágenes, a partir de ellas aprenden y codifican solamente lo necesario para hacer la detección de las características de interés.

1.5.3. PROBLEMAS COMUNES EN LA DETECCIÓN

Entre los problemas más comunes en la detección de rostros que se dan en el proceso de adquisición de imágenes, condiciones de luminosidad, bajo contraste, objetos extraños y distorsión de la imagen. A continuación se detallan:

- **Posición:** la orientación y posición de las imágenes varían debido al medio de captura o a la posición de las personas al momento de hacerse la captura de imagen.
- **Expresión facial:** puede haber la presencia de risa, enojo y otros gestos.
- **Presencia de estructuras:** la barba, bigote o lentes son elementos que pueden ser un problema.
- **Ocultación:** puede darse la superposición de imágenes.

1.5.4. ROTACIONES

Todo objeto que se desee detectar en una foto o en un video, es una proyección bidimensional de un objeto tridimensional. Por esta razón las rotaciones pueden ser un gran reto para la detección. Se clasifican en:

- **Rotaciones en el plano:** El objeto rota entre 8 y 360° entorno a su centro.

- **Rotaciones verticales fuera del centro:** Cuando la persona la persona está mirando hacia arriba hacia abajo se da una rotación fuera del plano.

- **Rotaciones horizontales fuera del plano:** Esta se produce cuando se ven los objetos de perfil o de frente y si el objeto a detectar no es uniforme, se vuelve en dificultad para la detección.

1.5.5. TÉCNICAS BASADAS EN RASGOS

Estas técnicas usan propiedades aparentes del rostro como el color de la piel y la geometría facial. Sigue una metodología clásica de detección en la que los rasgos de más bajo nivel se extraen de un análisis a priori basado en el conocimiento.

En estas técnicas la detección del rostro se resuelve manipulando medidas de distancia, ángulos y áreas de los rasgos visuales en la imagen. El punto determinante en este tipo de técnicas es decidir que rasgos del rostro interesan para su estudio [6].

1.5.5.1. ANÁLISIS DE BAJO NIVEL

Son técnicas que trabajan a nivel de pixel. En este grupo tenemos:

- **Detección de bordes:** Los bordes son los rasgos más primitivos de una imagen y se usan para detectar rasgos faciales. Se puede utilizar para detectar si la persona lleva gafas.

- **Niveles de gris:** Rasgos faciales como cejas, pupilas y labios aparecen más oscuros que las regiones de su alrededor.

- **Color:** El color es una herramienta poderosa porque triplifica la dimensionalidad y hay más información. Dos formas que en una imagen de blanco y negro aparecen iguales pueden ser diferentes en una imagen de color. El color de la piel humana ha permitido desarrollar algunas técnicas que detentan la raza.

- **Video:** Con una secuencia de video, localizar objetos en la imagen es más sencillo. Una de las mejores formas es mediante fotogramas. Existen técnicas que miden variaciones verticales y horizontales para encontrar los ojos. Detectar contornos de una escena en movimiento es más fácil.

- **Medidas generalizadas:** Se basa en que el rostro es simétrico y por esta razón utilizan la medida de simetría de la imagen.

1.5.5.2. ANÁLISIS DE RASGOS

El problema del análisis a bajo nivel es que pueden proporcionar información equívoca, por ejemplo, si aparecen en la imagen objetos que tengan color similar al modelo de piel utilizado [6].

El análisis de rasgos se fundamenta en la geometría del rostro para caracterizar y posteriormente verificar rasgos a fin de evitar ambigüedad. Existen dos estrategias:

- **Búsqueda de rasgos:** Estas técnicas buscan rasgos prominentes que permiten localizar rasgos menos prominentes partiendo de hipótesis geométricas. Por ejemplo una pequeña área sobre un área alargada puede corresponderse al

escenario <<cabeza sobre los hombros>> y un par de regiones encontradas en el área facial incrementa la probabilidad de que aquello sea realmente el rostro [6].

- **Análisis de constelación:** Usa un modelo probabilístico que estudia la posición espacial de los rasgos faciales, intentando buscar patrones que se asemejen al rostro.

1.5.5.3. ANÁLISIS DE FORMAS ACTIVAS

Estas técnicas representan la imagen en rasgos de alto nivel para posteriormente interactuar con rasgos locales de la imagen (ojos, brillo) y gradualmente deformarla hasta adaptarla a la forma de rasgos.

- **Serpientes (snakes):** Consiste en usar serpientes para localizar el contorno del rostro. Se inicializa la serpiente ante un rostro inminente. Se va cerrando a los contornos de la imagen asumiendo la forma de la cabeza.

- **Plantillas deformables:** Usar serpientes para encontrar más rasgos faciales además del contorno del rostro.

- **Modelo de distribuciones de puntos:** Se trata de una descripción compacta parametrizada de las formas del rostro basada en estadística.

1.5.5.4. TÉCNICAS BASADAS EN LA IMAGEN

En estas técnicas el objeto de estudio es la imagen misma. El conocimiento previo se incorpora implícitamente en esquemas de entrenamiento. A la representación de la imagen se le aplica algoritmos de entrenamiento y análisis.

- **Métodos basados en subespacios:** Consideran a las imágenes de rostros humanos como un subespacio lineal de un espacio mayor (de todas las imágenes). Al representarlas así pueden utilizarse muchos métodos para tratar los datos.

- **Redes neuronales:** Se han convertido en una técnica popular para el reconocimiento de patrones. Se usan generalmente para la clasificación de imágenes según patrones establecidos. La red se entrena usando un conjunto de imágenes que representan rotros de todo tipo y otro conjunto de imágenes que no

representan rostros, de forma que la red neuronal pueda establecer el criterio adecuada acerca de lo que es y no es un rostro.

- **Métodos estadísticos:** Son métodos basados en la teoría de la información. Consiste en calcular la varianza entre dos funciones probabilísticas de densidad, correspondientes a la probabilidad de que la imagen sea un rostro, y a la probabilidad de que no lo sea.

1.5.6. ALGORITMO DE DETECCIÓN DE ROSTROS VIOLA-JONES

Paul Viola y Michel Jones en su paper 'Robust real-time Object Detection', presentan un trabajo para la realización de un detector de objetos visuales que es capaz de realizar el procesamiento de imágenes en forma extremadamente rápida, logrando además una alta tasa de detección.

Este trabajo estaba motivado en gran parte por la tarea de realizar un detector de rostros eficiente y rápido. El objetivo de este detector es posibilitar la detección de rostros sobre video en tiempo real, sin que para ello sea necesario la utilización de computadoras especialmente potentes [26].

Una forma de implementar un algoritmo de detección de rostros consiste en ir recorriendo la imagen mediante una ventana de un determinado tamaño, la cual puede contener un rostro. Después de recorrer la imagen, la ventana puede ser escalada y repetir el proceso con el objeto de detectar posibles rostros de mayor tamaño que no fueron detectados con tamaños de ventana inferiores.

El algoritmo de detección de rostros propuesto por P. Viola y M. Jones, basado en la utilización de ventanas deslizantes, es sin duda el más utilizado, no sólo en tareas de detección de rostros sino también de detección de diversos objetos en general [27].

1.5.6.1. BASES DEL ALGORITMO

- **Rasgos de clasificación**

Los rasgos de clasificación son las formas geométricas utilizadas por el algoritmo para detectar zonas de una imagen (o ventana) que pueda contener partes de un rostro. Los rasgos de clasificación utilizados por el algoritmo Viola-Jones son estructuras

simples compuestas por dos, tres o cuatro rectángulos grises y blancos, como los mostrados en la Figura 16. Se observa que los rasgos pueden tener 6, 8 o 9 puntos significativos, correspondientes a las esquinas de cada rectángulo.

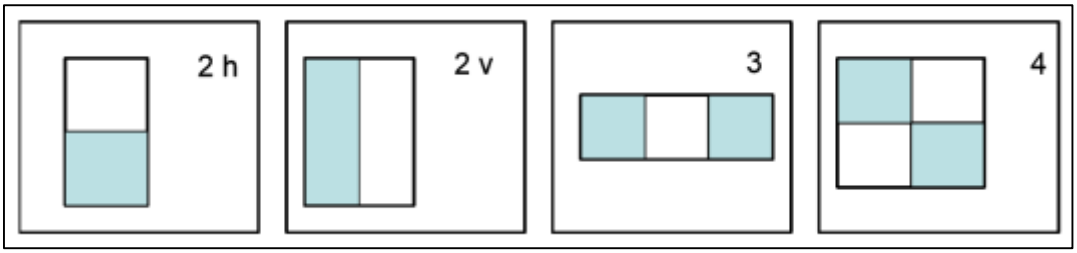

Figura 16. Rasgos de clasificación de 2, 3, y 4 rectángulos utilizados para la detección de rostros. Fuente: [26].

La resolución base con la que trabaja el detector es de 24 por 24 píxeles, el set total de características para este tamaño de imagen es extremadamente alto 45396, mucho mayor que la cantidad de píxeles contenidos en la imagen de 24 por 24 píxeles, 576 (la misma cantidad de píxeles) serían las necesarias para describir completamente la imagen [26].

Estas estructuras simples pueden ser asociadas a partes comunes de un rostro, como las correspondientes a los ojos, la nariz, la frente, el pelo, etc. La Figura 17 ilustra la correspondencia de dos posibles rasgos de clasificación con partes de un rostro [27].

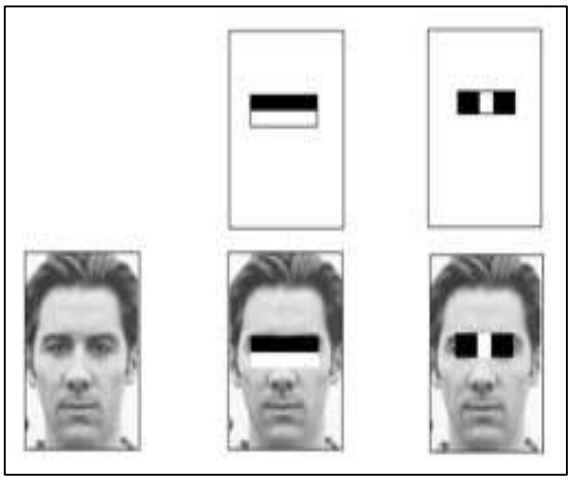

Figura 17. Imagen de dos rasgos de clasificación y su correspondencia con partes de un rostro. Fuente: [27].

Como el número de rasgos que se pueden corresponder con partes de un rostro en una imagen de un determinado tamaño es considerablemente elevado, Viola y Jones utilizaron un algoritmo de entrenamiento (conocido como AdaBoost) sobre un gran

conjunto de imágenes con y sin rostros, para seleccionar aquellos rasgos que mejor distinguen las zonas de una imagen que contienen un rostro [27].

- **Imagen integral**

Una contribución fundamental del algoritmo de Viola-Jones consiste en acelerar el cálculo del valor del rasgo al trabajar no con la imagen original sino con una imagen integral. La imagen integral es una imagen intermedia de la imagen original. Esta no es más que una transformación de la imagen original en donde cada punto de la misma toma el valor de la suma de todos los puntos que están ubicados por encima y a su izquierda. Así, se puede definir una imagen integral ii según la ecuación 1:

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

En donde ii(x, y) es el punto de la imagen integral en las coordenadas (x,y) e i(x,y) es el punto de la imagen original en las mismas coordenadas. La Figura 18a ilustra una imagen original, mientras que la 18b ilustra la imagen integral correspondiente [27].

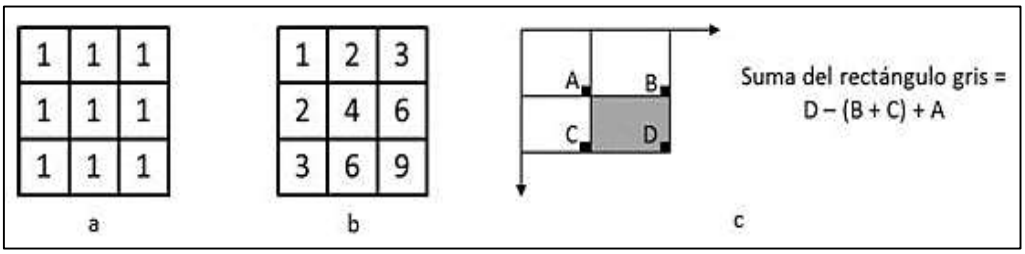

Figura 18. Cálculo de la imagen integral. a) Imagen origina. b) Imagen integral. c) Valor del rectángulo utilizando la imagen integral. Fuente: [27].

La imagen integral se puede calcular en un solo recorrido de la imagen original realizando las operaciones descritas en las ecuaciones 2 y 3:

$$s(x,y) = s(x,y-1) + i(x,y) \qquad (2)$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \qquad (3)$$

En donde s(x,y) es la suma de toda una fila considerando que s(x,-1) = 0 e ii(-1, y) = 0.

A partir de la imagen integral se puede calcular rápidamente la suma de todos los puntos contenidos en un rectángulo cualquiera de la imagen utilizando sólo los cuatro valores asociados a sus esquinas, tal como se ilustra en la Figura 18c. Esta característica permite que el cálculo de la suma de los puntos contenidos en un rectángulo de tamaño arbitrario pueda ser realizado en un tiempo constante utilizando sólo cuatro operaciones, por lo que el cálculo del valor de un rasgo se reduce considerablemente [27].

A partir de la imagen integral es muy simple la determinación del valor de un rasgo. Conociendo los valores de la imagen integral en los puntos significativos de un rasgo y las características del rasgo, la determinación del valor de un rasgo se reduce a simples operaciones de multiplicación (por coeficientes constantes) y suma. La Figura 19 ilustra este proceso, en donde se muestra un rasgo formado por dos rectángulos y se detalla la determinación de su valor. Se observa que los coeficientes por los cuales es necesario multiplicar los valores de la imagen integral de los seis puntos significativos de este rasgo son constantes. Al conjunto de estas constantes (6, 8 ó 9 valores según el rasgo) se denomina vector de pesos del rasgo (W) y es otro de los parámetros que forman parte de un clasificador [27].

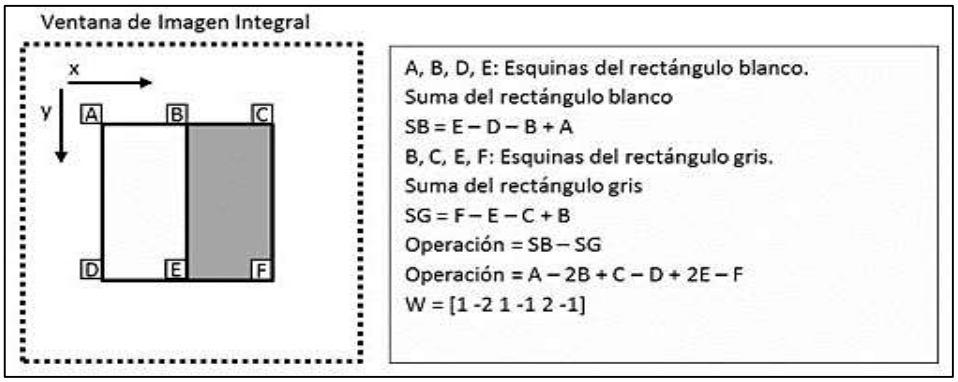

Figura 19. Cálculo del valor de un rasgo y del vector W de pesos del rasgo. Fuente: [27].

- **Organización en cascada de los clasificadores**

La tercera característica del algoritmo Viola-Jones es la utilización de una combinación en cascada de grupos de clasificadores con la finalidad de lograr un altísimo nivel de detección a la vez que una reducción drástica de la necesidad del cálculo y por ende un bajo costo computacional.

En las primeras ventanas se ponen clasificadores sencillos y de bajo coste computacional que rechazan casi todas las sub-ventanas y dejan pasar todos los casos positivos, es decir descartan las ventanas que no contienen rostros, concentrando el esfuerzo computacional en las etapas siguientes en aquellas con mayor probabilidad de contener un rostro.

La Figura 20 muestra una posible distribución de clasificadores en una cascada de cuatro etapas. La cantidad de etapas y de clasificadores en cada etapa dependen de la base de datos de clasificadores utilizada para la el proceso de detección de rostros [27].

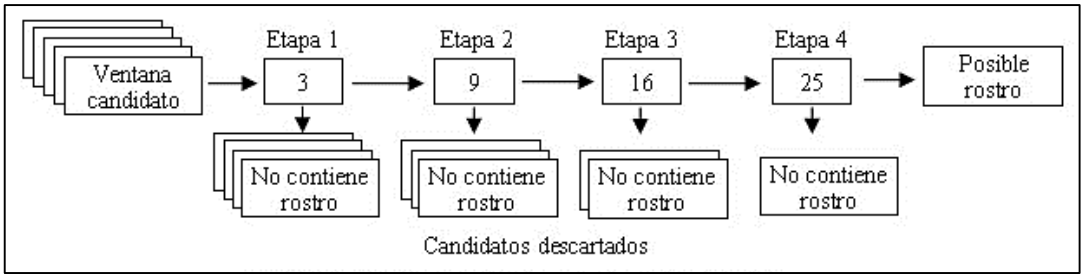

Figura 20. Cascada de clasificadores. Fuente: [27].

En la primera etapa, para cada una de las ventanas de la imagen se calculan sus tres clasificadores y se suman los aportes de los mismos. Si este valor es inferior al umbral de la etapa, se descarta que la ventana contenga un rostro y se procesa la siguiente. Si es superior, la ventana se procesa con los nueve clasificadores de la segunda etapa, y así sucesivamente hasta que la ventana sea descartada en alguna de las etapas o, si las supera todas, sea identificada como un posible rostro. Los valores del umbral de cada etapa se obtienen también del proceso de entrenamiento [27].

1.5.6.2. ENTRENAMIENTO DE LA CASCADA DE CLASIFICADORES

Para el diseño de la cascada es necesario fijar las metas de detección y performance. Lo usual sería lograr niveles de detección entre el 85 y 95%, con tasa de falso positivos muy bajas, del orden de $(10)^{-5}$ o $(10)^{-6}$ [26].

La idea es alcanzar esos registros pero con significativamente menor esfuerzo computacional. Dado un clasificador en cascada entrenado, la tasa F de falsos positivos viene dada por la siguiente expresión:

$$F = \prod_{i=1}^{K} fi \quad (4)$$

Donde fi es la tasa de falsos positivos de la etapa i y K es el número de etapas del clasificador. Por otro lado la tasa de detección D viene dada por la expresión:

$$D = \prod_{i=1}^{K} di \quad (5)$$

Por ejemplo para una tasa de detección de 90% y un detector de 10 etapas, cada etapa debe alcanzar una tasa de detección de 0.99, ya que $0.99^{10} \approx 0.9$.

Si bien esto parece en principio un poco complicado de lograr, vemos que la tarea se simplifica por la poco exigente tasa de falsos positivos F, ya que $6\,x\,10^{-6} \approx 0.3^{10}$, por lo que podemos permitirnos una tasa de falsos positivos de 30% [26].

El número esperado de características a evaluar en un proceso de estos es

$$N = n_0 + \sum_{i=1}^{K} \left(n_i \prod_{j<i} p_j \right) \quad (6)$$

Donde K es el número de etapas o clasificadores, p_j es la tasa de detección del i-ésimo clasificador y n_i es la cantidad de características de i-ésimo clasificador [26].

1.5.6.3. DETECCIÓN DE ROSTROS CON OPENCV

OpenCV (Open Source Computer Vision) es una biblioteca de visión por computadora, iniciado por Intel en 1999. La biblioteca multiplataforma inicia su especialización en el procesamiento de imágenes en tiempo real y cuenta con las implementaciones libres de patentes de los últimos algoritmos de visión por computador.

OpenCV nos facilita la tarea de detectar rostros pues ya cuenta con clasificadores entrenados para esta tarea almacenados en archivos xml, pero en caso de que lo necesitemos permite crear nuestros propios clasificadores, además de contar con todas las funciones necesarias para esta tarea.

OpenCV para detección de rostros tiene la clase cv::CascadeClassifier. La clase CascadeClassifier tiene dos funciones principales que son las que permiten cargar las características de Haar del objeto a detectar, y otra para detectar propiamente dicha. Concretamente, la función que permite cargar las características de Haar es la función load, que necesita como parámetro de entrada el fichero xml que contiene la información generada en un entrenamiento previo. Una vez cargado este fichero, lo siguiente es detectar en una imagen los rostros que haya, para lo cual se usa la función detectMultiScale. Esta función necesita como parámetros de entrada la imagen en la que se van a buscar los rostros, una variable en la que se almacenaran la lista de rostros detectadas, el tamaño al que se escalará la imagen y el tamaño mínimo posible de los rostros a detectar [28].

1.6. RECONOCIMIENTO FACIAL

1.6.1. DEFINICIÓN DE RECONOCIMIENTO FACIAL

El reconocimiento facial es una tarea fácil para los seres humanos. Incluso se ha demostrado que los bebés son capaces de distinguir entre los rostros conocidos. David Hubel y Torsten Wiesel, demostraron que nuestro cerebro ha especializado las células nerviosas y estas responden a las características locales específicas de la escena, tales como líneas, bordes, ángulos o movimiento. Dado que no vemos el mundo como piezas dispersas, nuestra corteza visual debe combinar de alguna manera las diferentes fuentes de información en patrones útiles.

El reconocimiento automático del rostro tiene que ver con la extracción de las características significativas de una imagen, ponerlos en una representación útil y la realización de algún tipo de clasificación en ellos.

Los algoritmos que existen para el reconocimiento facial se pueden clasificar en dos grupos:

- **Métodos holísticos:** Utilizan toda la imagen del rostro como entrada al sistema de reconocimiento, siendo ésta la unidad básica de procesamiento [6].

- **Métodos basados en características locales:** Se extraen características locales, como ojos, nariz, boca, etc. Sus posiciones y estadísticas locales constituyen la entrada al sistema de reconocimiento [6].

También existen métodos híbridos que combinan técnicas holísticas y locales.

1.6.2. MÉTODOS HOLÍSTICOS

1.6.2.1. ANÁLISIS DE COMPONENTE PRINCIPAL: PCA (PRINCIPAL COMPONENT ANALYSIS)

El análisis de componentes principales es un método típico en el análisis de datos multivalentes y tiene como objetivo la reducción de la dimensionalidad de los mismos. Intuitivamente PCA sirve para hallar las causas de la variabilidad de un conjunto de datos y ordenarlas por importancia. Lo que trata de hacer es analizar si dadas c muestras de un conjunto de n valores se puede representar la información por un número menor de variables, construidas como combinaciones lineales de las originales [6].

El funcionamiento de PCA puede resumirse en el siguiente algoritmo:

1. Obtener un conjunto de datos de dimensión *n*.
2. Calcular la media de los datos y restársela a cada uno de ellos, de esta forma se tiene unos datos cuya media es cero.
3. Calcular la matriz de covarianza.
4. Calcular los eigenvectores (vectores propios) y eigenvalores (valores propios) de la matriz de covarianza.
5. Elegir las componentes y formar un vector característico. Se ordenan los eigenvalores de mayor a menor, y se eligen los *p* eigenvectores (p<n) correspondientes a los mayores eigenvalores. Así se tiene un espacio de menor dimensión.
6. Obtener el nuevo conjunto de datos. Los datos originales se multiplican por el vector característico, así se tendrán los datos en términos de los eigenvectores elegidos

PCA es un método general de análisis de datos y se aplica en el reconocimiento facial con alguna variación en el método, dando lugar a eigenfaces.

1.6.2.2. ANÁLISIS DE LA COMPONENTE INDEPENDIENTE: ICA (INDEPENDENT COMPONENT ANALYSIS)

ICA es una generalización del método PCA. El análisis de la componente principal realiza la búsqueda de vectores de proyección en los que las características son más independientes entre sí.

Este método trata de descomponer una señal observada en una combinación lineal de fuentes independientes.

ICA tiene un enfoque estadístico y parte de la hipótesis de que las señales originales son estadísticamente independientes y los procedimientos que lo siguen se basan en propiedades de las fuentes y de las observaciones (mezclas), caracterizadas por sus distribuciones de probabilidad. Se pretende que si las señales originales son estadísticamente independientes, las señales recuperadas también deben de serlo.

Se tiene una matriz de variables independientes (fuentes) : $S = (S1, \dots, Sn)$, y una matriz de observaciones *X* . En esta matriz de observaciones, cada columna es el resultado de un experimento aleatorio, y en cada fila se tiene el valor de una prueba de ese experimento. Como se ha dicho el método ICA trata de descomponer la señal observada en una combinación de fuentes independientes, la matriz de combinación (desconocida) se llamará *A: X = A.S* [6].

Con el algoritmo ICA se busca la matriz de separación W, que cumple: U = W.X = W.A.S, donde U es la estimación máxima de probabilidad de las componentes independientes [6].

Hay dos formas de implementar el método ICA para el reconocimiento de rostros:

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

1.6.2.3. ANÁLISIS DE LA LÍNEA DISCRIMINANTE: LDA (LINEAR DISCRIMINANT ANALYSIS)

Este método es una técnica de aprendizaje supervisado para clasificar datos. La idea central de LDA es obtener una proyección de los datos en un espacio de menor (o incluso igual) dimensión que los datos entrantes, con el fin de que la separación de las clases sea la mayor posible. Es una técnica supervisada ya que para poder buscar esa proyección se debe entrenar el sistema con patrones etiquetados [6].

El escenario de trabajo necesario para aplicar este método al reconocimiento de rostros se basa en que se dispone de un conjunto de rostros de entrenamiento (xi) compuesto por un grupo de personas con distintas expresiones faciales y con diferentes vistas. Por definición todas las instancias de rostros de la misma persona están en una clase (de tamaño Nc), y los rostros de otras personas diferentes pertenecen a distintas clases (habrá c clases), teniendo así el espacio de entrenamiento separado en grupos. Además todas las instancias en el conjunto de entrenamiento deben estar etiquetadas [6].

1.6.2.4. MÉTODOS BASADAS EN KERNELS

Estos métodos son una generalización de los métodos de análisis de componentes (PCA, ICA, LDA). Los métodos basados en Kernels se tienen en cuenta momentos de mayor orden sin tener un costo computacional excesivamente grande.

Se lleva el problema de clasificación a un espacio de mayor dimensión donde las clases sean linealmente separables. Sigue estos pasos:

1. Se mapean los vectores de entrenamiento a través de una función no lineal que lleva los puntos a un espacio de mayor dimensión [6].
2. Se plantea un problema equivalente al problema de PCA, ICA o LDA en dicho espacio [6].
3. Se resuelve el problema equivalente, utilizando el kernel trick, que es una manera simplificada de resolver el problema de PCA, ICA o LDA en el espacio de mayor dimensión [6].

1.6.2.5. PERSECUCIÓN EVOLUTIVA: EP(EVOLUTIONARY PURSUIT)

Este método es un tipo de algoritmo generico que trata de encontrar una base de rostros a través de la rotación de ejes definidos en un espacio blanco PCA adecuado.

El algoritmo EP se utiliza para buscar entre las diferentes rotaciones y vectores base para encontrar un subconjunto de vectores óptimo (que tenga buena precisión en la clasificación y habilidad para generalizar) [6].

1.6.2.6. MÁQUINA DE VECTOR DE SUPORTE: SVM (SUPPORT VECTOR MACHINE)

SVM es un método genérico para resolver problemas de reconocimiento de patrones. Dado un conjunto de puntos en un determinado espacio que pertenecen a dos clases distintas, SVM encuentra el hiperplano que separa la mayor cantidad de puntos de la misma clase del mismo lado. Esto se realiza maximizando la distancia de cada clase al hiperplano de decisión, denominado OSH (Optimun Separating Hyperplane).

Los puntos más cercanos al hiperplano, son los llamados vectores soporte (support vectors) [6].

1.6.3. MÉTODOS BASADOS EN CARACTERÍSTICAS LOCALES

1.6.3.1. CORRESPONDENCIA ENTRE AGRUPACIONES DE GRAFOS ELÁSTICOS: EBGM (ELASTIC BUNCH GRAPH MATCHING)

EBGM es un algoritmo en la visión por computadora para el reconocimiento de objetos o clases de objetos en una imagen basada en una representación gráfica extraído de otras imágenes. Es utilizado en el reconocimiento y análisis de rostro, así como para gestos y otras clases de objetos.

En un sistema de reconocimiento de rostros que utilice este método, las imágenes de los rostros deben ser sometidas a un proceso de normalización en el que se centran los valores de los píxeles en la media de la imagen original para obtener una señal de media nula. También se realiza una normalización geométrica, de forma que las coordenadas de los ojos pasan a tener un valor predeterminado, y se ajustan los valores de los píxeles para tener una señal de media nula y desviación estándar igual a uno. Además se suavizan los bordes de la imagen [6].

Luego es necesario realizar la localización de los puntos principales. Estos puntos definen un grafo como en la figura 21 (conjunto de vértices o nodos unidos por aristas) sobre el rostro cuyos nodos son puntos característicos que se pueden haber definido previamente.

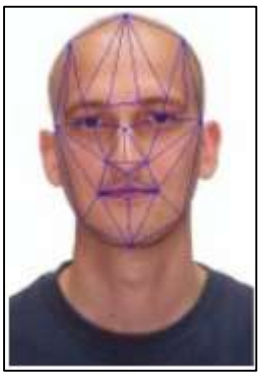

Figura 21. Grafo de puntos principales. Fuente: [6].

1.6.3.2. PATRONES BINARIOS LOCALES: LBP (LOCAL BINARY PATTERN)

LPB es una función que se utiliza para la clasificación de la textura. Es un operador que recorre la imagen y etiqueta los píxeles de la misma mediante un umbral de la diferencia entre el píxel central y sus vecinos, considerando el resultado como un número binario. La concatenación de las etiquetas de los vecinos puede utilizarse como descriptor. Para el reconocimiento facial LPB construye varios descriptores locales que se combinan en un descriptor global.

En el caso de la imagen de un rostro es importante mantener información sobre la relación espacial. Se divide la imagen en varias regiones y se extraen los descriptores de cada región independientemente. Estos descriptores son concatenados para formar un descriptor global del rostro [6].

1.6.3.3. MODELO DE APARIENCIA ACTIVA: AAM (ACTIVE APPEARANCE MODELS)

AAM es un modelo estadístico de la forma y la apariencia en niveles de gris del objeto de interés que se puede generalizar a casi cualquier ejemplo válido de dicho objeto.

Los modelos son entrenados con imágenes de rostros en un rango de puntos de vista. Por ejemplo para cubrir una rotación de 180º se necesitan sólo cinco modelos, para

-90º, -45º, 0º (vista frontal), 45º y 90º. Como los modelos de ±45º y ±90º son el reflejo el uno del otro (asumiendo que los rostros son simétricas), en realidad es suficiente con tres modelos. Utilizando el algoritmo AAM se puede encajar una nueva imagen en cualquiera de los modelos [6].

1.6.3.4. MODELO OCULTO DE MARKOV: HMM (HIDDEN MARKOV MODELS)

HMM (Modelo oculto de Markov) es un conjunto de modelos estadísticos utilizados para caracterizar las propiedades estadísticas de una señal. HMM consiste en dos procesos interrelacionados [6]:

1. Una cadena de Markov subyacente con un número finito de estados, una matriz de probabilidad de transición de estados y una distribución de probabilidad de estados inicial.

2. Un conjunto de densidades de probabilidad asociadas con cada estado.

Cada estado tiene una distribución de probabilidad sobre las posibles señales de salida. Por lo tanto la secuencia de símbolos generados por un HMM da alguna información acerca de la secuencia de estados. El adjetivo “oculto” se refiere a la secuencia de estado a través de la cual el modelo pasa, no a los parámetros del modelo.

1.6.3.5. MÉTODOS 3D

Los métodos de reconocimiento de rostros de dos dimensiones son sensibles a las condiciones de iluminación, a la orientación, a la expresión facial, etc. Estas limitaciones provienen de la limitación de la información que hay en una imagen en dos dimensiones. Esto ha hecho que aumente el uso de datos de rostros en tres dimensiones, ya que proporcionan mayor información de la orientación y las condiciones luminosas [6].

La idea general de los métodos 3D es buscar un modelo general de un rostro que luego debe ajustarse a cada rostro particular. Los rostros se tratan como superficies tridimensionales, hay métodos que se basan en producir gradientes de la superficie y para los que no es necesario reconstruir después la superficie del rostro [6].

1.6.4. EIGENFACES

Eigenface, es el PCA (Análisis de Componentes Principales) aplicado en imágenes faciales sobre un espacio representado por un conjunto de rostros grande [29]. Las características más significativas son conocidas como eigenfaces y estos son los eigenvectores (componentes principales) del conjunto de rostros; aunque no necesariamente, correspondan a rasgos tales como los ojos, oídos o nariz.

El reconocimiento se realiza proyectando una nueva imagen sobre el espacio formado por las "eigenfaces", y clasificándola comparando su posición en ese espacio con las posiciones de los individuos ya conocidos [30].

En términos matemáticos se requiere encontrar los componentes principales de la distribución de rostros, o los eigenvectores de la matriz de covarianza del conjunto de imágenes faciales, tratando una imagen como un punto (vector) en un espacio dimensional infinito. Posteriormente, se ordenan los eigenvectores de acuerdo a una cantidad relativa a la variación entre las imágenes faciales, seleccionando un número finito de estos, los más representativos en peso de acuerdo a su eigenvalor y que contribuyen a recuperar la imagen [29].

Cada uno de los eigenvectores es como un pequeño espectro del rostro original, y recibe el nombre de eigenface. Cada rostro individual se puede representar en términos de una combinación lineal de los eigenfaces. Por lo que, un rostro individual se puede aproximar empleando solo los "mejores" eigenfaces, aquellos que tienen los eigenvalores más largos o pesados, y que más contribuyen en la variación del conjunto de imágenes faciales [29].

La imagen de cada individuo puede ser representada mediante una combinación lineal de las "eigenfaces". Cada rostro puede aproximarse utilizando únicamente las "mejores eigenfaces", es decir, aquellas que tienen los mayores valores propios y por tanto, los de máxima varianza. Estas M "mejores eigenfaces" crean un subespacio M-dimensional de todas las imágenes posibles [30].

Las aproximaciones al reconocimiento facial implican las siguientes operaciones iniciales:

- **Fase de entrenamiento**
  1. Adquisición de la colección de imágenes iniciales ("conjunto de entrenamiento").

  2. Calcular las "eigenfaces" del set de entrenamiento, teniendo en cuenta las M imágenes que corresponden a los mayores valores propios. Estas M imágenes definen el "face space".

  3. Calcular la correspondiente distribución en el espacio M-dimensional para cada individuo, proyectando las imágenes en el "face space".

Una vez realizada la fase de entrenamiento se procede al reconocimiento de nuevos rostros.

- **Fase de test.**
  1. Proyectar la nueva imagen de entrada sobre el "face space" y determinar si éste es un rostro, comprobando si es próxima a este espacio.

  2. Si es un rostro, utilizar un clasificador para determinar la clase más próxima a la que pertenece la nueva imagen.

1.6.4.1. RESOLUCIÓN DE LAS IMÁGENES

Sea F un conjunto de M imágenes o fotografías de tamaño m x n, donde:

$$F = f_1, f_2, \dots, f_m \qquad (7)$$

Las operaciones de cálculo que se proponen en la sección siguiente, se reducen si cada imagen se reordena de manera que el tamaño o resolución de las mismas se lleven a la resolución Nx1 (todas las columnas de la imagen se reagrupan en una sola) donde N = mxn. De esta forma se obtendrá:

$$T = T_1, T_2, \dots, T_M \qquad (8)$$

que es un conjunto de M imágenes de resolución Nx1.

1.6.4.2. CÁLCULO DE EIGENFACES

El rostro promedio,Ψ, del conjunto de fotografías estará dado por:

$$\Psi = \frac{1}{M}\sum_{i=1}^{M}\Gamma_i \qquad (9)$$

Sea:

$$\Phi = T_i - \Psi \qquad (10)$$

la diferencia entre la i-esima imagen o fotografía y el rostro promedio. Entonces la matriz de covarianza estará dada por la relación:

$$C = \frac{1}{M}\sum_{k=1}^{M}\Phi_k\Phi\Gamma_k = AA^T \qquad (11)$$

donde:

$$A = [\Phi_1, \Phi_2, \dots, \Phi_M] \ (12)$$

es una matriz de tamaño NxM. A partir de la matriz de covarianza se pueden obtener los eigenfaces (eigenvectores) y los eigenvalores. Aplicando:

$$AA^T v_i = \mu_i \, \mathrm{v}_i \qquad (13)$$

donde $\mathrm{v}_i$es el vector de eigenfaces y μ_i el conjunto de eigenvalores. Notemos que la resolución de esta operación estará dada por:

$$AA^T = (NxM)^T (NxM) = (MxM) \qquad (14)$$

De esta forma el procedimiento para calcular los eigenvectores y eigenvalores será el siguiente.Consideremos la ecuación:

$$A^T A\mathrm{u}i = \mu_i \, \mathrm{u}_i \qquad (15)$$

De esta ecuación obtener los eigenvectores u_i y eigenvalores μ_i. A continuación ajustar el eigenvector $\mathrm{u}_i$obtenido multiplicandolo por A ya que:

$$A\,A^T(A\mathrm{u}i) = \mu_i \, (A\mathrm{u}i) \qquad (16)$$

donde:

$$v_i = Au_i \qquad (17)$$

nos da el conjunto de eigenfaces buscado.

1.6.4.3. CÓMO UTILIZAR LOS EIGENFACES PARA RECONOCIMIENTO DE ROSTROS

Los Eigenfaces son adecuados para describir imágenes faciales bajo condiciones controladas. Un valor pequeño M' (M'<M donde M' es el número de fotografías y M' ≅ 0.4M) es suficiente para la realización la identificación.

Un nuevo rostro (T) se transforma en sus componentes eigenface (proyectado sobre el "espacio facial") por la operación:

$$\omega_k = (u_k)^T (T - \Psi) \qquad (18)$$

para k = 1,…,M'. Estos componentes, pesos, forman un vector:

$$\Omega^T = [\omega_1, \omega_2 \ldots \omega_{M'}] \qquad (19)$$

que describen la contribución de cada eigenface en la representación del nuevo rostro de entrada. Este vector se puede emplear en un algoritmo de reconocimiento de patrones para determinar a qué número de clase de rostro predefinido pertenece, si existe alguno. El método, simple, para determinar este número de clase es encontrar la clase κ que minimiza la distancia euclidiana:

$$\epsilon_k{}^2 = \| (\Omega - \Omega_k) \|^2 \qquad (20)$$

donde Ω_k es el vector que describe la *k-ésima* clase facial. Diferentes fotografías de una persona se clasifican en una clase Ω_i promdiando los resultados de su representación en eigenfaces. Un rostro se clasifica como perteneciente a la clase *k* cuando el mínimo ϵ_k está por debajo de un umbral seleccionado de θ_c. De otra forma el rostro se clasifica como "desconocido" y opcionalmente se puede emplear para crear una nueva clase facial.

Crear el vector de pesos es equivalente a proyectar sobre la imagen facial original sobre el espacio facial de baja dimensión, las imágenes se proyectarán sobre un vector patrón. La distancia ϵ entre la imagen y el espacio facial estará dada por:

$$\epsilon^2 = \| \Phi - \Phi_f \| \qquad (21)$$

donde

$$\Phi_f = \mathrm{T} - \Psi \qquad (22)$$

y

$$\Phi_f = \sum_{i=1}^{M'} \omega_i u_i \qquad (23)$$

De lo anterior hay cuatro posibilidades para una imagen de entrada y su vector patrón: (1) espacio facial cercano y clase facial cercana, (2) espacio facial cercano pero ninguna clase facial cercana conocida, (3) lejana del espacio facial y clase facial cercana, y (4) lejana del espacio facial y ninguna clase facial cercana. En el primer caso, un individuo es reconocido e identificado. En el segundo caso, un individuo desconocido está presente. Los dos últimos casos indican que la imagen no es un rostro. El caso tres muestra un caso típico de falso positivo que es detectado dada la distancia significativa entre la imagen de entrada y el espacio facial. Es decir, un falso positivo corresponde a una imagen detectada como si fuera un rostro o siendo un rostro como si formara parte de la base de datos. Un falso negativo corresponde a un rostro que no es detectado y que sí forma parte de la base de datos.

CAPÍTULO 2

2. DISEÑO Y DESARROLLO DEL SISTEMA

2.1. ARQUITECTURA DEL SISTEMA

El objetivo de este proyecto es el diseño de un sistema biométrico basado en reconocimiento facial. El sistema utiliza imágenes y por ende se tiene el inconveniente de que se utiliza proyecciones en dos dimensiones (imagen) de un objeto que es tridimensional (rostro humano). Por esta razón se tiene pérdida de información. Pero tenemos como logro una mayor sencillez en los algoritmos necesarios para esta tarea.

Todos los sistemas de reconocimiento facial utilizan la misma secuencia de etapas:

1. Detección del rostro en la imagen.
2. Representación del rostro (rasgos visuales).
3. Clasificación del rostro (etapa de reconocimiento).

El sistema diseñado y desarrollado en esta tesis va a seguir las mismas etapas. En la figura 22 se muestra el diagrama de bloques en el que se representan las etapas de un sistema de reconocimiento facial.

Figura 22. Diagrama de bloques del sistema. Fuente: [La Autora].

- **Preprocesado:** Recibe como entrada una imagen desde una webcam y se le aplica ciertas técnicas para mejorar la imagen.
- **Detección:** Recibe la imagen preprocesada y localiza el rostro detectado en la imagen, si existe.
- **Extracción de características:** Recibe el rostro detectado y devuelve los eigenvectores de características.
- **Clasificación:** Recibe como entrada los eigenvectores de características y devuelve el rostro de la base de datos a la que más se aparece.

En la figura 23 se observa que el sistema de reconocimiento facial está formado por 3 etapas, esto con la finalidad de hacer más sencillo el desarrollo y optimizar los resultados, estas etapas son: preprocesado, la detección y el reconocimiento que serán vistos a fondo más adelante.

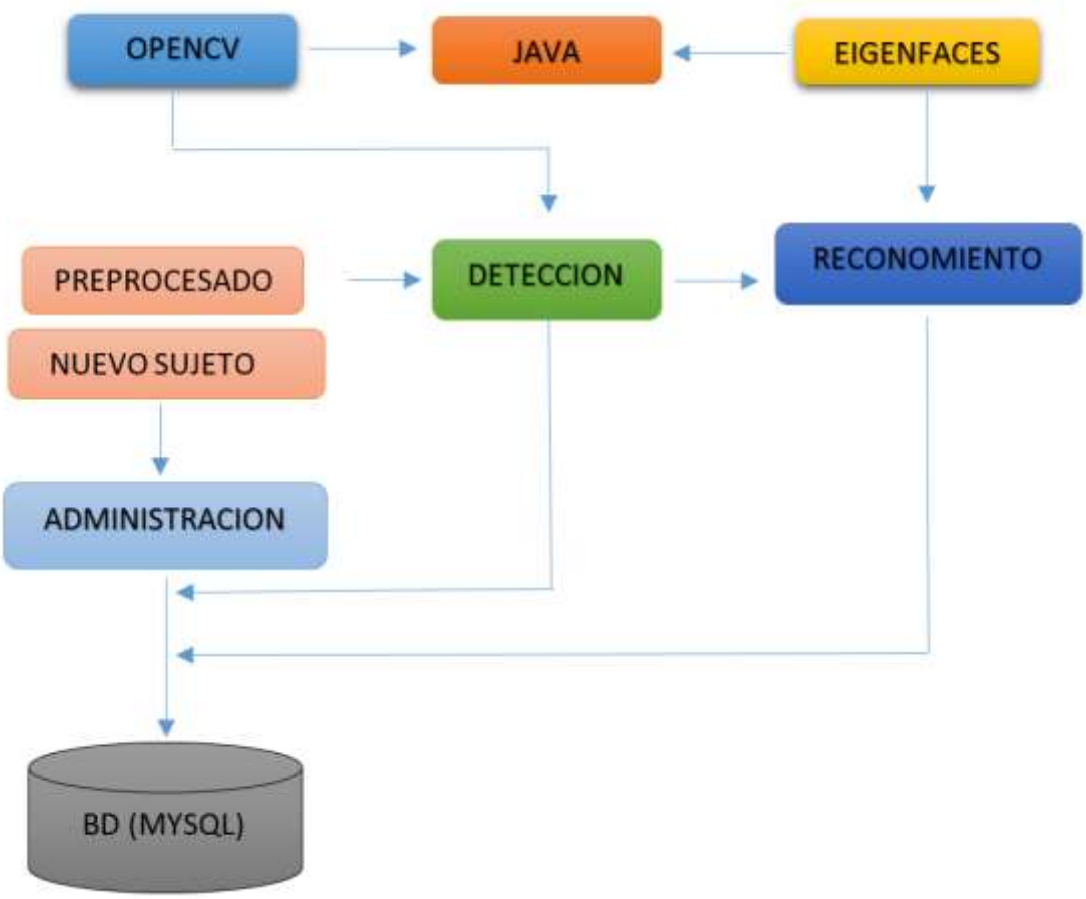

Figura 23. Diagrama general del sistema de reconocimiento facial. Fuente: [La Autora].

2.1.1. PREPROCESADO DE LA IMAGEN

El preprocesado de la imagen tiene como objetivo mejorar las propiedades de la imagen para facilitar las etapas de detección y reconocimiento.

2.1.1.1. PREPROCESADO PARA LA DETECCIÓN

El preprocesado en la detección se utilizó para cambiar una imagen RGB a escala de grises y cambiar la iluminación de la misma.

1. Convertir la imagen a escala de grises es necesario para el correcto funcionamiento del algoritmo de detección creado por Viola-Jones.

2. Si la iluminación no es la adecuada se la adapta para lograr un mejor resultado en la detección.

2.1.1.2. PREPROCESADO PARA EL RECONOCIMIENTO

El preprocesado que se hizo para el reconocimiento fue el siguiente: escala de grises, manipulación del contraste y normalizar el tamaño de la imagen.

1. Zona del rostro utilizado: el rostro se recorta de manera que no aparezcan zonas del fondo de la imagen ni zonas de cabello, cuyo aspecto es muy variable y perjudicaría la robustez del sistema. Es por esta razón que el tamaño de la imagen es de 125 x 150 pixeles. Para esto usamos la siguiente línea de código:

```
image = image.getScaledInstance(125, 150, Image.SCALE_SMOOTH);
```

2. La imagen a escala de grises es con el propósito de hacer rápido y más preciso el reconocimiento. Está representada la imagen por 8 bits, esto hace que el valor mínimo de los pixeles pase a 0 y el máximo a 255. Esta etapa debe realizarse después del recorte del rostro. Para esta tarea tenemos la siguiente línea de código:

```
BufferedImage bi =
utilC.escalaGrisesBufferedImage(utilC.convertirMatABufferedImage(imagen));
```

3. Aumentar el contraste de la imagen se hace con el objetivo de que las condiciones de iluminación no afecten el reconocimiento. Logramos esto con la siguiente línea de código:

```
mage = uc.contrastePhoto(image, 10)
```

2.1.2. DETECCIÓN DEL ROSTRO

La fase de la detección del rostro se la realizó con el algoritmo Viola-Jones pues permite la detección robusta y en tiempo real. Además es un algoritmo con gran rapidez porque la clasificación se realiza mediante características en vez de pixel a pixel.

Este algoritmo tiene 5 etapas bien definidas como se observa en la figura 24 y que son explicadas a continuación:

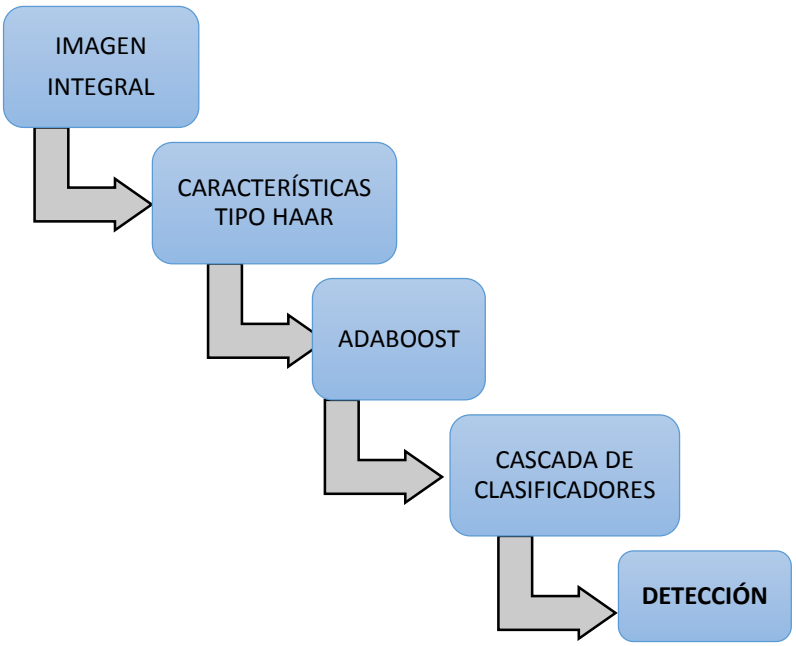

Figura 24. Proceso de detección. Fuente: [La Autora].

- **Imagen integral:** Es una representación de la imagen original y se la obtiene realizando operaciones por pixel. Esta imagen posibilita realizar una rápida evaluación de las características.

- **Características tipo Haar (Haar-like-features):** Permite obtener información de una zona específica mediante una operación aritmética. El valor de la característica está dado por la resta entre el valor de una o más subzonas dentro de la zona analizada, es decir las resta entre las subzonas grises y las zonas blancas de la imagen.

- **AdaBoost:** Es un algoritmo clasificador eficiente para seleccionar grupos de características, que serán las que se utilice para llevar a cabo la detección. Elige entre un gran conjunto, las características de Haar, para luego seleccionar las características que se ajusta mejor, para que clasifique correctamente.

- **Cascada de clasificadores:** Son clasificadores fuertes donde cada etapa corresponde a un clasificador y está entrenada con todos los ejemplos que la etapa anterior no clasificó correctamente. En la etapa de entrenamiento, cada etapa entrena con un conjunto de características. Las primeras etapas se encargan de descartar sub-imágenes que son muy diferentes de una cara y las últimas etapas en cambio pueden rechazar ejemplos más complejos (pelota).

2.1.2.1. IMÁGENES DE ENTRADA

Para esta etapa se consideran como imágenes de entrada las imágenes que son capturadas por la webcam para realizar el reconocimiento. En el sistema tenemos dos tipos de imágenes: las primeras son las imágenes que forman la base de datos y las segundas son las imágenes de prueba que deben ser comparadas con las imágenes de la base de datos. Las imágenes de la base de datos permiten verificar que el usuario es reconocido por el sistema, por tanto, se espera que en la fase de pruebas se obtenga un 100% de aciertos en el reconocimiento sin embargo, esto no siempre es así como se observará en la sección de resultados del capítulo siguiente.

2.1.2.2. LOCALIZACIÓN FACIAL

Dada una imagen de entrada cómo saber si existe, o no, un rostro en esa imagen. Para esto el algoritmo Viola-Jones utiliza grupos de características simples. Las características son evaluaciones de la intensidad de conjuntos de píxeles.

2.1.3. RECONOCIMIENTO

Para el reconocimiento de rostros se utilizó el algoritmo Eigenfaces. Este es un algoritmo matemático usado en problemas de visión por computadora para el reconocimiento de rostros basándose en los vectores propios de una imagen realizando así una comparación entre las imágenes de entrada y una base de datos previamente analizada.

El algoritmo ve a la imagen como un vector, las filas de las imágenes forman un conjunto. Todos los rostros tienen dos ojos, una nariz y una boca, por esta razón existen similitudes y estas se representan en el espacio vectorial, pues tiene posiciones iguales. Para determinar las propiedades características de la imagen utilizamos PCA (Análisis de Componentes principales) que es un método óptimo para reducir el número de dimensiones necesarias para representar la imagen como un conjunto de vectores. Finalmente obtenemos una matriz de covarianza.

Una vez que se encuentran cargadas todas las imágenes de referencia o entrenamiento el algoritmo tiene las siguientes etapas o fases (figura 25):

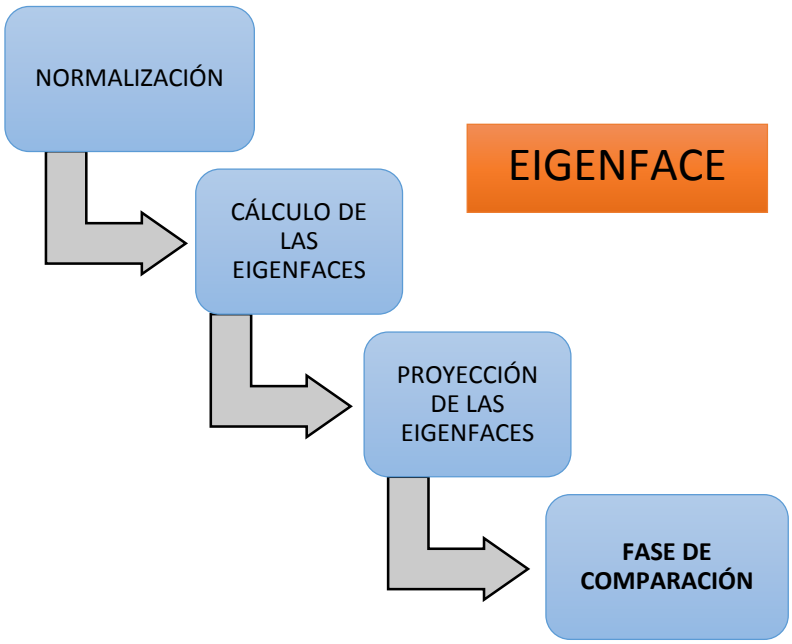

Figura 25. Proceso de reconocimiento. Fuente: [La Autora].

- **Normalización:** Se hace esto para compensar los factores que me permiten determinar las similitudes de dos rostros, como es el tamaño y la rotación.

- **Cálculo de Eigenfaces:** En esta fase se realiza el PCA (Análisis de Componentes Principales) con el que se extraen las Eigenfaces. El procedimiento de cálculo de las eigenfaces que se realiza es el siguiente:

    1. Obtener un conjunto de entrenamiento de imágenes $I_1, I_2, \dots, I_M$. Las de muestra deben ser centradas y escaladas a una misma resolución.

    2. Representar cada imagen I_i de resolución NxN como un vector columna Γ_i de tamaño N^2 como se observa en la figura 26. De ahora en adelante toda imagen será considerada como un vector.

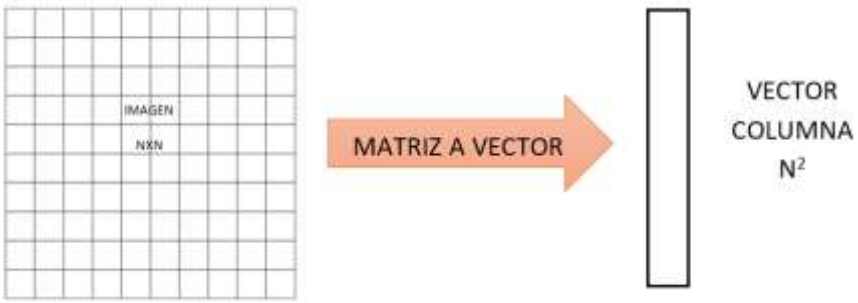

Figura 26. Representación de una imagen matriz como vector. Fuente: [La Autora].

3. Calcular la media Ψ.

$$\Psi = \frac{1}{M}\sum_{i=1}^{M} T_i \quad (vector\ N^2\ x1) \quad (9)$$

4. Restar la imagen media de cada imagen del conjunto.

$$\Phi_i = T_i - \Psi \quad \forall_i = 1 \dots M\ (vector\ N^2\ x1) \quad (10)$$

Donde Φ_i es una imagen normalizada con respecto a la imagen media Ψ.

5. Calcular la matriz de covarianza C.

$$C = \frac{1}{M}\sum_{i=1}^{M} \Phi_i \Phi T_i = AA^T \ \ (Matriz\ N^2 x\ N^2) \quad (11)$$

donde

$$A = [\Phi_1, \Phi_2, \dots, \Phi_M]\ (matriz\ N^2\ x\ M)$$

6. Calcular los vectores propios u_i de $C = AA^T$. Como la matriz de covarianza *C* es de dimensión $N^2 x\ N^2$, determinar los N^2 vectores propios. Hay que recordar que los vectores obtenidos deben ser normalizados.
7. De los *M* vectores propios calculados en el paso anterior, escoger solo *K* vectores propios correspondientes a los *K* más altos valores propios. Los valores propios escogidos son las eigenfaces que servirán para el reconocimiento.
8. Representar cada imagen del conjunto de entrenamiento en el "espacio de rostros" proyectando dicha imagen en cada una de las *K* eigenfaces. Para lograrlo, es conveniente representar las eigenfaces en una matriz u de dimensión $N^2 x\ K$ de modo que sus columnas sean las eigenfaces u_i. Entonces para obtener los pesos del conjunto de entrenamiento se usa la ecuación:

$$\Omega = u^T A \quad (24)$$

Donde Ω es una matriz de dimensión $K\ x\ M$ cuyas columnas contienen los *K* coeficientes de la proyección de cada imagen del conjunto de entrenamiento; es decir, que es el conjunto de imágenes de muestra (de dimensión N^2), ahora representados en una nueva dimensión *K*.

- **Proyección de las Eigenfaces:** Las Eigenfaces forman un conjunto ortogonal, por lo tanto para hallar la proyección de una imagen sobre ellas basta con realizar el producto escalar de la imagen sobre cada una de las Eigenfaces. Esto se conoce como entrenamiento de Eigenfaces.

- **Fase de comparación y decisión:** Se determina qué imagen del conjunto de entrenamiento es más parecida a la imagen capturada, a partir de la representación obtenida del Eigenfaces. Para ello se compara el vector formado por las proyecciones de la imagen de prueba sobre las eigenfaces con cada uno de los vectores. El criterio que se utiliza es la menor distancia euclidiana; esto es, mientras más aproximada a cero es la distancia euclidiana más acertado es el reconocimiento. Como se muestra en la siguiente la tabla:

Tabla 4. Eigenfaces de una imagen.

EIGENFACES	DISTANCIA
Eigenface 1 	0.8836845311480976
Eigenface 2 	0.42438538643308304

Fuente: [La Autora].

De los valores mostrados en la tabla 4 el Eigenface 2 es el que más se aproxima a 0 lo que quiere decir, que es la que tiene mayor similitud a la imagen de comparación.

2.2. BASE DE DATOS

En este proyecto se utilizaran 2 bases de datos: una de imágenes para comparar con los rostros capturados y otra que tendrá todos los registros de cada uno de los usuarios.

2.2.1. BASE DE DATOS DE IMÁGENES

Esta base de datos deberá estar formada por imágenes de los usuarios, 3 imágenes de cada uno para un mejor reconocimiento. El nombre de las imágenes para cada individuo es único.

- Las imágenes deberán estar tomadas en un ambiente similar, todas las personas se encuentran a una distancia prudente de la cámara, el fondo es plano, las variaciones de iluminación deberán ser mínimas y todas las imágenes del mismo tamaño (125 x 150 pixeles). En cada una de las imágenes solo debe existir una cara que ocupe la mayor parte de la imagen y esté centrada en ella. Las diferencias entre las imágenes de un mismo individuo son mínimas, se reduce a cambios en la expresión facial. Además el formato de las imágenes son PNG pues tienen una alta compresión y funciona muy bien para las imágenes con colores planos y bordes bien definidos.

2.2.2. BASE DE DATOS DE REGISTROS DEL SISTEMA

Un usuario es toda persona que tiene contacto con el sistema y es el encargado de realizar todos los registros en la base de datos. En el presente sistema la base de datos contiene lo siguiente:

- Datos del empleado: nombre, apellido, número de cédula, teléfono, dirección domiciliaria y el cargo que ocupa en la empresa.
- Datos del usuario: nombre de usuario, password y cargo.
- Horarios del empleado: Hora de entrada, hora de salida y tipo de horario.
- Registro de horas: Se encuentran todas las horas de trabajo registradas de cada uno de los empleados.

2.3. DIAGRAMAS DE LA METODOLOGÍA DE DESARROLLO XP

2.3.1. DIAGRAMA DE DOMINIO

Es el diagrama inicial, en el cual se dispone la estructura del sistema y es el punto de partida para el desarrollo de las demás etapas.

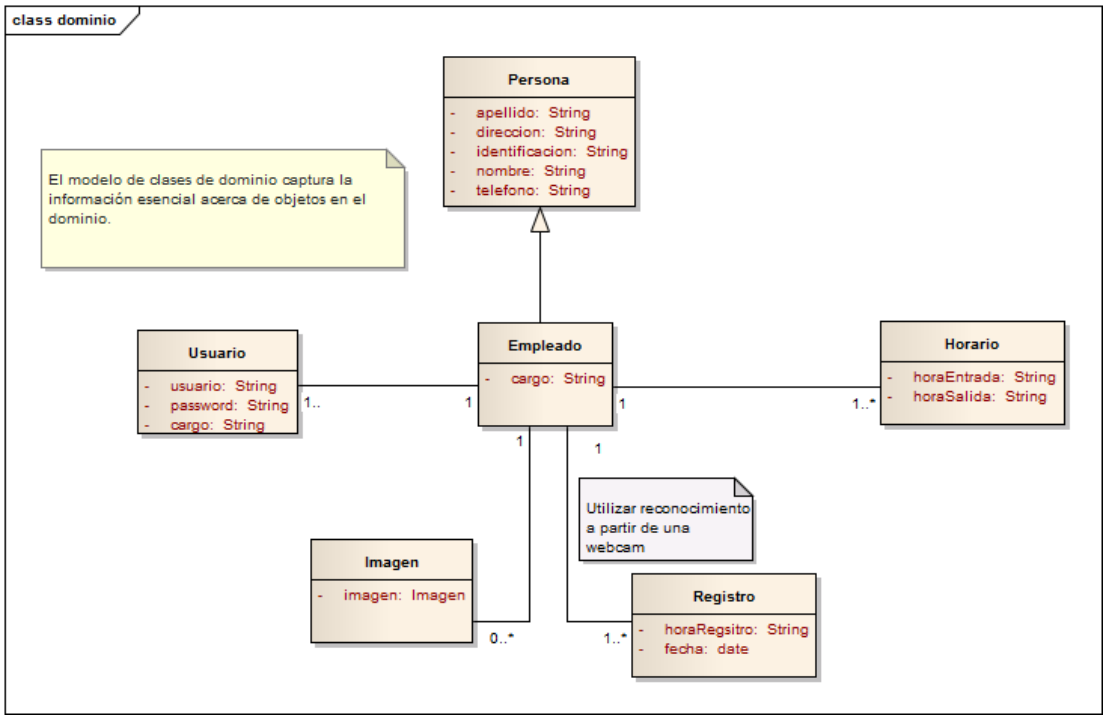

Figura 27. Diagrama de dominio del sistema. Fuente: [La Autora].

2.3.2. DIAGRAMA DE CLASES POR PAQUETES

En este diagrama están todas las clases, atributos, métodos y relaciones que se han desarrollado en este sistema, siguiendo la arquitectura Modelo Vista Controlador (MVC).

- **Paquete reco.dominio**

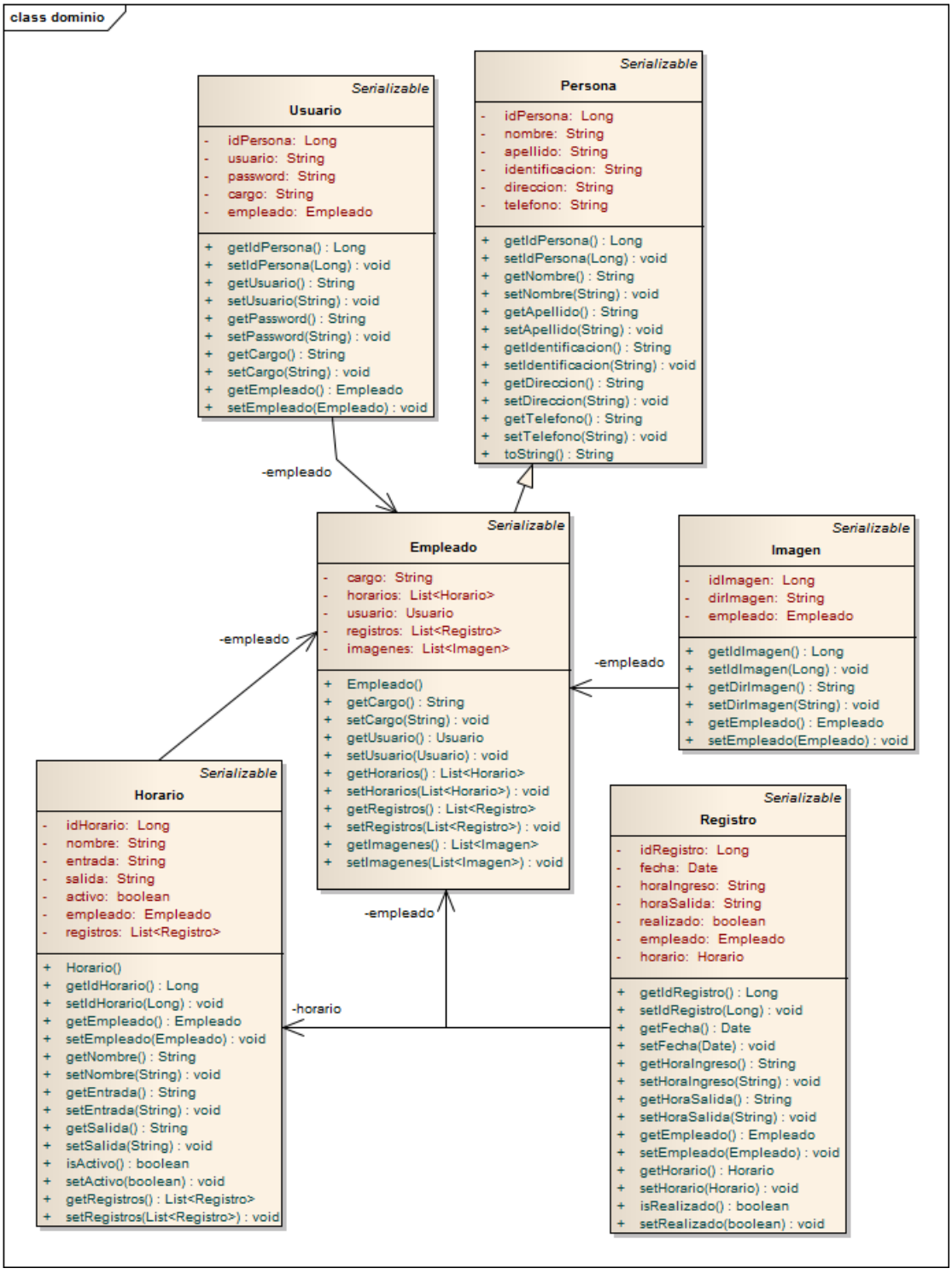

Figura 28. Dominio de clases. Fuente: [La Autora].

- **Paquete reco.vista**

Se muestra la información que se tiene como salida del sistema, pues es la presentación del modelo.

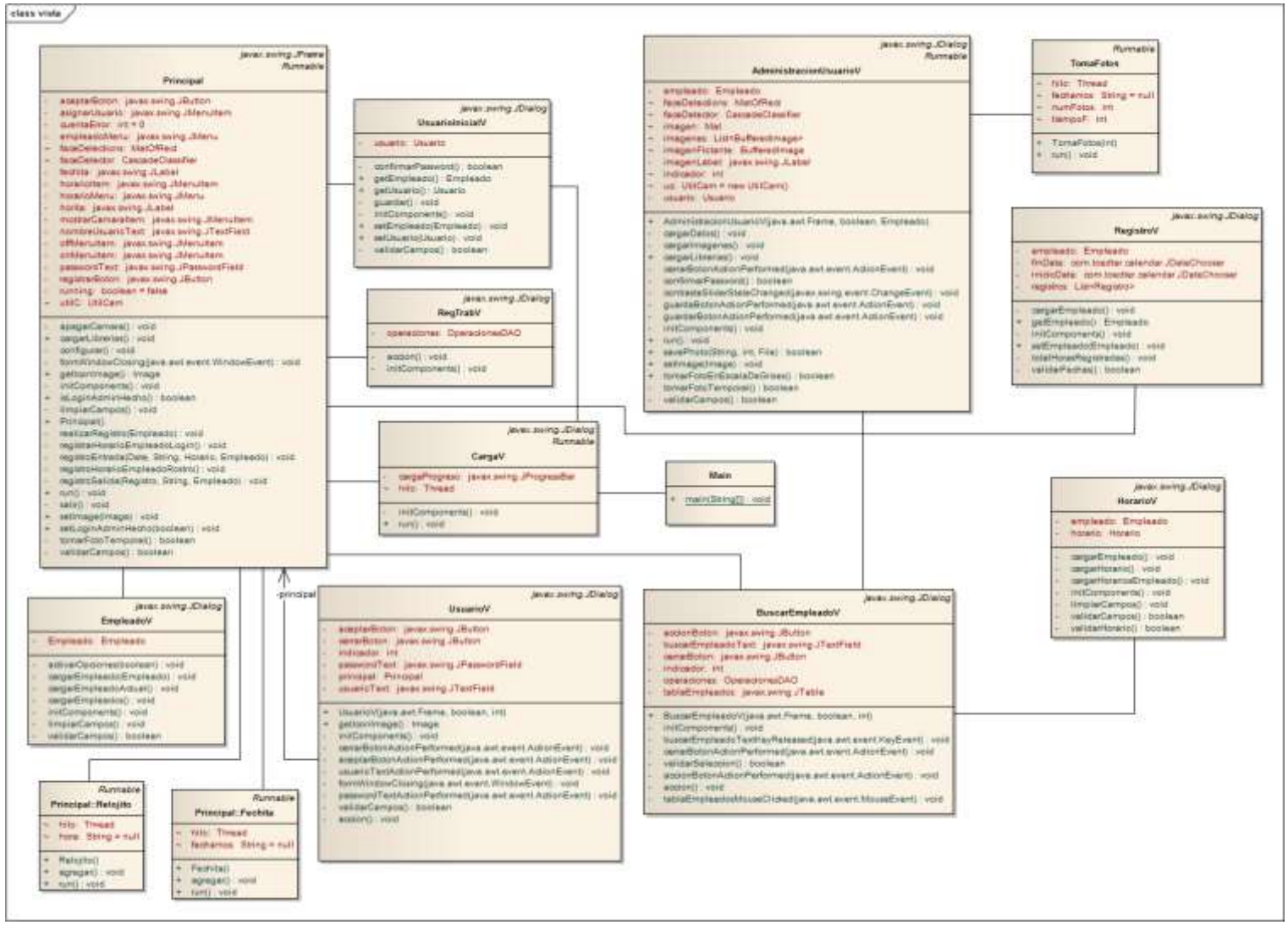

Figura 29. Vista del sistema. Fuente: [La Autora].

- **Paquete reco.ucc.utilidades (Controlador)**

El controlador hace de intermedio entre la vista y el modelo. Aquí se encuentran todas las operaciones del sistema.

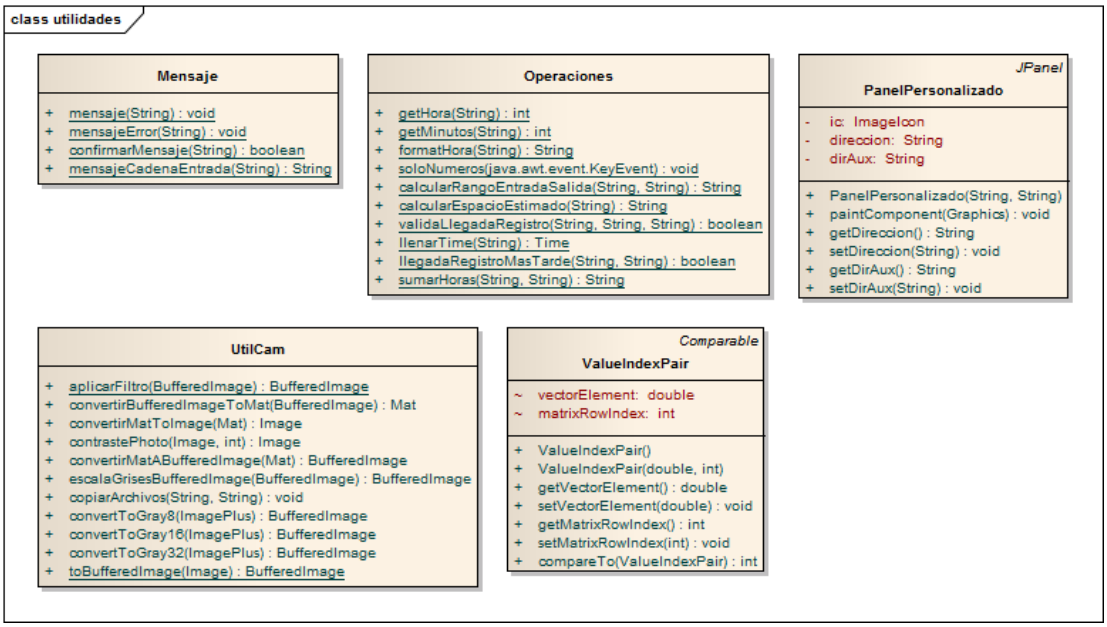

Figura 30. Diagrama del controlador del sistema. Fuente: [La Autora].

- **Paquete reco.dao**

Objeto de Acceso a Datos (DAO, Data Access Object) suministra una interfaz común entre la aplicación y la base de datos.

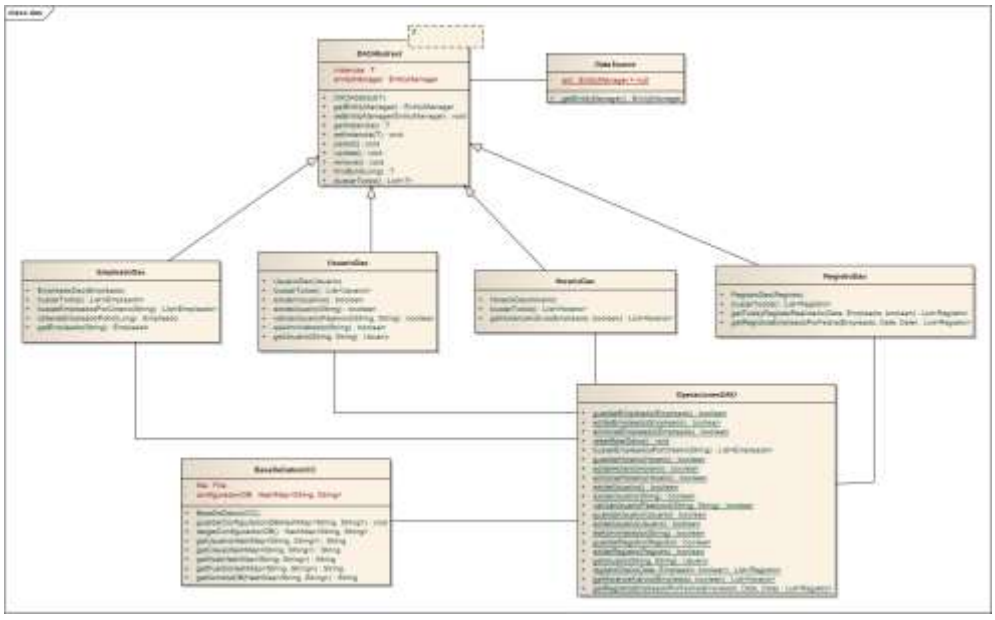

Figura 31. Objeto de Acceso a Datos del sistema de reconocimiento facial. Fuente: [La Autora].

2.3.3. DIAGRAMA DE COMPONENTES

Así como se muestra en la figura 32, están divididas las diferentes etapas del modelo vista controlador, para una mejor organización y fácil accesibilidad a cada uno de ellos.

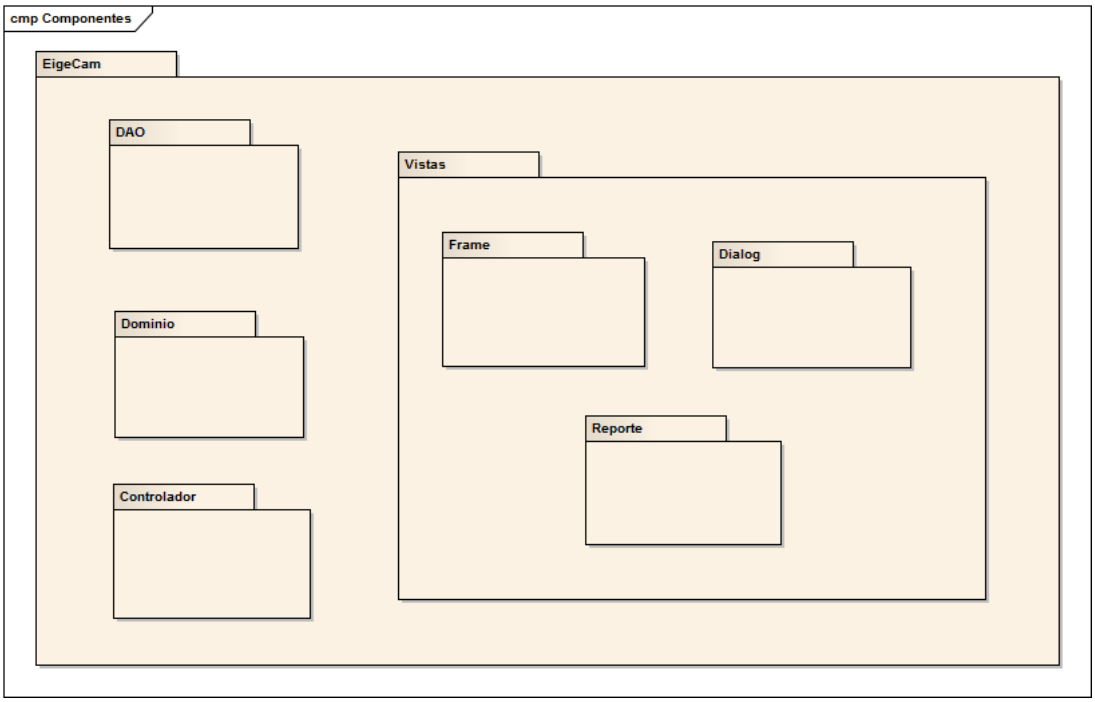

Figura 32. Diagrama de paquetes del sistema. Fuente [La Autora].

2.3.4. MODELAMIENTO DE LA ARQUITECTURA

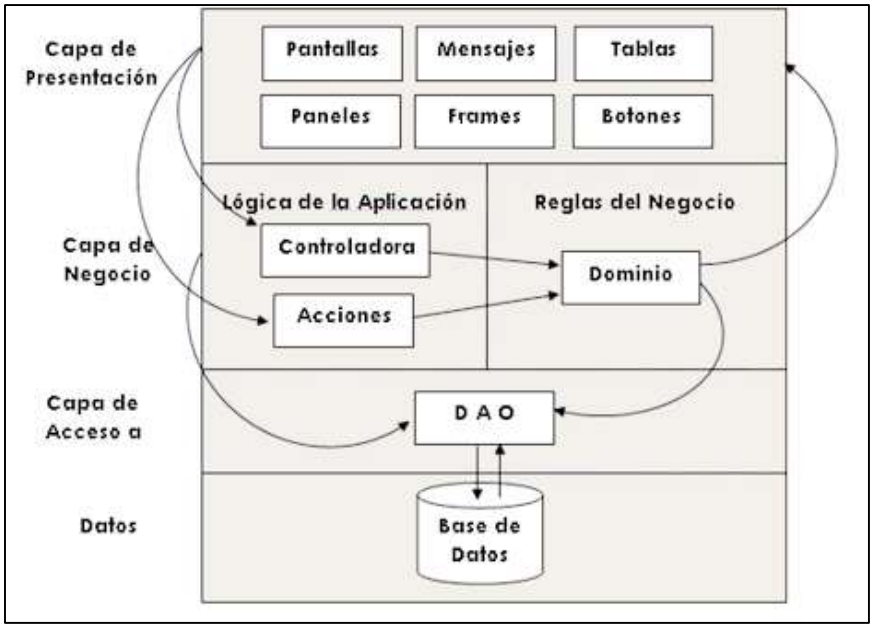

Figura 33. Modelamiento de la arquitectura del sistema. Fuente: [La Autora].

2.4. HERRAMIENTAS UTILIZADAS

Para el desarrollo del sistema biométrico basado en reconocimiento facial fue necesario hacer uso de tres herramientas:

1. **Java:** El lenguaje que se utilizó es Java, considerando las siguientes características básicas requeridas:
    - Programación de alto nivel.
    - Tener un lenguaje potente para el procesamiento matemático y gráfico.
    - Orientado a objetos para dar flexibilidad a las distintas etapas del procesamiento de imágenes y visualización de los resultados.

Para determinar el IDE (ambiente de desarrollo) más adecuado, se analizaron algunas alternativas especialmente de código abierto, descartando el uso de software dependientes de licencias pagadas.

El IDE seleccionado es NetBeans IDE 8.0 por las siguientes razones:

- Es uno de los IDE de código abierto que posee más años de existencia.
- Cuenta con versiones que permite suponer un nivel de madurez.
- La organización de su presentación es más amigable e intuitiva para el desarrollo.
- Su adaptabilidad a nuevas tecnologías lo hace un IDE potente.

2. **OpenCV:** Para el proceso de detección se utilizó Opencv 2.4.8. porque es una librería Java de código abierto para desarrollo de visión artificial. Las razones que justifican esta elección son las siguientes:
    - Es compatible con Windows, Linux, Mac OS, iOS y Android.
    - Fue diseñado para la eficiencia computacional y con un fuerte enfoque en las aplicaciones en tiempo real.
    - Posee un alto grado de madurez y tiene constante actualización.

3. **MySQL:** Para guardar los datos personales y registro de usuarios se optó por la base de datos MySQL por los siguientes características:
    - Un servidor de base de datos muy rápido, multi-usuario y robusto.
    - Tener una licencia GPL (General Public Licence).
    - Ser compatible con Java.

CAPITULO 3

3. PRUEBAS Y RESULTADOS

Una vez terminado el desarrollo del sistema, que finalizó con la producción de una aplicación de software llamada EigeCam 1.0, se procede a la realización de un conjunto de pruebas de reconocimiento para determinar la eficiencia del sistema.

En este capítulo se van a mostrar y analizar los resultados obtenidos en las pruebas del sistema tanto en la fase de detección como en la fase de reconocimiento.

3.1. INICIALIZACIÓN DE LA BASE DE DATOS

Para hacer reconocimiento de rostros se requiere de la bases de datos que permita la identificación con la imagen de entrada. En esta sección se describen las pruebas necesarias para generar las bases de datos del sistema de reconocimiento de rostros.

3.2. FASE DE DETECCIÓN

En capítulos anteriores se detalló la fase de detección de rostros dentro de una imagen. Aquí se van a mostrar los resultados obtenidos en el sistema.

- **Condiciones ideales**

Las condiciones ideales para la detección son:

- Adecuada iluminación.
- El rostro esté centrado a la cámara.
- El fondo no debe tener irregularidades.
- Interiores

Figura 34. Rostro detectado por el sistema. Fuente: [La Autora].

Figura 35. Dos rostros detectados por el sistema. Fuente: [La Autora].

Tabla 5. Eficiencia de la detección del sistema usando una cámara de 1.0 MP.

RESOLUCIÓN DE CÁMARA	# DE ROSTROS	# DE ROSTROS DETECTADOS	EFICIENCIA
1.0 MP	0	0	100%
	1	1	100%
	2	2	100%
	3	3	100%

Fuente: [La Autora]

Tabla 6. Eficiencia de la detección del sistema usando una cámara de 2.0 MP.

RESOLUCIÓN DE CÁMARA	# DE ROSTROS	# DE ROSTROS DETECTADOS	EFICIENCIA
2.0 MP	0	0	100%
	1	1	100%
	2	2	100%
	3	3	100%

Fuente: [La Autora].

$$Eficiencia\ de\ la\ deteción = \frac{\#\ de\ rostros}{\#\ de\ caras\ detectadas} x\ 100$$

$$Eficiencia\ de\ la\ deteción = \frac{3}{3} x\ 100 = 100\%$$

En la tabla 5 y 6 se puede apreciar la eficiencia de la detección del sistema usando una cámara de 1.0 y 2.0 MP obteniendo en ambas tablas unas eficiencia del 100% esto en condiciones ideales.

- **Condiciones extremas**

Las condiciones extremas para la detección son:

- Mala iluminación.
- El fondo es irregular.
- Exteriores.

Tabla 7. Eficiencia del sistema en condiciones extremas.

RESOLUCIÓN DE LA CÁMARA	# DE ROSTROS	# DE ROSTROS DETECTADOS	EFICIENCIA
1.0 MP	0	1	75%
	1	3	
	2	1	
	3	3	
2.0 MP	0	2	57.14%
	1	2	
	3	3	

Fuente: [La Autora].

$$Eficiencia\ de\ la\ deteción = \frac{6}{8} x\ 100 = 75\%$$

$$Eficiencia\ de\ la\ deteción = \frac{4}{7} x\ 100 = 57.14\%$$

La eficiencia de la detección en condiciones extremas, como era de esperarse es inferior a la eficiencia en condiciones ideales; 75% con la cámara de 1.0 MP y 57.14% con 2.0 MP.

3.3. FASE DE RECONOCIMIENTO

Las pruebas de reconocimiento se las realizó con 25 individuos con 3 imágenes de cada uno obteniendo una base de datos de 75 imágenes. En este caso las imágenes han sido tomadas en un ambiente controlado que contienen un rostro centrado en la imagen y ocupa la mayor parte de esta como se observa en la figura 36.

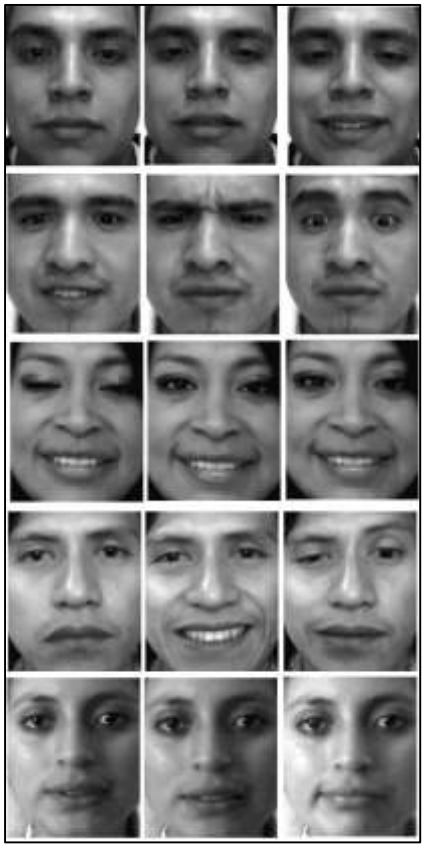

Figura 36. Algunas imágenes de la base de datos. Fuente: [La Autora].

Cabe mencionar que mediante pruebas experimentales se determinó que el rango de similitud máximo es 0.59 (distancia euclidiana), si es menor o igual la similitud es más

aproximada. El reconocimiento de rostros fue realizado para las condiciones físicas siguientes:

- **Condiciones ideales:** son las mismas condiciones que se usó para la detección.

Tabla 8. Eficiencia del reconocimiento facial en condiciones ideales.

# DE ROSTROS	# ROSTROS RECONOCIDOS	EFICIENCIA
25	23	92%

Fuente: [La Autora].

- **Condiciones extremas:** son las mismas que se usó para la detección de rostros.

Tabla 9. Eficiencia de reconocimiento facial en condiciones extremas.

# DE ROSTROS	# ROSTROS RECONOCIDOS	EFICIENCIA
25	7	28%

Fuente: [La Autora].

En las tablas 8 y 9 se tiene la eficiencia en condiciones ideales y extremas, en la cuales se observa que en las mejores condiciones del sistema se tiene una eficiencia del 92%.

3.3.1. TIEMPO DE RECONOCIMIENTO

El tiempo que tarda en hacer el reconocimiento desde que se da click en el botón registrar hasta que reconoce o no al individuo es 0.15 segundos. El tiempo de reconocimiento puede aumentar como disminuir, pues está directamente relacionado al número de imágenes que existe en la base de datos.

3.4. RENDIMIENTO

El rendimiento del sistema está calculado en condiciones ideales:

$$Rendimiento = \frac{\#\ veces\ que\ fall\acute{o}}{\#\ veces\ intent\acute{o}\ identificarse}$$

$$Rendimiento = \frac{19}{20} = 95\%$$

3.5. PARÁMETROS BIOMÉTRICOS

- **Verdaderos Positivos (VP):** Rostro SI está en el sistema y SI es identificado por el programa.

Tabla 10. Verdaderos positivos en el reconocimiento.

# DE ROSTROS	VP	PRESICIÓN
25	24	96%

Fuente: [La Autora].

- **Verdaderos Negativos (VN):** Rostro NO está en el sistema y NO es identificado por el programa.

Tabla 11. Verdaderos negativos en el reconocimiento facial.

# DE ROSTROS	VP	PRESICIÓN
25	3	12%

Fuente: [La Autora].

- **Falsos Positivos (FP):** Rostro No está en el sistema y SI es identificado.

Tabla 12. Falsos Positivos en el reconocimiento.

# DE ROSTROS	FP	PRESICIÓN
10	1	10%

Fuente: [La Autora].

- **Falsos Negativos (FN):** Rostro SI está en el sistema y NO es identificado por el programa.

Tabla 13. Falsos Negativos en el reconocimiento.

# DE ROSTROS	FP	PRESICIÓN
25	2	8%

Fuente: [La Autora].

Una vez obtenidas las tablas de parámetros biométricos procedemos a medir la efectividad del sistema biométrico con la siguiente fórmula:

$$Efectividad = \frac{FN\ (Falsos\ Negativos)}{FP(Falsos\ Positivos)}$$

$$Efectividad = \frac{8}{10} x100 = 80\%$$

3.6. CONSUMO DEL PROCESADOR

Al ejecutar el programa el consumo de recursos del computador son los describe en la tabla 14 y se puede apreciar en la figura 37.

Tabla 14. Consumo de recursos del computador.

Memoria RAM	200 MB
# de Subprocesos	48
CPU (Procesador)	34%
Disco Duro	300 KB/s

Fuente: [La Autora].

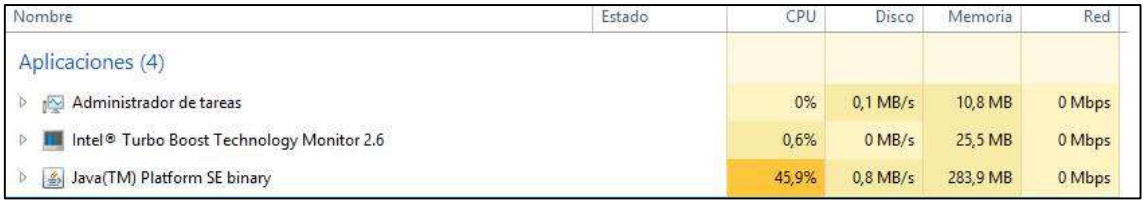

Figura 37. Consumo del procesador. Fuente: [La Autora].

3.7. TIEMPO DE INICIAR LA APLICACIÓN

El tiempo que se demora el sistema en cargar todas las librerías y componentes así, como la base de datos necesaria para inicializar la aplicación oscila entre 2 a 2.5 segundos, esto depende del tamaño de la base de datos.

3.8. REQUERIMIENTOS DEL SISTEMA

Esta aplicación para su eficiente funcionamiento necesita cumplir con los siguientes requerimientos del sistema:

- Java 1.7
- Disco Duro con al menos 100 MB de espacio libre para la instalación de EigeCam 1.0 (10 GB de espacio libre recomendado para almacenamiento de registros en la base de datos).
- Mínimo: Dual Core con 1GB RAM, o superior.
- Cámara o webcam 1.0 MP o superior.
- MYSQL 5.0, para gestionar la base de datos.
- Windows /VISTA/7/8, LINUX

CAPÍTULO 4

4. ANÁLISIS TÉCNICO Y ECONÓMICO DEL SISTEMA

El sistema ha sido desarrollado e implementado de manera satisfactoria debido a que los recursos de hardware y software utilizados para la puesta en ejecución de la aplicación son los indispensables.

Como se explicó anteriormente, las herramientas utilizadas para el desarrollo del software son de libre distribución, por lo que la obtención de los mismos no tuvo mayores complicaciones, ya que se encuentran disponibles en internet para cualquier persona. Las herramientas utilizadas son java 1.7, NetBeans 8.0 y MySQL 5.0.

La utilización de librerías Java como Opencv permiten utilizar su código fuente y adaptarlo a nuestras necesidades y todas ellas son libres.

Los costos reales asumidos por la autora de la tesis se detallan a continuación:

Tabla 15. Recursos Humanos

DESCRIPCIÓN	**CANTIDAD**	**# DE HORAS**	**V/u $**	**V/t $**
RECURSOS HUMANOS				
Tesista	1	400	3.00	1200.00
Director de tesis	1	0	0	-------
			TOTAL	**1200.00**

Fuente: [La Autora].

Tabla 16. Recursos Técnicos

RECURSOS TÉCNICOS				
HARDWARE	**CANTIDAD**	**# DE HORAS**	**V/u $**	V/t $
Computadora para Desarrollo e implementación	1	--	900.00	900.00
Impresora	1	--	50.00	1000
SOFTWARE				
Microsoft office	1	--	--	--
Java	1	--	--	--
Mysql	1	--	--	--
Opencv 2.4.8	1	--	--	--
Net Beans 8.0	1	--	--	--
			TOTAL	**950.00**

Fuente: [La Autora].

Tabla 17. Recursos Materiales.

RECURSOS MATERIALES			
Descripción	**Cantidad**	**Valor Unitario ($)**	**Valor Total ($)**
Copias	--	10.00	10.00
Resma de papel	4	3.50	14.00
Anillados	3	1.50	4.50
Empastado	4	10.00	40.00
Recarga de tinta negra	3	3.00	9.00
Recarga de tinta a color	3	3.00	9.00
Transporte	--	50.00	50.00
Internet	50	0.60	30.00
		TOTAL	**166.50**

Fuente: [La Autora].

Tabla 18. Presupuesto Final.

Presupuesto Final	**Subtotal**
Humanos	1200.00
Técnicos	950.00
Materiales	166.50
Subtotal del Proyecto	**2316.50**
Imprevistos 10 %	231.56
Total del Proyecto	**2548.15**

Fuente: [La Autora]

El análisis del costo del sistema de reconocimiento facial está basado en los precios actuales del hardware y software que fueron necesarios para el desarrollo del mismo que está detallado en las tablas 16 y 17.

CAPÍTULO 5

5. CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

- El sistema biométrico basado en reconocimiento facial desarrollado tiene una eficiencia de 100% para la detección y 92% para el reconocimiento, esto en condiciones ideales.

- Luego de estudiar los diferentes algoritmos para la detección y reconocimiento facial se optó por realizar el sistema con el algoritmo de Viola-Jones para la detección porque es capaz de procesar imágenes de manera rápida alcanzando altos índices de detección y con Eigenfaces el reconocimiento ya que esta técnica utiliza el análisis de componentes principales para un rápido y preciso reconocimiento de rostros.

- Se definió como herramientas para el desarrollo de la aplicación denominada EigeCam 1.0 a Java pues es un lenguaje de programación código abierto y MySQL para el almacenamiento de los datos.

- Se desarrolló una interfaz hombre-máquina de fácil manejo, amigable e intuitivo para el usuario.

- El sistema desarrollado sirve de ayuda para el control de las horas laborables del personal de una empresa, generando los reportes de estos registros.

- Una vez implementado el sistema se obtuvo una buena aceptación por parte de los usuarios, pues no es invasivo, ya que se requiere de una computadora y una cámara.

- Mientras mayor sea las veces de entrenamiento va a disminuir el margen de error, para el siguiente reconocimiento, así como también mientras mayor sea el número de individuos aumentará el tiempo de respuesta del reconocimiento.

- La etapa de pre-procesado tiene vital importancia en el sistema, pues su labor es adaptar lo mejor posible la imagen para la extracción de características.

- Mientras menor sea la distancia euclidiana entre la imagen de entrada y las imágenes de entrenamiento, mayor es la similitud y por lo tanto el reconocimiento es más aproximado.

5.2. RECOMENDACIONES

- Utilizar herramientas de software libre en la construcción de aplicaciones permite optimizar los procesos, disminuir los costos y el tiempo de desarrollo.

- Realizar las pruebas de validación para asegurar la entrega de un software de calidad, que brinde información confiable.

- Se recomienda dividir el sistema en módulos, con el fin de ir realizando las pruebas por separado para identificar y corregir rápidamente los errores del sistema.

- Ubicar la cámara en un lugar donde las variaciones de iluminación sean mínimas para evitar sombras que interfieran en el reconocimiento. Además el uso de una cámara de alta resolución permite obtener mejores resultados.

- Se recomienda al administrador captar como mínimo 3 imágenes de cada individuo para que forme parte de la base de datos así como también en un ambiente de las mismas características para obtener mejores resultados de reconocimiento.

- Para un mejor reconocimiento se debe estar sin accesorios que puedan obstruir el rostro como gorra y gafas.

- En el caso de que haya cambios en el rostro como es el caso barba o bigote pedir al administrador del sistema que tome nuevamente las imágenes para mantener el buen rendimiento del sistema.

- Para que el sistema sea aún más seguro en caso de que sea necesario, se podría hacer que una vez identificado el usuario se logee es decir, ingrese su nombre de usuario y contraseña.

- Se recomienda implementar el sistema de autenticación biométrica basado en reconocimiento facial porque es un sistema poco intrusivo y tiene una alta aceptación por parte de los usuarios en comparación al reconocimiento por retina y huella dactilar que tiene una aceptación media.
- El sistema de reconocimiento facial sirve para registrar las horas laborables del personal de una entidad y es adecuado para funcionar en empresas e instituciones educativas como la Universidad Nacional de Loja para el registro del personal docente y administrativo en cada una de sus áreas.

BIBLIOGRAFÍA:

[1] Diseño de materiales multimedia. Web 2.0. *Imagen. Conceptos básicos de imagen digital*. [En línea]. [Fecha de consulta: 7 de agosto de 2013]. Disponible en: http://www.ite.educacion.es/formacion/materiales/107/cd/imagen/pdf/imagen01.pdf

[2] masadelanta.com. *¿Qué es un Pixel? - Definición de Pixel.* [En línea]. [Fecha de consulta: 7 de agosto de 2013]. Disponible en: http://www.masadelante.com/faqs/pixel

[3] FOTONOSTRA. *La imagen digital.* [En línea]. [Fecha de consulta: 7 de agosto de 2014]. Disponible en: http://www.fotonostra.com/digital/imagendigital.htm

[4] Diseño de Materiales Multimedia _Web 2.0. *Profundidad de color*. [En línea]. 2008. [Fecha de consulta: 9 de agosto de 2013]. Disponible en: http://www.ite.educacion.es/formacion/materiales/107/cd/imagen/imagen0103.html

[5] Diseño de Materiales Multimedia _Web 2.0. *Modos de color*. [En línea]. 2008. [Fecha de consulta: 13 de agosto de 2013]. Disponible en: http://www.ite.educacion.es/formacion/materiales/107/cd/imagen/imagen0104.html

[6] GÁMEZ, Jiménez Carmen Virginia. *Diseño y Desarrollo de un Sistema de Reconocimiento de Caras*. [En línea]. 2009. [Fecha de consulta: 15 de agosto de 2013]. Disponible en: http://e-archivo.uc3m.es/bitstream/handle/10016/5831/PFC_CarmenVirginia_Gamez_Jimenez.pdf?sequence=1

[7] HERRERO, Tamara. *SISTEMA AUTOMÁTICO DE DETECCIÓN Y ETIQUETADO DE CARAS EN IMÁGENES*. [En línea]. Marzo, 2010. [Fecha de consulta: 12 de septiembre de 2013]. Disponible en: http://e-archivo.uc3m.es/bitstream/handle/10016/11170/PFC%20Tamara%20Herrero%20Vez.pdf?sequence=1

[8] *Tema 6. Técnicas de filtrado.* [En línea]. Marzo, 2010. [Fecha de consulta: 7 de octubre de 2013]. Disponible en: http://www.um.es/geograf/sigmur/teledet/tema06.pdf

[9] CASTRO, Luis. ¿Qué es Java? [En línea]. [Fecha de consulta: 10 de noviembre de 2013]. Disponible en: http://aprenderinternet.about.com/od/Glosario/g/Que-Es-Java.htm

[10] LEMAY, Laura y CADENHEAD, Rogers. *Aprendiendo Java 2 en 21 días.* [En línea]. [Fecha de consulta: 15 de noviembre de 2013]. Disponible en: http://es.scribd.com/doc/164940910/Java-en-21-Dias

[11] BELMONTE, Oscar. *Introducción al lenguaje de programación Java*. [En línea]. [Fecha de consulta: 15 de noviembre de 2013]. Disponible en: http://www3.uji.es/~belfern/pdidoc/IX26/Documentos/introJava.pdf

[12] BLOG Buhos.com. *Lenguajes de Programación cuadro comparativo.* [En línea]. [Fecha de consulta: 17 de noviembre de 2013]. Disponible en: http://blog.buhoos.com/lenguajes-de-programacion-cuadro-comparativo/

[13] PEREIRA, Manuel. *Diez motivos para programar en Java*. [En línea]. 10 de diciembre de 2010. [Fecha de consulta: 22 de noviembre de 2013]. Disponible en: http://manuelpereiragonzalez.blogspot.com/2010/12/diez-motivos-para-programar-en-java.html

[14] HOMINI. *Plataforma Biométrica Homini*. [En línea]. 2004. [Fecha de consulta: 3 de agosto de 2013]. Disponible en: http://www.homini.com/new_page_5.htm

[15] INBIOSYS Biometria. *Historia de la Biometría*. [En línea]. Septiembre, 2009. [Fecha de consulta: 13 de julio de 2013]. Disponible en: http://inbiosys.wordpress.com/2009/09/16/historia-de-la-biometria/

[16] Blog de control de accesos. *Biometría: conceptos básicos.* En línea]. 17 de abril, 2008. [Fecha de consulta: 17 de julio de 2013]. Disponible en: http://control-accesos.es/lectores/lectores-biometricos/biometria-conceptos-basicos

[17] BRAVO, Miguel. *Funcionamiento y rendimiento de la Biometría.* [En línea]. 17 de noviembre, 2008. [Fecha de consulta: 13 de julio de 2013]. Disponible en: http://cliover.blogspot.com/2008/11/funcionamiento-y-rendimiento-biometria.html

[18] umanik. *Huella dactilar*. [En línea]. [Fecha de consulta: 13 de julio de 2013]. Disponible en: http://www.umanick.com/wp-content/uploads/2013/11/tecnologiasbiometricas_masusadas_huelladactilar.pdf?4f89ec

[19] umanik. *Reconocimiento facial*. [En línea]. [Fecha de consulta: 13 de julio de 2013]. Disponible en: http://www.umanick.com/wp-content/uploads/2013/11/tecnologiasbiometricas_masusadas_reconocimientofacial.pdf?4f89ec

[20] umanik. *Reconocimiento de voz*. [En línea]. [Fecha de consulta: 13 de julio de 2013]. Disponible en: http://www.umanick.com/wp-content/uploads/2013/11/tecnologiasbiometricas_masusadas_reconocimientovoz.pdf?4f89ec

[21] umanik. *ADN*. [En línea]. [Fecha de consulta: 10 de febrero de 2014]. Disponible en: http://www.umanick.com/wp-content/uploads/2013/11/tecnologiasbiometricas_masusadas_ADN.pdf?4f89ec

[22] Kimaldi. *Biometría*. [En línea]. [Fecha de consulta: 05 de julio de 2013]. Disponible en: http://www.kimaldi.com/kimaldi/area_de_conocimiento/biometria/parametros_biometricos

[23] GOIT. *Biometría*. [En línea]. [Fecha de consulta: 02 de julio de 2013]. Disponible en: http://www.goit.cl/biometria.html

[24] ARDFOMACION. *Curso de Adobe Illustrator CS4.* [En línea]. [Fecha de consulta: 04 de marzo de 2014]. Disponible en: http://www.adrformacion.com/cursos/illustcs4/leccion1/tutorial2.html

[25] Procesamiento digital de imágenes (y más) en VB. NET/Java. *Histograma acumulado. Tratamiento de imágenes. Parte XIV.* [En línea]. 4 Mayo, 2013. [Fecha de consulta: 21 de febrero de 2014]. Disponible en: http://algoimagen.blogspot.com/2013/05/histograma-acumulado-tratamiento-de.html]

[26] PAZ, Néstor. *Adaboost con aplicación a detección de caras mediante el algoritmo de Viola-Jones.* [En línea]. Febrero, 2009. Fecha de consulta: 04 de febrero de 2014]. Disponible en: https://eva.fing.edu.uy/file.php/514/ARCHIVO/2008/TrabajosFinales2008/NestorPaz2008_informe.pdf

[27] HERNANDEZ, Ernesto, CABRERA, Alejandro. *Impacto de la memoria cache en la aceleración de la ejecución de algoritmo de detección de rostros en sistemas empotrados.* [En línea]. 2012. Fecha de consulta: 04 de febrero de 2014]. Disponible en: http://scielo.sld.cu/scielo.php?pid=S1815-59282012000200008&script=sci_arttext

[28] deteccin_de_caras. *OpenCV: Introducción.* [En línea]. 23 de Noviembre, 2012. Fecha de consulta: 09 de febrero de 2014]. Disponible en: http://ocw.unia.es/ciencias-tecnologicas/tecnologia-del-ocio/materiales-basicos-folder/html/B2_U3/deteccin_de_caras.html/skinless_view

[29] SERVÍN, Jorge. *Localización y reconocimiento de rostros en imágenes monoculares de frente con variación en escala.* [En línea]. 2009. Fecha de consulta: 12 de marzo de 2014]. Disponible en: http://newton.azc.uam.mx/mcc/01_esp/11_tesis/tesis/terminada/Tesis%20-%20Localizaci%C3%B3n%20y%20Reconocimiento%20de%20Rostros%20en%20Im%C3%A1genes%20Monoculares%20de%20Frente%20con%20Variaci%C3%B3n%20.pdf

[30] RODRÍGUEZ, Sergio. *"Análisis del preprocesado de imágenes en el reconocimiento de caras basado en PCA".* [En línea]. 2010. Fecha de consulta: 13 de marzo de 2014]. Disponible en: http://lcsi.umh.es/docs/pfc_sergio/Memoria_Sergio_Rodriguez.pdf

ANEXOS

ANEXO A. CÓDIGO DE LA APLICACIÓN

- **DETECCIÓN**

A continuación se muestra el código que permite realizar la detección de rostros del sistema, aquí también se muestra la manera como se manipula la cámara para poder realizar las capturas de las imágenes en tiempo real:

```
public void run() {
    try {
        //se fija el número de cámara que se va a utilizar
      OpenCVFrameGrabber grabber = new OpenCVFrameGrabber(numeroCamara);

      grabber.setImageWidth(camaraLabel.getWidth());
      grabber.setImageHeight(camaraLabel.getHeight());
      grabber.start();
      opencv_core.IplImage image = grabber.grab();
      camarasItem.setEnabled(true);

       //mientras running sea true, se mostrará la cámara y se realizará la detección.
      while (running) {
        opencv_core.IplImage originalImage = grabber.grab();
        imagen = utilC.convertirBufferedImageToMat(originalImage.getBufferedImage());

        if (!imagen.empty()) {
          //detectame los rostros
          faceDetector.detectMultiScale(imagen, faceDetections);

          for (Rect rect : faceDetections.toArray()) {
            int ancho = rect.width;
            int anchoPorc = (15 * ancho) / 100;
            ancho = rect.width - anchoPorc;

            //dibujando rectangulo de los rostros detectados.
            Core.rectangle(imagen, new Point(rect.x + anchoPorc, rect.y + anchoPorc), new
Point(rect.x + ancho, rect.y + rect.height + (anchoPorc / 2)), new Scalar(0, 255, 0));
          }
          setImage(utilC.convertirMatToImage(imagen));
        }

        camarasItem.setEnabled(false);
      }
      grabber.stop();
      grabber.release();
    } catch (Exception ex) {
    }
  }
```

- **RECONOCIMIENTO**

A continuación se enuncia el código que se encarga del reconocimiento de rostros del sistema, el cual utiliza la técnica eigenfaces para su funcionamiento.

```
/* devuelve el resultado del reconocimiento realizado, si se reconoció o no al individuo*/
   public Resultado findResultado(String direccionImagen, int eigenfacesSeleccionadas, double
umbral) {
      boolean encontrado = false;
      String message = null;
      String coincidencia = "";
      double distanciaMinima = 0.0;
      try {
         //checkImageSizeCompatibility
         validarCompatibilidadTamanoImagen(direccionImagen);
         Matrix2D inputFace = getNormalisedInputFace(direccionImagen);

         inputFace.subtract(new Matrix2D(bundle.getAvgFace(), 1));
         Matrix2D inputWts = getInputWeights(eigenfacesSeleccionadas, inputFace);
         //distancias de todas las eigenfaces comparadas con la imagen de entrada/
         double[] distances = getDistances(inputWts);
         ImageDistanceInfo distanceInfo = getMinimumDistanceInfo(distances);
         distanciaMinima = Math.sqrt(distanceInfo.getValue());
         coincidencia = getMatchingFileName(distanceInfo);

         System.out.println("distancia minima: "+distanciaMinima);
         if (distanciaMinima > umbral) {
            message = "No se ha encontrado coincidencia, intente un valor inicial (threshold) más
alto";
         } else {
            encontrado = true;
            message = "Coincidencia de imagen encontrada";
         }
      } catch (Exception e) {
         return new Resultado(false, "", Double.NaN, e.getMessage());
      }
      return new Resultado(encontrado, coincidencia, distanciaMinima, message);
   }

/*Este método obtiene las imágenes de entrada normalizadas*/
private Matrix2D getNormalisedInputFace(String imageFileName) throws ErrorFaceRec {
      double[] inputFaceData = getImageData(imageFileName);
      Matrix2D inputFace = new Matrix2D(inputFaceData, 1);
      //logger.info("inputface");
      //logger.info(inputFace.toString());
      inputFace.normalise();
      //logger.info("normalised inputface");
      //logger.info(inputFace.toString());
```

```
        return inputFace;
    }

/*hace los cálculos de acuerdo a la eifenfaces seleccionadas y el conjunto de imágenes de
entrada*/
private void doCalculations(String dir, List<String> imglist, int selectedNumOfEigenFaces)
throws ErrorFaceRec, IOException {
        FaceBundle b = createFaceBundle(imglist);
        double[][] wts = calculateWeights(b, selectedNumOfEigenFaces);
        this.bundle = b;
        this.weights = wts;
        writeCache(dir, b);
    }

/*crea el rostro de entrenamiento a partir de las direcciones de las imágenes de los
individuos*/
public FaceBundle createFaceBundle(List<String> filenames) throws ErrorFaceRec, IOException
{
        BufferedImage[] bufimgs = getGrayScaleImages(filenames);
        checkImageDimensions(filenames, bufimgs);
        Matrix2D imagesData = getNormalisedImagesData(bufimgs);
        double[] averageFace = imagesData.getAverageOfEachColumn();
        imagesData.adjustToZeroMean();
        EigenvalueDecomposition egdecomp = getEigenvalueDecomposition(imagesData);
        double[] eigenvalues = egdecomp.getEigenValues();
        double[][] eigvectors = egdecomp.getEigenVectors();

        TreeSet<ValueIndexPair> pairList = getSortedPairs(eigenvalues, eigvectors);
        eigenvalues = getSortedVector(pairList);
        eigvectors = getSortedMatrix(eigvectors, pairList);

        Matrix2D eigenFaces = getNormalisedEigenFaces(imagesData, new Matrix2D(eigvectors));
        int imageWidth = bufimgs[0].getWidth();
        createEigenFaceImages(eigenFaces, imageWidth);
        int imageHeight = bufimgs[0].getHeight();
        FaceBundle b = new FaceBundle(filenames, imagesData.toArray(), averageFace,
eigenFaces.toArray(), eigenvalues, imageWidth, imageHeight);
        return b;
    }

/*crea por cada imagen de entrada 2 eigenfaces*/
    public TreeSet<ValueIndexPair> createPairs(double[] aVector, double[][] aMatrix) {
        TreeSet<ValueIndexPair> pList = null;
        if (aVector.length != aMatrix.length) {
            printError("matrix rows don't match items in vector ");
        } else {
            pList = new TreeSet<ValueIndexPair>();
            for (int i = 0; i < aVector.length; i++) {
                ValueIndexPair dp = new ValueIndexPair(aVector[i], i);
                pList.add(dp);
```

```
        }
      }
      return pList;
    }
/*Se encarga de la creación de las eigenfaces*/
    public void createEigenFaceImages(Matrix2D eigenfaces, int imgwidth) throws IOException {
      System.out.println("creating eigenfaces");
      double[][] eigenfacesArray = eigenfaces.toArray();
      String fldrname = System.getProperty("user.dir") + "/eigen";

      makeNewFolder(fldrname);
      String prefix = "eigen";
      String ext = ".png";
      for (int i = 0; i < eigenfacesArray.length; i++) {
        double[] egface = eigenfacesArray[i];
        String filename = fldrname + File.separator + prefix + i + ext;
        createImageFromArray(filename, egface, imgwidth);
      }
      System.out.println("created eigenfaces");
    }

/*obtiene todas la eigenfaces normalizadas*/
    private Matrix2D getNormalisedEigenFaces(Matrix2D imagesData, Matrix2D eigenVectors) {
      Matrix2D eigenFaces = eigenVectors.multiply(imagesData);
      double[][] eigenFacesData = eigenFaces.toArray();
      for (int i = 0; i < eigenFacesData.length; i++) {
        double norm = Matrix2D.norm(eigenFacesData[i]);
        for (int j = 0; j < eigenFacesData[i].length; j++) {
          double v = eigenFacesData[i][j];
          eigenFacesData[i][j] = v / norm;
        }
      }
      return new Matrix2D(eigenFacesData);
    }

/*convierte en escala de grises a la imagen de entrada*/
    public BufferedImage convertToGray(BufferedImage img) throws ErrorFaceRec {
      BufferedImage gray = null;
      try {

        gray = new BufferedImage(img.getWidth(), img.getHeight(),
BufferedImage.TYPE_BYTE_GRAY);
        ColorConvertOp op = new ColorConvertOp(img.getColorModel().getColorSpace(),
gray.getColorModel().getColorSpace(), null);
        op.filter(img, gray);
        return gray;
      } catch (Exception e) {
        //logger.log(Level.SEVERE, "grayscale conversion failed", e);
        throw new ErrorFaceRec("conversion a escala de grises fallida:\n" + e.getMessage());
      }
```

```
  }


/*valida el numero de eigenfaces seleccionadas.*/
  private void validarNumeroEigenFacesSeleccionadas(int selectedNumOfEigenFaces,
      List<String> newFileNames) throws ErrorFaceRec {
    int numImgs = newFileNames.size();
    if (selectedNumOfEigenFaces <= 0 || selectedNumOfEigenFaces >= numImgs) {
      //logger.log(Level.SEVERE, "incorrect number of selectedeigenfaces" +
selectedNumOfEigenFaces + "used" + "allowed btw 0-" + numImgs);
      throw new ErrorFaceRec("incorrecto numero de eigenfaces seleccionadas usadas..\nuse
un numero entre 0 y hasta " + (numImgs - 1));
    }
  }

/*crea una imagen a partir de un arreglo  de decimales que son de entrenamiento*/
  private void createImageFromArray(String filename, double[] imgdata, int wd) throws
IOException {
    BufferedImage bufimg = new BufferedImage(wd, imgdata.length / wd,
BufferedImage.TYPE_BYTE_GRAY);
    Raster rast = bufimg.getData();
    WritableRaster wr = rast.createCompatibleWritableRaster();
    double maxValue = Double.MIN_VALUE;
    double minValue = Double.MAX_VALUE;
    for (int i = 0; i < imgdata.length; i++) {
      maxValue = Math.max(maxValue, imgdata[i]);
      minValue = Math.min(minValue, imgdata[i]);
    }
    for (int j = 0; j < imgdata.length; j++) {
      imgdata[j] = ((imgdata[j] - minValue) * 255) / (maxValue - minValue);
    }
    wr.setPixels(0, 0, wd, imgdata.length / wd, imgdata);
    bufimg.setData(wr);
    File newfile = new File(filename);
    ImageIO.write(bufimg, "png", newfile);

  }

/*reconstruye las imagenes a partir de los eigenfaces*/
  public void reconstruccionRostros(int selectedeigenfaces) throws IOException {
    double[][] eigenfacesArray = bundle.getEigenFaces();
    Matrix2D egnfacesMatrix = new Matrix2D(eigenfacesArray);
    Matrix2D egnfacesSubMatrix = egnfacesMatrix.getSubMatrix(selectedeigenfaces);
    double[] eigenvalues = bundle.getEigenValues();
    Matrix2D eigenvalsMatrix = new Matrix2D(eigenvalues, 1);
    Matrix2D eigenvalsSubMatrix =
eigenvalsMatrix.transpose().getSubMatrix(selectedeigenfaces);

    double[][] phi = getPhiData(egnfacesSubMatrix, eigenvalsSubMatrix);
    double[][] xnew = addAverageFaceData(phi);
```

```
        String reconFolderName = System.getProperty("user.dir") + "\\eigen";

        String ext = ".png";
        reconstructPhiImages(phi, reconFolderName, ext);
        reconstructOriginalImages(xnew, reconFolderName, ext);
        System.out.println("reconstruccion terminada");
    }


    /*Devuelve el resultado del reconocimiento, si se reconoció o no al individuo*/
    public Resultado procesoSelecciones(String faceImageName, String directorio, String
numRostros, String inicio) {
        Resultado resultado = null;
        int numeroRostros = 0;
        double umbral = 0.0;
        try {
            validarImagenArchivoSeleccionado(faceImageName);
            validarCarpetaSeleccionada(directorio);
            numeroRostros = getNumRostrosVal(numRostros);
            umbral = getUmbral(inicio);
            String extension = getExtensionArchivo(faceImageName);
            checkCache(directorio, extension, numeroRostros);
            reconstruccionRostros(numeroRostros);
            resultado = findResultado(faceImageName, numeroRostros, umbral);
        } catch (Exception e) {
            resultado = new Resultado(false, null, Double.NaN, e.getMessage());
        }
        return resultado;
    }

    /*devuelve la direccion de la imagen encontrada como similitud*/
    public String accion(String dirImagen, String dirImagenes, String numRostros, String umbral)
{
        long inicio = System.currentTimeMillis();

        List<String> conf = null;
        if(LecturaDeArchivos.dirExiste("config/config.eig")){
            conf = LecturaDeArchivos.leerArchivo("config/config.eig");
            if(conf.size() != 3){
                Mensaje.mensaje("Realice la configuración");
                return "";
            }
        }else{
            Mensaje.mensaje("Realice la configuración");
            return "";
        }


        FaceRec face = new FaceRec();
```

```
      Resultado r = face.procesoSelecciones(dirImagen, dirImagenes, numRostros, umbral);

      String dirIma = "";

      long finReco = System.currentTimeMillis();
         debug("tiempo transcurrido de reconocimiento = " + (finReco - inicio) / 1000.0 + "
segundos");

      if (r.getMatchSuccess()) {

         System.out.println("distancia: "+r.getDistanciaEncontrada());
         if (r.getDistanciaEncontrada() <= Double.parseDouble(conf.get(1))){
            debug(dirImagen + " se asemeja a " + r.getDireccionCoincidencia() + " precision=" +
r.getDistanciaEncontrada());
            //javax.swing.JOptionPane.showMessageDialog(null, "Similitud", "FOTO",
JOptionPane.INFORMATION_MESSAGE, new ImageIcon(r.getDireccionCoincidencia()));
            dirIma = r.getDireccionCoincidencia();

         } else {
            printError("similitud fallida: " + r.getMensajeCoincidencia());
            Mensaje.mensajeError("Sujeto no encontrado");
         }
//          System.out.println("akiiiiiiiiiii");
      } else {
         printError("similitud fallida: " + r.getMensajeCoincidencia());
         Mensaje.mensajeError("Sujeto no encontrado");
      }
      long fin = System.currentTimeMillis();
      debug("tiempo transcurrido = " + (fin - inicio) / 1000.0 + " segundos");
      return dirIma;
   }
```

ANEXO B. IMÁGENES DE LA IMPLEMENTACIÓN

Figura 38. Logo de la aplicación. Fuente: [La Autora].

Figura 39. Pruebas de funcionamiento del sistema. Fuente: [La Autora].

Figura 40. Pruebas de reconocimiento facial. Fuente: [La Autora].

Figura 41. Sistema en funcionamiento. Fuente: [La Autora].

IDPERSONA	DTYPE	APELLIDO	DIRECCION	IDENTIFICACION	NOMBRE	TELEFONO	CARGO
1	Empleado	Yanangomez	Celi Roman	1104540754	Michael	0997645469	Administrador
51	Empleado	Garrochamba	San Agustin	1900743707	Sandra E.	2555555	Jefe de Personal
1351	Empleado	Davila	Belen	1104193345	Fernando	0997645469	Programador
1451	Empleado	Abad	Angelia	1104469461	Galo	0997645469	Programador
1701	Empleado	Tinizaray	Bolivar entre Miguel Riofrio y Rocafuerte	1104350440	Julyssa	2584706	Contadora
1705	Empleado	Martinez	Bolivar entre Miguel Riofrio y Rocafuerte	1104097983	Cecilia	584706	Contadora
2101	Empleado	Tinizaray	Belen	1103740054	Diana	0996457893	Secretaria

Figura 42. Empleados en MySQL. Fuente: [La Autora].

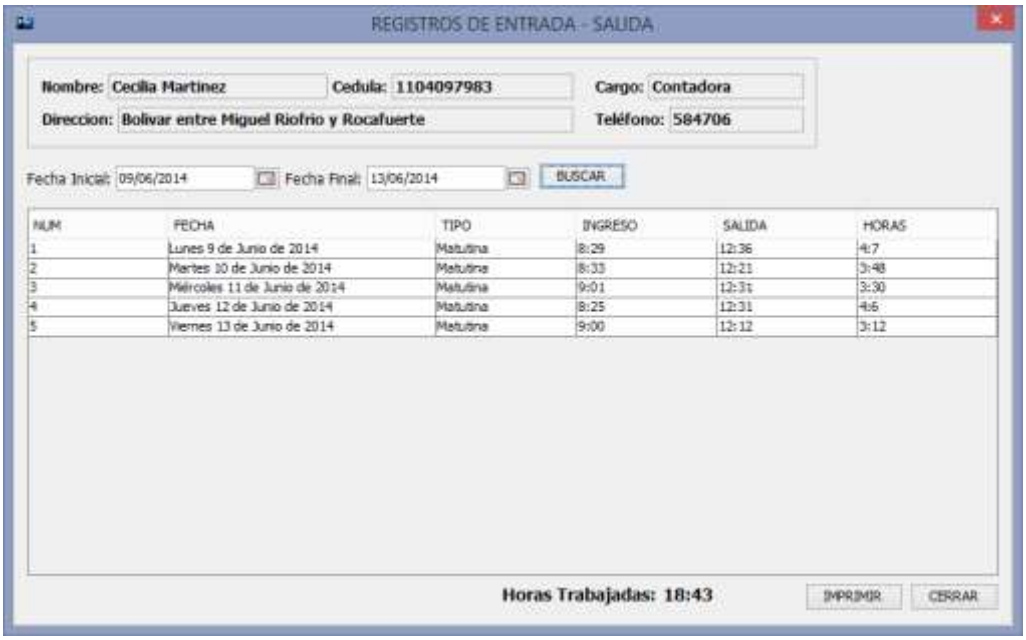

Figura 43. Registros de entrada-salida de un usuario. Fuente: [La Autora].

Figura 44. Registro de ingreso-salida listo para ser impreso. Fuente: [La Autora].

ANEXO C. MANUAL DE USUARIO

2014

UNIVERSIDAD NACIONAL DE LOJA

ÁREA DE ENERGÍA, LAS INDUSTRIAS Y LOS RECURSOS NATURALES NO RENOVABLES

MANUAL DEL USUARIO

RESPONSABLE:

- ✓ SANDRA ELIZABETH GARROCHAMBA SÁNCHEZ

Índice

Requerimientos del sistema.....1

Introducción.....1

Manual.....2

REQUERIMIENTOS DEL SISTEMA

Esta aplicación para su eficiente funcionamiento necesita cumplir con los siguientes requerimientos del sistema:

- Java 1.7
- Disco Duro con al menos 100 MB de espacio libre para la instalación de EigeCam 1.0 (10 GB de espacio libre recomendado para almacenamiento de registros en la base de datos).
- Mínimo: Dual Core con 1GB RAM, o superior.
- Cámara o webcam 1.0 MP o superior.
- MYSQL 5.0, para gestionar la base de datos.
- Windows /VISTA/7/8, LINUX

INTRODUCCIÓN A EIGECAM 1.0

EigeCam 1.0 es un software cuya función principal es administrar el tiempo de horas laboradas de cada uno de los individuos de una empresa. Además tiene una interfaz de usuario fácil de manejar permitiendo que los usuarios no tengan problemas al momento de usar el software, también le ofrece la posibilidad de tener todos los reportes en una base de datos con la opción de imprimir si se lo desea. En caso de que el sistema no reconozca al usuario ofrece la opción de registrarse en el sistema con su nombre de usuario y contraseña.

EigeCam 1.0 características:

- Administrar (Nuevo, editar, eliminar) Usuario.
- Login Usuario
- Login Administrador
- Acerca de…

MANUAL DE EIGECAM 1.0

1. Cuando se ejecuta el programa se observa que se están cargando todas las librerías y si es utilizado por primera vez, pide que primero se registre el usuario administrador como se en las ilustraciones 1 y 2.

Ilustración 1. Cargando el programa.

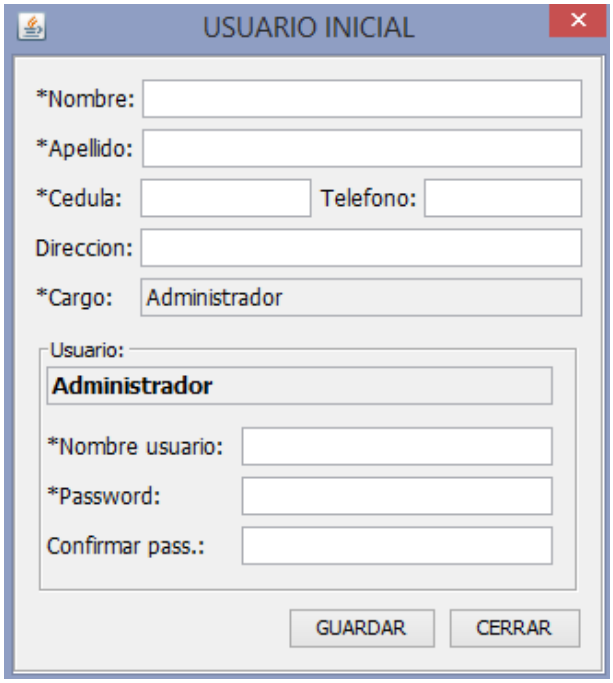

Ilustración 2. Registro de datos personales del administrador.

2. Desde la pantalla principal de EigeCam 1.0 vamos a utilizar todas las opciones del sistema.

Ilustración 3. Pantalla principal.

3. Antes de realizar cualquier acción en EigeCam 1.0 siempre va a pedir que el usuario-administrador escriba su nombre de usuario y password. Esta herramienta permite que solo el administrador puedan realizar la manipulación de horarios, crear, editar o eliminar un usuario así como también tener acceso a los reportes.

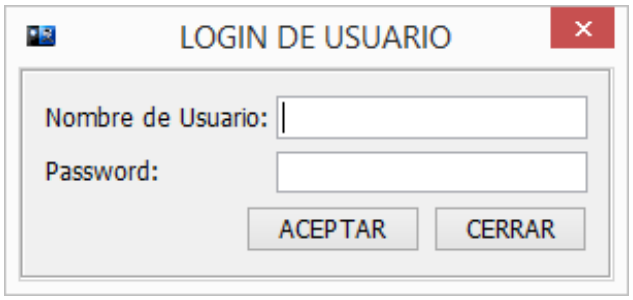

Ilustración 4. Login de usuario-administrador antes de hacer un cambio en el sistema.

4. En el caso de que el administrador escriba mal el nombre de usuario, la contraseña o ambos saldrá un mensaje de error.

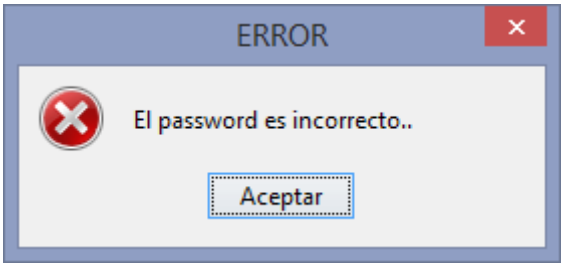

Ilustración 5. Error

5. Para crear, editar, o eliminar un usuario del sistema, vamos a archivo luego empleado y seleccionamos administrar.

Ilustración 6. Crear, editar o eliminar un usuario.

6. Antes de realizar cualquier cambio en los usuarios siempre nos aparecerá la opción buscar empleados, donde basta con poner una letra para que se muestren los usuarios que tienen dicha letra y seleccionamos al que buscamos.

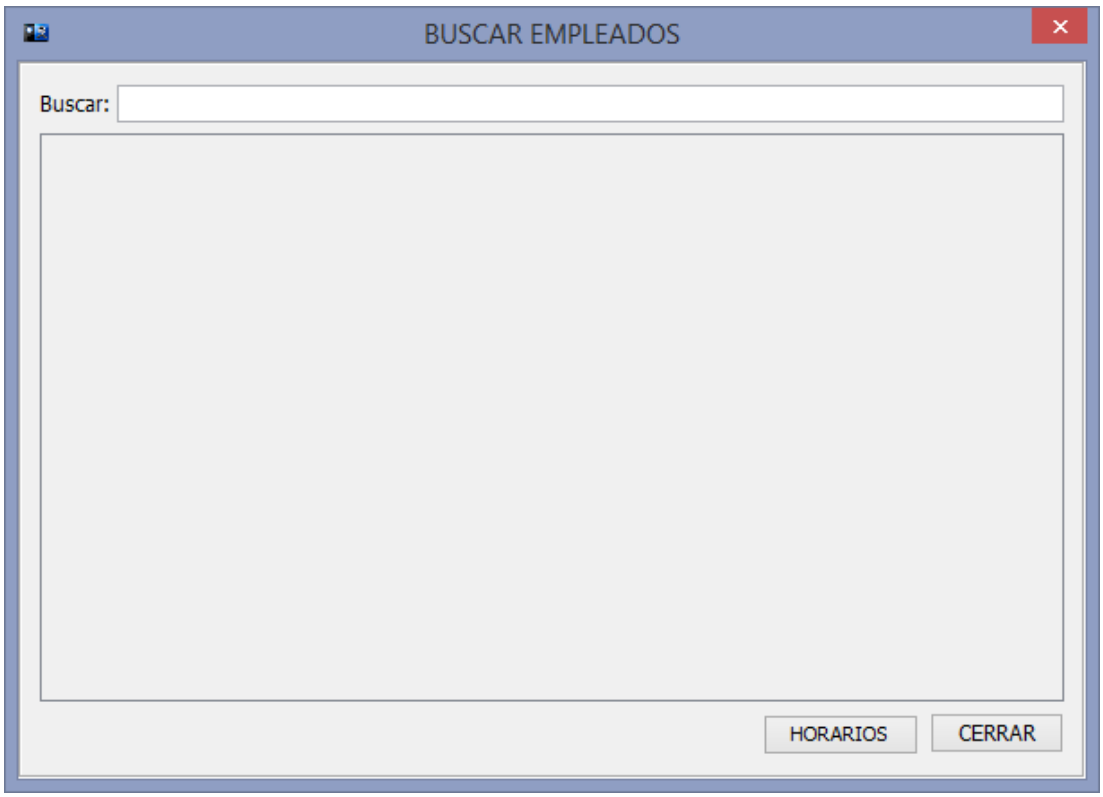

Ilustración 7. Buscar empleados.

7. Para asignar a un usuario las imágenes nos dijimos a archivo, luego a empleado y asignar usuario. Se presentará la siguiente pantalla:

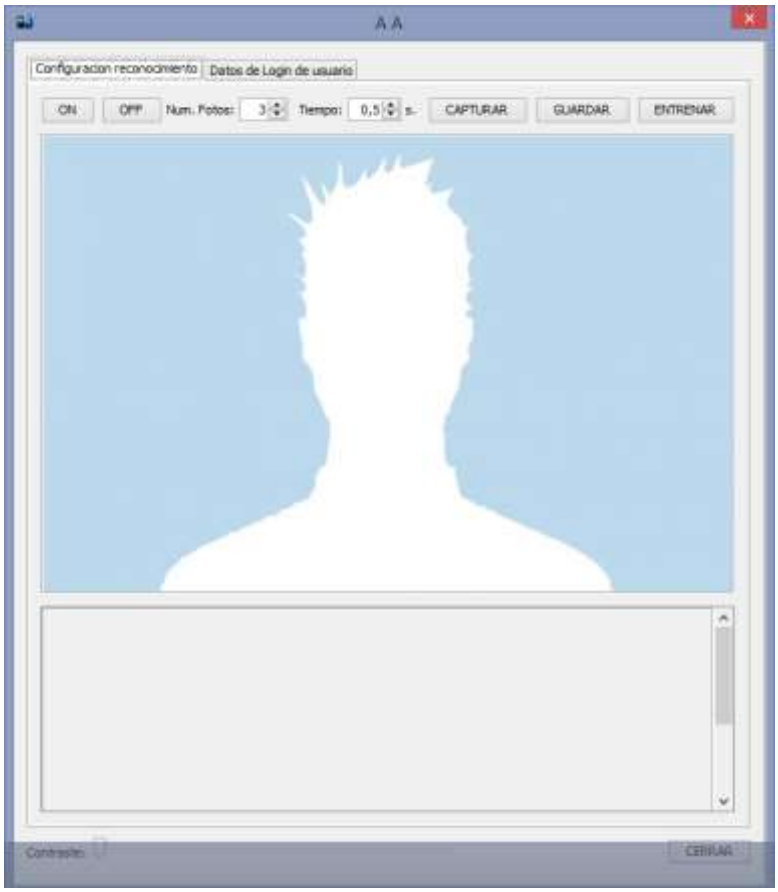

Ilustración 8. Asignar imagen al usuario.

Además tenemos las opciones para elegir el tiempo entre cada imagen, el número de imágenes y una barra para subir el contraste en caso de ser necesario. Tenemos datos de login de usuario que serán útiles en caso de que al hacer tres intentos de registro con el rostro el sistema no reconozca.

8. Una vez que se tiene al usuario con las imágenes respectivas, pasamos a asignarle el horario u horarios respectivos. Para ello nos ubicamos en archivo, horario y luego horarios. Es necesario que esté marcada activo para poder hacer uso del horario.

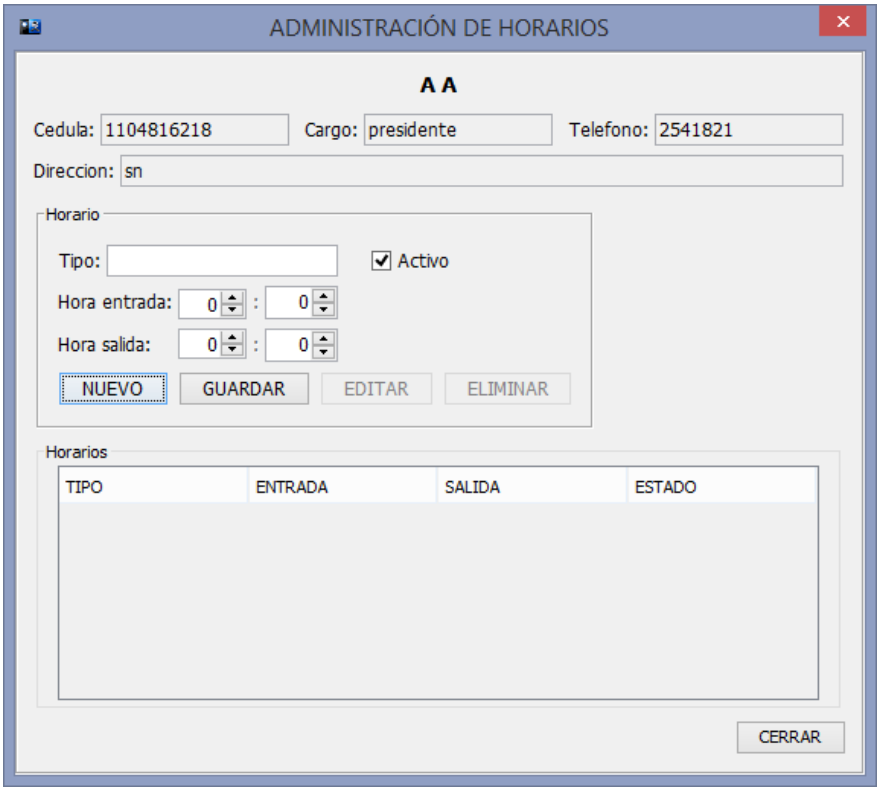

Ilustración 9. Asignar el horario a los usuarios.

9. Para ver las horas registradas de los usuarios tenemos la opción registro de entrada-salida, en la que podemos seleccionar desde y hasta que fecha queremos el reporte. Además existe la opción para imprimir el reporte.

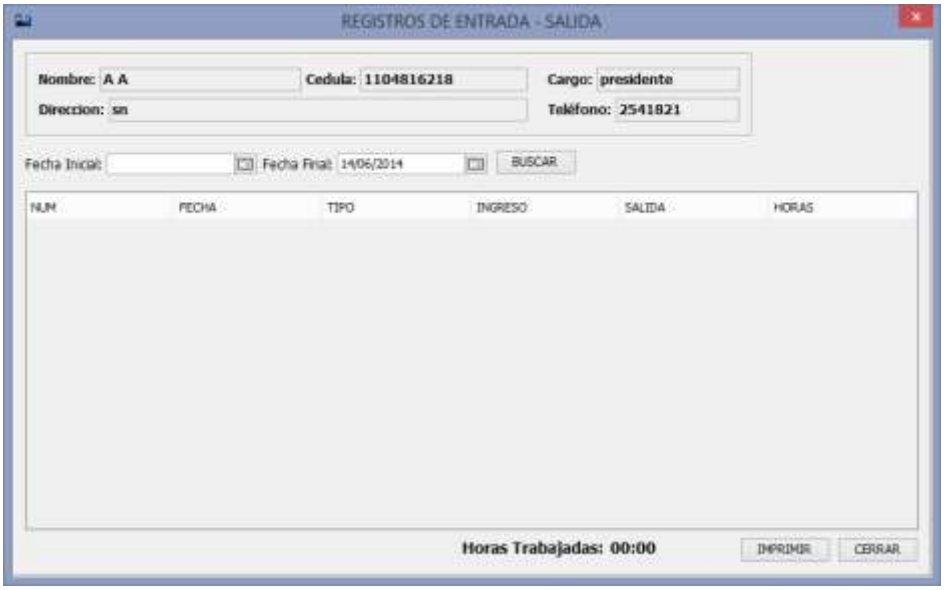

Ilustración 10. Reporte de registros de entrada-salida de usuarios.

10. En cámara tenemos las opciones de elegir la cámara en caso de que exista más de una. Además encender, apagar y mostrar cámara. Mostrar cámara permite activar la cámara cuando este desactivada.

Ilustración 11. Seleccionar la cámara con la que se desea trabajar.

11. En configuración tenemos:

- Número de cámaras, donde se ubica el número de cámaras que tenemos en el sistema pero no podemos trabajar con todas a la vez.
- Rango de similitud Eigenface permite que el administrador ubique este valor de acuerdo a las características del lugar donde son capturadas las imágenes para el reconocimiento.

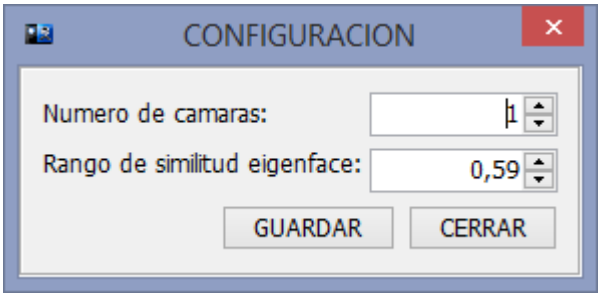

Ilustración 12. Número de cámaras y rango de similitud Eigenface.

12. En acerca de tenemos información personal de la desarrolladora de la aplicación.

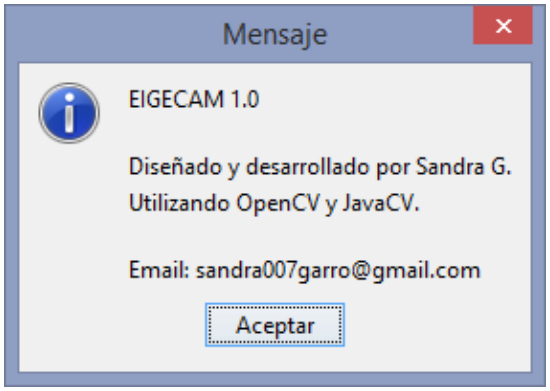

Ilustración 13. Información de la desarrolladora.

13. Para registrar la hora entrada-salida la cámara tiene que estar encendida, el rostro centrado en la imagen y dar clic en registrar.

Ilustración 14. Pantalla con cámara encendida.

14. Para poder acceder a los reportes de los registros de horarios vamos a archivo, horario y registros. Aquí buscamos al usuario y luego seleccionamos la fecha del reporte. Además tenemos la opción de guardar el reporte en algunos formatos o imprimirlo directamente.

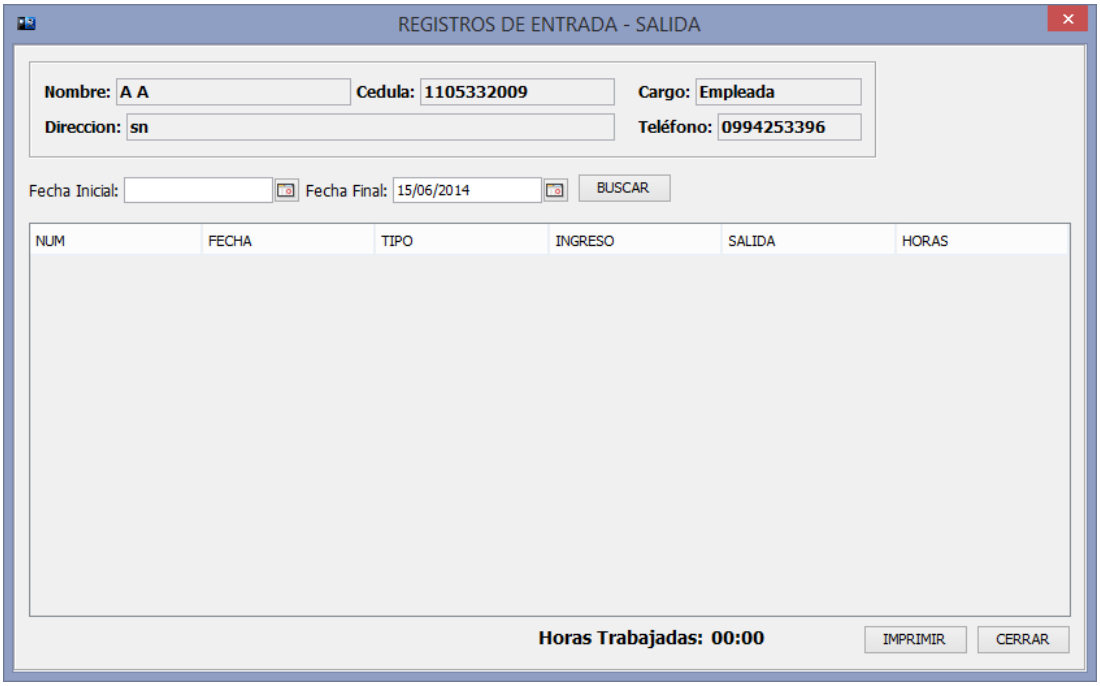

Ilustración 15. Registro de entrada-salida.

ANEXO D. ANTEPROYECTO